Michael K. Bergman

A Knowledge Representation Practionary

Guidelines Based on Charles Sanders Peirce

Springer

A Knowledge Representation Practionary

Michael K. Bergman

A Knowledge Representation Practionary

Guidelines Based on Charles Sanders Peirce

Springer

Michael K. Bergman
Cognonto Corporation
Coralville, IA, USA

ISBN 978-3-319-98091-1 ISBN 978-3-319-98092-8 (eBook)
https://doi.org/10.1007/978-3-319-98092-8

Library of Congress Control Number: 2018954477

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

To
Wendy

Preface

Human language is not the starting point for knowledge representation. Our utterances or our symbols are not the basis for what we desire to convey; they are only representations. Knowledge, the actionable side of information, is rooted in something more fundamental than language. What that something may be is what this book is about.

Competing factions have claimed truth since at least the beginning of communication. Who knows, maybe bees, whales, dingoes, and apes also have communities believing different things as true, perhaps even leading to conflict. As humans, we know from wars, missed opportunities, and personal misunderstandings the tragedy that different premises of truth may bring. We have to admit if we want to represent human knowledge to computers that we humans have not done such a hot job representing knowledge to ourselves. Since we are starting out on a journey here to explore knowledge representation (KR) for knowledge management, artificial intelligence, and other purposes, more than a bit of humility seems in order.

Information, by no means a uniformly understood concept, arises from a broader context than gestures, symbols, or sounds. For some, information is energy or when missing is entropy, the nuts-and-bits of messages. For some, information is meaning. That we continue to use 'information' in these senses, and more, in fact, tells us these senses are properly within the boundaries of the concept. Still, even if we can clear the hurdle of grokking information, we have the next obstacle of deciphering what is knowledge, that which next lies directly on our path. Further, of course, we then need to record somehow and convey all of this if we are to represent the knowledge we have gained to others. Like I say, if we have a hard time communicating all of this to other humans, what can we say about our ability to do so to machines and AI?

But maybe I overthink this. Any tasks us humans do using information that we can automate with acceptable performance may lead to more efficiency and perhaps more job satisfaction for the workers involved, maybe even more wealth. Conversely, maybe this automation leads to loss of jobs for the workers. I do know, however, if we are ever to rely upon machines to work on our behalf, requiring little or no oversight, then we need to figure out what this knowledge is and how to represent it to the machine. Such is the task of KR. What I try to provide in this book is a way to

think and a practical guidebook of sorts for how to approach the questions of computers and knowledge.

The world is real. It exists independent of us or how we may think about it, though our thoughts are also part of our reality. Human history fills but a small thimble yet through the application of reason and truth-testing, including, since the Enlightenment, the scientific method, we humans have increasingly unveiled the truths of Nature, in the process creating wealth and comfort never before seen. Artificial intelligence (AI) will undoubtedly accelerate this trend. How fast that acceleration occurs is, in part, a function of how good we get at representing our knowledge. These representations are the encodings by which intelligent machines will work on our behalf. My quest in this treatise is to help promote this trend. I believe this quest to be noble and, in any case, inevitable. I believe there is something in our nature that compels us to pursue the path of useful information leading to knowledge.

The past decade was a golden one in advances in AI. We can now voice commands and requests to our phones and devices acting as virtual assistants. We are on the verge of self-driving vehicles and automation of routine knowledge worker tasks. Still, the deep learning that underlies many of these advances is an opaque, black box of indecipherable inferences. We don't know why some of this magic works or what the representations are upon which machines draw these inferences. For further advances to occur, for general AI or cognition to arise in silico, I believe we will need better ways to represent knowledge, reflective of the nature of information and its integral role in the real world.

I have had a passion for the nature and role of information throughout my professional life. I originally trained as an evolutionary biologist and population geneticist. Since my graduate days, I have replaced my focus on biological information with one based on digital information and computers. My passion has been on the role of information—biological or cultural—to confer adaptive advantage to deal with an uncertain future and as a means of generating economic wealth. My intuition—really, my underlying belief—is that there are commonalities between biological and cultural information. I have been seeking insights into this intuition for decades.

One of my first forays into information technology was a data warehousing venture, where the idea was to find ways to connect structured databases that, in native form, were stand-alone and unconnected. This venture coincided with the explosive growth of the initial Internet. To support the exploding content, we observed that large content suppliers were populating their web sites with searchable, dynamic databases, hidden from the search engines of that time (before Google's inception). We named this phenomenon the 'deep web' and did much to define its huge extent and figure out ways to mine it. We saw that, in aggregate, the web was becoming a giant, global data warehouse, though largely populated by text content and less so by structured data. We shifted our venture emphasis to text and discovery. This shift raised the perplexing question of how to place information in text onto a common, equal basis to the information in a database, such as a structured record. (Yeah, I know, kind of a weird question.)

Tim Berners-Lee, inventor of the World Wide Web, and colleagues put forward a vision of the Semantic Web in a *Scientific American* article in 2000.[1] The article painted a picture of globally interconnected data leveraged by agents or bots designed to make our lives easier and more automated. The late Douglas Adams, of *Doctor Who* and *The Hitchhiker's Guide to the Galaxy* fame, had presciently produced a fascinating and entertaining TV program on the same topic for BBC2 about 10 years earlier. Called *Hyperland*, you can see this self-labeled 'fantasy documentary' from 1990 in its entirety on YouTube. The 50-min presentation, written by and starring Adams as the protagonist having a fantasy dream, features Tom, the semantic simulacrum (actually, Tom Baker from *Doctor Who*). Tom is the "obsequious, and fully customizable" personal software agent who introduces, anticipates, and guides Adams through what is a Semantic Web of interconnected information. Laptops (actually an early Apple), pointing devices, icons, and avatars sprinkle this *tour de force* in an uncanny glimpse into the (now) future.

One of the premises of the Semantic Web is to place what we now call unstructured, semi-structured, and structured information onto a common footing. The approach uses the RDF (Resource Description Framework) data model. RDF provided an answer to my question of how to combine data with text. I am sure there were other data models out there at the time that could have perhaps given me the way forward, but I did not discover them. It took RDF and its basic *subject-predicate-object* (*s-p-o*) 'triple' assertion to show me the way ahead. It was not only a light going on once I understood but the opening of a door to a whole new world of thinking about knowledge representation.

The usefulness of ideas behind the Semantic Web and the semantic technologies supporting it lured me to switch emphasis again. I founded a new company with Frédérick Giasson, and we proceeded to provide semantic technology solutions to enterprises over the next 10 years. The Web today is almost unrecognizable from the Web of 15 years ago. If one assumes that Web technologies tend to have a 5-year or so period of turnover, we have gone through three to four generations of change on the Web since the initial vision for the Semantic Web.

Many of our engagements were proprietary, though we did provide three notable open source projects. We developed a general semantic platform for ontology (knowledge graph) and data management, the still-active Open Semantic Framework project. To help information interoperate, we created UMBEL, a subset of Cyc and a contributor to our current efforts, as a set of reference concepts that users can share across different Web datasets. Based on that experience, we designed a successor reference knowledge structure, KBpedia, a combination of upper knowledge graph and leading public knowledge bases. We talk much about KBpedia throughout since it is this book's reference knowledge structure.

The marrying of electronic Web knowledge bases—such as Wikipedia or internal ones like the Google search index or its Knowledge Graph—with improvements in machine learning algorithms is systematically mowing down what used to be called

[1] Berners-Lee, T., Lassila, O., and Hendler, J., "The Semantic Web," *Scientific American Magazine*, 2001.

the Grand Challenges of computing, such as machine translation or language understanding. Sensors are also now entering the picture, from our phones to our homes and our cars, that exposes the higher-order requirement for data integration combined with semantics. Natural language processing (NLP) kits have improved in accuracy and execution speed; many semantic tasks such as tagging or categorizing or questioning already perform at acceptable levels for most projects. We naturally call the marriage of these knowledge sources with AI 'knowledge-based artificial intelligence.' KBAI is one of the potential payoffs that would arise from better ways to represent knowledge and thus is a common theme throughout the book.

Combining information goes beyond the technical challenges of matching forms and formats. We need to tackle the question of meaning, inextricably entwined with context and perspective. Cinemaphiles will readily recognize Akira Kurosawa's *Rashomon* film of 1951. In the 1960s, one of the most popular book series was Lawrence Durrell's *The Alexandria Quartet*. Both, each in its way, tried to get at the question of what is the truth by telling the same story from the perspective of different protagonists. Whether you saw Kurosawa's movie or read Durrell's books, you know the punchline: truth is very different depending on the point of view and experience—including self-interest and delusion—of each protagonist.

All of us recognize this phenomenon of the blind man's view of the elephant. The problem we are trying to solve is how to connect information meaningfully. For that, we need to somehow capture the ideas of perspective and context, as well as the usual vagaries of imprecise semantics. Root cause analysis for what it takes to achieve meaningful, interoperable information suggests one pivotal factor is to describe source content adequately in context to its use. Capturing and reflecting context is essential if we are to get information sources to work together, a capability we give the fancy label of 'interoperability.' We also need to assemble and represent this information such that we can reason over it and test new knowledge against it, a structural form we call a 'knowledge graph.' All of this requires a logical and coherent theory—a *grounding*—for how to represent knowledge.

Our client efforts over the past decade were converging on design thoughts about the nature of information and how to signify and communicate it. The bases of an overall philosophy regarding our work emerged around the teachings of Charles Sanders Peirce and Claude Shannon, each explicating one of the boundary senses of information. Shannon emphasized the message and mechanical aspects of information; Peirce emphasized meaning in both breadth and depth. In the combination, we see semantics and groundings as essential to convey accurate messages. Simple forms, so long as they are correct, are always preferred over complex ones because message transmittal is more efficient and less subject to losses (inaccuracies). How we could represent these structures in graphs affirmed the structural correctness of our design approach. The now visible reawakening of artificial intelligence helps to put the Semantic Web in its proper place: a key subpart, but still a subset, of AI.

I first encountered Charles S. Peirce from the writings of John Sowa about a decade ago. Sowa's writings are an excellent starting point for learning about logic

and ontologies, especially his articles on Peirce and signs.[2] Early on it was clear to me that knowledge modeling needed to focus on the inherent meaning of things and concepts, not their surface forms and labels. Sowa helped pique my interest that Peirce's theory of semiotics was perhaps the right basis for getting at these ideas.

In the decade since that first encounter, I have based some writings on Peirce's insights. I have developed a fascination with his life and teachings and thoughts on many topics. I have become convinced that Peirce—an American philosopher, logician, scientist, and mathematician—was possibly one of the greatest thinkers ever. While the current renaissance in artificial intelligence can certainly point to the seminal contributions of George Boole, Shannon, Alan Turing, and John von Neumann in computing and information theory (among many others), my view, not alone, is that C.S. Peirce belongs in those ranks from the perspective of knowledge representation, the *meaning* of information, and hewing to reality.

The importance of studying Peirce for me has been to tease out those principles, design bases, and mindsets that can apply Peircean thinking to the modern challenge of knowledge representation. This knowledge representation is like Peirce's categorization of science or signs but is broader still in needing to capture the nature of relations and attributes and how they become building blocks to predicates and assertions. In turn, we need to subject these constructs to logical tests to provide a defensible basis for what is knowledge and truth given the current information. Then, all of these representations need to be put forward in a manner (symbolic representation) that is machine readable and computable.

In reading and studying Peirce for more than a decade, it has become clear that he had insights and guidance on every single aspect of this broader KR problem. My objective has been to take these piece parts (Peirce parts?) and recombine them into a whole consistent with Peirce's *architectonic*. How can Peirce's thinking be decomposed into its most primitive assumptions to build up a new KR representation? These are the points I argue in the book while also sharing the experience of how we may integrate these viewpoints into working knowledge management systems.

I have no intent for balance in this exposition. There are wonderful textbooks and handbooks available if you are seeking a neutral presentation on knowledge representation in computer and information science. The lens I use is strictly that of Peirce and his views that contribute to an understanding of knowledge representation, at least how I read and understand those views. Peirce further guides the scope and organization of this book. One of Peirce's signal contributions was the philosophy of pragmatism, according to a specific maxim and a recommended methodology to follow, what the Peirce scholar Kelly Parker calls a 'practionary.' To my knowledge, this book employs this Peircean methodology for the first time. Given this emphasis, we will by necessity need to tackle many Peircean concepts, some with arcane or jaw-breaking labels. That is a small price to pay to gain entry into Peirce's brilliant insights.

I also minimize math and equations in the book. I provide many salient references for exploring topics further. I try to emphasize how to think and organize.

[2] Use https://www.google.com/search?as_q="peirce"&as_sitesearch=jfsowa.com for a listing.

I avoid cookbook steps or prescriptive techniques or methods. I do not recommend specific tools. Rather, because of the coherence of Peirce's views, I use how I understand him and his writings, including interpretations by others, to bring a consistent approach, logic, and mindset to the question of knowledge representation. By straddling today's two separate worlds of Peirce scholarship and knowledge representation, I perhaps risk disappointing both camps. One of my points, though, is that the camps should be separated no longer.

I would first like to thank my colleague and business partner, Frédérick Giasson, for his creativity and effort in our commercial ventures over the past decade. He was not only the implementer of the many systems we developed, and a constant fount of ideas and innovation, but a great friend and a calm and cool influence during those engagements. Though I am the recorder of the results in this book, he deserves co-billing for why and how this book came into being.

I want to thank those who have encouraged me over many years to write this book, including from many commenters on my *AI3:::Adaptive Information* blog. I especially thank Fred, Steve Ardire, Alianna Maren, Alan Morrison, Amit Sheth, and Peter Yim for their encouragement. I further thank Amit for his kind efforts to help me find and secure a publisher.

I thank my former colleague, Jacquie Bokow, for early editorial assistance and advice. I much appreciate the complete and detailed reviews I got on the first draft from Michael Buckland, Scott David, Rob Hillard, John Huntley, and Jack Park. I am grateful for the commentary and errors found in my readings of Peirce from Jon Alan Schmidt and Edwina Taborsky, as well as insights I have gained from the Peirce-L discussion group. I further thank William Anderson, Andreas Blumauer, Fred Giasson, Alan Morrison, Amit Sheth, Aleksander Smywiński-Pohl, Bobbin Teagarden, and Tom Tiahrt for their reviews and commentary. Despite their best efforts to find and correct my errors and to make great suggestions, I am sure that errors remain, which are entirely my responsibility. I ask your forbearance for any errors or oversights. I lastly thank Susan Lagerstrom-Fife and Caroline Flanagan for helping to shepherd the manuscript through the publication process.

I find it wondrous that the human species has come to learn and master symbols. That mastery, in turn, has broken the shackles of organic evolution and has put into our hands and minds the very means and structure of information itself. The *lingua franca* for doing so is knowledge representation, best done, I believe, following the guidelines of Charles Sanders Peirce.

Coralville, IA
October 2018

Michael K. Bergman

In-line Citations

Here are the conventions and sources used for quotations for Peirce's writings as used in the book. Items separated by a period or colon are page or clause number.

Abbrv.	Example	Source
CP	CP 1.343	Peirce, C. S., *The Collected Papers of Charles Sanders Peirce, 8 Volumes*, Cambridge, MA: Harvard University Press, 1931.
EP	EP 2:43	Peirce, C. S., *The Essential Peirce: Selected Philosophical Writings, Vol 1 (1867–1893)*, Bloomington: Indiana University Press, 1992. Peirce, C. S., *The Essential Peirce: Selected Philosophical Writings, Vol 2 (1893–1913)*, Bloomington: Indiana University Press, 1998.
MS	MS 32	Robin, R. S., *Annotated Catalogue of the Papers of Charles S. Peirce*, Amherst, Massachusetts: The University of Massachusetts Press, 1967. Robin, R. S., "The Peirce Papers: A Supplementary Catalogue," *Transactions of the Charles S. Peirce Society*, 1971, pp. 37–57.
NEM	NEM 4:83	Peirce, C. S., *The New Elements of Mathematics by Charles S. Peirce*, Hague, Netherlands: The, Mouton Publishers, 1976 (four volumes).
W	W 3:266	Peirce, C. S., *The Writings of Charles S. Peirce—Chronological Editions Vo1 8,* compiler, Peirce Edition Project, Bloomington: Indiana University Press, 1982–2009 (by volume).

I have attempted to date each Peirce quote, given the current tendency in scholarship and its usefulness to place his views into a chronology. Unlike most practice, I list the year first and then the citation.

Permission
Significant portions of the material in this book were first published on the author's *AI3:::Adaptive Information* blog, at http://mkbergman.com. We thank the author for the permission to use this copyrighted material.

Contents

Chapter 1
Introduction

Knowledge representation, of course, deals with knowledge, itself based on information. Knowledge representation is shorthand for how to represent human symbolic information and knowledge to computers, preferably in the most effective manner. Formally, and the working definition for this book, *knowledge representation*[1] is a field of artificial intelligence dedicated to representing information about the world in a form that a computer system can utilize to solve complex tasks. KR applications range from semantic technologies and machine learning and artificial intelligence to information integration, data interoperability, and natural language understanding.

I am not even-handed in this book. My explicit purpose is to offer a fresh viewpoint on knowledge representation and ontology engineering, informed by a variety of projects over the past dozen years, and guided by the principles of Charles Sanders Peirce, as I best understand them. Many others have different perspectives on knowledge representation. For more balance and to understand this diversity, I recommend the excellent KR reference texts by van Harmelan [1] or Brachman and Levesque [2].

C.S. Peirce (1839–1914), pronounced 'purse,' was an American logician, scientist, mathematician, and philosopher of the first rank. His profound insights and writings spanned a half-century, and cover topics ranging from the nature of knowledge and epistemology to metaphysics and cosmology.[2] His universal categories of Firstness, Secondness, and Thirdness provide the mindset and theories that guide this book. Peirce, along with Gottlob Frege, is acknowledged as a founder of predicate calculus, to which Peirce provided a notation system, and which formed the basis of first-order logic. Peirce's theory of signs and sign-making, semiosis, is a seminal understanding of icons, indexes, and symbols, and the way we perceive and

[1] Many of the italicized terms in this book are defined when first used and listed in the *Glossary*.

[2] Appendix A, from which I borrow these two sentences, is a summary biography and reading suggestions for Charles Sanders Peirce. He is also referenced in the literature as Peirce, Charles Peirce, C.S. Peirce, or CSP.

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8_1

understand objects. Peirce's semiosis (*semeiosis*, his preferred spelling) and approach arguably provide the logical basis for description logics and other aspects underlying the semantic Web building blocks of the RDF data model and, eventually, the OWL language. Peirce is the acknowledged founder of pragmatism, the philosophy of linking practice and theory in a process akin to the scientific method. He was also the first formulator of existential graphs, a basis to the field now known as model theory [3], and the basis for conceptual graphs, a KR formalism. No aspect of knowledge representation exceeded his grasp.

This book also weaves the open-source knowledge artifact, KBpedia, through its later chapters and observations. KBpedia combines the information from multiple public knowledge bases, prominently including Wikipedia and Wikidata, under the conceptual structure of the KBpedia Knowledge Ontology (KKO), a *knowledge graph* organized according to the Peircean universal categories. KBpedia's 55,000 reference concepts, classified into 85 mostly separate *typologies*, and with access to millions of notable *entities* and *events*, is a modular resource that may be leveraged or expanded for particular domain purposes. However, the confederation between this book and KBpedia is loose. Each stands on its own without reliance on the other.

We have witnessed enormous and mind-boggling strides over the past decade in *artificial intelligence*. *Machine learning* has leveraged massive *knowledge bases* to deliver breakthrough capabilities in automated question answering and intelligent virtual assistants. Deep learning, with its mostly indecipherable black-box layers, has enabled automatic recognition of voice, images, and patterns at speeds and accuracies often exceeding that of humans.

Still, we struggle to integrate information, get data to interoperate, or discover or manage knowledge. Our current AI techniques appear close to reaching limits, including whether we even understand what those techniques are doing. Peircean ideas hold the tantalizing prospect to unlock better ways to represent knowledge. KR is the foundation upon which, I believe, next breakthroughs will come. I believe Peircean ideas provide the way to better represent human knowledge such that AI-powered computers can organize, index, reference, and cross-check information in any digital form. This prospect will obliterate current boundaries to information sharing. If the past is a guide, innovation, transformation, and wealth will follow.

Structure of the Book

This book is structured into parts and chapters. The central portion of the book (Parts II–IV) reflects C.S. Peirce's universal categories of Firstness, Secondness, and Thirdness. Across nearly five decades of writings, Peirce likens the universal categories to more than 60 different expressions (Table 6.2). The expression used for this central portion of the book is Peirce's logic triad of *grammar* (1ns), logics

and tools (or *critic*) (2ns), and methods (or *methodeutic*) (3ns).[3] We use this triadic organization to explain the what and how for a working knowledge representation system, with frequent reference to KBpedia.

Parts I and V are bookends around this central portion. Part I, the opening bookend, provides the context for why one should be interested in the topic of knowledge representation and what kind of functions KR should fulfill. Part V, the closing bookend, provides practical speculation for what kinds of benefits and applications may result from a working KR system built according to Peircean principles. A couple of chapters tee up this structure.

The structural approach of this book is consistent with Peirce's *pragmatic maxim* to achieve the "third grade of clearness of apprehension" (1878, W 3:266)[4] covering "all of the conceivable practical effects," regarding an understanding of something. If a *dictionary* is for the definition of terms, a *practionary* is for the definition of methods and potential applications resulting from an explication of a domain. In the case of this book, that domain is *knowledge representation*.[5]

To my knowledge, this is the only Peirce book dedicated solely to knowledge representation, and the only KR book exclusively devoted to Peirce.[6] Some reviewers of drafts of this book have suggested splitting the book into multiple parts. I admit there is some logic to that suggestion. Early chapters discuss contexts of information theory, economics, and social circumstances. Middle parts of the book are theoretical, even philosophical, that evolve into how-to and practice. The latter parts of the book are speculative and span potential applications in breadth and depth. My answer in keeping these parts together is to try to be faithful to this overall ideal of a Peircean *practionary*. I welcome you to a soup-to-nuts banquet of Peircean perspectives on the challenge of knowledge representation.

Overview of Contents

Before we start the formal structure of the book, we begin with this chapter and then Chap. 2 discussing the core concepts of *information*, *knowledge*, and *representation*. Gregory Bateson defined information as the "difference that makes a difference." Claude Shannon, the founder of information theory, emphasized the

[3] 1ns, 2ns, and 3ns are shorthand for Firstness, Secondness, and Thirdness, respectively.

[4] See the note on Abbreviations after the *Preface* for the citations format for the Peirce quotations used throughout.

[5] The term of *practionary* comes from Kelly Parker based on his study of Peirce [3]; I thank him for graciously allowing me to use the term.

[6] John Sowa's 1999 book, *Knowledge Representation: Logical, Philosophical, and Computational Foundations* (Brooks Cole Publishing Co., Pacific Grove, CA, 2000), was much influenced by Peirce. Sowa in his work on conceptual graphs builds directly from Peirce's existential graphs. However, Sowa's book is not based exclusively on Peirce, nor is his ontology (see http://www.jfsowa.com/ontology/toplevel.htm). Still, Sowa's is the closest Peirce-KR treatment to my knowledge without being solely based on him.

engineering aspect of information, defining it as a message or sequence of messages communicated over a channel; he specifically excluded meaning. Peirce emphasized meaning and related it to the triadic relationship between immediate object, representation, and interpretation. We associate knowledge and its discovery with terms such as open, dynamic, belief, judgment, interpretation, logic, coherence, context, reality, and truth. Peirce's pragmatic view is that knowledge is fallible information that we believe sufficiently upon which to act. I argue in this book, consistent with Peirce, that knowledge representation is a complete triadic sign, with the meaning of the information conveyed by its symbolic representation and context, as understood and acted upon by the interpreting agent. A challenge of knowledge representation is to find structured representations of information—including meaning—that can be simply expressed and efficiently conveyed.

We then begin the structural portions of the book. Part I and its three chapters attempt to place knowledge representation, as practiced today, in context. Chapter 3 describes the situation and importance of information to enterprises and society. Knowledge representation is a primary driver for using computers as a means to improve the economic well-being of all peoples. Solow, a student of Schumpeter, had the insight into two papers in the 1950s, for which he won a Nobel Prize, that technological change is the 'residual' leftover from empirical growth once we remove the traditional inputs of labor and capital. This residual is what we now call total-factor productivity. Romer's subsequent work internalized this factor as a function of information and knowledge, which in contrast became the endogenous growth model. Innovation and its grounding in knowledge had finally assumed its central, internal role in economists' understanding of economic growth. Unlike the historical and traditional ways of measuring assets—based on the tangible factors of labor, capital, land, and equipment—information is an intangible asset. If we are to improve our management and use of information, we need to understand how much value we routinely throw away.

Once we survey the situation, Chap. 4 begins to surface some of the opportunities. The path to knowledge-based artificial intelligence (KBAI) directly coincides with a framework to aid data interoperability and responsive knowledge management (KM). A knowledge graph (or ontology) provides the overall schema, and semantic technologies give us a basis to make logical inferences across the knowledge structure and to enable tie-ins to new information sources. We support this graph structure with a platform of search, disambiguation, mapping, and transformation functions, all of which work together to help achieve data interoperability. KBAI is the use of large statistical or knowledge bases to inform feature selection for machine-based learning algorithms. We can apply these same techniques to the infrastructural foundations of KBAI systems in such areas as data integration, mapping to new external structure and information, hypothesis testing, diagnostics and predictions, and myriad other uses to which researchers for decades hoped AI would contribute. We apply natural language processing to these knowledge bases informed by semantic technologies.

To complete the context, we discuss other vital precepts (or premises) in Chap. 5. Knowledge should express a coherent reality, to reflect a logical consistency and

structure that comports with our observations about the world. How we represent reality has syntactic variation and ambiguities of a semantic nature that can only be resolved by context. A hub-and-spoke design with a canonical data model is a superior way to organize, manipulate, and manage input information. By understanding the sources of semantic heterogeneity, we set the basis for extracting meaning and resolving ambiguities. Once we resolve ('disambiguate') the source information, we need to organize it into 'natural' classes and relate those classes coherently and consistently to one another. This organization takes the form of a knowledge graph. Traditional relational databases do not; they are inflexible and fragile when the nature (schema) of the world changes, and require expensive re-architecting in the face of new knowledge or new relationships.

We next embark on the central portion of our thesis, Parts II–IV. Part II covers the grammar of knowledge representation. I discuss in detail Peirce's universal categories of Firstness, Secondness, and Thirdness in Chap. 6. The ideas behind Peircean pragmatism are how to think about signs and representations (*semiosis*); logically reason and handle new knowledge (*abduction*) and probabilities (*induction*); make economic research choices (*pragmatic maxim*); categorize; and let the scientific method inform our inquiry. The connections of Peirce's sign theory, his threefold logic of deduction-induction-abduction, the importance of the scientific method, and his understanding about a community of inquiry have all fed my intuition that Peirce was on to some fundamental insights suitable to knowledge representation. We can summarize Firstness as unexpressed possibilities; Secondness as the particular instances that may populate our information space; and Thirdness as general types based on logical, shared attributes. Scholars of Peirce acknowledge how infused his writings on logic, semiosis, philosophy, and knowledge are with the idea of 'threes.' Understanding, inquiry, and knowledge require this irreducible structure; connections, meaning, and communication depend on all three components, standing in relation to one another and subject to interpretation by multiple agents in multiple ways.

Our next topic within the *speculative grammar* of the KR space addresses basic terminology, which we cover in Chap. 7. We begin our analysis with the relevant 'things' (nouns, which are *entities, events*, *types*, or *concepts*) that populate our world and how we organize them. We pair these things with three kinds of internal and external relations to other things. Attributes are the intensional characteristics of an *object*, event, entity, type (when viewed as an *instance*), or concept. External relations are actions or assertions between an event, entity, type, or concept and another particular or general. Representations are signs and the means by which we point to, draw attention to, or designate, denote, or describe a specific object, entity, event, type, or general. We now know that *attributes* are a Firstness in the universal categories; that Secondness captures all events, entities, and relations; and that Thirdness provides the types, context, meaning, and ways to indicate what we refer to in the world.

Chapter 8 presents the logic basis and introduces the actual vocabularies and languages to express this grammar. Knowledge graphs and knowledge bases need to be comprehensive for their applicable domains of use, populated with 'vivid'

knowledge. We use deductive logic to infer hierarchical relationships, create forward and backward chains, check if domains and ranges are consistent for assertions, assemble attributes applicable to classes based on member attributes, conform with transitivity and cardinality assertions, and test virtually all statements of fact within a knowledge base. We want a knowledge representation (KR) language that can model and capture intensional and extensional relations; one that potentially embraces all three kinds of inferential logic; that is decidable; one that is compatible with a design reflective of particulars and generals; and one that is open world in keeping with the nature of knowledge. Our choice for the knowledge graph is the W3C standard of OWL 2 (the Web Ontology Language), though other choices may be just as valid.

Using this grammatical and language foundation, Part III transitions to discuss the working components of a KR system. In Chap. 9, I argue the importance of openness and keeping an open design. Open content works to promote derivative and reinforcing factors in open knowledge, education, and government. Open standards encourage collaboration and make it easier for data and programs to interoperate. Open data in public knowledge bases are a driver of recent AI advances in knowledge. Open also means we can obtain our knowledge from anywhere. Our knowledge graphs useful to a range of actors must reflect the languages and labels meaningful to those actors. We use reference concepts (RCs) to provide fixed points in the information space for linking with external content. We now introduce KBpedia to the remainder of the discussion. We use RDF as a kind of 'universal solvent' to model most any data form. We match this flexible representation with the ability to handle semantic differences using OWL 2, providing an open standard to interoperate with open (or proprietary) content.

In Chap. 10, we shift the emphasis to modular, expandable typologies. The idea of a SuperType is equivalent to the root node of a typology, wherein we relate multiple entity types with similar essences and characteristics to one another via a natural classification. Our typology design has arisen from the intersection of (1) our efforts with SuperTypes to create a computable structure that uses powerful disjoint assertions; (2) an appreciation of the importance of entity types as a focus of knowledge base terminology; and (3) our efforts to segregate entities from other constructs of knowledge bases, including attributes, relations, and annotations. Unlike more interconnected knowledge graphs (which can have many network linkages), typologies are organized strictly along these lines of shared attributes, which is both simpler and also provides an orthogonal means for investigating type-class membership. The idea of nested, hierarchical types organized into broad branches of different entity typologies also offers a flexible design for interoperating with a diversity of worldviews and degrees of specificity.

Typologies are one component of our knowledge graphs and knowledge bases, to which we shift our attention in Chap. 11. Relations between nodes, different than those of a hierarchical or subsumptive nature, provide still different structural connections across the knowledge graph. Besides graph theory, the field draws on methods including statistical mechanics from physics, data mining and information visualization from computer science, inferential modeling from statistics, and social

structure from sociology. Graph theory and network science are the suitable disciplines for a variety of information structures and many additional classes of problems. We see the usefulness of graph theory to linguistics by the various knowledge bases such as WordNet (in multiple languages) and VerbNet. Domain ontologies emphasize conceptual relationships over lexicographic ones for a given knowledge domain. Furthermore, if we sufficiently populate a knowledge graph with accurate instance data, often from various knowledge bases, then ontologies can also be the guiding structures for efficient machine learning and artificial intelligence. We want knowledge sources, preferably knowledge bases, to contribute the actual instance data to populate our ontology graph structures.

We have now discussed all of the conceptual underpinnings to a knowledge representation *system*. Part IV, also spread over three chapters, presents how these components are now combined to build a working platform. In Chap. 12, we outline the basic KR platform and the accompanying knowledge management (KM) capabilities it should support. The platform should perform these tasks: insert and update concepts in the upper ontology; update and manage attributes and track specific entities as new sources of data are entered into the system; establish coherent linkages and relations between things; ensure that these updates and changes are done wholly and consistently while satisfying the logic already in place; update how we name and refer to things as we encounter variants; understand and tag our content workflows such that we can determine provenance and authority and track our content; and do these tasks using knowledge workers, who already have current duties and responsibilities. These requirements mean that use and updates of the semantic technologies portion, the organizing basis for the knowledge in the first place, must be part of daily routines and work tasking, subject to management and incentives.

Once a platform is available, it is time to build out the system, the topic of Chap. 13. Critical work tasks of any new domain installation are the creation of the domain knowledge graph and its population with relevant instance data. Most of the implementation effort is to conceptualize (in a knowledge graph) the structure of the new domain and to populate it with instances (data). In a proof-of-concept phase, the least-effort path is to leverage KBpedia or portions of it as is, make few changes to the knowledge graph, and populate and test local instance data. You may proceed to create the domain knowledge graph from pruning and additions to the base KBpedia structure, or from a more customized format. If KBpedia is the starting basis for the modified domain ontology, and if we test for logic and consistency as we make incremental changes, then we are able to evolve the domain knowledge graph in a cost-effective and coherent manner.

Before releasing for formal use, the system and its build-outs should be tested in various ways and developed using best practices. Chapter 14 addresses these needs. The problems we are dealing with in information retrieval (IR), natural language understanding or processing (NLP), and machine learning (ML) are all statistical classification problems, specifically in binary classification. The most common scoring method to gauge the 'accuracy' of these classification problems uses statistical tests based on two metrics: negatives or positives, and true or false. We discuss a variety of statistical tests using the four possible results from these two metrics

(e.g., false positive). We offer best practices learned from client deployments in areas such as data treatment and dataset management, creating and using knowledge structures, and testing, analysis, and documentation. Modularity in knowledge graphs, or consistent attention to UTF-8 encoding in data structures, or the emphasis on 'semiautomatic' approaches, or the use of literate programming and notebooks to record tests and procedures are just a few of the examples where lines blur between standard and best practices.

In the concluding Part V, the last bookend in our structured organization, we tackle the "conceivable practical effects" that may result from following these pragmatic Peircean approaches. As before, three chapters comprise this part. The first two chapters present what kind of benefits and practical effects can result from following these guidelines to KR. I offer each potential use as a 'mini-story' following the same structure as the book.[7] Chapter 15 speculates on 12 potential applications in *breadth*. Four of these are near-term applications in word sense disambiguation, relation extraction, reciprocal mapping, and extreme knowledge supervision. Four are logic and representation applications in automatic hypothesis generation, encapsulating KBpedia for deep learning, measuring classifier performance, and the thermodynamics of representation itself. The last four areas in Chap. 15 include new applications and uses for knowledge graphs. Two of these, self-service business intelligence and semantic learning, have been on wish lists for years. The last two apply Peirce's ideas and guidance to nature and questions of the natural world. These examples show the benefits of organizing our knowledge structures using Peirce's universal categories and typologies. Further, with its graph structures and inherent connectedness, we also have some exciting graph learning methods that we can apply to KBpedia and its knowledge bases.

Chapter 16 discusses three potential uses in *depth*. The three application areas are workflows and business process management (BPM), semantic parsing, and robotics. Workflows are a visible gap in most knowledge management. One reason for the gap is that workflows and business processes intimately involve people. Shared communication is at the heart of workflow management, a reason why semantic technologies are essential to the task. In semantic parsing, a lexical theory needs to handle word senses, sentences and semantics, cross-language meanings, common-sense reasoning, and learning algorithms. We can map the compositional and semantic aspects of our language to the categorial perspectives of Peirce's logic and semiosis, and then convert those formalisms to distributions over broad examples provided by KBpedia's knowledge. Cognitive robots embrace the ideas of learning and planning and interacting with a dynamic world. Kinesthetic robots may also be helpful to our attempts to refine natural language understanding.

In our last Chap. 17, we are now able to draw some conclusions looking across the broad sweep of our completed *practionary*. Peirce posited a "third grade of clearness of apprehension" to better understand a topic at hand, a part of his pragmatic maxim. As was first stated, knowledge representation is a field of artificial

[7] Namely, that structure is parts organized as context and practical outcomes that are the bookends surrounding the logic triad of grammar (1ns), modes of logic (2ns), and methods (3ns).

intelligence dedicated to representing information about the world in a form that a computer system can utilize to solve complex tasks. Peirce (at least how I interpret him) offers a fresh and realistic take on the question of KR. The foundation of the universal categories and other Peircean ideas offer unique and valuable insights into semantic technologies, knowledge representation, and information science. We need to better understand the nature of signs and representation in the use of semantic technologies. More minds and more scrutiny will improve our understanding and will increase the knowledge we may derive from Peirce's ideas.

I provide supplementary material in three appendices. Appendix A is a short bio of Charles S. Peirce, a most accomplished and fascinating person. Most Peircean scholars acknowledge changes in Peirce's views over time, from his early writings in the 1860s to those at the turn of the century and up until his death in 1914. In Peirce's cosmogony, the primitives of chance (Firstness), law (Secondness), and habit (Thirdness) can explain everything from the emergence of time and space to the emergence of matter, life, and then cognition. Synechism, which Peirce equated with continuity,[8] is the notion that space, time, and law are continuous and form an essential Thirdness of reality in contrast to existing things and possibilities. Peirce made a profound contribution to mathematical logic, where he pioneered many new areas. We can also point to a second area in probability theory, then known as the Doctrine of Chances. Peirce's universal categories of Firstness, Secondness, and Thirdness provide the mindset for how to think about and organize knowledge. The appendix concludes with an annotated list of resources for learning more about Peirce.

Appendix B provides overview information on the KBpedia knowledge artifact. KBpedia is structured to enable useful splits across a myriad of dimensions from entities to relations to types that can all be selected to create positive and negative training sets, across multiple perspectives. The disjointedness of the SuperTypes that organize the 55,000 entity types in KBpedia provides a robust selection and testing mechanism. We organize KBpedia using a knowledge graph, KKO, the KBpedia Knowledge Ontology, with an upper structure based on Peircean logic. KKO sets the umbrella structure for how we relate KBpedia's six constituent knowledge bases to the system. We split the KBpedia knowledge graph into concepts and topics, entities, events, attributes, annotations, and relations and their associated natural classifications or types.

Appendix C discusses the KBpedia features suitable for use by machine learners. This systematic view, coupled with the large-scale knowledge bases such as Wikipedia and Wikidata in KBpedia, provides a basis for faster and cheaper learners across a comprehensive range of NLP tasks. For natural language, a feature may be a surface form, like terms or syntax or structure (such as hierarchy or connections);

[8] Peirce states, "I have proposed to make *synechism* mean the tendency to regard everything as continuous" (1893, CP 7.565). He goes on to say, "I carry the doctrine so far as to maintain that continuity governs the whole domain of experience in every element of it. Accordingly, every proposition, except so far as it relates to an unattainable limit of experience (which I call the Absolute,) is to be taken with an indefinite qualification; for a proposition which has no relation whatever to experience is devoid of all meaning" (CP 7.566).

it may be derived (such as statistical, frequency, weighted, or based on the ML model used); it may be semantic (in terms of meanings or relations); or it may be latent, as either something hidden or abstracted from feature layers below it. I present and organize an inventory of more than 200 feature types applicable to natural language. They include lexical, syntactical, structural, and other items that reflect how we express the content in the surface forms of various human languages.

Throughout the book, I try to stick with more timeless concepts and guidelines, rather than current tools or specific methods. Tools and methods change rapidly, with current ones rather easily identified at implementation time.[9] I also try to limit mathematical notations or overly technical discussions. The abundance of references and endnotes provided at the conclusion of each chapter or appendix offers further entry points into these topics. A glossary of technical and Peircean terms and an index conclude the book.

Key Themes

Some themes recur throughout this book. Sometimes how I discuss these concepts may differ by context. To help reduce confusion, let's tackle some of these concepts early.

The first theme is the concept of Peirce's universal categories of Firstness, Secondness, and Thirdness. I devote Chap. 6 to this concept due to its importance and prominence. Peirce's penchant for threes and his belief in the universal categories perfuse his writings across all eras. Peirce's terminology for these 'threes' differs in the contexts of sign-making (semiosis), logic, thought, phenomenology, evolution, protoplasm, information, and so on. As I have tallied across his writings to date, Peirce employs the idea of the universal categories across more than 60 different contexts (see Table 6.2). OK, then, so what is an absolute universal category?

The answer, I think, is it still depends. As I suggest in Chap. 6, perhaps the base definition comes from *hypostatic abstraction* applied to the ideas of First, Second, and Third. Still, all my suggestion does is to substitute one abstract First for another slightly different abstract Firstness. Labels seem to twist us up into literalness and miss the broader point, the one I often harken to in this book about mindset. If we look to the most grounded primitives from which all things, ideas, and concepts are built, according to Peirce, nothing seems as irreducible as one, two, and three. If we further take the understanding of our signs as built from more primitive signs, which combine into more complicated statements and arguments, we can bring Peirce's conception of the universal categories into clear focus. They are meant to inform a process of investigation, refinement, and community, each new concept and term building upon others that came before it. If we reduce that process to its most reduc-

[9] For example, for nearly a decade I started and maintained a listing of semantic technology tools that eventually grew to more than 1000 tools, called Sweet Tools. I ultimately gave up on trying to maintain the listing because of the rapid creating and abandonment of tools. Only a small percentage of these tools lasted for more than a few years.

tive level, it is pretty hard to get more primitive than Firstness, Secondness, and Thirdness. In other words, we can represent anything that we can describe, perceive, or understand using the universal categories for a given context. Our and Peirce's different ways to describe these categories depend on where we are in the representational hierarchy, which is just another way of saying context.

Given the context of knowledge representation, then, what might be the best way to label these categories of Firstness, Secondness, and Thirdness? Many of the optional expressions shown in Table 6.2 approximate this answer. Since the context of knowledge representation is the real world and what we can know and verify, let's take that perspective.

Figure 1.1 is a working conception for what the base context may be for the knowledge representation domain. The unexpressed possibilities or building blocks that might contribute to a given knowledge category I term potentialities, a Firstness.[10] (One could argue that Peirce preferred the idea of possibilities as a Firstness over potential, and good scholarly bases exist to support that contention. However, in this context, it makes sense to limit our possible building blocks to those likely for the category at hand. I think 'potential' better conveys this restriction that some possibilities are more likely for a given topic category than others.) Potentialities include any unexpressed attribute, such as shape, color, age, location, or any characterization that may apply to something in our current category.

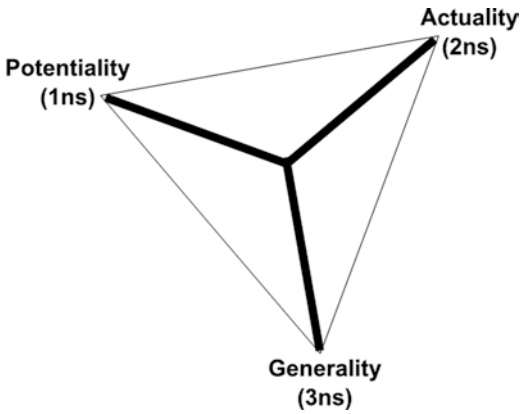


Fig. 1.1 A version of the "universal categories"

Potentialities, when expressed, are done so by the actualities of the world, a Secondness in the universal categories.[11] Actualities are the real, actual things that populate our domain, specifically including entities and events. These actual things may not have a corporal or physical existence (for example, Casper the friendly ghost, with 'fiction' being a legitimate attribute), but they can be pointed to, referred to, or described or characterized. What we find as commonalities or regularities

[10] In the KBpedia Knowledge Ontology, we term the Firstness (1ns) branch as Monads. Also, recall the earlier shorthand of 1ns, 2ns, and 3ns for the three universal categories.

[11] In the KBpedia Knowledge Ontology, we term the Secondness branch as Particulars.

across actual things we can call generalities,[12] a Thirdness in the universal categories. Generalities include types, laws, methods, and concepts that cut across many actuals or generals. Given a different context, the labeling of these universal categories may differ quite substantially, as Table 6.2 affirms. However, virtually any context invoking the universal categories would still retain some sense of these distinctions of potentiality, actuality, and generality.

(Another aspect to note in Fig. 1.1 is its central, heavier lined image, which we can describe as a three-pronged spoke or three-pointed star. Many Peirce scholars prefer this image. It is the form used by Peirce in his writings.[13] We can ascribe the lighter lined equilateral image in Fig. 1.1 to the 'meaning triangle' approach of Ogden and Richards in 1923, also apparently informed by Peirce's writings [4]. Most current Peircean practice favors the equilateral image, which I also tend to use. Though perhaps deep implications reside in the choice of image, I find either image acceptable.[14])

Given the variety of expressions for the universal categories, always ask yourself what the context is for a particular reference. As I state multiple times in the book, the universal categories are a mindset of how to decompose the signs of the world, and plumbing the use and application of the categories in different contexts is one way to better apprehend that mindset.

Another area of 'threes' in this book, but not directly related to the universal categories, is the idea of a *triple*. A triple—so named because it triply combines a *subject* to a *predicate* and to an *object* (*s-p-o*) that is the basic *statement* or assertion in the RDF and OWL languages that we use in this *practionary*. The triple is equivalent to what Peirce called a proposition. We often represent triples as barbells, with the subject and objects being the bubbles (or nodes), and the connecting predicate being the bar (or edge). Figure 1.2 is such a representation of a basic triple.

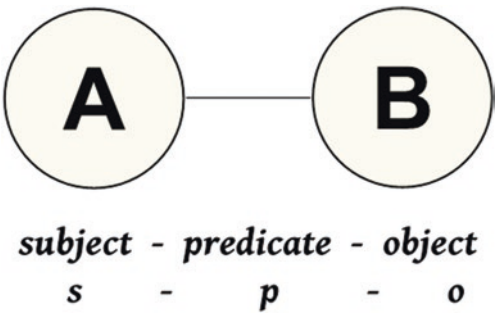


Fig. 1.2 Basic "triple"

[12] In the KBpedia Knowledge Ontology, we term the Thirdness branch as Generals.

[13] Edwina Taborsky is one vocal advocate for using the "umbrella spoke triad" image as she calls it, noting that it is open and not closed (equilateral triangle), and is the form used by Peirce.

[14] I skewed Fig. 1.1 10 degrees so as to not convey a preference or order to any of the three universal categories.

The triple statements are basic assertions such as 'ball is round' or 'Mary sister of John.' Sometimes an assertion may point to a value, such as "Mary age 8," but it also may be a true object, such as 'John citizen of Sweden.' Objects, then, in one triple statement might be the subject of a different one, such as 'Sweden located Northern Hemisphere.'

Note that I earlier likened the subject and object to *nodes* and predicates to *edges*. This terminology is the language of *graphs*. As one accumulates statements, where subjects of one statement may be an object in another or vice versa, we can see how these barbells grow linked together. When these accrete or accumulate as encountered, we have a bottom-up image of how graphs grow, as illustrated in Fig. 1.3, wherein a single statement grows to become a longer story:

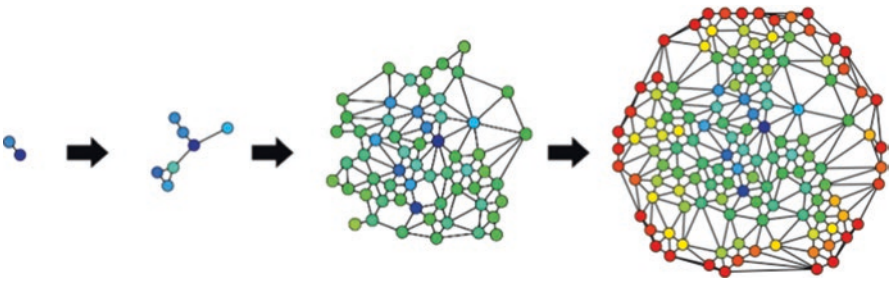


Fig. 1.3 A bottom-up view of graph growth

Of course, we can also create graphs in a top-down manner. An *upper ontology* is one example. We often intend top-down graphs to be a sort of coherent scaffolding of vetted (coherent) relationships upon which we can hang the statements for new instances. Graphs are a constant theme in this book. Chapter 11 is largely devoted to graphs and their uses. The specific kind of graph our knowledge structures assume is a DAG, a *directed acyclic graph*. This fancy term means that the edge relationships in the graph are not all transitive (both directions); one or more exhibit directionality, such as 'George father of Mary.'

Last, let me raise a crucial theme, *fallibility*. Our knowledge of the world is continually changing, and our understanding of what we believe justifies that belief may still be in error—both central tenets of Peirce. I believe that arming ourselves with how to think—and with logical methods to discover, test, select, and relate information—is the right adaptable and sustainable response to a changing world.

References

1. F. van Harmelen, V. Lifschitz, B. Porter (eds.), *The Handbook of Knowledge Representation* (Elsevier, Amsterdam, 2008)
2. R.J. Brachman, H.J. Levesque, *Knowledge Representation and Reasoning* (Morgan Kaufmann, Burlington, 2004)
3. K.A. Parker, *The Continuity of Peirce's Thought* (Vanderbilt University Press, Nashville, 1998)
4. C.K. Ogden, I.A. Richards, *The Meaning of Meaning* (Harcourt, Brace, and World, New York, 1923)

Chapter 2
Information, Knowledge, Representation

Practitioners of knowledge representation (KR) should have a shared working understanding of what the concepts of information, knowledge, and knowledge representation mean. That is the main thrust of this chapter.[1] As a symbolic species [1], we first used symbols as a way to convey the ideas of things. Simple markings, drawings, and ideograms grew into more complicated structures such as alphabets and languages. The languages came to embrace still further structure via sentences, documents, and ways to organize and categorize multiple documents, including ordered alphabets and categorization systems.

Grammar is the rules or structures that govern language. It is composed of syntax, including punctuation, traditionally understood as the sentence structure of languages, and morphology, which is the structural understanding of a language's linguistic units, such as words, affixes, parts of speech, intonation, or context. The field of linguistic typology studies and classifies languages according to their structural features. However, grammar is hardly the limit to language structure. In the past, semantics, the meaning of language, was held separate from grammar or structure. Via the advent of the thesaurus, and then linguistic databases such as WordNet and more recently concept graphs or *knowledge graphs* that relate words and terms into connected understandings, we have now come to understand that semantics also has structure. It is the marriage of the computer with language that is illuminating these understandings, enabling us to capture, characterize, codify, share, and analyze. From its roots in symbols, we are now able to extract and understand those very same symbols to derive information and knowledge from our daily discourse. We are doing this by gleaning the structure of language, which in turn enables us to relate it to all other forms of structured information.

[1] Some material in this chapter was drawn from the author's prior articles at the *AI3:::Adaptive Information* blog: "The Open World Assumption: Elephant in the Room" (Dec 2009); "Give Me a Sign: What Do Things Mean on the Semantic Web?" (Jan 2012); "The Trouble with Memes" (Apr 2012); "What is Structure?" (May 2012); "The Irreducible Truth of Threes" (Sep 2016); "The Importance of Being Peirce" (Sep 2016); "Being Informed by Peirce" (Feb 2017).

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8_2

What Is Information?

Many definitions of *information* may be found across the ages, often at variance because of what sense is primary. Some definitions are technical or engineering in nature; others emphasize intention, context, or meaning. Gregory Bateson offered one of the more famous definitions of information, claiming it the "difference that makes a difference" [2]. Claude Shannon, the founder of information theory, emphasized a different aspect of information, defining it as a message or sequence of messages communicated over a channel; he specifically segregated the meaning of information from this engineering aspect [3]. For Charles S. Peirce, information is equivalent to *meaning*, which is measurable as the breadth times the depth of the object. Despite this difference, I see both Shannon and Peirce talking broadly about the same underlying thing, though from different aspects of the universal categories. Shannon is addressing a Firstness of information, Peirce a Thirdness, as I will explain.[2]

Some Basics of Information

The idea of information has an ethereal quality. It is something conveyed that reflects a 'difference,' to use Bateson's phrase, from some state that preceded it. Indeed, Norbert Wiener, of cybernetics fame, stated in 1961 that "Information is information, not matter nor energy" [4]. By coincidence, that was also the same year that Rolf Landauer of IBM posited the physical law that all computing machines have irreversible logic, which implies physical irreversibility that generates heat. This principle sets theoretical limits to the number of computations per joule of energy dissipated. By 1991 Landauer was explicit that information was physical [5]. Physicists confirmed that data erasure is a dissipative heat process in 2012 [6]. The emerging consensus is that information processing does indeed generate heat [7]. By these measures, information looks to have a physical aspect.

The motivation of Shannon's 1948 paper on information theory was to understand information losses in communication systems or networks [3]. Much of the impetus for this came about because of issues in wartime communications and early ciphers and cryptography and the emerging advent of digital computers. The insights from Shannon's paper also relate closely to the issues of data patterns and data compression. In a strict sense, Shannon's paper was about the amount of information that could be theoretically and *predictably* communicated between a sender and a

[2] One might try to avoid the term "information" because it has multiple meanings or differing interpretations, substituting instead narrower definitions for each meaning. Moreover, multiple meanings do not have crisp boundaries, so that there is always a probability of misunderstanding or misassignment. However, when a term is commonly used, we need to accept the reality of its use. Thus, Peirce tried mightily, a practice to which I adhere, to define terms as understood and put forward. This consideration applies to any term or definition.

receiver in a message. The message communication implies no context or semantics, only the amount of information (for which Shannon introduced the term 'bits'[3]) and what might be subject to losses (or uncertainty in the accurate communication of the message). (Weaver, Shannon's later co-author of a popular version of the original paper, stated explicitly that use of the word "information must not be confused with meaning" [8].) What Shannon called 'information' is perhaps better understood by what we now call 'data.' (Of course, data has its own multiple interpretations. Bob Losee defines data as the product of a process [9]. Jonathan Furner likens data to datasets and then documents [10].)

Shannon labeled his measure of unpredictability, information entropy, as *H*. Shannon called *H entropy* because it resembled the mathematical form for Boltzmann's original definition of second law entropy (as elaborated by Gibbs, denoted as *S*, for Gibb's entropy).[4] The second law of thermodynamics expresses the tendency that, over time, differences in temperature, pressure, or chemical potential equilibrate in a closed (isolated) physical system. Thermodynamic entropy is a measure of this equilibration: for a given physical system, the highest entropy state is one at equilibrium. Fluxes or gradients arise when differences in state potentials occur. (In physical systems, these are *sources* and *sinks*; in information theory, they are *sender* and *receiver*.) Fluxes go from low to high entropy and are nonreversible—the 'arrow of time'—without the addition of external energy. Heat, for example, is a by-product of fluxes in thermal energy. In a closed system (namely, the entire cosmos), one can see this gradient as spanning from order to disorder, with the equilibrium state being the random distribution of all things. This perspective, and much schooling regarding these concepts, tends to present the idea of entropy as a 'disordered' state. Because these fluxes are directional in isolation, we see a perpetual motion machine as impossible.

Shannon's *H* is expressed as the average number of bits needed to store or communicate one symbol in a message. Shannon entropy thus measures the change in uncertainty transmitted and predictably received between the sender and receiver. The actual information that gets transmitted and received was formulated by Shannon as *R*, which he called rate, and expressed as

$$R = H_{before} - H_{after}$$

R, then, becomes a proxy for the amount of information accurately communicated. *R* can never be zero because all communication systems have losses. H_{before} and H_{after} are both state functions for the message, so this also makes *R* a function of state. While Shannon entropy (unpredictability) exists for any given sending or

[3] As Shannon acknowledges in his paper, the "bit" term was suggested by J. W. Tukey. Shannon can be more accurately said to have popularized the term via his paper.

[4] According to Wikipedia, the equation was originally formulated by Ludwig Boltzmann between 1872 and 1875, but later put into its current form by Max Planck in about 1900. To quote Planck, "the logarithmic connection between entropy and probability was first stated by L. Boltzmann in his kinetic theory of gases."

receiving state, the actual amount of 'information' (that is, data) that is transmitted is a change in state measured by a change in uncertainty between the sender (H_{before}) and the receiver (H_{after}). In the words of Thomas Schneider, who provides a clear discussion of this distinction, "[Shannon] Information is always a measure of the decrease of uncertainty at a receiver" [11].

Shannon's idea of information entropy has come to inform entropy in physics and the second law of thermodynamics [12]. According to Koelman, "the entropy of a physical system is the minimum number of bits you need to describe the detailed state of the system fully." Very random (uncertain) states have high entropy, and patterned states have low entropy. Work by individuals such as Jaynes suggested a reinterpretation of statistical mechanics to equate the concept of thermodynamic entropy with information entropy [13]. How others interpreted Jayne's work helped add to the confusion that somehow Shannon entropy is related to the 'disorder' of thermodynamic entropy. To unpack this confusion we need to introduce the ideas of scale and open systems with external inputs of energy.

At cosmic scales—that is, a closed system—we see the tendency to dispersal and disorder. However, at our local scale, we see order and life and the development of complex biological systems [14] and self-replication [15]. Erwin Schrödinger, of the cat thought experiment, in his famous 1943 lectures on "What is Life?" [16], tried to square life with what he knew then about the physical and chemical world. One insight he had was to introduce the idea of genetic material carried in an 'aperiodic crystal' (DNA as eventually discovered). Another assertion was that living matter evades the decay to thermodynamic equilibrium by feeding on what he called 'negative entropy,' a sort of reverse entropy toward order. Brillouin extended the idea to information and shortened the name to 'negentropy' [17]. Prigogine tried to get at the same questions with his minimum entropy dissipative structures [18]. Over time, Schrödinger and others changed from an entropy basis to the related Gibbs free energy basis, which is the maximum work potential of a system at constant pressure and temperature. What researchers have been trying to do is to take a static view of thermodynamics under ideal and closed conditions and relate it to the dynamic notions of life and information. Through the more recent work of Annila [19], Crooks [20], England [21], Karnani [7], Salthe [22], and many others, the starting assumptions of static and closed conditions have been reassessed under local and dynamic ones. We have seen a shift to questions of nonequilibrium thermodynamic conditions, such as life, and how maximum entropy production may be favored to dissipate high influxes of external energy. We now understand that open systems receiving fluxes of outside energy, such as Earth, favor order and structures that dissipate these external fluxes faster. Some, such as Annila, and separately England [23], relate these forces to evolution.

What appears as fundamental truths relating to information, entropy, dissipation, and structure in dynamic environments underlie these current strains of research. Some have "hinted at a possible deep connection between intelligence and entropy maximization" [24]. What we can say so far is that information is physical and perhaps energetic, with strong conceptual and deeper ties to the ideas of thermodynamic entropy. Messages are the ways information is conveyed, and

always incur losses. Order and structure seem to play a role here, perhaps in providing faster ways to dissipate energy toward equilibrium in high-energy local conditions. Still, we have yet to discuss meaning, and senses like information having economic value [25].

The Structure of Information

Structure is something, of tangible or intangible character, that refers to the recognition, observation, nature, or permanence of patterns and relationships of things. The concept may refer to an object, such as a built structure, or an attribute, such as the structure of society, or something abstract, like a data structure or language. Structure may thus be abstract, or it may be concrete. Its realm ranges from the physical to ideas and concepts. As a term, 'structure' is ubiquitous to every domain. We may find structure across every conceivable scale, from the most minute and minuscule to the cosmic. Even realms without any physical aspect at all—such as ideas and beliefs—are perceived by many to have structure. We apply the term to any circumstance in which things are arranged or connected to one another, as a means to describe the organization or relationships of things. We seem to know structure when we see it and to discern structure of very many kinds in contrast to unstructured or random backgrounds.

In this way, structure resembles patterns, and perhaps even is a synonym. Bates closely relates information to patterns, as well as provides a broad listing of other information aspects [26]. The structure of Peirce's universal categories implies, I believe, likely patterns in our information. Using thermodynamic insights, Bejan has devoted his career to outlining how the 'constructal law' of flows and related concepts such as order, organization, design, or form contribute to the structures we see in nature [27]. Information in relation to structure raises questions such as which structures are preferred and why do some of them perpetuate under the conditions of nature. An aspect of structure, which provides insight into its roles and importance, is that we can express it in shortened form as a mathematical statement. One could even be so bold as to say that mathematics is the language of structure.

Forms of Structure

The natural world is replete with structure. Patterns in nature are regularities of visual form found in the natural world. We may model such patterns mathematically. Typical mathematical forms in nature include fractals, spirals, flows, waves, lattices, arrays, Golden ratio, tilings, Fibonacci sequences, and power laws. We can see natural forms in clouds, trees, leaves, river networks, fault lines, mountain ranges, craters, animal spots and stripes, shells, lightning bolts, coastlines, flowers, fruits, skeletons, cracks, growth rings, heartbeats and rates, earthquakes, veining, snowflakes, crystals, blood and pulmonary vessels, ocean waves, turbulence,

beehives, dunes, and DNA. The mathematical expression of structures in nature is frequently repeated or recursive in nature, often in a self-organizing manner. The swirls of a snail's shell reflect a Fibonacci sequence, while natural landscapes or lifeforms often have a fractal aspect.[5] Fractals are typically self-similar patterns, generally involving some fractional or ratioed formula that is recursively applied. Another way to define it is a detailed pattern repeating itself.

Even though we can often express these patterns mathematically, and they often repeat themselves, their starting conditions can lead to tremendous variability and a lack of predictability. This lack makes them chaotic, as studied under chaos theory, though their patterns are often discernible. While we certainly see randomness in statistics, quantum physics, and Brownian motion, it is also striking that what gives nature its beauty is structure. As a force separate and apart from the random, there appears something within structure that guides the expression of what is natural and what is so pleasing to behold. Self-similar and repeated structures across a variety of spatial scales are an abiding aspect of nature. Such forms of repeated patterns or structure are also inherent in that unique human capability, language, a topic on which Warner has written extensively [28].

Some Structures Are More Efficient

The continuation of structure from nature to language extends across all aspects of human endeavor. I remember once excitedly describing to a colleague what likely is a pedestrian observation: pattern matching is a common task in many fields. (I had observed that pattern matching in very different forms was standard practice in most areas of industry and commerce.) My 'insight' was that this commonality was not widely understood, which meant that widely divergent pattern-matching techniques in one field were not often exploited or seen as transferable to other domains.

In computer science, pattern matching is the act of checking some sequence of tokens for the presence of the constituents of some pattern. It is closely related to the idea of pattern recognition, which is the characterization of some discernible and repeated sequence. These techniques, as noted, are widely applied, with each field tending to have favorite algorithms. Typical applications that one sees for such

[5] There are dynamic illustrations of many of these patterned phenomena, but it is also the case that animated images on the Web tend to have a short shelf life. There are some examples related to this section that may be found on Wikipedia, which likely means that similar images may be found by searching exact and related topics in that source, even should the exact links provided here prove dated. Examples of patterned examples include fractals such as http://upload.wikimedia.org/wikipedia/commons/f/f4/Animation_of_the_growth_of_the_Mandelbrot_set_as_you_iterate_towards_infinity.gif, http://upload.wikimedia.org/wikipedia/commons/f/fd/Von_Koch_curve.gif, and http://upload.wikimedia.org/wikipedia/commons/7/76/Feigenbaumzoom.gif, and cellular automata such as http://upload.wikimedia.org/wikipedia/commons/8/86/Oscillator.gif, with recursion and repeated patterns (such as in the whorls of flowers) providing examples of how simple stratagems may also result in rather complex behaviors.

pattern-based calculations include communications,[6] encoding and coding theory, file compression, data compression, machine learning, video compression, mathematics (including engineering and signal processing via such techniques as statistics or Fourier transforms), cryptography, NLP,[7] speech recognition, image recognition, OCR, image analysis, search, sound cleaning (that is, error detection, such as Dolby), and gene sequence searching and alignment, among many others.

To better understand what is happening here and the commonalities, let's look at the idea of compression. Data compression is valuable for transmitting any form of content in wired or wireless manners because we can transmit the same (or closely similar) message faster and with less bandwidth.[8] Compression uses both lossless (no loss of information) and lossy methods. Algorithms for lossless data compression usually exploit statistical redundancy—that is, a pattern match—to transmit data more concisely without losing information. Lossless compression is possible because most real-world data has statistical redundancy. In lossy data compression, some loss of information is acceptable by dropping detail from the data to save space. For instance, some frequencies are inaudible to people. A lossy audio recording may drop these frequencies without being noticed.

A close connection relates machine learning and compression. A system that predicts the posterior probabilities of a sequence given its entire history can be used for optimal data compression (by using arithmetic coding on the output distribution), while an optimal compressor can be used for prediction (by finding the symbol that compresses best, given the previous history). In contrast, cryptography seeks to construct messages that pattern matching is too time consuming to analyze.

Evolution Favors Efficient Structures

An example shows how Shannon entropy relates to patterns or data compression. Let's take a message of entirely random digits. To accurately communicate that message, we would need to transmit all digits (bits) in their original state and form. Absolutely no compression of this message is possible. If, however, patterns reside within the message (which, of course, now ceases to make the message random), we can express them algorithmically in a shortened form so that we only need to communicate the algorithm and not the full bits of the original message. If this 'compression' algorithm can then be used to reconstruct the bit stream of the original

[6] Communications is a particularly rich domain with techniques such as the Viterbi algorithm, which has found universal application in decoding the convolutional codes used in both CDMA and GSM digital cellular, dial-up modems, satellite, deep-space communications, and 802.11 wireless LANs.

[7] Notable areas in natural language processing (NLP) that rely on pattern-based algorithms include classification, clustering, summarization, disambiguation, information extraction, and machine translation.

[8] There is a massive list of "codecs" (compression/decompression) techniques available (http://en.wikipedia.org/wiki/List_of_codecs); fractal compression is one.

message, the data compression method is deemed lossless. The algorithm so derived is also the expression of the pattern that enabled us to compress the message in the first place. We can apply this same type of intuition to human language.

In open systems, structures (patterns) are a means to speed the tendency to equilibrate across energy gradients. This observation helps provide insight into structure in natural systems, and why life and human communications tend toward more order (less randomness). Structure will always continue to emerge because it is adaptive to speed the deltas across these gradients; structure provides the fundamental commonality between biological information (life) and human information. Of course, in Shannon's context, what we measure here is data (or bits), not information embodying any semantic meaning or context. However, it does show that 'structure'—that is, the basis for shortening the length of a message while still retaining its accuracy—is information in the Shannon context. This structure arises from order or patterns, often of a hierarchical or fractal or graph nature. Emergent structure that can reduce the energy gradient faster is favored.

These processes are probabilistic and statistical. Uncertainties in state may favor one structure at one time versus another at a different time. The types of chemical compounds favored in the primordial soup were likely greatly influenced by thermal and light cycles and drying and wet conditions. In biological ecosystems, huge differences occur in seed or offspring production or overall species diversity and ecological complexity based on the stability (say, tropics) or instability (say, disturbance) of local environments. These processes are inherently nondeterministic. As we climb up the chain from the primordial ooze to life and then to humans and our many information mechanisms and technology artifacts (which are themselves embodiments of information), we see increasing complexity using different structural mechanisms.

The mechanisms of information transfer in living organisms occur (generally) via DNA in genes, mediated by sex in higher organisms, subject to random mutations, and then kept or lost entirely as their host organisms survive to procreate or not. Those are harsh conditions: the information survives or not (on a population basis) with high concentrations of information in DNA and with a priority placed on remixing for new combinations via sex. Information exchange (generally) only occurs at each generational event. Human cultural information, however, is of an entirely different mediation. We can record our information and share it across individuals or generations, extended with innovations like written language or digital computers.

Common to all of these perspectives—from patterns in nature and on to life and then animal and human communications—we see that structure is information. Human artifacts and technology, though not 'messages' in a conventional sense, embody information within their structures [29]. We also see the interplay of patterns and information in many processes of the natural world [30]. Examples include complexity theory, emergence, autopoiesis, autocatalysis, self-organization, stratification, and cellular automata [31].

We, beings who can symbolically record our perceptions, seem to recognize patterns innately. We see beauty in symmetry. Bilateral symmetry seems deeply

ingrained in the perception by humans of the possible health or fitness of other living creatures. We also seem to recognize beauty in the simple. Seemingly complex bit streams reduced to a shorter algorithmic expression are always viewed as more elegant than lengthier, more complex alternatives. The simple laws of motion and Newtonian physics fit this pattern, as does Einstein's $E = mc^2$. This preference for the simple is a preference for the greater adaptiveness of the shorter, more universal pattern of messages, a lesson from Shannon's information theory.

These insights point to the importance of finding and deriving structured representations of information—including *meaning*—that can be simply expressed and efficiently conveyed. Building upon the accretions of structure in human and computer languages, the semantic Web and semantic technologies offer just such a prospect. These insights provide a guidepost for how and where to look for the next structural innovations. We find them in the algorithms of nature and language, and in making connections that provide the basis for still more structure and patterned commonalities.

The Meaning of Information

For Charles S. Peirce, signs convey all information. All signs are a triadic whole of the *object*, how it is perceived or signaled (*representamen*), and how it is understood or interpreted (*interpretant*), including meaning. Signs might be iconic, such as physical road signs or brand logos. Signs might be indexical, such as seeing the weather vane pointing the direction of the wind or hearing the whistle signaling the approaching train. Alternatively, the sign might be one of convention or patterns ('habits' or 'laws' in Peircean terms), as embodied in symbols. Examples of symbols include the stylus impression on clay, the crystalline structures of RNA and DNA,[9] the printed letters and words on the page, or the ordered magnetic charges on a hard drive.

No matter the medium or form, information is a physical sign that indicates some change in state, the 'difference' in Bateson's term. Because information is real, it can be theorized over and investigated empirically. In a letter to Lady Welby in 1902 Peirce says [32]:

> As for the 'meaning,' logicians have recognized since Abélard's day and earlier that there is one thing which any sign, external or internal, stands for, and another thing which it signifies; its denoted breadth, its 'connoted' depth. They have further generally held, in regard to the most important signs, that the depth, or signification, is intrinsic, the breadth extrinsic (CP 8.119).

[9] Michael Buckland, among many others, questions treating physical characteristics as messages (pers. comm.). However, at a certain level of Thirdness, inanimate patterns like crystalline structures do represent the "best" mediating organization at the time of formation to maximize entropy production. At the level of symbology, of course, human language is much more derived. It is the idea of mediation and Thirdness that brings crystals and language together, not the nature of the symbols used.

Peirce specifically defined *information* as the *breadth* × *depth* of a concept (1867, CP 2.407-8) or what he also called the *area* (CP 2.419). He affirmed the same view more than 35 years later (1903, EP 2:305). The *breadth* refers to all of the external things regarding the concept, spanning its extensions or denotations, the things to which it connects. The *depth* applies to the intensions or comprehension about the concept, what it *is*, including internal properties or qualities. Peirce preferred *extension v comprehension* when referring to these two respective ideas. Peirce uses these terms in their absolute senses. That is, for a given thought or concept at hand, complete information would mean correctly comprehending all of the context and aspects of that given thing. This complete understanding is the 'truth' about the subject, in all of its absolute, dynamic elements. In fact, for signs, Peirce is repeatedly clear about distinguishing what the sign is about, what he calls the *immediate object*, which is what the sign conveys, and the actual *dynamic object*, the real thing that is the (inadequately signified) object of the sign:

> We must distinguish between the Immediate Object,—i.e. the Object as represented in the sign,—and the Real (no, because perhaps the Object is altogether fictive, I must choose a different term, therefore), say rather the Dynamical Object, which, from the nature of things, the Sign *cannot* express, which it can only *indicate* and leave the interpreter to find out by *collateral experience* (1909, CP 8.314).

For example, no matter how cleverly or comprehensively we try to convey the idea of a general type called *diamonds* (the dynamic object), the object we signify to convey this reality (immediate object) can never be complete. There is never enough breadth, depth, perspective, and completeness to capture the dynamic *diamond*, similar to the territory map in Jorge Luis Borges's "On Exactitude of Science." These concepts are no different from the Shannon idea that losses always occur between what is sent and what is received.

If one applies these breadth and depth measures to a domain, we begin to get into massively scaled senses of information. All objects and their connections, no matter how tenuous, and their characteristics, no matter how subtle, constitute the entirety of the possible information space. This expansion is not tractable, which means we must find pragmatic ways to handle the combinatorial challenge, as well as to filter what is useful based on context and relevance.

Information is thus a very lossy concept. We have the real world, and all that it is. We represent what is in this world, imperfectly and incompletely. The messages we convey are subject to loss. We perceive or try to signify what we understand from these messages. Our representations are understood or not, and interpreted via circumstance and context. Higher losses across this circuit lower trust in the information and decrease our ability to act.

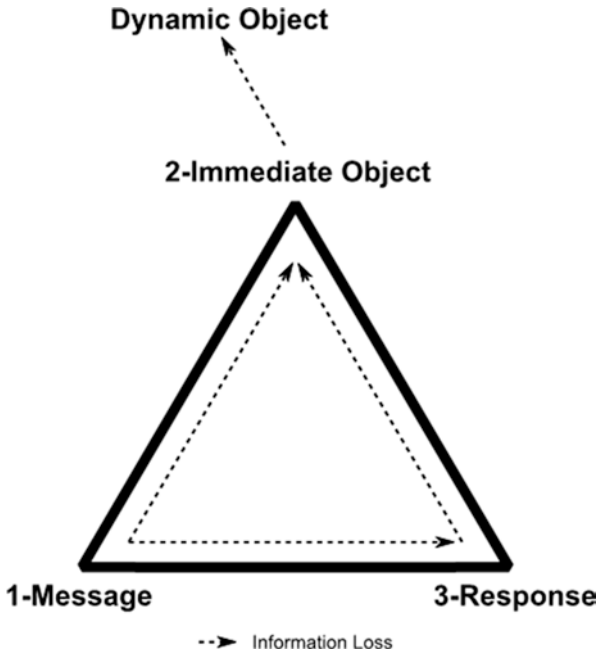


Fig. 2.1 Semiotic information triad

We can, however, take Peirce's views on sign-making (*semiosis*) and information and derive a somewhat integrative picture of how all of these piece parts may fit together. Figure 2.1 is not a standard presentation because, first, I merge Shannon information constructs into the standard Peircean interpretation. Second, also in keeping with Shannon, I show arrows indicating information loss.[10] In Fig. 2.1 we first stipulate a given domain and scope of inquiry (not shown). Real things occupy this space, never, unfortunately, wholly understandable nor transmittable as fully correct messages. The dynamic object represents the total information theoretic potential. It is all that one might say about the real object. The dynamic object may be a singular thing or collections of ideas or things. In representing our dynamic objects, we can only convey them as somewhat incomplete immediate objects, which are in Secondness based on Peirce's universal categories (see Chap. 6).

How these objects are pointed to or signified is also an abstraction. Maybe we convey something iconic like an image, or perhaps we describe it in words. In all cases, our signification is imperfect. In Shannon terms, this is a message, which we can see as an analog of what Peirce called the *representamen*.[11] This message is a Firstness regarding the universal categories.[12] Structure affects how the message is

[10] These arrows should not be confused with diagrams from other authors that depict the flows of understanding or meaning in Peircean semiosis.

[11] Not all Peirce scholars agree with this view. A key passage is CP 8.332 (1904), one of Peirce's letters to Lady Welby.

[12] Note this is 'message' in the sense of Shannon, not the 'meaning' of the transmission, which is in Thirdness.

initially *encoded* for transmittal and then *decoded* at the receiver (that is, the response level).

Then, as *interpretants*—that is, the response level for those who receive the messages and respond to them—we also understand the object based on our perspectives and contexts. We may grasp and perceive many aspects of the signified object, or we may not. As the representation of the object by the sign, loss also arises from the interpretation of the sign by the responder to the object. Some of that loss, of course, may also be due to a loss of clarity from the sign to the interpretant, or what the interpretant can perceive and process, all subject to circumstance or context.

Peirce posed three different kinds of interpretants:

> It is likewise requisite to distinguish the Immediate Interpretant, i.e. the Interpretant represented or signified in the Sign, from the Dynamic Interpretant, or effect actually produced on the mind by the Sign; and both of these from the Normal Interpretant, or effect that would be produced on the mind by the Sign after sufficient development of thought (1908, CP 8.343).

The *immediate interpretant* is the sense, or quality of the impression, invoked by the sign; Peirce also likened it to a schema. The immediate interpretant is a kind of Firstness. The *dynamic interpretant* is the meaning of the sign for a given concrete instance, an "act of the Mind" (1909, CP 8.315). It is a kind of Secondness. The *normal interpretant*, also called the *final* or *ultimate*, is the full significance of the sign, what it 'means' in all of its various aspects. The normal interpretant is a kind of Thirdness. It is the 'sum of lessons' learned from the sign and is a basis for action. I understand the normal interpretant to embrace all of the *breadth* and *depth* of information knowable to the interpreting agent. Though never expressed as such, I interpret Peirce to liken the immediate interpretant as the sense of something or its impression; the dynamic interpretant as taking note of something, recognizing it as information; and the normal interpretant as something we know and are willing to act upon with all that that means.

I discuss more the transition from information to knowledge in the next section. With Shannon's information theory, we have a technical way to understand and quantify information concerning entropy and its potential. That theory also gives us a robust framework for understanding and evaluating information losses, and how it is that we lose fidelity and truth as we move from the real to the perceived and communicated. As information theory gets better understood from the standpoints of the statistical mechanics of dynamic, nonequilibrium systems—that is, the circumstances of life and humans—I think we will begin to further understand the role of structures and patterns as favored dissipation systems. We are still building awareness that information is a rich environment, one which we may use Peirce's universal categories and semiosis to represent. We are still at the cusp of unpeeling these perspectives into an integrated information whole.

What Is Knowledge?

As a book about knowledge representation, we have been sneaking up on what this concept of *knowledge* means. We see that it is grounded in information somehow, but it is also different. Significant terms we associate with knowledge and its discovery include *open*, *dynamic*, *belief*, *judgment*, *observation*, *process*, *representation*, *signification*, *interpretation*, *logic*, *coherence*, *context*, *reality*, and *truth*. These were all topics of Peirce's deep inquiry and explicated by him via his triadic worldview. To get at the question, I begin with some of our common sense understandings of 'knowledge.' I then supplement these notions with what Peirce himself had to say about the nature of knowledge. We conclude this section by looking at the critical question of *doubt*, and why that is the basis for stimulating inquiry and our search for new knowledge.

The Nature of Knowledge

Let's take the statement: *the sky is blue*. We can accept this as a factual statement. However, if we know the sky is dark or black, we know it is the night. Alternatively, the sky may be gray if it is cloudy. When we hear the statement that *the sky is blue*, if we believe the source or can see the sky for ourselves, then we can readily infer whether the observation is correct, occurring during daylight, under a clear sky. Our acceptance of an assertion as factual or being true carries with it the implications of its related contexts. On the other hand, were we simply to state *le ciel est bleu*, and if we did not know French, we would not know what to make of the statement, true or false, with context or not, even if all of the assertions were still correct.

This simple example carries with it two profound observations. First, context helps to determine whether we believe or not a given statement, and, if we believe it, what the related context implied by the statement might be. Second, we convey this information via symbols—in this case, the English language, but applicable to all human and artificial and formal notations like mathematics as well—which we may or may not be able to interpret correctly. If I am monolingual in English and I see French statements, I do not know what the symbols mean.

Knowledge may reside solely in our minds, and not be part of 'common knowledge.' However, ultimately, even personal beliefs not held by others only become 'knowledge' that we can rely upon in our discourse once others have 'acknowledged' the truth. Forward-looking thinkers like Copernicus or Galileo or Einstein may have understood something in their minds not yet shared by others, but we do not 'acknowledge' those understandings as knowledge until we can share and discuss the insight (that is, what scientists would call independent verification). In this manner, knowledge, like language and symbol creation, is inherently a social phenomenon. If I coin a new word, but no one else understands what I am saying, that is not part of knowledge; that is gibberish.

None of this denies that individuals may 'know' things or have insights not shared with others. Perhaps we could call this 'personal knowledge.' My larger point, as it was for Peirce, is to advocate a more elevated understanding of knowledge that has the essences of being shared, valid to some degree, and supported by a community one respects. Indeed, the process of sharing knowledge with communities is to test and reflect on the shared understanding, thereby honing and improving our knowledge of the subject.

Peirce maintained, in part, we have to *believe* information for it to become knowledge. Put another way, we need to believe information to act upon it.[13] As our prior discussion also showed, we also see that the information upon which our judgments may depend may differ at all levels of human experience, perceptions, and language. We have a variety of viewpoints on any topic of ordinary human discourse. One criterion we apply when evaluating a viewpoint is whether it is *coherent*. Coherence is a state of logical, consistent connections, a logical framework for intelligently integrating diverse elements. Another criterion for evaluating information is whether it is ambiguous. *Ambiguity* is a frequent source of error, as when we wrongly identify the object connections can get drawn that are in glaring error. This potential error is why *disambiguation* is such a big deal in semantic systems. *Context* is thus an essential basis for resolving disambiguities. The same information may be used differently or given different importance depending on circumstance. One immediate implication of these italicized points is that we need to embed our information in a pragmatic semantics that reflects these realities.

Besides semantics, let's also look at some of the other common sense characteristics we associated with knowledge, and how these senses may affect what we need in a knowledge management system:

- *Knowledge is never complete*—Gaining and using knowledge is a process, and is never complete. A completeness assumption around knowledge is by definition inappropriate.
- *Knowledge may reside in multiple forms*—Structured databases represent only a portion of structured information (spreadsheets and other non-relational data stores are other structured forms). Further, general estimates are that 80% of information available resides in documents, with growing importance to metadata, Web pages, markup documents, and other semi-structured sources. A proper system for knowledge representation should be equally applicable to these various information forms.
- *Knowledge occurs anywhere*—Relevant information about customers, products, competitors, the environment, or virtually any knowledge-based topic may arise from internal and external information. The emergence of the Internet and the universal availability and access to mountains of public and shared information demand its thoughtful incorporation into knowledge management systems.

[13] Actions, of course, are not all premised on belief. Actions might be coerced or unconsciously reflexive.

- *Knowledge is about connections*—The epistemological nature of knowledge can be argued endlessly, but I submit that much of what distinguishes knowledge from information is that knowledge makes the connections—that is, asserts relations—between disparate pieces of relevant information, and it does so truly and believably. As these relationships accrete, the knowledge base grows. We need knowledge systems that enable us to add new connections as discovered without adversely impacting our existing knowledge characterizations.
- *Knowledge structures evolve with the incorporation of more information*—Our ability to describe and understand the world or our problems at hand requires inspection, description, and definition. Birdwatchers, botanists, and experts in all domains know well how investigation and study of specific domains lead to more discerning understanding and 'seeing' of that domain. Before learning, everything is just a shade of green or a herb, shrub, or tree to the incipient botanist; eventually, she learns how to discern entire families and individual plant species, all accompanied by a rich domain language. We need to explicitly recognize in our KM systems how increased knowledge leads to more structure and more vocabulary.
- *Knowledge is about what is agreed upon*[14]—Since knowledge is a state of understanding by practitioners and experts in a given domain, it is also vital that those very same users be active in its gathering, organization (structure), use, and consensus of what it is and what it means. The adjudication of knowledge is ultimately a social and community phenomenon. We should build KM systems around and for its users.

Of course, we may ascribe other senses to knowledge. Peirce, for example, notes that all knowledge comes to us via observation (1897, CP 2.444). He notes that different knowledge may have different economic value (1902, CP 7.158). He also separates out 'acquaintance' knowledge in his discussion of how to evaluate signs. 'Acquaintance' knowledge comes from 'collateral observation,' which goes beyond mere context to also include the meaning of the background knowledge applied to recognizing and interpreting the sign:

> Now let us pass to the Interpretant. I am far from having fully explained what the Object of a Sign is; but I have reached the point where further explanation must suppose some understanding of what the Interpretant is. The Sign creates something in the Mind of the Interpreter, which something, in that it has been so created by the sign, has been, in a mediate and *relative* way, also created by the Object of the Sign, although the Object is essentially other than the Sign. And this creature of the sign is called the Interpretant. It is created by the Sign; but not by the Sign quâ member of whichever of the Universes it belongs to; but it has been created by the Sign in its capacity of bearing the determination by the Object. It is created in a Mind (how far this mind must be real we shall see). All that part of the understanding of the Sign which the Interpreting Mind has needed collateral observation for is outside the Interpretant. I do not mean by 'collateral observation' acquaintance with the system of signs. What is so gathered is *not* COLLATERAL. It is on the contrary the prerequisite for getting any idea signified by the sign. But by collateral observation, I mean previous acquaintance with what the sign denotes (1909, CP 8.179).

[14] Per the more expansive definition used above, which reduces the role of 'personal knowledge.'

We communicate shared knowledge via symbols. That means we communicate these assertions as arguments, which require *judgment* as to whether and how to act:

> That is the first point of this argument; namely, that the *judgment*, which is the sole vehicle in which a concept can be conveyed to a person's cognizance or acquaintance, is not a purely representitious event, but involves an act, an exertion of energy, and is liable to real consequences, or effects (1908, CP 5.547).

Prior knowledge, or 'collateral observation', helps inform the judgment. The role of prior knowledge suggests that local context, broadly defined as the locality in a *knowledge graph*, needs to play a role in the characterization of knowledge.

In another vein, Joel Mokyr, whom we will have a chance to discuss in Chap. 3, proposes a useful split between propositional knowledge (descriptive knowledge) from the know-how of procedural language [30]. A search will turn up many other assertions in the literature across disciplines about the nature and classification of knowledge.

Knowledge as Belief

Information is a proposition or assertion conveyed to us via signs, most often symbols, about objects in our domain (or world). The breakpoint from information to knowledge, based on our evaluation so far, occurs when we believe the information, and are willing to act upon it. We observe and evaluate the sign; if our response is action or a willingness to act, we can consider the sign as knowledge. I suspect under this interpretation that the act of merely recording the assertion, storing it for later use or inspection, does not qualify as an act of belief. This interpretation seems to conform with Peirce's idea of the dynamic interpretant.

The centrality of the idea of belief to knowledge goes back to at least Plato and then the Enlightenment in a formulation known as 'justified, true belief (JTB)' [33]. The proposition must be believed as true with justification to qualify as knowledge. Peirce essentially endorsed this notion (though with some caveats and expansions as I suggest below):

> Plato is quite right in saying that a true belief is not necessarily knowledge. A man may be willing to stake his life upon the truth of a doctrine which was instilled into his mind before his earliest memories without knowing at all why it is worthy of credence; and while such a faith might just as easily be attached to a gross superstition as to a noble truth, it may, by good luck, happen to be perfectly true. But can he be said to *know* it? By no means: to render the word knowledge applicable to his belief, he must not only believe it, but must know,—I will not say, with the ancients, the rationale of the real fact, as a reality,—but must know what justifies the belief, and just WHY and HOW the justification is sufficient. I beg that the reader will turn this over in his mind and satisfy himself as to how far what I am saying is true. For it is not a very simple point but is one that I intend to insist upon. Before knowledge of any subject can be put to any extensive use, it is almost indispensable that it should be made as thorough and complete as possible, until every detail and feature of the matter is spread out as in a German handbook. But if I am asked to what the wonderful success of modern science is due, I shall suggest that to gain the secret of that, it is necessary to consider science as living, and therefore not as knowledge already acquired but as the

> concrete life of the men who are working to find out the truth. Given a body of men devoting the sum of their energies to refuting their present errors, doing away with their present ignorance, and that not so much for themselves as for future generations, and all other requisites for the ascertainment of truth are insured by that one (1902, CP 7.49-50).

Peirce describes in this passage the nature of truth, the means of justification, and the role of doubt. Let's first understand more precisely what Peirce means by *belief:*

> A cerebral habit of the highest kind, which will determine what we do in fancy as well as what we do in action, is called a belief. The representation to ourselves that we have a specified habit of this kind is called a judgment. A belief-habit in its development begins by being vague, special, and meagre; it becomes more precise, general, and full, without limit. The process of this development, so far as it takes place in the imagination, is called thought. A judgment is formed; and under the influence of a belief-habit this gives rise to a new judgment, indicating an addition to belief. Such a process is called an inference; the antecedent judgment is called the premiss; the consequent judgment, the conclusion; the habit of thought, which determined the passage from the one to the other (when formulated as a proposition), the leading principle (1880, CP 3.160).

Interestingly, though, Peirce closely ties belief to probability:

> Probability and chance undoubtedly belong primarily to consequences, and are relative to premisses; but we may, nevertheless, speak of the chance of an event absolutely, meaning by that the chance of the combination of all arguments in reference to it which exist for us in the given state of our knowledge. Taken in this sense it is incontestable that the chance of an event has an intimate connection with the degree of our belief in it. Belief is certainly something more than a mere feeling; yet there is a feeling of believing, and this feeling does and ought to vary with the chance of the thing believed, as deduced from all the arguments. Any quantity which varies with the chance might, therefore, it would seem, serve as a thermometer for the proper intensity of belief. Among all such quantities there is one which is peculiarly appropriate. When there is a very great chance, the feeling of belief ought to be very intense. Absolute certainty, or an infinite chance, can never be attained by mortals, and this may be represented appropriately by an infinite belief (1878, CP 2.676).

These views are one reason why Peirce contributed so much to probability theory over his career. Peirce's views are strongly tied to his belief in fallibilism: while truth exists, it can never be known absolutely, but as an approximation moving toward its limit function through testing and inquiry (namely, the scientific method).

> … I used for myself to collect my ideas under the designation *fallibilism*; and indeed the first step toward *finding out* is to acknowledge you do not satisfactorily know already; so that no blight can so surely arrest all intellectual growth as the blight of cocksureness; and ninety-nine out of every hundred good heads are reduced to impotence by that malady—of whose inroads they are most strangely unaware! (1897, CP 1.13)

A core consistency underlying Peirce's views of knowledge is his belief in reality. Reality exists outside of the mind or the individual; it exists whether minds exist or not to consider it; and it can be unveiled or discovered over time through observation and inquiry. In all of his writings, except when dedicated to the topic, Peirce attempted to look outside of psychology for his premises and logic. Objective truth exists, even if not absolutely knowable at its limits:

> There are Real things, whose characters are entirely independent of our opinions about them; those Reals affect our senses according to regular laws, and, though our sensations

> are as different as are our relations to the objects, yet, by taking advantage of the laws of perception, we can ascertain by reasoning how things really and truly are; and any man, if he have sufficient experience and he reason enough about it, will be led to the one True conclusion. The new conception here involved is that of Reality (1903, CP 5.384).

Knowledge is what we know of the past that influences our expectations about the future. It produces judgments and enables us to act, based on our believed probabilities. Unlike standard JTB, though, Peirce's version of knowledge asserts the fallibility of truth, is more rigorous in proposing how to justify and test for it, and includes the role for community adjudication. Thus, while neither truth nor justification may ever be absolute, and may change based on what we discover about objective reality, the weighing of the evidence gives us the belief in knowledge upon which to act.

Doubt as the Impetus of Knowledge

Another difference from JTB that Peirce emphasizes regards the drive or the quest for knowledge. So long as we doubt, we have an impetus to inquiry and knowledge. Upon attaining what Peirce called 'full belief,' once doubt has been removed and its irritation sated, the impetus to acquire knowledge on that topic abates:

> Doubt is an uneasy and dissatisfied state from which we struggle to free ourselves and pass into the state of belief (1892, CP 5.372).

> The irritation of doubt causes a struggle to attain a state of belief. I shall term this struggle *Inquiry*, though it must be admitted that this is sometimes not a very apt designation (1892, CP 5.374).

The 'irritation of doubt,' how to remove it, and how to make ideas clear were the subjects of a series of papers by Peirce in *Popular Science Monthly* in the late 1870s. One of the papers, "The Fixation of Belief" [34], critically reviewed what Peirce claimed were the only four methods for obtaining belief and removing doubt. The four methods are the (1) method of tenacity, wherein one repeats or wills to believe something; (2) the method of authority, wherein governments or external forces insist upon certain beliefs; (3) the *a priori* method, wherein precedent or social consensus determines beliefs; or (4) the method of science, obtained from the scientific method and the application of observation, testing, and logic. Peirce noted logical and other pitfalls for the first three methods. Only the fourth method can fix (and refix!) belief in an objective reality based on fallible truth. It is noteworthy given Peirce's lifelong devotion to science and the scientific method that he also makes the explicit point that belief, however, is not the objective of science:

> We believe the proposition we are ready to act upon. *Full belief* is willingness to act upon the proposition in vital crises, *opinion* is willingness to act upon it in relatively insignificant affairs. But pure science has nothing at all to do with *action* …. There is thus no proposition at all in science which answers to the conception of belief (1898, CP 1.635).

Through science, we gain probabilities on our information that constitutes conditioned or provisional knowledge. Belief, how Peirce defines it, is not a matter of science, but of action:

> But in vital matters, it is quite otherwise. We must act in such matters; and the principle upon which we are willing to act is a belief. Thus, pure theoretical knowledge, or science, has nothing directly to say concerning practical matters, and nothing even applicable at all to vital crises. Theory is applicable to minor practical affairs; but matters of vital importance must be left to sentiment, that is, to instinct (1898, CP 1.636-7).

Peirce perhaps does not say all that he could regarding doubt and the scientist's quest for truth. What we do glean from these perspectives, though, is the importance of the scientific method to inquiry, the driving force of doubt in our seeking more information, and the role that belief plays in elevating information to knowledge.

What Is Representation?

'Representation' is the second part of knowledge representation (KR). One dictionary sense is that 'representation' is the act of speaking or acting on behalf of someone else. This sense is the one, say, of a legislative representative (the Thomas Hobbes view, a dominant theme in classical empiricism [35]). Another sense is a statement made to some formal authority communicating an assertion, opinion, or protest, such as a notarized document. The sense applicable to KR, however, according to the *Oxford Dictionary of English*, is the one of 'representing,' that is, "the description or portrayal of someone or something in a particular way or as being of a certain nature."[15]

Peirce bases his representational view of the world on semiotics, the study and logic of signs. In his seminal writing on this in 1894, "What is in a Sign?" [36], Peirce wrote that "every intellectual operation involves a triad of symbols" and "all reasoning is an interpretation of signs of some kind." Do not confuse Peirce's semiosis with that of Ferdinand de Saussure, which was for many years better known but lacks the perspective of Thirdness (mediation, continuity) in Peirce's version.

After the advent of computers, knowledge representation (and reasoning) was formalized as a sub-discipline of artificial intelligence and received more focused attention. Davis et al. wrote an influential piece in 1993 that stipulated five requirements for knowledge representation [37], all of which are captured in one way or another by the approach recommended in this book. We may study KR through standard texts such as by Brachman and Levesque [38], or van Harmelan [4]. We may understand KR via the language of thought hypothesis of Jerry Fodor [39] or via set theory (such as from Zhou [38]) or Fred Dretske's representational thesis (which pays particular attention to phenomenology but does not mention Peirce).[16]

Clearly, for the reasons cited throughout this book, Peirce is the polestar I have chosen to guide my thinking on knowledge representation. We have already seen his

[15] See https://en.oxforddictionaries.com/definition/representation.

[16] See CP 1.568, wherein Peirce provides "The author's response to the anticipated suspicion that he attaches a superstitious or fanciful importance to the number three, and forces divisions to a Procrustean bed of trichotomy."

insights into information and knowledge. His semiosis takes dead aim at the questions of how to represent knowledge, and the role and the unique triadic relationship of signs. Indeed, the sheer consistency and coherence of his logics and philosophical views dovetail directly with the needs of conveying information and knowledge to computers. Except for John Sowa's book [40], now nearly 30 years old, it is time to bring Peirce back into the knowledge representation fold of AI.

The Shadowy Object

When we see something, or point to something, or describe something in words, or think of something, we are, of course, using proxies in some manner for the actual thing. If the something is a 'toucan' bird, that bird does not reside in our head when we think of it. The 'it' of the toucan is a 'representation' of the real, dynamic toucan. The representation of something is never the actual something but is itself another thing that conveys to us the idea of the real something. In our daily thinking we rarely make this distinction, thankfully; otherwise, our flow of thoughts would be wholly jangled. Nonetheless, the difference is real, and we should be conscious of it when inspecting the nature of knowledge representation.

How we 'represent' something is also not uniform or consistent. For the toucan bird, perhaps we make caw-caw bird noises or flap our arms to indicate we are referring to a bird. Perhaps we point at the bird. Alternatively, perhaps we show a picture of a toucan or read or say aloud the word "toucan" or see the word embedded in a sentence or paragraph, as in this one, that also provides additional context. How quickly or accurately we grasp the idea of 'toucan' is partly a function of how closely associated one of these signs may be to the idea of toucan bird. Probably all of us would agree that arm flapping is not nearly as useful as a movie of a toucan in flight or seeing one scolding from a tree branch to convey the 'toucan' concept. There's a reason why we love the game of charades.

The question of what we know and how we know it fascinated Peirce over the course of his intellectual life. He probed this relationship between the real or actual thing, the *object*, and how that thing is represented and understood. This triadic relationship between immediate object, representation, and interpretation forms a *sign* and is the basis for the process of sign-making and understanding that Peirce called *semiosis*.[17]

Even the idea of the *object*, in this case, the toucan bird, is not necessarily so simple. The real thing itself, the toucan bird, has characters and attributes. How do we 'know' this real thing? Bees, like many insects, may perceive different coloration for the toucan and adjacent flowers because they can see in the ultraviolet spectrum, while we do not. On the other hand, most mammals in the rainforest

[17] Peirce actually spelled it "*semeiosis*." However it is true that other philosophers such as Ferdinand de Saussure also employed the shorter term "semiosis." I also use this more common term due to greater familiarity.

would also not perceive the reds and oranges of the toucan's feathers, which we readily see. Perhaps only fellow toucans could perceive by gestures and actions whether the object toucan is healthy, happy, or sad (in the toucan way). Humans, through our ingenuity, may create devices or technologies that expand our standard sensory capabilities to make up for some of these perceptual gaps, but technology will never make our knowledge fully complete. Given limits to perceptions and the information we have on hand, we can never completely capture the nature of the dynamic object, the real toucan bird.

Alternatively, let's take another example more in keeping with the symbolic nature of KR, in this case, the word for 'bank.' We can see this word, and, if we speak English, even recognize it, but what does this symbol mean? A financial institution? The shore of a river? Turning an airplane? A kind of pool shot? Tending a fire for the evening? In all of these examples, an actual object is the focus of attention. What we 'know' about this object depends on what we perceive or understand and who or what is doing the perceiving and the understanding. We can never fully 'know' the object because we can never encompass all perspectives and interpretations.

Peirce well recognized these distinctions. As we noted before, he termed the object of the representations as the *immediate object* while also acknowledging that this representation does not fully capture the underlying, real *dynamical object*:

> Every cognition involves something represented, or that of which we are conscious, and some action or passion of the self whereby it becomes represented. The former shall be termed the objective, the latter the subjective, element of the cognition. The cognition itself is an intuition of its objective element, which may therefore be called, also, the immediate object (1868, CP 5.238).

> Namely, we have to distinguish the Immediate Object, which is the Object as the Sign itself represents it, and whose Being is thus dependent upon the Representation of it in the Sign, from the Dynamical Object, which is the Reality which by some means contrives to determine the Sign to its Representation (1906, CP 4.536).

> As to the Object, that may mean the Object as cognized in the Sign and therefore an Idea, or it may be the Object as it is regardless of any particular aspect of it, the Object in such relations as unlimited and final study would show it to be. The former I call the *Immediate Object*, the latter the *Dynamical Object* (1909, CP 8.183).[18]

One imperative of knowledge representation—within reasonable limits—is to try to ensure that our immediate representation of the objects of our discourse is in close correspondence to the dynamic object. This imperative, of course, does not mean assembling every minute bit of information possible to characterize our knowledge spaces. Instead, we need to seek a balance between what and how we characterize the instances in our domains with the questions we are trying to address, all within limited time and budgets. Peirce's pragmatism, as expressed through his *pragmatic maxim* discussed in Chap. 14, helps us reach this balance.

[18] See further the prior Figs. 1.1 and 2.1.

Three Modes of Representation

Representations are signs (1905, CP 8.191), and the means by which we point to, draw, or direct attention to, or designate, denote, or describe a particular object, entity, event, type, or general. In Peirce's mature theory of signs, he characterizes signs according to different typologies, which we cover in this and later sections. One of his better known typologies is how we may denote the object, which, unlike some of his other typologies, he kept relatively constant throughout his life. Peirce formally splits these denotative representations into three kinds: *icons*, *indexes*, or *symbols* (CP 2.228, CP 2.229 and CP 5.473).

> … there are three kinds of signs which are all indispensable in all reasoning; the first is the diagrammatic sign or *icon*, which exhibits a similarity or analogy to the subject of discourse; the second is the *index*, which like a pronoun demonstrative or relative, forces the attention to the particular object intended without describing it; the third [or *symbol*] is the general name or description which signifies its object by means of an association of ideas or habitual connection between the name and the character signified (1885, CP 1.369).

The *icon*, which may also be known as a *likeness* or *semblance*, has a quality shared with the object such that it resembles or imitates it (see Table 2.1). Portraits, logos, diagrams, and metaphors all have an iconic denotation. Peirce also views algebraic expressions as icons since he believed (and did much to prove) that mathematical operations can be expressed through diagrammatic means (as is the case with his later existential graphs).

An *index* denotes the object by some form of linkage or connection. An index draws or compels attention to the object by this genuine connection, and does not require any interpretation or assertion about the nature of the object. A finger pointed at an object and a weathervane indicating which direction the wind is blowing are indexes, as are keys in database tables or Web addresses (IRIs or URLs[19]) on the Internet. Pronouns, proper names, and figure legends are also indexes.

Table 2.1 Three ways to denote objects of signs

An icon	}	is a sign fit to be used as such because	{	it possesses the quality signified.
An index				it is in real reaction with the object denoted.
A symbol				it determines the interpretant sign.

[19] The URI "sign" is best seen as an *index*: the URI is a pointer to a representation of some form, be it electronic or otherwise. This representation bears a relation to the actual thing that this referent represents, as is true for all triadic sign relationships. However, in some contexts, again in keeping with additional signs interpreting signs in other roles, the URI "sign" may also play the role of a symbolic "name" or even as a signal that the resource can be downloaded or accessed in electronic form. In other words, by virtue of the conventions that we choose to assign to our signs, we can supply additional information that augments our understanding of what the URI is, what it means, and how it is accessed.

Symbols, the third kind of denotation, represent the object by accepted conventions or 'laws' or 'habits' (Peirce's preferred terms). Symbols are an understood interpretation, gained through communication and social consensus. All words are symbols, plus their combinations into sentences and paragraphs. All symbols are generals, though we express them as individual instances or tokens. For example, 'the' is a single symbol (type), but it is expressed many times (tokens) on this page. Knowledge representation, by definition, is based on symbols, which are interpreted by either humans or machines based on the conventions and shared understandings we have given them. When Peirce returned to the investigation of signs later in his career, he attempted many times to help clarify how to best distinguish between these three. For example:

> There is an infallible criterion for distinguishing between an index and an icon. Namely, although an index, like any other sign, only functions as a sign when it is interpreted, yet though it never happened to be interpreted, it remains equally fitted to be the very sign that would be if interpreted. A symbol, on the other hand, that should not be interpreted, would either not be a sign at all, or would only be a sign in an utterly different way. An inscription that nobody ever had interpreted or ever would interpret would be but a fanciful scrawl, an index that some being had been there, but not at all conveying or apt to convey its meaning (1904, NEM 4:256).

Peirce confined the word *representation* to the operation of a sign or its relation to the interpreter for an object. The three possible modes of denotation—that is, icon, index, or symbol—Peirce collectively termed the *representamen*:

> A very broad and important class of triadic characters [consists of] representations. A representation is that character of a thing by virtue of which, for the production of a certain mental effect, it may stand in place of another thing. The thing having this character I term a *representamen*, the mental effect, or thought, its *interpretant*, the thing for which it stands, its *object* (1897, CP 1.564).

Symbols are in Thirdness, one of the universal categories we discuss at length in Chap. 6. As a preview, though, understand that these symbols are themselves representations, which build in an ever-growing cascade, to convey deeper and more complicated representations, each with a meaning to its interpretant:

> The easiest of those [ideas in which Thirdness is predominant] which are of philosophical interest is the idea of a sign, or representation. A sign stands *for* something *to* the idea which it produces, or modifies. Or, it is a vehicle conveying into the mind something from without. That for which it stands is called its *Object*; that which it conveys, its *Meaning*; and the idea to which it gives rise, its *Interpretant*. The object of a representation can be nothing but a representation of which the first representation is the interpretant. But an endless series of representations each representing the one behind it may be conceived to have an absolute object at its limit. The meaning of a representation can be nothing but a representation (1893, NEM4:309-310; MS 717).

Again, note that representation is the complete triadic sign, while meaning is the understanding conveyed by the symbolic representation, as understood and acted upon by the interpreting agent.

Peirce's Semiosis and Triadomany

In the same early 1867 paper in which Peirce laid out the three modes of denotation of icon, index, and symbol [39], he also presented his three phenomenological categories for the first time, what I (and others) have come to call his *universal categories* of Firstness, Secondness, and Thirdness.[20] This seminal paper also provides the contextual embedding of these categories, which is worth repeating in full:

> "The five conceptions thus obtained, for reasons which will be sufficiently obvious, may be termed *categories*. That is,
>
> BEING,
>
> Quality (reference to a ground),
>
> Relation (reference to a correlate),
>
> Representation (reference to an interpretant),
>
> SUBSTANCE.
>
> The three intermediate conceptions may be termed accidents (1896, EP 1:6, CP 1.55).

Note the commas, suggesting the order, and the period, in the listing. In his later writings, Peirce ceases to discuss Being and Substance directly, instead focusing on the 'accidental' categories that became the first expression of his universal categories. Being, the starting point, is the absolute, most abstract beginning for Peirce's epistemology.[21] The three 'accidental' categories of quality, relation, and representation are one of the first expressions of Peirce's universal categories of Firstness, Secondness, and Thirdness, as applied to substance. "Thus substance and being are the beginning and end of all conception. Substance is inapplicable to a predicate, and being is equally so to a subject" (1867, CP 1.548).

These two, early triadic relations—one, the denotations in signs, and, two, the universal categories—are examples of Peirce's lifelong fascination with trichotomies (see footnote 16). He used triadic thinking in dozens of areas in his various investigations,[22] often in a recursive manner (threes of threes). It is not surprising, then, that Peirce also applied this mindset to the general characterization of signs themselves.

Peirce returned to the idea of sign typologies and notations at the time of his Lowell Institute lectures at Harvard in 1903 [41]. Peirce expanded upon his first triad of icons, indexes, and symbols with two additional trichotomies.

---

[20] See entire Chap. 6, especially Table 6.2.

[21] The polymath Buckminster Fuller also maintained that the triad was the most stable structure in the universe; see https://bookofthrees.com/buckminster-fuller-building-blocks/.

[22] Table 6.2 lists more than 60 examples.

In one of these additions, that of the second trichotomy, Peirce proferred three ways to describe the use of signs. These three uses are *qualisigns* (also called *tones*, *potisigns*, or *marks*), which are signs that consist of a quality of feeling or possibility, and are in Firstness; *sinsigns* (also called *tokens* or *actisigns*), which consist of action/reaction or actual single occurrences or facts, and are in Secondness; or *legisigns* (also called *types* or *famisigns*), which are signs that consist of generals or representational relations, and are in Thirdness. Instances (tokens) of legisigns are replicas and thus are a sinsign. All symbols are legisigns. Synonyms, for example, are replicas of the same legisign, since they mean the same thing, but are different sinsigns.

In the second of these additions, the third trichotomy, Peirce described three ways to interpret signs (*interpretant*) based on possibility, fact, or reason. A *rheme* (also called *sumisign* or *seme*) is in Firstness and is a sign that stands for its object for some purpose, expressed as a character or a mark. Terms are rhemes, but they also may be icons or indexes. Rhemes may be diagrams, proper nouns, or common nouns. A proposition expressed with its subject as a blank (unspecified) is also a rheme. A *dicisign* (also called *dicent* or *pheme*) is the second interpretation of a sign. A dicent is in Secondness and is a fact of actual existence. Icons cannot be dicisigns. Dicisigns may be either indexes or symbols and provide indicators or pointers to the object. Standard propositions or assertions are dicisigns. An *argument* (also called *suadisign* or *delome*) is the third way of a reasoning sign, in Thirdness, and stands for the object as a generality, law, or habit. A sign itself is an argument, including major and minor premises and conclusions. Combinations of assertions or statements, such as novels or works of art, are arguments. Context resides in Thirdness.

One might expect these three Peircean sign trichotomies to result in 27 different possibilities ($3 \times 3 \times 3$). However, the nature of the monadic, dyadic, and triadic relationships embedded in these trichotomies only logically leads to ten variants ($1 + 3 + 6$).[23] Table 2.2 summarizes these ten sign types and provides some examples of how to understand them. The $1 + 3 + 6$ variants include Sign I, Signs II to IV, and Signs V to X, respectively, as shown in the table.

[23] Understand that each trichotomy is comprised of three elements, A, B, and C. The monadic relations are a singleton, A, which can only match with itself and A variants. The dyadic relations can only be between A and B and derivatives. And the triadic relations are between all variants and derivatives. Thus, the ten logical combinations for the three trichotomies are A-A'-A"; B-A'-A"; B-B'-A"; B-B'-B"; C-A'-A"; C-B'-A"; C-B'-B"; C-C'-A"; C-C'-B"; and C-C'-C", for a total of ten options.

Table 2.2 Ten classifications of signs[a]

	Sign by use	Relative to object	Relative to interpretant	Sign name (redundancies)	Some examples
I	Qualisign	Icon	Rheme	(Rhematic Iconic) Qualisign	A feeling of 'red'
II	Sinsign	Icon	Rheme	(Rhematic) Iconic Sinsign	An individual diagram
III		Index	Rheme	Rhematic Indexical Sinsign	A spontaneous cry
IV			Dicisign	Dicent (Indexical) Sinsign	A weathercock or photograph
V	Legisign	Icon	Rheme	(Rhematic) Iconic Legisign	A diagram, apart from its factual individuality
VI		Index	Rheme	Rhematic Indexical Legisign	A demonstrative pronoun
VII			Dicisign	Dicent Indexical Legisign	A street cry (identifying the individual by tone, theme)
VIII		Symbol	Rheme	Rhematic Symbol (Legisign)	A common noun
IX			Dicisign	Dicent Symbol (Legisign)	A proposition (in the conventional sense)
X			Argument	Argument (Symbolic Legisign)	A syllogism

[a]From CP 2.254-263, EP 2:294-296, and MS 540 of 1903

The schema in Table 2.2 is the last one fully developed by Peirce. We will next return to this schema in Chap. 16 (specifically Table 16.3) when we turn to the topic of semantic parsing of natural language. However, also realize that, in Peirce's last years, he also developed 28-class and 66-class sign typologies, though incomplete in important ways and details. These expansions reflected sign elaborations for various subclasses of Peirce's more mature trichotomies, such as for the immediate and dynamic objects previously discussed (*c.f.*, 1904, CP 8.342–379).

A symmetrical and recursive beauty exists in these incomplete efforts, with sufficient methodology suggested to enable informed speculations as to where Peirce may have been heading [33, 42–44]. Twenty-five years ago Nathan Houser opined that "… a sound and detailed extension of Peirce's analysis of signs to his full set of ten divisions and sixty-six classes is perhaps the most pressing problem for Peircean semioticians" [45]. I somewhat agree, but applying the *pragmatic maxim* suggests that it is not the next priority. True, with much digging the archeology of Peirce's intent at the time may be discerned to some degree. However, Peirce himself would likely have reconsidered and revised his views, as he was wont to do over time, especially in light of massive changes in knowledge over the past century. Such is the nature of knowledge, and how we dynamically respond to it.

A significant portion of the Peircean community believes that signs and semiosis are the central aspects underlying Peirce's philosophy. Passages in Peirce's writings support this interpretation. However, I agree that Peirce's 'theory of categories,' to

use Siosifa Ika's phrase, is the better key to understanding Peirce's metaphysical and epistemological realism [46]. Besides Ika's well-reasoned thesis, I argue three additional reasons to see the universal categories as the more fundamental driver. First, Peirce, as we noted, conducted his thought in threes and tried to reason in threes. Second, similar and compelling passages in Peirce (see throughout and in Appendix A) support the primacy of the universal categories in contradistinction to signs. Third, the categories prescind[24] both signs and logic, indicating their superordinate position.

Thus, in this book, we take a different path. Rather than engaging in the archeology of Peirce's intended sign schemas, I have chosen to try to fathom and plumb Peirce's mindset. His explication of the centrality and power of signs, his fierce belief in logic and reality, and his commitment to discovering the fundamental roots of *epistêmê* guide how to think about knowledge representation attuned to today. I believe that Peirce's triadomany (see footnote 16), especially as expressed through the universal categories, provides the illuminating light to this guidance.

References

1. T.W. Deacon, *The Symbolic Species: The Co-Evolution of Language and the Brain* (W.W. Norton, New York, 1997)
2. G. Bateson, *Steps to an Ecology of Mind: Collected Essays in Anthropology, Psychiatry, Evolution, and Epistemology* (University of Chicago Press, Chicago, 1972)
3. C.E. Shannon, A mathematical theory of communication. Bell Syst. Tech. J. **27**, 379–423 (1948)
4. R. Landauer, Irreversibility and heat generation in the computing process. IBM J. Res. Dev. **5**, 183–191 (1961)
5. R. Landauer, Information is physical. Phys. Today **44**(5), 23–29 (1991)
6. A. Bérut, A. Arakelyan, A. Petrosyan, S. Ciliberto, R. Dillenschneider, E. Lutz, Experimental verification of Landauer's principle linking information and thermodynamics. Nature **483**, 187–189 (2012)
7. M. Karnani, K. Paakkonen, A. Annila, The physical character of information. Proc. R. Soc. A Math. Phys. Eng. Sci. **465**, 2155–2175 (2009)
8. C.E. Shannon, W. Weaver, *The Mathematical Theory of Communication* (University of Illinois Press, Urbana, IL, 1998)
9. R.M. Losee, *Information from Processes: About the Nature of Information Creation, Use, and Representation* (Springer Science & Business Media, Berlin, 2012)
10. J. Furner, in *Information Cultures in the Digital Age*, ed. by M. Kelly, J. Bielby. 'Data': The data (Springer, Wiesbaden, 2016), pp. 287–306
11. T.D. Schneider, Information is Not Entropy, Information is Not Uncertainty! *Molecular Information Theory and the Theory of Molecular Machines*. https://schneider.ncifcrf.gov/information.is.not.uncertainty.html
12. J. Koelman, What Is Entropy? *Science 2.0*. http://www.science20.com/hammock_physicist/what_entropy-89730
13. E.T. Jaynes, Information theory and statistical mechanics. Phys. Rev. **106**, 620–630 (1957)
14. B. Zimmer, The surprising origins of life's complexity. *Quanta Magazine*. https://www.quantamagazine.org/the-surprising-origins-of-lifes-complexity-20130716/

[24] See the section on *Hierarchies* in Chap. 7.

15. S. Sarkar, J.L. England, Sufficient physical conditions for self-replication. *arXiv:1709.09191 [cond-mat, physics:physics]* (September 2017)
16. E. Schrödinger, *What Is Life?: The Physical Aspect of the Living Cell* (University Press, Trinity College, Dublin, 1944)
17. L. Brillouin, The Negentropy principle of information. J. Appl. Phys. **24**, 1152–1163 (1953)
18. I. Prigogine, G. Nicolis, Biological order, structure and instabilities. Q. Rev. Biophys. **2–4**, 107–148 (1971)
19. A. Annila, S. Salthe, Physical foundations of evolutionary theory. J. Non-Equilib. Thermodyn. **35**, 301–321 (2010)
20. G.E. Crooks, The entropy production fluctuation theorem and the nonequilibrium work relation for free energy differences. Phys. Rev. E **60**, 2721–2726 (1999)
21. J.L. England, Dissipative adaptation in driven self-assembly. Nat. Nanotechnol. **10**, 919–923 (2015)
22. S.N. Salthe, Naturalizing information. Inf. Dent. **2**, 417–425 (2011)
23. J.L. England, Statistical physics of self-replication. J. Chem. Phys. **139**, 121923 (2013)
24. A.D. Wissner-Gross, C.E. Freer, Causal entropic forces. Phys. Rev. Lett. **110**, 168702 (2013)
25. J.J. Eaton, D. Bawden, What kind of resource is information? Int. J. Inf. Manag. **11**, 156–165 (1991)
26. M.J. Bates, Information and knowledge: An evolutionary framework for information science. Inf. Res. Int. Electron. J. **10**, n4 (2005)
27. A. Bejan, S. Lorente, The constructal law and the evolution of design in nature. Phys. Life Rev. **8**, 209–240 (2011)
28. J. Warner, *Human Information Retrieval* (MIT Press, Cambridge, MA, 2010)
29. K. Kelley, *What Technology Wants* (Viking/Penguin, New York, 2010)
30. J. Mokyr, *The Gifts of Athena: Historical Origins of the Knowledge Economy* (Princeton University Press, Princeton, 2002)
31. S. Wolfram, *A New Kind of Science* (Wolfram Media, Champaign, IL, 2002)
32. C.S. Peirce, L.V. Welby, *Semiotic and Significs: The Correspondence Between Charles S. Peirce and Lady Victoria Welby* (Indiana University Press, Bloomington, 1977)
33. R.W. Burch, Peirce's 10, 28, and 66 sign-types: The simplest mathematics. Sem. Ther. **2011** (2011)
34. C.S. Peirce, The fixation of belief. Pop. Sci. Monthly **12**, 1–15 (1877)
35. T. Hobbes, *Elements of Law, Natural and Political* (Routledge, Abingdon, 2013)
36. C.S. Peirce, *What is in a Sign?* (Indiana University Press, Bloomington, IN, 1894)
37. R. Davis, H. Shrobe, P. Szolovits, What is a knowledge representation? AI Mag. **14**, 17 (1993)
38. Y. Zhou, A set theoretic approach for knowledge representation: The representation part. *arXiv*, vol. 1603 (March 2016)
39. C.S. Peirce, On a new list of categories. Proc. Am. Acad. Arts Sci. **VII**, 287–298 (1867)
40. J.F. Sowa, *Knowledge Representation: Logical, Philosophical, and Computational Foundations* (Brooks/Cole, Pacific Grove, 2000)
41. C.S. Peirce, The Peirce Edition Project, Nomenclature and divisions of triadic relations, as far as they are determined, The Essential Peirce: Selected Philosophical Writings, Volume 2 (1893–1913) (Indiana University Press, Bloomington, IN, 1998), pp. 289–299.
42. Borges P. A visual model of Peirce's 66 classes of signs unravels his late proposal of enlarging semiotic theory. In Model-Based Reasoning in Science and Technology 2010 (pp. 221–237). Springer, Berlin, Heidelberg.
43. P. Farias, J. Queiroz, On diagrams for Peirce's 10, 28, and 66 classes of signs. Sem. Ther. **147**, 165–184 (2003)
44. T. Jappy, *Peirce's Twenty-Eight Classes of Signs and the Philosophy of Representation: Rhetoric, Interpretation and Hexadic Semiosis* (Bloomsbury Academic, London, 2017)
45. C.S. Peirce, *The Essential Peirce: Selected Philosophical Writings, Vol 1 (1867–1893)* (Indiana University Press, Bloomington, 1992)
46. S. Ika, *A Critical Examination of the Philosophy of Charles S. Peirce: A Defence of the Claim that his Pragmatism is Founded on his Theory of Categories*. Ph.D., University of Notre Dame Australia (2002)

Part I
Knowledge Representation in Context

Chapter 3
The Situation

Since Adam Smith, a focus of economics has been its attempt to explain the basis of growth. This emphasis is not surprising since the birth of the field of economics corresponded to a historically unprecedented inflection point in economic growth. Smith ascribed this growth to productivity resulting from the division of labor, using his famous example of the pin factory. However, it is only within the past 50 years or so that economists have begun unpacking growth from the other factors of production.[1] In this chapter, we talk specifically about the role of information in growth, and how it may contribute further.

Growth is a percent increase from a prior state. Economic growth compounded over a period has the virtuous reward of resulting in increased wealth. We measure economic growth through such means as revenues (for the individual firm) or GDP (for regions or countries). Net worth (for the firm or individuals) or GDP per capita measure the wealth associated with the current stock of economic goods at any given point in time. Such measures, while useful proxies, still do not account for other changes in comfort, convenience, freedom, choice, leisure, and mobility that may accompany growth and transcend the material. On the other hand, growth may also create 'externalities,' some of which may be negative such as pollution or traffic congestion. Wealthier societies have tended to regulate against such harmful effects over time. We should include all of these factors in the value equation.

[1] Some material in this chapter was drawn from the author's prior articles at the *AI3:::Adaptive Information* blog: "Untapped Assets: The $3 Trillion Value of US Enterprise Documents" (Jul 2005); "A Trillion is a Large Number (12 zeros)" (Jul 2005); "Why Are $800 Billion in Document Assets Wasted Annually? V Summary" (Mar 2006); "Climbing the Data Federation Pyramid" (May 2006); "Knowledge: Unravelling the X Factor in Growth and Wealth" (Jun 2006); "Historical Origins of the Knowledge Economy" (Jul 2006); "Information is the Basis for Economic Growth" (Aug 2007); "When Linked Data Rules Fail" (Nov 2009); "Declining IT Innovation in the Enterprise" (Jan 2011); "Spring Dawns on Artificial Intelligence" (Jun 2014); "Innovation, Information, Growth and Wealth" (Jun 2014); "Big Structure and Data Interoperability" (Aug 2014); "Logical Implications of Interoperability" (Jun 2015); "Hidden Expenses Underneath Machine Learning" (Feb 2016).

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8_3

Throughout history, we have seen discontinuities in growth (and then wealth) for individuals, families, firms, industries, cities, regions, and nations. Growth thus has immense importance across the entire economic spectrum. This chapter makes the argument that access to information—and impediments to that—is a significant determinant of wealth and economic growth. Better knowledge representation using computers is one means to improve the economic well-being of all peoples.

Information and Economic Wealth

If we toil, year by year, doing the same activity, like growing wheat and we gain the same harvest for the same inputs of labor and land, we are not surprised. Sometimes, the weather or rainfall patterns may differ, or we may have more children helping us in the fields, or a mule to help plow. Money helps us buy more of the essential inputs, maybe more land, seed, or mules, or the comfort to have more children. These are the traditional factors of production: that is, land, capital, and labor.

If we add more of these factors to the mix, we still understand we have merely tweaked the regular basis of our wheat production. Differences in the amount of these factors of production, throughout most of human history, are what accounted for the differences between rich and poor, landlord and serf. If by having more land or children, we are now able to feed more people, we are by definition more wealthy, and if we can accumulate more of this wealth, we can leverage these standard factors even more. Control and exploitation have been common paths to much wealth creation.

These factors are pretty easy to observe and track. We intuitively understand that more inputs of labor, land, or capital can result in growth, but one that feels and appears somewhat fixed based on the change in these inputs. This kind of growth has a more or less trending return based on changes in these inputs. These types of inputs may also be subject to diminishing returns, wherein adding more of a given factor reduces payoff. For example, adding more fertilizer to the wheat crop produces less per unit output yield after some optimum, eventually lowering yields by chemically burning the crop. Alternatively, while a computer increases the productivity of an individual worker, giving her more computers may degrade her overall performance.

Historical Breakpoints

Still, a different kind of growth is not constrained to a fixed return based on inputs. Perhaps we have a neighbor who raises more wheat, possibly on drier, more marginal land, or with less water or fertilizer. His yield exceeds our own. These differences occur because our neighbor is doing something different and is producing more given his inputs.

Many of us (now) older people can recall grandparents talking about their first sight of a car or an airplane. In my own life (born 1952) I can remember the first instance of color TVs, electronic calculators, personal computers, the Internet, and smartphones. The fact is that the pace of development and technological change is now so constant that its very existence seems unremarkable—part of the daily background noise. For 99.5% of human history, this has not always been so.[2]

In our daily lives we are bombarded by statistics: quarterly economic growth rates, sports scores, weather precipitation likelihoods, and daily temperatures, in a constant and thus background stream of numeric immersion. It is interesting to note that *statistics* (originally derived from the concept of information about the *state*) only began in France in the 1700s. The first actual population census, as opposed to enumerations in biblical times or the land and tax recordings of the Domesday Book in England in 1086, occurred in Spain in that same century, with the United States being the first country to set forth a decennial census beginning around 1790 [1].

Because the state collected no data—indeed, the idea of data and statistics did not exist—attempts in our modern times to re-create economic and population assessments in earlier centuries are a heroic exercise, laden with estimation. Nonetheless, the renowned economic historian Angus Maddison and his team,

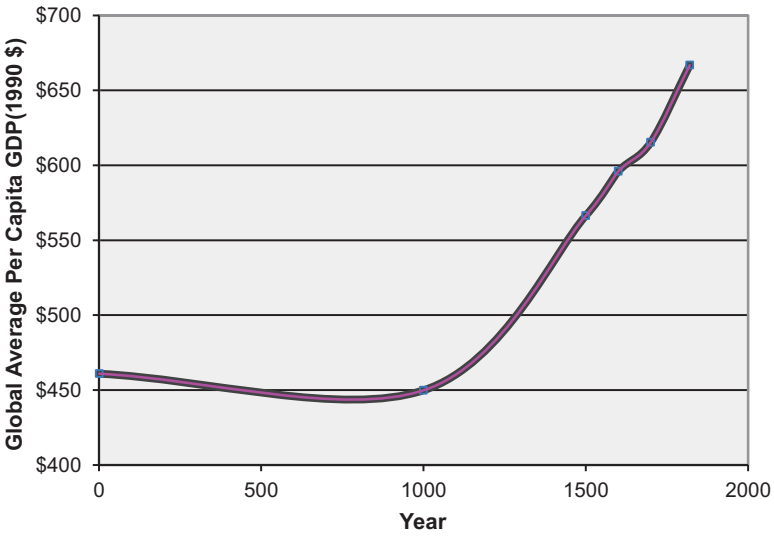


Fig. 3.1 Global average per capita GDP, 1–1800 AD

[2] If we take the first growth breakpoint as 1820, and a nominal estimate of 40,000 years for human history (fudging a bit on the estimated occurrence of cave paintings to account for oral histories), we end up with exactly 99.5%.

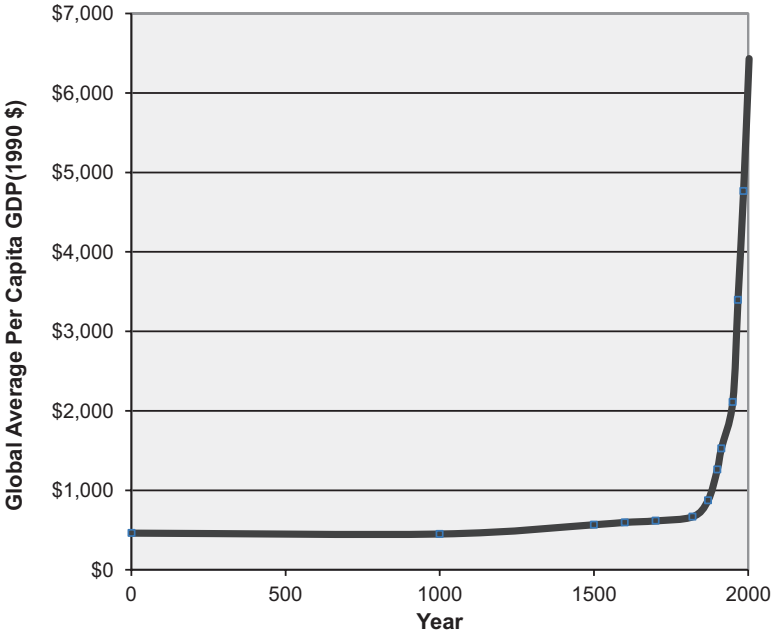


Fig. 3.2 Growth skyrockets about 1820 AD

written in some definitive OECD studies, prepared economic and population growth estimates for the world and various regions going back to AD 1 [1].[3]

Through at least 1000 AD global economic growth per capita (as well as population growth) was approximately flat. Maddison estimated that a doubling of economic well-being per capita only occurred every 3000 to 4000 years. A historical shift occurred about 1000 AD when flat or negative growth began to accelerate slightly.[4] The growth trend looks comparatively impressive in Fig. 3.1, but growth over a period of about 800 years to 1750 AD only totals 45%, a doubling of per capita wealth that still requires 1000–2000 years. These are annual growth rates

[3] The historical data were originally developed in three books by Angus Maddison: Monitoring the World Economy 1820–1992, OECD, Paris 1995; The World Economy: A Millennial Perspective, OECD Development Centre, Paris 2001; and The World Economy: Historical Statistics, OECD Development Centre, Paris 2003. All these contain detailed source notes. Figures for 1820 onwards are annual, wherever possible. For earlier years, benchmark figures are shown for 1 AD, 1000 AD, 1500, 1600, and 1700. These figures have been updated to 2003 and may be downloaded by spreadsheet from the Groningen Growth and Development Centre (GGDC), a research group of economists and economic historians at the Economics Department of the University of Groningen headed by Maddison. See http://www.ggdc.net/.

[4] This inflection point in approx. 1000 AD corresponds somewhat to the adoption of raw linen paper v. skins and vellum, among other correlations that might be drawn.

about *30 times lower* than today, which, with compounding, prove anemic indeed over such long historical periods.

By 1820 or so onward, this doubling accelerated at warp speed to every 50 years or so, as shown in Fig. 3.2. Historically flat income averages skyrocketed, as this famous figure showing global changes in per capita (person) GDP from Maddison illustrates (see footnotes 2, 3, 4).[5]

William Nordhaus captured a similar discontinuity looking at the price of light, normalized according to the labor effort needed to obtain 1000 lumens. His study, too, shows an exponential decrease in the price of lighting beginning about 1800 [2]. More recent trends show an additional upward blip in growth shortly after the turn of the twentieth century, corresponding to electrification, but then a more massive discontinuity begins after World War II, as next shown in Fig. 3.3. Growth rates accelerated to a doubling of wealth every 40–45 years.

These comparatively abrupt changes in growth rates and concomitant changes in wealth were more than two orders of magnitude higher than what had been experienced before in human history, and thus garnered the attention of economists and economic historians as never before. Something huge *did* happen in the early 1800s.

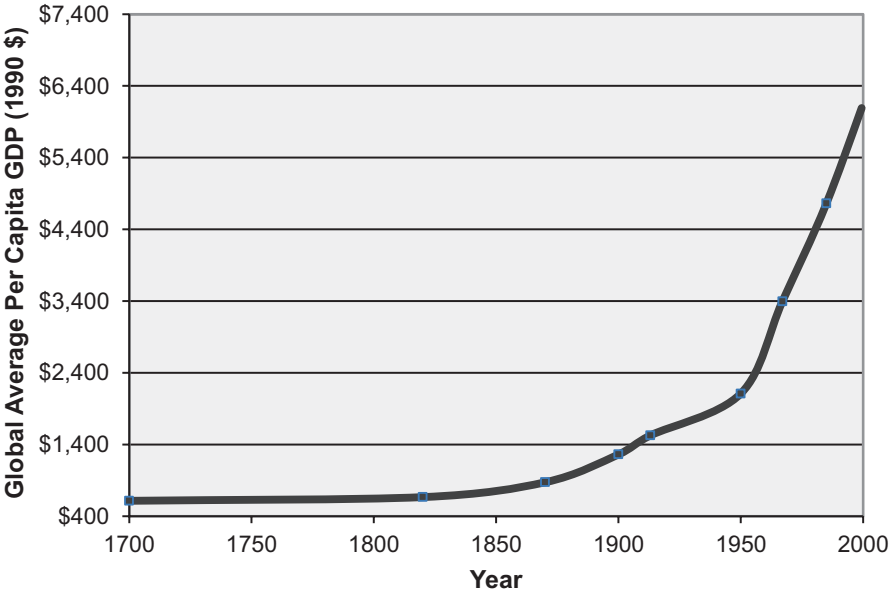


Fig. 3.3 Growth skyrockets again in 1950s AD

[5] In Paul Starr's *Creation of the Media: Political Origins of Modern Communications*, Basic Books, 1-2005, he notes how in 15 years from 1835 to 1850 the cost of setting up a mass-circulation paper increased from $10,000 to over $2 million (in 2005 dollars). True, mechanization was increasing costs, but from the standpoint of consumers, the cost of information content was dropping to zero and approaching a near-time immediacy. The concept of "news" was coined, delivered by the "press" for a now-emerging "mass media."

Since their occurrence, many have attributed the inflection points in growth rates of the 1820s and 1950s to 'technological change,' but the specific causes of this change lack consensus. The prior era of the Enlightenment suggested some fundamental shift in thinking. Had a notable transition occurred in the mid-1400s to 1500s it would have been obvious to ascribe more modern economic growth trends with the availability of the printing press. While the printing press had massive effects as Elizabeth Eisenstein has shown [3], the empirical record of changes in economic growth is not coincident with its adoption. The closer concurrence with the Industrial Revolution lent credence to the adoption of machines, prime movers, and harnessing of energy as a likely explanation. Cultural and religious factors have also been posited to explain why Britain and then the United States were the original centers of growth. Earlier, I noted the invisible hand of the market and division of labor and specialization, as advocated by Adam Smith. Education, followed by literacy, and support for basic and applied research have their advocates. Financial and banking innovations and the rule of law and patents and other intellectual property rights are other possible causes.

Common sense tells us that all of these factors, and perhaps more, can all work as force multipliers to the traditional inputs to the economic function. However, I posit that one element reigned supreme in these trends—information.

The X Factor of Information

Joel Mokyr provides a sweeping and comprehensive account of the period from 1760 (what he calls the 'Industrial Enlightenment') through the Industrial Revolution beginning roughly in 1820 and then continuing through the end of the nineteenth century [4]. Mokyr centers his explanation for these growth changes on 'useful knowledge,' a phrase first coined by Simon Kuznets, as expanded upon by Michael Polanyi and others. Mokyr argues how *propositional knowledge,* the base of knowledge such as science, combines with *prescriptive knowledge*, the 'recipes' for applying knowledge, with *discovery*, to create the innovations that fueled the observed growth acceleration.

One of Mokyr's key points is that both kinds of knowledge reinforce one another, with that timeframe being a period of unprecedented growth in such knowledge. Another point, easily overlooked since 'discoveries' are the most visible, is that *techniques* and *practical applications* of knowledge provide a multiplier effect to knowledge growth. Mokyr notes that the inventions of writing, paper, and printing not only significantly reduced access costs but also materially affected human cognition, including the way people thought about their environment. Mokyr notes but does not adequately pursue, "In the decades after 1815, a veritable explosion of technical literature took place. Comprehensive technical compendia appeared in every industrial field." Statements such as these in his outstanding book, *The Gifts of Athena*, hint at these fundamental drivers.

The industrialization that proceeded apace in the Americas and Europe is the engine that produced the wealth reflected in the earlier figures. However, from where did that mechanization and know-how come? It came from innovations and improved methods, for sure, but the more direct cause, I believe, was the broader dissemination of information. Our first inflection point in the 1820s roughly corresponds to the innovation of cheaper 'pulp' paper (and the genesis of 'pulp' fiction by serialized writers like Dickens or Hugo) (see footnote 5); the second inflection point in the 1950s corresponds to the beginning use of the computer and digital information. Change was everywhere, and many factors were at work. It is hard to deny that information and greater access to it must surely have been central factors for increased innovation, literacy, and social and political change.

Knowledge and Innovation

Until the mid-1950s, economists ascribed the sources of this notable growth to 'technological change' and other vague factors, often argued in anecdotal ways. Empirical datasets were few and far between to test hypotheses, and quantitative means of reasoning over economic problems were only beginning. Growth theory was becoming an economic discipline in its own right.

Joseph Schumpeter, in *The Theory of Economic Development*, first published in 1911 [5], argued that innovation is central to economic growth and continuously disrupts the general equilibrium of market exchange. Innovation gains the firm a temporary monopoly status in which to charge higher rents, thereby providing an incentive for further innovation. Schumpeter's emphasis on entrepreneurship and his popularization of 'creative destruction' recognized that new innovative market entrants might cause older firms to become obsolete. He tied these ideas into his basic views on business cycles, also driven by technological change. Innovation was central to Schumpeter's economic worldview.

The theoretical story begins in earnest after World War II when the concept of *total factor productivity* came to the fore. Robert Solow is an American economist mainly known for his work on the theory of economic growth; we name the exogenous growth model after his work. Solow took courses from Schumpeter at Harvard and was influenced by his views on innovation and technological change, though Solow was also part of the generation of economists embracing the new discipline of mathematical or quantitative economics, which was foreign to Schumpeter [6]. Solow's insight in two papers in 1956 and 1957, for which he won a Nobel Prize, was that technological change, what he called 'technological progress,' must be the "residual" leftover from empirical growth once we remove the traditional inputs of labor and capital.[6] Total factor productivity (TFP) is the 'residual' in total output not

[6] Solow's endogenous model of economic growth is also known as the Solow-Swan neoclassical growth model as the model was independently discovered by Trevor W. Swan and published in "The Economic Record" in 1956, and allows the determinants of economic growth to be separated out into increases in inputs (labor and capital) and technical progress.

credited to the traditional inputs of labor and capital.[7] Solow calculated that 87.5% of the growth in US output per worker was due to technical progress [7]. In 1954, Solomon Fabricant similarly estimated the amount as 90% [8]. But these were 'lumpy' measures; factors like a changing composition of the workforce (especially the growth of women and two-earner families) were also at play.

Fritz Machlup's seminal 1962 book, *The Production and Distribution of Knowledge in the United States*, was the first to coin the terms 'knowledge industry' and 'knowledge worker' [9]. It noted that the knowledge industry generated 29% of the US GNP in 1958 [10]. Marc Porat updated Machlup's efforts for 1967 using a different methodology based on national income accounts, an approach that is less comprehensive than Machlup's, but which has the advantage of relying on standard data collection [27]. This effort, *The Information Economy*, authored with Michael Rubin in 1977, was also adapted as the methodology for cross-country comparisons by the OECD in the 1980s [11]. Another influential paper of this era was by Kenneth Arrow in 1962, in which he introduced the concept and evidence for what he called 'learning by doing' [12], what is now more formally understood and accepted as the learning curve. Unlike a specific innovation, the idea of the learning curve captured that experience and practice led to efficiencies and productivity as we master our tasks.

By the 1960s and 1970s, it was becoming clear that developed economies were becoming information economies, increasingly staffed by knowledge workers. These forces needed explicit attention within quantitative economic models. Robert Lucas, now a Nobel laureate from the University of Chicago, probed the questions of rational expectations and internal factors promoting growth. By the mid-1980s, a group of growth theorists had become increasingly dissatisfied with standard accounts of exogenous factors determining long-run growth. The focus shifted to the needs for quantitative models that made these 'technological' or 'information' factors explicit. In other words, these 'X' factors are not a lump, residual consideration as defined by TFP, but are an internal one within the models with multipliers and feedbacks. In short, these new growth factors needed an explicit and *endogenous* (internal) specification in the model, not left as some *exogenous* (external) residual.

Arguably, the field of information economics began with David Lawrence's book, *The Economic Value of Information*, in 1999 [10]. A book by David Warsh in 2007, *Knowledge and the Wealth of Nations: A Story of Economic Discovery* [13], is an explicit account of the transition from TFP to an internal growth model. The book focuses on Paul Romer, then of Stanford University, a recent chief economist of the World Bank, but earlier a colleague of Lucas, pivoting on his seminal paper, "Endogenous Technological Change" [14]. By bringing the consideration internal to the model, it could be inspected and broken into parts. The first Romer insight is that information and its artifacts are also products and outputs of the economic function. Romer's second insight is that once produced, information or knowledge assets

[7] By definition, TFP cannot be measured directly.

may be provided or distributed at essentially zero marginal cost. Romer had added a new dimension of 'rival' and 'non-rival' goods to the growth theory lexicon. Information and knowledge were becoming both inputs and outputs to the economic function. Romer's papers provided the concepts to analyze further the role of information in growth.

For example, between 2000 and 2005, estimates at the industry level indicated that almost half of the aggregate productivity was due to productivity growth originating from information technology [15], though the IT industries themselves only accounted for a little over 3% of nominal aggregate value [16]. Jorgenson, Ho, and Samuels [17] explicitly separated out innovation from the diffusion of prior innovations due to information. The study by Apte and Nath, mostly an update of the earlier analyses by Porat [27], found that by 1997 two-thirds of the US economy was an information one [18].

By 2009, Romer and Jones were able to claim proof for the endogenous growth model, and they put forward six research questions to look for in the coming 25 years, including the role of human capital, differential growth rates between countries, and accelerated growth [11]. Innovation and its grounding in knowledge had finally assumed its central, internal role in economists' understanding of economic growth.

What Schumpeter had referred to as 'innovation' is now understood as too narrow; innovation is but a part of the overall growth effect due to information. What is helpful from these more recent studies is to separate out innovation from information dissemination. The next step, for which we have not yet developed useful datasets, would be to unpack the ideas of innovation and information into the categories from Mokyr [4], namely, propositional and prescriptive knowledge.

Innovation is an individual affair in its discovery, but a communal one in its application, at which point we call it *knowledge*. We mimic innovations that produce real differences. Farming innovations may include better ways of planting or spacing the wheat, perhaps using a plow; selecting specific wheat strains for next year's plantings; irrigating the land; providing harnesses to the mules; or dividing and specializing the responsibilities between the children. Some of these innovations are new devices, such as harnesses or plows. Some of these innovations are new practices, such as tilling or irrigation methods or specializations in tasks or labor. Not every farmer must innovate on his own. Copying and imitation diffuse these changes across farms and workers.

Indeed, for millennia, this is how human progress took place. Some innovations, such as fire, the wheel, iron and bronze, the arch, alphabets, the plow, and the yoke, had material benefits to all who encountered them. These innovations were fundamental and diffused at the pace of human movement. However, one could argue, each arose as a flash of insight and not from a process of systemic information. Further, innovations tended to diffuse slowly, following the pace and concentration of trade routes. The innovative event was quite rare, and most practices had been

stable for centuries. It is not at all surprising that early economic ideas tended to focus on the traditional factors of production of land, labor, and capital. These had been the steady constants for what had been flat growth for centuries.

If the nature of the biological organism is to contain within it genetic information from which adaptations arise that it can pass to offspring via reproduction—an information volume that is inherently limited and only transmittable by single organisms—then the nature of human cultural information is a massive breakpoint. With the fixity and permanence of printing and cheap paper—and now cheap electrons—all prior information across the entire species can be accumulated and passed on to subsequent generations. Our storehouse of available information is thus growing geometrically, and accessible to all, factors that make the fitness of our species indeed a shift from all prior biological beings, including early humans.

It is silly, of course, to point to single factors or offer simplistic slogans about why this growth occurred and when. Indeed, the scientific revolution, industrial revolution, increase in literacy, electrification, printing press, reformation, rise in democracy, and many other plausible and worthy candidates have been brought forward to explain these historical inflections in accelerated growth. For my lights, I believe each of these factors had its role to play. Still, at the most fundamental level, I think the drivers for this growth came from prior human information. Undoubtedly, the printing press helped to increase total volumes, but it was declining paper costs that made information access affordable and (nearly) universal.

Information, specifically nonbiological information passed on through cultural means, is what truly distinguishes us humans from other animals. We have been easily distracted looking at the tangible when it is the information artifacts ('symbols') that make us human. So, the confluence of cheaper machines (steam printing presses) with cheaper paper (pulp) brought information to the masses. In that process, more people learned, more people shared, and more people could innovate. Now, with computers and the Internet, we can also digitize and place nearly all of the accumulated human knowledge into anyone's hands. What will that bring?

Untapped Information Assets

Today, in the advanced knowledge economy of the United States, the information contained within documents represents about a third of total gross domestic product. Some 25% of the annual trillions of dollars spent on document creation could benefit from actionable improvements [19]. If we are to improve our management and use of information, we need to understand how much value we routinely throw away.

Valuing Information as an Asset

For an enterprise, we can define intangible assets as private expenditures on assets that are intangible and necessary to the creation and sale of new or improved products and processes, including designs, software, blueprints, ideas, documents, know-how, artistic expressions, recipes, branding, and the like. Nakamura made one of the first economy-wide investigations of intangible assets in 2000 [20]. He presented direct and indirect empirical evidence that US private firms invested at least $1 trillion annually in intangible assets in that year. This amount was nearly equal to the amount spent on plant and equipment. Firms also held a capital stock of intangibles with a market value of at least $5 trillion, representing a third of the amount of US corporate assets.

Another group—Carol Corrado, Charles Hulten, and Daniel Sichel, known as 'CHS' across their many studies—also began systematically to evaluate the extent and basis of intangible assets [21]. They estimated that spending on long-lasting knowledge capital—not just intangibles broadly—grew relative to other major components of aggregate demand during the 1990s. CHS was the first to show that by the turn of the millennium fixed US investment in intangibles was at least as large as business investment in traditional, tangible capital. Surveys of more than 5000 companies in 25 countries confirmed these trends and showed that most of these assets did not get reflected in financial statements. A large portion of this value was due to 'brands' and other market intangibles [22]. The total 'undisclosed' portion appeared to equal or exceed total reported assets. In 2009 the National Academies in the United States reported on their investigation into policy questions related to intangible assets [17], with much relevant information. The study contained an update by CHS confirming and extending their prior findings. In 2010 Nakamura also revisited his earlier analysis and found that intangible values had finally exceeded expenditures on plant and equipment, with intangible investments now being on the order of 8–10% of GDP annually in the United States [20].

In parallel, these groups and others began to decompose the intangible asset growth by country, sector, or asset type. The specific component of 'information' received a great deal of attention. Apte, Karmarkar, and Nath, in particular, conducted a couple of important studies during the 2000 decade [18, 23, 24]. They found that nearly two-thirds of recent US GDP was due to information or knowledge industry contributions, a percentage that had been growing over time [18]. They found that a secondary sector of information internal to firms constituted well over 40% of the information economy or some 28% of the entire economy. So the information activities that are internal to organizations represent a considerable part of the economy.

Today, intangibles now equal or exceed the value of tangible assets in advanced economies. The methodological and conceptual issues of how to explicitly account for information on a company's books are, of course, matters best left to economists and professional accountants. However, with the growing share of information related to intangible assets, this is a matter of great importance to national policy. For example, accounting for R&D efforts, one possible component of intangible assets, as an asset versus a cost, has been estimated to add on the order of 11% to US national GDP estimates [17].

The mere generation of information is not necessarily an asset. Some of the information has no value, and some indeed represent a net sunk cost. What we can say, however, is that valuable information that is created by the enterprise but remains unused or created anew means that what was an asset has now been turned into a cost[8]—sometimes a cost repeated many times over. Information that *is* used *is* an asset, intangible or not. The value of this information depends on its nature and use. We may value the information by *cost* (historical cost or what it costs to develop it), *market value* (what others will pay for it), or *utility* (what is its present value as benefits, broadly accounted for, accrue into the future). Traditionally the historical cost method has been applied to information. However, since information can both be sold and still retained by the organization, it may have both market value and utility value, with its total value being the sum or a portion thereof.[9]

Researchers estimated in the early 2000s that enterprises adequately use only 5–7% of existing information and the total value of information in enterprises is in the range of 10–33% of US GDP [18, 19]. Among all enterprise resources and assets, information is the least understood and the least managed. Managers are overlooking the value of their information.

More than a decade ago Moody and Walsh put forward a seminal paper on the seven 'laws' of information [25]. Unlike other assets, information has some unique characteristics that make understanding and valuing it more difficult, which leads to lower perceived importance. I have taken some liberty with the Moody and Walsh 'laws' to reflect my experience:

1. Information is (infinitely) shareable; it is not necessarily a depletable resource (though sharing may reduce proprietary advantage).
2. The benefit of information often increases with use, such as through the learning curve.[10]
3. The value of information increases with accuracy.
4. The value of information increases in combination.[11]
5. The value of information is situational and perishable, with varying shelf life.
6. More is not necessarily better; the question is one of relevancy.
7. Information builds upon prior information, the combinations of which often stimulate new insights.[12]

[8] That is because time and effort are required to generate unique information.

[9] Of course, information can also have a multiplicative effect, especially in those areas Mokyr calls prescriptive knowledge; but, that is not applicable to this specific point, since we are talking about *reuse*.

[10] A corollary is that it is an asset only if it provides future economic value, another is that awareness of the information's existence is an essential requirement in order to obtain this value, and a third corollary is that information requires an understanding of where it fits and how to take advantage of it.

[11] Network effects are particularly important here; see discussion of the Viking algorithm in Chap. 10.

[12] This propagation results from summations, analysis, unique combinations, and other ways that basic datum get recombined into new information. Thus, while the first law noted that information cannot be consumed (or depleted) by virtue of its use, we can also say that information tends to reproduce and expand itself via use and inspection.

Lost Value in Information

Information—more specifically, knowledge management—has bedeviled enterprises for decades. As the prior section indicates, information has enormous importance to most organizations and the overall economy. Why do these disheartening statistics keep cropping up concerning information management?

- 65% of data integration or KM projects 'fail.'
- A typical organization only uses 5–7% of the information it already has on hand.
- 20–25% of a knowledge worker's time is spent trying to find information.
- We waste 25% of all document creation costs.
- IT now consumes 4% of all enterprise expenditures and employs 6% of enterprise workers.

These are statistics I have encountered, or about which I have researched and written [1]. As rough figures or averages, they say nothing about what an individual enterprise or project may experience—there are, after all, good managers out there—but they do provide a pretty fair metric for the typical experience.

About a decade ago I began a series of analyses looking at how we spend money on preparing documents within US companies, and how much of that investment was being wasted or not reused [19]. The total benefit from improved information access and use to the US economy may be on the order of 8% of GDP. For the 1000 largest US firms, benefits from these improvements can approach nearly $250 million annually per firm (2002 basis). About three-quarters of these benefits arise from *not* re-creating the intellectual capital already invested in prior document creation. About one-quarter of the benefits are due to reduced regulatory noncompliance or paperwork, or better competitiveness in obtaining solicited grants and contracts. Finding and reusing information for compliance purposes as well as avoiding duplicate content creation are areas amenable to waste reductions. Note that new initiatives, as discussed in the next Chap. 4, are *not* included in this analysis.

This overall value of document use and creation is in line with the analyses of intangible and information assets noted above, and which arose from entirely different analytical bases and data. This triangulation brings some confidence that the estimates are approximately accurate. In any case, the potential benefits to the better use of existing information assets likely exceed what most managers currently believe; otherwise we would see better performance trends.

IT departments seem to have particular difficulty with information and knowledge management projects. Transaction and relational data systems require a different set of skills and viewpoint than for information sharing and open nature of knowledge.[13] Relational database systems, which embody a closed-world design, work well for environments where the information domain is known and bounded, but do not work well with knowledge and changing information. Moreover, the schema that governs closed-world designs is brittle and hard to improve and manage. It is this fact that has often led to IT delays and frustrations. Re-architecting or adding new schema views to an existing closed-world system, as knowledge systems demand, can be fiendishly difficult. Here are some other areas of frustration with IT regarding information and knowledge management:

[13] See Chap. 8.

Table 3.1 Enterprise IT weaknesses in relation to KM

IT problems for KM	Comments
Inflexible reports	• Reports are rarely 'self-service' • New requests need to be placed in a queue • 90% of stored report templates are never used • Unlimited "slicing and dicing" not available
Inflexible analysis	• The analysis is rarely 'self-service' • New requests need to be placed in a queue • Many requests not accepted due to schema rigidities, cascading changes needed • Analysis options are 'pre-canned,' inflexible
Schema bottlenecks	• Brittleness of relational data model and typical star schema • Crossing across schema or databases difficult • Load and re-indexing cycles can limit access, impose expensive back-end requirements • Cannot (often) accommodate new data, structures
ETL bottlenecks	• Getting data into the system needs to be placed in queue • New external data requires extract, transform, and load (ETL) routines to be written • Schedule and update cycles can be a mismatch to access needs
Reliance on intermediaries	• All problems above work through intermediaries • There is a disconnect between those with need and decision-makers and those who implement the solutions • Inherent issues in communicating requirements to implementers • Related time delays to implementation exacerbate the communication of requirements
Specialized expertise required	• Expertise and skill sets needed to implement solutions different from those of the knowledge consumer • Inherent issues in communicating requirements to implementers • High costs for attracting necessary expertise • Expertise is inherently an overhead function
Slow response time	• All problems above lead to delays, slow response • Timely communications, analysis, decisions suffer • Delays mean knowledge management is not an active 'contact sport,' becomes mired and unresponsive • Some needs are just not requested because of these problems
Dependence on external apps	• New apps need to be identified, procured • Design and configuration of apps require external expertise, programming skills • Multiple sourcing of apps leads to frequent incompatibilities, high costs for integration, poor interoperability
Unmet needs	• Many KM needs are not requested • By the time responses are forthcoming, needs and imperatives have moved on • Communications, analysis, and decisions become hassles • The 'contact sport' of active discovery and learning is unmet
High opportunity costs	• Many KM insights are not discovered • Delays and frustration add to costs, friction, and inefficiencies • No way to know the opportunity costs of what is not learned—but, surely is high
High failure rates	• The net impact of all of the problems above is to lead to high failure rates (~60% to 70%) and unacceptable costs • Reliance on IT for KM has generally failed

The problems raised in Table 3.1 show that losses in information and its poor organization and handling lead to a decline in business value. Removing unnecessary mediation roles by IT and placing the knowledge management function directly into the hands of the knowledge worker present a huge opportunity to recapture that lost value. Much of what I discuss throughout the remainder of this book is geared directly to this aim.

The Information Enterprise

One can probably clock the start of enterprise information technology (IT) to the first use of mainframe computers in the early 1950s, or nearly 70 years ago.[14] The earliest mainframes were massive and expensive machines that required their own specially air-conditioned rooms because of the heat they generated. The first use of 'information technology' as a term occurred in a Harvard Business Review article from 1958.[15] Architectures progressed from mainframes to minis and then personal computers with networks, leading to today's dominance of the Internet. Relational database designs won out for the enterprise in the 1970s and 1980s, continuing into today's dominance, but with the recent adoption of graph and NoSQL data stores.

The apogee for enterprise software and apps occurred in the 1990s. Whole classes of new applications (most denoted by three-letter acronyms) such as enterprise resource planning (ERP), business intelligence (BI), customer relationship management (CRM), enterprise information systems (EIS), and the like came to the fore. These systems also began as proprietary software, which resulted in the 'stovepiping' or creating of information silos. In reaction and with market acceptance, vendors such as SAP arose to provide comprehensive, enterprise-wide solutions, though often at high cost, with disappointing results not uncommon. Software revenues as a percent of IT vendor revenues peaked in about the mid-1990s. The plateau for IT expenditures as a portion of GDP appears to have occurred somewhat around 2000.

More significantly, the 1990s also saw the innovation of the World Wide Web with its basis in hypertext links on the Internet. Greatly facilitated by the Mosaic Web browser, the basis of the Netscape browser (ultimately Firefox), and the HTML markup language and HTTP transport protocol, millions began experiencing the benefit of creating Web pages and interconnecting. By the mid-1990s, enterprises were on the Web in force, bringing with them larger content volumes, dynamic databases, and enterprise portals. The ability for anyone to become a publisher led to a focus and attention on the new medium that led to still further innovations in

[14] Punch cards were an early form of automation, and extend back to the 1700s and then famously to the Jacquard loom. Though calculating machines and others extend back to Charles Babbage and more relevant efforts during World War II, the first UNIVAC was delivered to the US Census Bureau in 1951, and the first IBM to the US Defense Department in 1953. Many installations followed thereafter. See, for example, Lectures in the History of Computing: Mainframes.

[15] As provided by "information technology" (subscription required), Oxford English Dictionary (2 ed.), Oxford University Press, 1989, http://dictionary.oed.com/, retrieved 12 January 2011.

e-commerce and online advertising, creating entirely new categories of business. New languages and uses of Web pages and applications emerged, creating a convergence of design, media, content, and interactivity. Venture capital and new startups with valuations independent of revenues led to a frenzy of hype and eventually the dot-com crash of 2000. The growth companies of the past 15 years have not had the traditional focus on enterprises but the use and development of the Web. From search (Google) to social interactions (Facebook and Twitter) to media and video (Flickr, YouTube) and information (Wikipedia), the engines of growth have shifted away from the enterprise.

Meanwhile, the challenges of data integration and interoperability that were such a keen focus going back to initial enterprise computerization remain. Now, however, these challenges are even higher, as we see that images, documents (unstructured data), and Web pages, markup, and metadata (semi-structured data) become first-class information citizens. What was a challenge in integrating structured data in the 1980s and 1990s via data warehousing remains daunting for the enterprise today in the face of unprecedented scale and scope. Services have drifted to the largest IT vendors, and open source is now a primary source of innovation and challenge. We have climbed the data federation pyramid sufficient to overcome most obstacles of hardware, protocols, and data formats, but are stuck at the levels of semantics and trust (provenance).

Roughly in 1997 or so, the number of public enterprise software vendors peaked as did venture funding [26]. There was an uptick in preparing for Y2K and a significant downtick due to the dot-com bubble, and then later the financial crisis. However, change is coming about from the shift of expenditures from license and maintenance fees to services. Some software vendors began to see revenues from services overcome that from licensing in the 1990s. By the early 2000s, this was true for the enterprise software sector as a whole [26]. Today, service revenues account for 70% or so of aggregate sector revenues. Combined with the emergence of open source and other alternatives such as software as a service (SaaS) and cloud computing in general, I think it is fair to say that the era of proprietary software with exceedingly high margins from monopoly rents has come to an end.[16] According to Gartner, in the United States, more than 70% of IT expenditures are devoted to running existing systems, with only about 11% of budgets dedicated to innovation. This relative lack of support for innovation and high percentages for running existing systems have held true for a decade. Meanwhile, IT's contribution to US productivity has been declining since 2001.[17]

Arguably the emphasis on consumer and Internet technologies means that is where the best developers gravitate. Developing apps for smartphones or working at one of the cool Internet companies or joining a passionate community of open-source developers is now attracting the best developers. Open-source and Web-

[16] While we may occasionally see some vendors successfully buck this trend, I suspect these will only occur for established vendors with established platform advantages or for isolated applications where the innovating vendors have a significant first-mover advantage.

[17] PricewaterhouseCoopers, "Why Isn't IT Spending Creating More Value?" Available: http://www.pwc.com/en_US/us/increasing-it-effectiveness/assets/it_spending_creating_value.pdf.

based systems also lead to faster development cycles. The very best developers are often the founders of the next-generation startups and Web and software companies, as startup costs plunge.[18]

The shift in innovation away from the enterprise has been structural, not cyclical. That means that very fundamental forces are at work to cause this change in innovation focus. Every knowledge-oriented organization must learn to support and nurture its information enterprise. These structural shifts need to affect priorities, mindsets, budgets, and staffing. In an environment of cost pressures and the need for quantifiable results, we need to make pragmatic choices, Peirce's dominant message. The rest of this book, in part, talks about machine learning and various other aspects of artificial intelligence. These are all exciting topics, the shiny new thing. Still, the pragmatic viewpoint insists that in the process of making expenditures for these purposes, we should include in our design more fundamental and, perhaps, useful applications in information sharing and knowledge management. We will try to weave this practical viewpoint through our narrative as well.

Impediments to Information Sharing

Our survey of the current situation suggests a few things. Better use of information will be a significant factor in future economic growth. Growth is vital to wealth creation. Leveraging our existing information assets through reuse and connections is one immediate source of growth, with surprisingly large upsides. Innovations using artificial intelligence will continue the virtuous cycle to help support healthy growth. Individuals and enterprises need to grasp the challenge of knowledge management and need to place those functions into the hands of the knowledge worker. In an information-driven economy, education and access to information and knowledge management resources are essential foundations.

Cultural Factors

Since the widespread adoption of the Internet, which marked a passage beyond hardware, protocols, and formats as limits to interoperate data, the main impediments to information sharing have become cultural. Awareness of the importance of information as an asset has been lacking. Knowing how to interoperate across information stores is not a sought skill. Rewards are geared to information hoarding and gatekeeping over openness and sharing, which are rarely a formal part of performance evaluations. A lack of enlightened understanding about the importance of information leads to a lack of vision and an absence of a knowledge strategy. Since many bosses don't know what to do, they make second-rate decisions

[18] Open-source and Internet-based systems have reduced the capital necessary for a new startup by an order of magnitude or so over the past decade. It is now possible to get a new startup up and running for tens to hundreds of thousands of dollars, as opposed to the millions of years past.

about information and knowledge assets. Budget and operational fiefdoms continue a climate that guarantees inefficiencies, with a high opportunity cost, in how to manage information and knowledge. This book tries to provide a consistent viewpoint and logic for why information sharing is so important to enterprises. A change in perspective is required to unleash new growth, one that focuses on management and mindset.

The fact that an overlay of semantic technologies is required, as I discuss in Chap. 5, is good news from a cultural standpoint. A critical aspect of shared knowledge schema within an enterprise is the need for relevant stakeholders to have a role in bounding and defining the terminology of the domain. The first dictum of effective messages and reasoning is to communicate with a shared grammar and semantics. The absolute wrong strategy is to try to find or impose 'official' terminologies. Rather, we need to capture language as we use it daily in our tasks and find ways to relate these uses to a shared knowledge graph. Relevant stakeholders need to interact to document current terms and tasks, using the language of their daily work. With sufficient top management commitment, not often easy, such first steps can help set a new cultural tone for sharing. Given the potential incremental nature of deploying semantic technologies, early efforts should focus on prototypes at the level of workgroups or departments. The open nature of semantic technologies means we can readily expand the vocabularies and relate them to what already exists as we bring new stakeholders into the process. One can start small and grow as results become evident. I have called this the 'pay as you benefit' strategy.[19]

No fundamental technical roadblocks are preventing any enterprise from moving to a vision of shared information, providing useful knowledge support. Despite decades of trying, enterprises have still not broken down their data stovepipes. Rather, they continue to proliferate. In the process, the enterprise has failed to unlock 80% of its information value in documents (unstructured data) and has continuously wasted money in unneeded duplication and lost opportunities.

Tooling and Technology

The semantic technologies recommended in this book are open standards with years of implementation experience; still, their state of tooling is weak. Knowledge graph (ontology) editors and development environments exist, but all of the tools to create, edit, manage, update, delete, map, and validate could be improved. Each operation would benefit from being streamlined from the standpoint of the user, starting with subject matter experts (SMEs). Rather than comprehensive integrated development environments (IDEs), many of these functions are better separated out as options embedded within current workflows. Such a function, say, might be to add a new synonym for a concept in the knowledge base when encountered in a relevant document. Many individual functions would benefit from being split apart and

[19] See Chap. 13.

incorporated in a distributed way across multiple steps in information handling, from creating to using, validating, updating, or archiving.

Another weak area is in user permissions and authorizations. Optimally, a single sign-on should be sufficient to grant access or not to various datasets, records, and applications. Methods and protocols exist, mostly IP based, with some working installations from which to draw lessons, but more robust and secure options are needed, likely using third-party applications. When relying on Internet protocols, we need to manage unauthorized access and hacking.

Perspectives and Priorities

I think it is fair to say, in general, that we do not have a broad and informed view on the value of information. We do not know what information we have and cannot find it, and we waste much time looking for it. We misallocate resources for generating, capturing, and storing information because we do not understand its value and potential and don't know what we already have. We do not manage the use of information or its reuse. We do a lousy job of using information to bridge communication differences across our stakeholders. We inadequately leverage what information we have and often miss valuable insights. What we have we do not connect. We do not know how to turn our information into knowledge.

Fundamentally, because we do not understand information in our bones as central to the well-being of our enterprises, we continue to view the generation of information as a 'cost' and not an 'asset.' Perhaps, akin to the perspective of Thirdness in Peirce's universal categories, which we discuss in Chap. 10, we need to bring new perspectives to our understanding and appreciation of information.

Peirce defended and is known as a realist. Within that realism and subject to his pragmatism, I believe that he can also be called an idealist. Real and practical ways exist to achieve meaningful visions of information sharing, which can release hidden value within any information enterprise. This foundation can then be extended with knowledge bases and artificial intelligence to mine further still the value contained in that information.

References

1. A. Maddison, *The World Economy: Historical Statistics* (OECD Development Centre, Paris, 2003)
2. W.D. Nordhaus, Do real-output and real-wage measures capture reality? The history of lighting suggests not, in *The Economics of New Goods*, ed. by T.F. Bresnahan, R.J. Gordon (University of Chicago Press, Chicago, 1996), pp. 27–70
3. E.L. Eisenstein, *The Printing Press as an Agent of Change* (Cambridge University Press, Cambridge, 1980)
4. J. Mokyr, *The Gifts of Athena: Historical Origins of the Knowledge Economy* (Princeton University Press, Princeton, 2002)

5. J.A. Schumpeter, *The Theory of Economic Development: An Inquiry Into Profits, Capital, Credit, Interest, and the Business Cycle* (Harvard University Press, Cambridge, MA, 1955)
6. R. Solow, *Economist's View: Robert Solow on Joseph Schumpeter*. http://economistsview.typepad.com/economistsview/2007/05/robert_solow_on.html
7. R.M. Solow, Technical change and the aggregate production function. Rev. Econ. Stat. **39**, 312–320 (1957)
8. S. Fabricant, *Economic Progress and Economic Change, a part of the 34th annual report of the National Bureau of Economic Research*, New York (1954)
9. F. Machlup, *The Production and Distribution of Knowledge in the United States* (Princeton University Press, Princeton, 1962)
10. D.B. Lawrence, *The Economic Value of Information* (Springer-Verlag, New York, 1999)
11. C.I. Jones, P.M. Romer, The new Kaldor facts: Ideas, institutions, population, and human capital. Am. Econ. J. Macroecon. **2**, 224–245 (2010)
12. K. Arrow, Economic Welfare and the Allocation of Resources for Invention, *The Rate and Direction of Inventive Activity: Economic and Social Factors*, Universities-National Bureau, Committee for, Economic Research, Committee on Economic Growth of the Social, and Science Research Council, eds. (Princeton University Press, Princeton, 1962), pp. 609–626.
13. D. Warsh, *Knowledge and the Wealth of Nations: A Story of Economic Discovery* (WW Norton & Company, New York, 2007)
14. P.M. Romer, Endogenous technological change. J. Political Econ. **98**, S71–S102 (1990)
15. Rosenthal, S., Russell, M., Samuels, J. D., Strassner, E. H., Usher, L., et al., Integrated industry-level production account for the United States: Intellectual property products and the 2007 NAICS, in *3rd World KLEMS Conference (May), Tokyo, Japan* (2014)
16. D. Jorgenson, M. Ho, J. Samuels, Long-term estimates of US productivity and growth, in *Third World KLEMS Conference* (2014), pp. 19–20
17. C. Mackie, *Intangible Assets: Measuring and Enhancing Their Contribution to Corporate Value and Economic Growth* (The Board on Science, Technology, and Economic Policy (STEP) Committee on National Statistics (CNSTAT), 2009)
18. U.M. Apte, H.K. Nath, *Size, Structure and Growth of the US Information Economy* (UCLA Anderson School of Management on Business and Information Technologies, 2004)
19. M.K. Bergman, Untapped assets: The $3 trillion value of U.S. Enterprise Documents, in *AI3:::Adaptive Information* (July 2005)
20. L.I. Nakamura, Intangible assets and national income accounting. Rev. Income Wealth, 56, 135–155 (2010)
21. C.A. Corrado, C.R. Hulten, How do you measure a 'technological revolution'? Am. Econ. Rev. **100**, 99–104 (2010)
22. K.P. Jarboe, R. Furrow, *Intangible Asset Monetization: The Promise and the Reality* (Athena Alliance, Washington, DC, 2008)
23. U.M. Apte, U.S. Karmarkar, H.K. Nath, Information services in the US economy: Value, jobs and management implications. Calif. Manage. Rev. **50**, 12–30 (2008)
24. U. Apte, The U.S. information economy: Value, employment, industry structure, and trade. Foundations Trends® Technol. Inf. Oper. Manag. **6**, 1–87 (2012)
25. D. Moody, P. Walsh, Measuring the value of information: An asset valuation approach, in *Seventh European Conference on Information Systems (ECIS'99)* (Copenhagen Business School, Frederiksberg, 1999)
26. M.A. Cusumano, The changing software business: Moving from products to services. Computer **41**, 20–27 (2008)
27. M.U. Porat, *The Information Economy: Definition and Measurement* (Office of Telecommunications, Department of Commerce, 1977)

Chapter 4
The Opportunity

Charles S. Peirce, the intellectual founder of *pragmatism*, a uniquely American contribution to philosophy, advocated that we obtain knowledge by balancing research effort with a likelihood of results. To do so, we should first consider all of the 'practical effects' posed by alternatives. We select what deserves more detailed attention using a mode of logical inference he called *abduction*. Wherever we have doubt, we should be open to and pursue the path of inquiry to unveil further potential 'practical effects,' enhancing our knowledge. In this way, we continually modify what we believe about the world, and therefore how we act within it.

Today, we have the ability and information to query nearly the entire storehouse of accumulated human knowledge. By combining general knowledge storehouses with representations of our organizations and domains, we have paths of inquiry leveraging computers and machine learning to test what we think we know, and to discover previously hidden anomalies or falsities to propel our knowledge further. As we have seen with earlier breakpoints in humanity's abilities to share and process information, this quest for new truth will bring significant financial benefit across the full spectrum of economic actors, from individuals and small groups to enterprises and governments.[1]

In this chapter, I discuss these opportunities under three broad tents. The first tent, more of a foundation, embraces general applications in *knowledge management* (KM). This broad tent may not be the motivating interest, but it does reside on the path to other capabilities, and it addresses important needs in their own right. The second of the tents, more of a process, are the approaches and applications that

[1] Some material in this chapter was drawn from the author's prior articles at the *AI3:::Adaptive Information* blog: "Search and the '25% Solution'" (Sep 2005); "Climbing the Data Federation Pyramid" (May 2006); "structWSF: A Framework for Data Mixing" (Jun 2009); "The Open World Assumption: Elephant in the Room" (Dec 2009)' "Changing IT for Good" (Mar 2010); "Ontology-Driven Apps Using Generic Applications" (Mar 2011); "Democratizing Information with Semantics" (Apr 2011); "Leveraging Intangible Assets Using Semantic Technologies" (May 2011); "Knowledge Supervision as a Grounding for Machine Learning" (Jun 2015); "Why the Resurgence in AI?" (Jan 2016).

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8_4

enable *data interoperability*. The techniques of data interoperability are essential for ingesting relevant information leading to knowledge and for unleashing the value of existing information assets across the organization. The third tent, more an expression of potential, is *knowledge-based artificial intelligence*. Via KBAI we can cost-effectively create labeled training sets (supervised learning) and training corpora (unsupervised) for machine learning to support a variety of tasks from entity and relation recognition and extraction to categorization, natural language understanding, sentiment analysis, and much more. Like the lizard eating its tail, we can also apply KBAI to our initial knowledge bases and knowledge graphs that drive these applications, producing a virtuous cycle of knowledge expansions and better learning accuracy.

KM and a Spectrum of Applications

Many of the problems of wasted information assets and lack of connections, some described in the prior chapter, and most with real economic costs, can be ascribed to a failure of *knowledge management.* KM is the practice of creating, sharing, finding, annotating, connecting, and extending information and knowledge for a given domain [1]. The practice includes applications and management platforms; shared workflows, vocabularies, and organizational schema; training and best practices; and roles for practitioners. The practice is managed and possibly encouraged by rewards and incentives. Software is an essential component of knowledge management, but woefully inadequate alone to accomplish it.

Some Premises

The nature of knowledge helps set some parameters for what a knowledge management system should encompass. First, knowledge is 'open' and needs an architecture and design that embrace this openness. This consideration has logical and epistemic importance that gets further treatment in Chap. 9. Second, knowledge is ultimately a community reality, since knowledge is what we believe and upon which we act. Because our means of communicating within the community is via symbols, we need methods for defining, clarifying, and reconciling the meanings of those symbols, such that we are effectively communicating within the community. This imperative means that we should look to semantic technologies as our representation and messaging frameworks; Chap. 5 covers this topic. Moreover, third, we need to design our knowledge management systems to get maximum pragmatic leverage from what already exists and what we can support with such a system. We need to design our systems for knowledge *uses*, with management a contributing component to that.

Potential Applications

KM includes such applications as business intelligence, data warehousing, data integration and federation, enterprise information integration and management, competitive intelligence, workflow systems, knowledge representation, and so forth. Information management is a bit broader category and adds such functions as document management, data management, enterprise content management, enterprise or controlled vocabularies, systems analysis, information standards, and information asset management to the functions of KM. Knowledge management also importantly includes pruning (deleting) dated, inaccurate, or otherwise wasteful information. An absolute essential for an effective KM system is bridging vocabulary, concept, and representation differences.

These are all important and legitimate knowledge management functions, but we often pursue them in isolation or under different databases, vocabularies, or conceptual approaches. In point, one could reasonably argue that much of the challenge that has faced KM has been a lack of coherence or a shared conceptual grounding to the efforts. The decades-long literature into KM supports such a view of fragmentation.

A broadly useful KM framework should support a minimum of four application areas:

- First, some form of governing conceptual and terminological schema is required by which to reference and ground disparate information sources. In KM systems based on semantic technologies, this schema takes the form of a *knowledge graph* (or ontology).
- Second, given that on average about 80% of an organization's information base resides in documents, *natural language processing* should be an integral part of the mix. NLP uses computers to extract meaningful information from natural language input or produce natural language output. NLP is one method for assigning structured data characterizations to text content; without NLP, all such assignments are manual, which does not scale.[2]
- Third, as part of these NLP capabilities, we need various extractors. *Entity recognition*, the means for identifying specific *entities* in text, is the first among equals here. Concept and relation extractors may supplement that. Extraction methods involve parsing and tokenization, and then generally the application of one or more information extraction techniques or algorithms.
- Fourth, *tagging* is a needed adjunct to extraction. The *tag* is a keyword or term we assign to a piece of information (*e.g.*, a picture, article, portion of text, or video clip). Tags describe the item and enable keyword-based classification of

[2] Even with natural language processing (NLP), we do not see fully automated systems. We need some manual inspection to remove errors in NLP processing and to ensure quality commitments to the *knowledge base*. Such managed NLP systems are best characterized as "semiautomatic" and place human editors into critical parts of the workflow.

the information.[3] The resulting representation is a form of *semi-structured data*. Like extractors, we may use tags for entities, concepts, attributes, or relations. When a knowledge graph is employed, we recommend *ontology-based information extraction* (*OBIE*), which is the use of an ontology to inform this tagging process.

These are the essential functions required to 'ingest' new content and provide a shared vocabulary via the schema for placing content onto a common footing. This shared representation is the basis for a series of specific KM format conversions from multiple external sources, and in functions such as search, retrieval, analysis, and visualization. As we add multiple input sources to the system, we assign *metadata* by source (such as title, provenance, workflow dates, formats) to the content, providing still additional means for searching, filtering, and aggregating the content.

If the KM system is also a precursor to more knowledge- and intelligence-oriented tasks, we advise including reasoners and mappers. *Reasoning* is one of many logical tests using *inference* rules as commonly specified using an ontology language, and often a description logic. Many reasoners use first-order predicate logic to perform reasoning. Inference commonly proceeds by forward chaining or backward chaining (see Chap. 8). *Mapping* connects *objects* in two different sources to one another, using a specific *property* to define the relation. A linkage is a subset of possible mappings where the connections may be traced and followed. Mappings are the means by which we bring in multiple information sources leading to a federation of sources, so that we may use, analyze, or reason over all of them. It is the central task of *data federation*. Pairwise mappings result in a combinatorial explosion of connections as the number of sources increases. A hub-and-spoke design is the only practical architecture to overcome this problem since it scales linearly, with a reference set of concepts, such as KBpedia, providing the hub.

A Minimal Scaffolding

We could stop with this initial configuration and merely deploy the knowledge management system for generic KM tasks. This basis, the minimal scaffolding, is sufficient to address the lost opportunities and waste described in the prior chapter. However, we have our sights set higher than recovering lost opportunities.

The general development path this book recommends is to first address these lost opportunities, perhaps on a small or departmental basis (see 'pay as you benefit' in Chap. 13). As we gain confidence and climb the learning curve, it is then appropriate to bridge out to encompass more departments and to begin deploying machine learning to develop bespoke extractors and classifiers, tuned for the relevant nature of your growing knowledge base.

[3] Tagged information is one of the main sources of *semi-structured data*; see Chap. 5.

We introduce the role of KBpedia here as a lead-in to later chapters. KBpedia is an open-source knowledge graph with maps to leading knowledge bases. Parts III and IV cover design and deployment topics in detail. Appendix B is a broad overview of KBpedia. Appendix C discusses the features available in KBpedia for machine learning.

Data Interoperability

Data integration is the bringing together of data from heterogeneous and often physically distributed data sources into a single, coherent view. Sometimes this is the result of searching across multiple sources, in which case it is called federated search. However, it is not limited to search. Data integration is a crucial concept in business intelligence and data warehousing and a driver behind master data management (MDM). Data integration first became a research emphasis within the biology and computer science communities in the 1980s [2, 3]. At that time, extreme diversity in physical hardware, operating systems, databases, software, and immature networking protocols hampered the sharing of data. Data *interoperability* extends beyond integration to add unified views for analysis and reasoning across its sources.

By its nature, data integration means that we combine data across two or more datasets. Such integration brings to light the myriad aspects of semantic heterogeneities, precisely the kinds of issues why we use semantic technologies. However, resolving semantic differences, which we probe in detail in Chap. 5, cannot be fulfilled by semantic technologies alone. While semantics can address the basis of differences in meaning and context, resolution of those differences or deciding between differing interpretations (that is, ambiguity) also requires many of the tools of artificial intelligence or natural language processing (NLP). By decomposing this content into its various sources of semantic heterogeneities—as well as the work required to provide for such functions as search, disambiguation, mapping, and transformations—we can begin to understand how all of these components can work together to help achieve data interoperability.

The Data Federation Pyramid

It is easy to forget just how far data federation has progressed in the last four or five decades. Before the introduction of the IBM personal computer in 1981, the hardware landscape was diverse and fragmented. There were mainframes from weird 36-bit Data General systems to DEC PDP minicomputers to the PCs themselves. Even on PCs, there were multiple operating systems, and many then claimed that CP/M was ascendant, let alone the upstart MS-DOS or the gorilla threat of IBM's

OS/2 (in development). Hardware differences were manifest, and operating systems were diverse; nothing worked with anything else.

'Data federation' at that time needed to first look at issues at the iron or silicon or OS level. Those problems were pretty daunting, though the clever folks behind Ethernet and Novell with PCs were about to show one route around the traffic jam. Client server and all of the 'N-tier' networking speak soon followed. It was an era of progress, but still, one of costly and proprietary answers to get devices to talk to one another. That is where the Internet, specifically the Web protocols of HTTP and HTML and the Mozilla (then commercially Netscape) browser, came in. Within 5 years (actually less) from 1994, the Internet took off like a rocket, doubling in size every 3–6 months.

In the early years of trying to find standards and conventions for representing semi-structured data (though not yet called that), the primary emphasis was on data representation and transfer protocols. In the financial realm, one standard dating from the late 1970s was electronic data interchange (EDI). In information and library science, the MARC communications format for sharing catalog metadata arose in the 1960s and remains well used in many countries today. In science, there were tens of exchange formats proposed with varying degrees of acceptance. Notable examples are the abstract syntax notation (ASN.1), TeX (a typesetting system created by Donald Knuth and its variants such as LaTeX), hierarchical data format (HDF), CDF (common data format), and the like, as well as commercial formats such as PostScript, PDF (portable document format), and RTF (rich text format). One of these formats was the 'standard generalized markup language' (SGML), first published in 1986. SGML was flexible enough to represent either formatting or data exchange. However, with its flexibility came complexity. Only when its two simpler progenies arose, namely HTML (HyperText Markup Language) for describing Web pages and, later, XML (eXtensible Markup Language) for data exchange, did variants of the SGML form emerge as widely used common standards. JSON has also now joined this group as a leading data representation. The Internet and its TCP/IP and Web HTTP protocols and XML standards, in particular, have been major contributors to overcoming respective physical and syntactical and data exchange heterogeneities.

I illustrate this historical progression over the decades, from the bottom-up, using the data federation pyramid in Fig. 4.1. Current progress and adoption place us, today, with a stack that has boundaries at the data (and knowledge) representation, semantics, and pragmatics layers in the Fig. 4.1 pyramid. We show only a part of the progress in Web standards. The TCP/IP and HTTP protocols were essential to overcome the network bottlenecks; we show OWL and RDF due to their importance to our story and their role in addressing semantics issues. We cannot integrate information as knowledge until we overcome the semantic challenges. Pragmatics covers understanding the kinds of practical needs and implications resulting from our integration of information. Trust refers to the ability to identify and track the provenance of our information to judge whether we use and integrate it or not. These upper layers of the stack are some of the unresolved issues we attend to over the rest of this book.

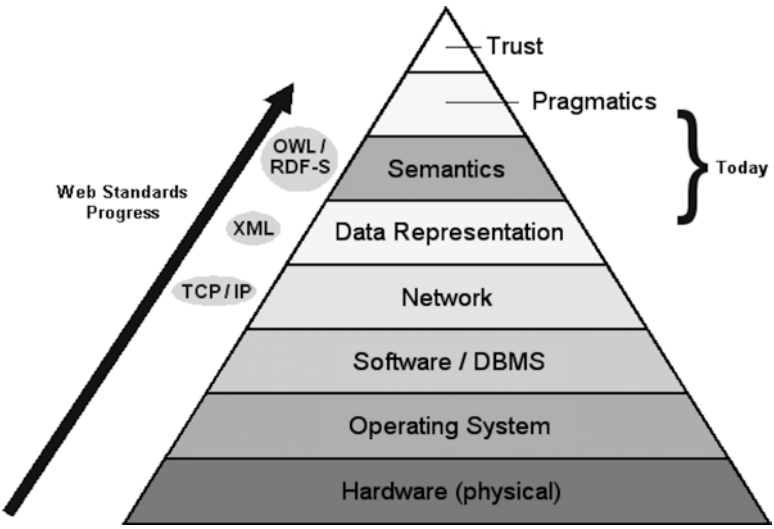


Fig. 4.1 Climbing the data federation pyramid

Benefits from Interoperability

Data interoperability should be one of the chief emphases of a knowledge management initiative because of these challenges, many of which have remained unsolved for decades:

- 80% of all available information is in text or documents (unstructured).
- 40% of standard IT project expenses are devoted to data integration in one form or another, due to the manual effort needed for data migration and mapping.
- Information volumes are now doubling in fewer than 2 years.
- Current information resides in individual stores, stovepiped from one another, with few or no connections to external knowledge sources.
- Other trends including smartphones and sensors are further accelerating information growth.
- Effective business intelligence requires the use of quality, integrated data.

The problem is creating a focus and then beginning to implement a data interoperability initiative. We know it promises to deliver some key, measurable benefits:

- *Efficiency*—Trillions of dollars are spent each year globally in the research, creation, reuse, publishing, storing, and browsing of information. Relevant information is hard to find, and sometimes we overlook useful but obscure information. The lack of reuse of prior good content because it is not discoverable is unconscionable given today's technologies.
- *Cost*—Missed information or lack of awareness of relevant information leads to increased time, increased direct costs (labor and material), and increased indirect costs to re-create it. Awareness, understanding, and reuse of existing information

would save millions or more for large firms annually if we could overcome these interoperability gaps.
- *Insight*—Drawing connections between previously unconnected things and enabling discovery are essential inputs to innovation, itself the overall driver of productivity (and, therefore, wealth) gains. The reinforcing leverage of interoperability resides in its ability to bring new understandings and insights.
- *Capture*—We benefit by capturing the many fields, data streams, APIs, mappings, DBs, datasets, Web content, on-the-fly discoveries, and device sensors available through the connectedness of the Web and the Internet of Things (IoT).

For decades, the vision of data interoperability has mostly remained unfulfilled. Though significant progress has occurred in climbing the data federation pyramid, only when one is at the very topmost layers can we achieve actual data interoperability. The semantics are an absolute threshold. A few practitioners and a few exemplary organizations have demonstrated the worth of semantic technologies to leverage this next step. Doing so adheres to Peirce's *pragmatic maxim*, the understanding of a topic or object by an apprehension of all of the practical consequences potentially arising from it.

Adopting knowledge graphs is a prerequisite for applying semantic technologies to the fullest. Once adopted with the graph mindset embraced, it is then straightforward to extend the scope of the graphs a bit to encompass labels for user interfaces and calls to small, external Web applications. We discuss these so-called *administrative ontologies* in Chap. 11. These practical techniques cost little to implement and can be a useful adjunct to standard knowledge maintenance.

Material progress on the data interoperability challenge will bring us one step closer to self-service information management. The benefits and flexibilities from doing so will extend from creating data and content to publishing and deploying it. The fact that any source—internal or external—or format—unstructured, semi-structured, and structured—can be brought together with semantic technologies is a qualitative boost over existing KM approaches. Further, since we represent all information via simple text formats, we can readily manipulate and manage that information with easy-to-understand tools and applications. Reliance on open standards and languages by semantic technologies also leads to greater use and availability of open-source systems. In short, self-service information management could be one of the great benefits from interoperability. These are the kinds of opportunities that will enable knowledge management to fulfill its vision.

A Design for Interoperating

Ultimately, since we express all of our content and information with human language, we need to start there to understand the first sources of semantic differences. Like the differences in human language, we also have differences in worldviews and experience. These differences are often conceptual and reveal distinctions in

real-world perspectives and experiences. From there, we encounter differences in our specific realms of expertise or concern, or the relevant domain(s) for our information and knowledge. Then, as we probe details, we give our observation and characterization data and values to specify and quantify our observations. The attributes of these data are subject to the same semantic vagaries as concepts. Attributes also pose challenges in how we measure and express units.

The current challenge is to resolve differences in meaning, or *semantics*, between disparate data sources. Your '*glad*' may be someone else's '*happy*' and you may organize the world into countries while others organize by regions or cultures. From the conceptual to actual data, then, we see differences in perspective, vocabularies, measures, and conventions. Only by systematically understanding these sources of heterogeneity—and then explicitly addressing them—can we begin to try to put different information on a common footing. Only by reconciling these differences can we begin to get data to interoperate. Some of these differences and heterogeneities are intrinsic to the nature of the data at hand. Some of these heterogeneities also arise from the basis and connections asserted between datasets, as misuse of the *sameAs* predicate showed in early linked data applications. Fortunately, in many areas, we are transitioning due to technological progress to overcome many of these sources of semantic heterogeneity. Semantic Web approaches where data items are assigned unique IRIs are another source of making integration easier. Moreover, whether all agree from a cultural aspect if it is right, we also see English becoming the *lingua franca* of research and data.

To bring about a basis for data interoperability, John Blossom argues the importance of Web approaches and architectures; incorporation of external data; leverage of Web applications; and use of open standards and APIs to avoid vendor lock-in [4]. Much, if not all of this, can be aided by open-source software. Open source is not indispensable: commercial products that embrace these approaches can also be compatible components across the stack. Further, we need to resolve semantic heterogeneities. Though there is only a single layer of the pyramid in Fig. 4.1, resolving semantics is a complicated task and may involve *structural conflicts* (such as naming, generalization, aggregation), *domain conflicts* (such as schemas or units), or *data conflicts* (such as synonyms or missing values). Researchers have identified nearly 40 distinct types of possible semantic heterogeneities, to which we delve into more detail in Chap. 5.

Semantic technologies give us the basis for understanding differences in meaning across sources, specifically geared to address real-world usage and context. Semantic tools are essential for providing common bases for relating structured data across various sources and contexts. These same semantic tools are also the basis by which we can determine what unstructured content 'means,' thus providing the structured data tags that also enable us to relate documents to conventional data sources (from databases, spreadsheets, tables, and the like). These semantic technologies are thus the key enablers for making information—unstructured, semi-structured, and structured—understandable to both humans and machines across sources. Such understandings are then the basis for powering the artificial intelligence applications involving human language.

An initial embrace of semantic technologies for knowledge management often naturally leads to adopting knowledge graphs. These ontologies provide a means to define and describe these different worldviews. Referentially integral languages such as RDF (Resource Description Framework) and its schema implementation (RDF-S) or the Web ontological description language (OWL) is a leading standard among other emerging ones for machine-readable means to communicate the semantics of data. You can read more about the languages of these semantic technologies in Chap. 8.

Adoption of semantic technologies does not necessarily mean open data nor open source (though they are suitable for these purposes with many open-source tools available). We can apply the techniques equivalently to internal, closed, proprietary data and structures. We can use these techniques as a basis for bringing external information into the enterprise. The use of 'open' here refers to the critical use of the open-world assumption (Chap. 9). Moreover, the design practices we recommend here do not require replacing current systems and assets; they can be applied equally to public or proprietary information; and, they can be tested and deployed incrementally at low risk and cost. The very foundations of our recommended practice encourage a learn-as-you-go approach and active and agile adaptation. While embracing semantic technologies can lead to quite disruptive benefits and changes, we can do so as a layered initiative with minimal disruption. Incremental adoption is one of the most compelling aspects of semantic technologies.

Knowledge-Based Artificial Intelligence

Artificial intelligence (AI) is the use of computers to do or assist complex human tasks or reasoning. AI has many, broad subfields from pattern recognition to robotics, and sophisticated planning and optimizations. Knowledge-based artificial intelligence, or KBAI, is the use of large statistical or knowledge bases to inform feature selection for machine-based learning algorithms used in AI. Correctly expressed KBs can support creating positive and negative training sets, promote feature set generation and expression, and generate reference standards for testing AI learners and model parameters. The use of knowledge bases to train the features of AI algorithms improves the accuracy, recall, and precision of these methods. These improvements lead to better information queries, including for pattern recognition. Further, in a virtuous circle, KBAI techniques can also be applied to identify additional possible facts within the knowledge bases themselves, improving them further still for KBAI purposes. Lastly, we hope that better ways to represent knowledge (with richer feature sets) may help unlock some of the black-box aspects typical of neural nets and deep learning.

Knowledge-based artificial intelligence is not a new idea. Its roots extend back perhaps to one of the first AI applications, Dendral, more than a half-century ago in 1965. Edward Feigenbaum initiated Dendral, which became a 10-year effort to develop software to deduce the molecular structure of organic compounds using scientific instrument data. Dendral was the first expert system and set the outline for

knowledge-based systems, which are one or more computer programs that reason and use knowledge bases to solve complex problems. Indeed, it was in the area of expert systems that AI first came to the attention of most enterprises. Expert systems spawned the idea of knowledge engineers, whose role was to interview and codify the logic of the chosen experts. However, expert systems proved expensive to build and difficult to maintain and tune.

The specific identification of 'KBAI' was (to my knowledge) first made in a Carnegie-Mellon University report to DARPA in 1975 [5]. The source knowledge bases were broadly construed, including listings of hypotheses. The first known patent citing knowledge-based artificial intelligence is from 1992.[4] Within the next 10 years there were dedicated graduate-level course offerings on KBAI at many universities, including at least Indiana University, SUNY Buffalo, and Georgia Tech. In 2007, Bossé et al. devoted a chapter to KBAI in their book on information fusion, but still, at that time, the references were more generic [6]. However, by 2013, as a report by Hovy et al. indicates, collaborative, semi-structured information stores such as Wikipedia were assuming a prominent position in AI efforts [7]. It has been the combination of KB + AI that has led to the notable AI breakthroughs for knowledge purposes of the past, say, decade. It is in this combination that we gain the seeds for sowing AI benefits in other areas, from tagging and disambiguation to the complete integration of text with conventional data systems. Further, the structure of all of these systems can be made inherently multilingual, meaning that context and interpretation across languages can be brought to our understanding of concepts.

KBAI is part of the AI branch that includes knowledge-based systems. Besides areas already mentioned, knowledge-based systems also include:

- Knowledge models—formalisms for knowledge representation and reasoning.
- Reasoning systems—software that generates conclusions from available knowledge using logical techniques such as deduction and induction.

As the influence of expert systems waned, another branch emerged, that of knowledge-based engineering and its support for CAD- and CASE-type systems. Still, we can charitably describe the overall penetration to date of most knowledge-based systems as disappointing.

It is different today. Structured information and the means to query it now give us a powerful, virtuous circle whereby our knowledge bases can drive the feature selection of AI algorithms, while those very same algorithms can help find still more features and structure in our knowledge bases (see Fig. 4.2). Once we reach this threshold of feature generation, we now have a virtuous dynamo for knowledge discovery and management. We can use our AI techniques to refine and improve our knowledge bases (the top loop of Fig. 4.2), which then makes it easier to improve our AI algorithms and incorporate still further external information (the bottom loop). Effectively utilized KBAI (knowledge-based artificial intelligence) thus becomes a generator of new information and structure.

[4] See "Method for converting a programmable logic controller hardware configuration and corresponding control program for use on a first programmable logic controller to use on a second programmable logic controller," US Patent No. 5142469 A, August 25, 1992.

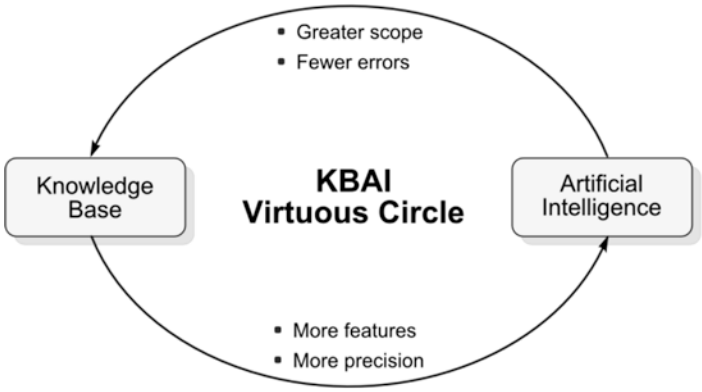


Fig. 4.2 Virtuous knowledge-based artificial intelligence

This virtuous circle has not been applied fully, seen mostly to date in the adding of new facts to Wikipedia or Wikidata. Importantly, we can apply these same basic techniques to the very infrastructural foundations of KBAI systems in such areas as data integration, mapping to new external structure and information, hypothesis testing, diagnostics and predictions, and myriad other uses to which researchers for decades hoped AI would contribute. The virtuous circle between knowledge bases and AIs does not require us to make leaps and bounds improvements in our core AI algorithms. Instead, we need to only stoke our existing AI engines with more structure and knowledge fuel to keep the engine churning.

KBAI has two primary knowledge sources: recognized knowledge bases, such as Wikipedia, and statistical corpora. Knowledge bases are coherently organized information with instance data for the concepts and relationships covered by the domain at hand, all accessible in some manner electronically. Knowledge bases can extend from the nearly global, such as Wikipedia, to particular topic-oriented ones, such as restaurant reviews or animal guides. Some electronic knowledge bases are designed explicitly to support digital consumption, with defined schema and standard data formats and, increasingly, APIs. Others may be electronically accessible and highly relevant, but the data is hard to consume and requires extraction and processing before use. Hundreds of knowledge bases are suitable for artificial intelligence, most of a restricted domain nature [6]. Chap. 11 is devoted to this topic.

The use and role of statistical corpora are harder to describe. Statistical corpora provide relationships or rankings to aid the processing of (mostly) textual information. Uses can range from entity extraction to machine language translation. Huge sources, such as search engine indexes or massive crawls of the Web, are most often the sources for these knowledge sets. The statistical corpora or databases have

a precise focus. While lists of text corpora and many other things may contribute to this category, the ones actually in commercial use are huge and designed for bespoke functionality. A good example is the Web 1T 5-gram dataset.[5] This dataset, contributed by Google for public use in 2006, contains English word n-grams and their observed frequency counts. N-grams capture word tokens that often coincide with one another, from single words to phrases. The length of the n-grams ranges from unigrams (single words) to five-grams. Google generated the database from approximately 1 trillion word tokens of text from publicly accessible Web pages.

Another example of statistical corpora is what Google's Translate uses. According to Franz Josef Och, a former lead manager at Google for its translation activities, a solid base for developing a usable language translation system for a new pair of languages should consist of a bilingual text corpus of more than a million words, plus two monolingual corpora each of more than a billion words [20]. Statistical frequencies of word associations form the basis of these reference sets. Google initially seeded its first language translators using multiple language texts from the United Nations [5]. If we add structure to statistical corpora, they may evolve to look more like a knowledge base. NELL, for example, contains a relatively flat listing of assertions extracted from the Web for various entities. NELL goes beyond frequency counts or relatedness but does not have the full structure of a general knowledge base like Wikipedia [8]. We thus can see that statistical corpora and knowledge bases reside on a continuum of structure, with no bright line to demark the two categories.

Created using both statistical techniques and results from machine learning, we are using these methods to extract massive datasets of entities, relationships, and facts from the Web. Some of these efforts, like NELL, or its academic cousins such as KnowItAll or Open IE (UWash), involve extractions from the open Web. Others, such as the terabyte (TB) n-gram listings from Google, are derived from Web-scale pages or Google books. Word, sentence, or graph vectors are other types. These examples are but a sampling of various datasets and corpora available. These various statistical datasets may be used directly for research on their own or may contribute to further bootstrapping of still further refined AI techniques. Similar datasets are aiding advertising placements and search term disambiguation. In some cases, while the full datasets may not be available, open APIs may be available for areas such as entity identification or tabular data.

The Web is the reason these sources—both statistical corpora and knowledge bases—have proliferated, so the dominant means of consuming them is via Web services with the information defined and linked to IRIs. The availability of electronically accessible knowledge bases, exemplified and stimulated by Wikipedia, has been the telling factor in recent artificial intelligence advances. For example, at least a thousand different papers cite using Wikipedia for various natural language

[5] The Web 1T 5-gram dataset is available from the Linguistic Data Corporation, University of Pennsylvania.

processing, artificial intelligence, or knowledge base purposes. These papers began to stream into conferences about 2005–2006, and have not abated since. In turn, researchers are applying the various techniques innovated for extracting more and more structure and information from Wikipedia to other semi-structured knowledge bases, resulting in a renaissance of knowledge-based processing for AI purposes. These knowledge bases are emerging as the information substrate under many recent computational advances, such as for virtual agents we command by voice. The agents use pattern recognition at the front and back ends of the workflow based on statistical datasets derived from phonemes and text. The agents apply natural language processing, as informed by knowledge bases and represented by semantic technologies, to the text sandwiched between these bookends to conduct question understanding and answer formulation.

This remarkable chain of processing is now almost taken for granted, though its commercial use in virtual agents is fewer than 10 years old. For different purposes with different workflows, we see useful question answering and diagnosis with systems like IBM's Watson [9] and structured search results from Google's Knowledge Graph [10]. Try posing some questions to Wolfram Alpha and then stand back and be impressed with the data visualizations. Behind the scenes, pattern recognition from faces to general images or thumbprints is further eroding the distinction between man and machine. Google's Knowledge Vault extends the Knowledge Graph using probabilistic methods to add facts gleaned from the Web [11]. Google Translate now effectively covers language translation between more than 100 human languages [12]. All major Web players are active in these areas, from Amazon's recommendation system [13] to Facebook, Microsoft, Twitter, or Baidu. Unfortunately, the sponsors required significant effort to reorganize and characterize the source knowledge bases as coherent inputs to KBAI. All of the impressive advances we have seen to date in distant supervised machine learning applications result from bespoke, manually trained efforts, repeated numerous times across providers.

We now understand how content-rich electronic knowledge bases may help power machine learning for natural language understanding and information processing. The usefulness is apparent to re-express the KBs to maximize the features available for machine learning, including disjointedness assertions to enable selection of positive and negative training sets. Specific aspects of the KBs, for which such reorganization is appropriate, include concepts, types, entities, relations, events, attributes, and statements. As we build these frameworks, they can facilitate mappings to other knowledge structures, and aid in data interoperability and information integration. We may apply these same principles for building a general structure to new domains or new knowledge bases. Three significant aspects—in machine learning, knowledge supervision, and feature engineering—intersect to re-express knowledge bases for KBAI purposes. Let's investigate each in turn.

Machine Learning

Machine learning is the construction of algorithms that can learn from and make predictions on data by building a model from example inputs. A wide variety of techniques and algorithms may be employed—such as Markov chains, neural networks, conditional random fields, Bayesian statistics, and many other options—that can be characterized by many dimensions. Some are supervised, meaning we need to train them against a standard labeled corpus to estimate parameters; others require little or no training—that is, are unsupervised—but may be less accurate as a result. Some are statistical; others use pattern matching of various forms. *Supervised learning* is a machine learning task of inferring a function from labeled training data, which optimally consists of positive and negative training sets. The supervised learning algorithm analyzes the training data and produces an inferred function that is used to determine the correct class labels for unseen instances. In supervised learning, we present positive and (often) negative training examples to the learning algorithm. *Unsupervised learning* is a different form of machine learning, in that the approach attempts to find meaningful, hidden patterns without the use of labeled data. We require no training examples in unsupervised learning. Supervised methods are more accurate than unsupervised methods, and nearly universally so in the realm of content information and knowledge.

Deep learning is a recent trend to combine multiple techniques. In this approach, the algorithm models the problem set as a layered hierarchy of distributed representations, with each layer using (often) neural network techniques for unsupervised learning, followed by supervised feedback (often termed 'back-propagation') to fine-tune parameters. While computationally slower than other techniques, this approach has the advantage of automating the supervised learning phase and is effective across a range of AI applications. The major disadvantage is that deep learning creates 'hidden' statistical features within its intermediate layers; it is impossible to interpret how the technique determines its final results. Deep learning is nonetheless producing amazing results in recognizing images, audio, video or sensory perception, and language translation. Effectiveness in knowledge areas has been less satisfactory, with the lack of explanatory power a further detriment.

In supervised learning, the main drawback is the effort and expense associated with labeling the positive or negative training examples (sets). The maximum effort occurs from constructing the training sets entirely by hand. We can reduce the effort by constructing them in a semiautomatic manner or by letting knowledge bases provide the labels. These techniques are known as semi-supervised, weak supervision, or distant supervision [14, 15].[6] The accuracy of the eventual models is only as good as the trueness of the input training sets, with traditionally the best results coming from manually determined training sets. We call the most accurate of these sets 'gold standards.' The creation of manual training sets may consume as much as

[6] Distant supervision was earlier called self-supervision, indirect supervision, or weakly supervised.

80% of overall machine learning efforts and is always a time-consuming task whenever employed.

One way to help overcome the costs of developing manual training sets is by a subclass of supervised learning called *distant supervision*, which is a method to use knowledge bases to label entities or other types automatically in text, which is then used to extract features and train a machine learning classifier. The knowledge bases provide coherent positive training examples and avoid the high cost and effort of manual labeling. When we use knowledge bases for distant supervision, we only use a portion of the structure as features. Still, other distant supervision efforts may be geared to other needs and use a different set of features. Indeed, broadly considered, knowledge bases have a rich diversity of possible features. These potential features arise from the text, and its content, syntax, semantics, and morphology; use vectors of co-occurring terms or concepts; categories; conventions; synonyms; linkages; mappings; relations; attributes; content placement within its knowledge graph; and disjointedness. Appendix C shows just how broadly diverse these types of features may be.

State-of-the-art machine learning for natural language processing and semantics uses distant supervision and knowledge bases like Freebase[7] or Wikipedia to extract training sets for supervised learning. We can create relatively clean positive and negative training sets with much reduced effort over manually created ones. However, as employed to date, distant supervision has mostly been a case-by-case, problem-by-problem approach, and most often applied to entity or relation extraction. The effort has heretofore not been systematic in approach nor purposefully applied across a range of ML applications. How to structure and use knowledge bases across a range of machine learning applications with maximum accuracy and minimum effort is what we call *knowledge supervision*, which I discuss more in a moment.

Besides supervised and unsupervised learning, a third broad category of machine learning is *reinforcement learning*. Unlike the first two categories where prior examples are used to learn a statistical prediction for new cases, reinforcement learning focuses on the learning process itself. Reinforcement learning is an active, iterative process where rewards associated with a given set of objectives are used to select from and optimize next actions. "Although one might be tempted to think of reinforcement learning as a kind of unsupervised learning because it does not rely on examples of correct behavior, reinforcement learning is trying to maximize a reward signal instead of trying to find hidden structure" [16]. Because we have not heretofore linked knowledge bases with models of action, we have limited our use of KBs to static questions and applications. Insofar as we may be able to stage and embed our knowledge bases into a true action model, a topic of Chaps. 7 and 16, we may be able to see them inform models of reinforcement learning as well.

[7] Freebase was retired from service by its owner, Google, in 2016. Many of its knowledge assertions have been transferred to Wikidata.

Knowledge Supervision

Whatever the combination of method, feature set, or training sets, the ultimate precision and accuracy of the machine learning require the utmost degree of true results in both positive and negative training sets. Training to inaccurate information merely perpetuates inaccurate information. As anyone who has worked extensively with source knowledge bases may attest, assignment errors and incomplete typing and characterizations are all too familiar. Further, few existing knowledge bases provide disjointedness assertions. Though early efforts in artificial intelligence understood that capturing and modeling common sense was both an essential and surprisingly tricky task —the impetus, for example, behind the 30-year attempt of the Cyc knowledge base—what is new in today's circumstance is how these massive knowledge bases can inform and guide symbolic computing. The literally thousand research papers regarding the use of Wikipedia data alone show how these massive knowledge bases are providing base knowledge around which AI algorithms can work. Unlike the early years of mostly algorithms and rules, AI has now evolved to explicitly embrace Web-scale content and data and the statistics that we may derive from global corpora.

The innovation of distant supervision has been to leverage knowledge bases to overcome the costs of labeling data and creating positive and negative training sets for supervised learning. Wikipedia, as noted, has been leveraged for these purposes by such players as IBM, Google, Facebook, Baidu, Microsoft, Amazon, and others [19]. However, each of these players has done their own massaging of Wikipedia from scratch to support these purposes. None of this is free. Much purposeful work is necessary to configure and stage the data structures and systems that support the broad application of distant supervision. The idea of *knowledge supervision*, our third component to KBAI along with feature engineering and machine learning, is to take distant supervision one step further.

To achieve these aims for knowledge supervision, we purposefully stage our source knowledge bases. We structure the KBs to maximize information extraction of concepts, entities, relations, attributes, and events because we have provided such structure in the central knowledge graph of KBpedia. We use these structures for linking and mapping to still additional knowledge sources. We support this entire process with methods for codifying self-learning such that our systems continue to get more accurate. We test continuously to improve the assignments and the accuracy of the system.

Table 4.1 Knowledge-based AI applications

• Attribute 'slot filling'	• Mapping
• Bespoke analysis	– Data mapping
• Bespoke platforms	– Knowledge base mapping
• Classifiers	– Ontology mapping
– Concept tagging	• Master data management
– Document categorization	• Natural language processing
– Entity classifiers	– Artificial writing
• Cluster analysis	– Autocompletion
– Concept clustering	– Entity linking
– Data clustering	– Language translation
• Cognitive computing	– Multi-language versions
• Converters	– Phrase (n-gram) identification
– Data conversion	– Speech recognition
– Format converters	– Speech synthesis
• Disambiguators	– Spell correction
– Word sense disambiguation	– Text generation
• Duplicates removal	– Text summarization
• Entity dictionaries	• Ontologies
– Gazetteers	– Ontology development
• Information extraction	– Ontology matchers
– Attribute extractors	– Ontology mappers
– Entity recognizers	• Pattern recognition[a]
– Event extractors	– Computer vision
– Relation extractors	– Facial recognition
– Subgraph extraction	– Image recognition
• Knowledge base improvements	– Optical character recognition
• Knowledge base population	• Reasoning
• Machine learning	– Inferencing
– Deep learning	– Question answering
– Distant supervision	– Recommendation systems
– Knowledge supervision	– Semantic relatedness analysis
– Supervised learning	– Sentiment analysis
– Reinforcement learning	• Search and information retrieval
– Unsupervised learning	• Semantic publishing

[a]Not a *knowledge supervision* ML option [17]

Table 4.1 provides a listing of some of those areas to which knowledge supervision may apply; some already use distant supervision or have been shown useful in academic research, and others we have not yet exploited.

Knowledge supervision is thus the purposeful structuring and use of knowledge bases to provide features and training sets for multiple kinds of machine learners that we may apply to multiple artificial intelligence outcomes. While distant supervision also uses knowledge bases, it does so passively, taking the knowledge bases as is, rather than re-expressing them in a purposeful, directed manner across multiple machine learning problems. Knowledge supervision is thus the better method to achieve KBAI.

Feature Engineering

Feature engineering is the process of creating, generating, and selecting the features used in machine learning, based on an understanding of the underlying data and choosing ones likely to impact learning results and effectiveness. A feature is a measurable property of the analyzed system. A feature is equivalent to what statistics calls an explanatory variable. The ML algorithms tend to favor features with high explanatory power independent of other features (that is, they are *orthogonal*) because each added feature adds a computational cost. Many features correlate with one another; in these cases, we need to find the strongest signals and exclude the other correlates. Tuning and refinement are also more difficult with too many features, what has sometimes been called the curse of dimensionality. Overfitting by using too many features is also often a problem, which limits the ability of the model to generalize to other data. Still, using too few features results in inadequate explanatory power.

Features and training sets are the major determinants of how successful the machine learning is. *Training sets* are a set of data used to discover potentially predictive relationships. In supervised learning, a positive training set provides data that meet the training objectives; a negative training set fails to meet the objectives. Features also pose trade-offs and require skill in selection and use. Though it is hard to find a discussion of best practices in feature extraction, many practitioners note that striking this balance is an art [18]. We might also need multiple learners to capture the smallest, independent (non-correlated) feature set with the highest explanatory power.[8]

An understanding of what features are possible within knowledge bases is the first hurdle toward more purposeful knowledge supervision. We stage the structured information as RDF triples and OWL ontologies, which we can select and manipulate via APIs and SPARQL. We also stage the graph structure and text with the support of a search engine,[9] which gives us powerful faceted search and other advanced NLP manipulations and analyses. These same features may also be utilized to extend the feature set available from the knowledge base through such actions as extracting new entities, attributes, or relations; fine-grained entity typing[10]; creation of word vectors or tensors; results of graph analytics; forward or backward chaining; and efficient processing structures. Appendix C overviews the features available to KBAI from using the KBpedia knowledge structure.

Because all features are selectable via either structured SPARQL query or faceted search, it is also possible to more automatically extract positive and negative training

[8] A rich literature provides guidance on feature selection and feature extraction. Feature extraction creates new features from functions of the original features, whereas feature selection returns a subset of the available features. It is also possible to apply methods—the best known and simplest being principal component analysis, among many—to reduce feature size (dimensionality) with an acceptable loss in accuracy.

[9] We have used the open-source systems Lucene and Solr in our platform work, though options such as ElasticSearch would work as well.

[10] Pattern recognition and its sub-methods are included in the table because they can be trained against labeled metadata for identification and captioning purposes. The input data differs from knowledge sources because they are digitized media. For this reason, these should not be understood as *knowledge supervised* sources.

sets. Attention to proper coverage and testing of disjointedness assertions is another purposeful step useful to knowledge supervision since it aids in identification of negative examples for the training. We get into such operational details in Chaps. 12–14.

These opportunities do not exhaust those available from applying Peircean guidelines to knowledge representation, backed by knowledge bases. However, knowledge management, data interoperability, and knowledge-based AI form the leading edge of promising new capabilities.

References

1. J. Girard, J. Girard, Defining knowledge management: Toward an applied compendium. Online J. Appl. Knowl. Manage. **3**, 1–20 (2015)
2. P. Buneman, Semistructured data, in *ACM Symposium on Principles of Database Systems (PODS)*, Tucson, Arizona (1997), pp. 117–121
3. S.B. Davidson, C. Overton, P. Buneman, Challenges in integrating biological data sources. J. Comput. Biol. **2**, 557–572 (1995)
4. J. Blossom, Enterprise Publishing and the 'New Normal' in I.T.—Are You Missing the Trend? (Mar. 2010)
5. L.D. Erman, V.R. Lesser, *A Multi-level Organization for Problem Solving Using Many Diverse, Cooperating Sources of Knowledge* (Carnegie-Mellon University, Pittsburgh, PA, 1975)
6. É. Bossé, J. Roy, S. Wark (eds.), *Concepts, Models, and Tools for Information Fusion* (Artech House Publishers, Norwood, 2007)
7. E. Hovy, R. Navigli, S.P. Ponzetto, Collaboratively built semi-structured content and artificial intelligence: The story so far. Artif. Intell. **194**, 2–27 (2013)
8. A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E.R. Hruschka Jr, T.M. Mitchell, Toward an architecture for never-ending language learning, in *AAAI* (2010), p. 3
9. IBM, This is Watson. IBM J. Res. Dev. **56**, 1:1–1:15 (2012).
10. A. Singhal, Introducing the Knowledge Graph: Things, Not Strings, *Official Google Blog* (May 2012)
11. X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, W. Zhang, Knowledge vault: A web-scale approach to probabilistic knowledge fusion, in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, 2014), pp. 601–610
12. T. Schulz, Google's Quest to End the Language Barrier, *Spiegel Online* (Sep. 2013)
13. G. Linden, B. Smith, J. York, Amazon. com recommendations: item-to-item collaborative filtering. IEEE Internet Comput. **7**, 76–80 (2003)
14. M. Craven, J. Kumlien, Constructing biological knowledge bases by extracting information from text sources, in *International Conference on Intelligent Systems for Molecular Biology (ISMB)* (1999), pp. 77–86
15. M. Mintz, S. Bills, R. Snow, D. Jurafsky, Distant supervision for relation extraction without labeled data, in *Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP*, Suntec, Singapore, 2–7 (2009), pp. 1003–1011
16. R.S. Sutton, A.G. Barto, *Reinforcement Learning: An Introduction, 2nd Edition (pre-release draft)* (MIT Press, Cambridge, MA, 2017)
17. X. Ling, D.S. Weld, Fine-grained entity recognition, in *Proceedings of the 26th AAAI Conference on Artificial Intelligence* (2012)
18. P. Domingos, A few useful things to know about machine learning. Commun. ACM **55**, 78–87 (2012)
19. F.M. Suchanek, G. Weikum, Knowledge bases in the age of big data analytics, in *Proceedings of the VLDB Endowment* (2014), pp. 1713–1714
20. F.J. Och, Statistical machine translation: Foundations and recent advances, in *The Tenth Machine Translation Summit*, Phuket, Thailand (2005)

Chapter 5
The Precepts

To gain the opportunities in artificial intelligence and knowledge management, we need to look the world squarely in the eye and tackle realities as they exist. As a philosopher, Charles S. Peirce was a confirmed and staunch defender of realism, though he was also an idealist in his belief that truth, while not perhaps knowable in its absolute limits, could be increasingly discovered through the application of logic and the scientific method. Pragmatism is the way forward to approach this ideal.

The world is a messy place. Not only is it complicated and richly diverse, but our ways of describing and understanding it are made more complex by differences in language and culture. We know the world is interconnected and interdependent. Effects of one change can propagate into subtle and unforeseen consequences. Not only is the world always changing, but so is our understanding of what exists in the world and how it affects and is affected by everything else. This continuous flux means we are always uncertain to a degree about how the world works and the dynamics of its working. Through education and research we continually strive to learn more about the world, but often in that process find what we thought was true is no longer so and even our human existence is modifying our world in manifest ways.[1]

[1] Some material in this chapter was drawn from the author's prior articles at the AI3:::Adaptive Information blog: "What is Linked Data?" (Jun 2008); "When is Content Coherent?" (Jul 2008); "'Structs': Naïve Data Formats and the ABox" (Jan 2009); "The Law of Linked Data" (Oct 2009); "When Linked Data Rules Fail" (Nov 2009); "The Bipolar Disorder of Linked Data" (Apr 2010); "Practical P-P-P-Problems with Linked Data" (Oct 2010); "What is Structure?" (May 2012); "The Rationale for Semantic Technologies" (Jul 2012); "Making Text a First-Class Citizen" (Jan 2013); "Three Leading Arguments for Semantic Technologies" (Jan 2013); "Seven Arguments for Semantic Technologies" (Feb 2013); "The Primacy of Search in the Semantic Enterprise" (Feb 2013); "'Natural Classes' in the Knowledge Web" (Jul 2015); "SWEETpedia" (available at http://www.mkbergman.com/sweetpedia/).

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8_5

Knowledge is very similar to this nature of the world. We find that knowledge is never complete and we can see it anywhere and everywhere. We capture and codify knowledge in structured, semi-structured, and unstructured forms, ranging from 'soft' to 'hard' information. We find that the structure of knowledge evolves with the incorporation of more information. We often see that knowledge is not absolute, but contextual. That does not mean truth does not exist; rather knowledge should be coherent, to reflect a logical consistency and structure that comport with our observations about the physical world. Knowledge, like the world, is continually changing; we thus must adapt to what we observe and learn.

Chapter 3 pointed to the importance of information to economic growth. We saw the breakpoint accelerations in growth tied to historical changes in the cost and access to information. Future generations will surely come to see the Internet phenomenon as one of those transitions. Massive storehouses of information, under free and open licenses, are available at our fingertips. None of these sources were designed for interoperability at a concept or knowledge level, and each has its context, format, and terminology. Here is a practically unlimited source of useful information that, by applying our approaches and principles to semantic technologies and interoperability, we can tap for digital reasoning and learning.

To tap this storehouse of information, to connect and make the information usable with other information, we need to understand what makes that information in its raw form a Tower of Babel. To overcome these differences we need to embrace some premises—or *precepts*—about how our information exists in its native forms, and then to adopt still further propositions for how to put that information on a common footing. These precepts relate to the nature of data, semantic heterogeneities in what that data means, and how we organize and classify that data. These precepts help set the ground rules for our actions going forward.

Equal Class Data Citizens

Knowledge representation, by our definition, operates in an electronic medium with messages conveyed in bits, which makes all information represented in the system as data. To deal in the realm of knowledge and belief, the purpose of our KR systems, we must be able to ingest and process any electronic data in any form that can contribute to our knowledge.[2] We include any digital information artifact in this category, including 'soft' or 'hard' information, social information, information of varying certainty, and information of diverse provenance. We thus define *content* as information that has the potential to contribute to knowledge.

[2] While streaming media alone does not meet this definition, transcripts or tags associated with the content do.

These variations are what would be called syntactic, or the structure or form of the information, though content ambiguities also lead to an entirely different plane of differences, those of a semantic nature. Whatever these differences of structure, format, or content, as long as the information represented is a possible contributor to knowledge, we must be able to ingest and process it. Knowledge management systems must treat all data forms with a potential to contribute to knowledge as equal class citizens.

The Structural View

A favorite, and I think useful, split of content is according to its native structure, that is, the structure it assumes when created for its primary purpose. One of these groupings is *structured data*, what we most often think of when we hear the term 'data.' This classification is where the information presented is according to a defined data model, commonly found in relational databases or other forms of tabular data, such as even an electronic spreadsheet. This information includes any managed by database management software, but it can also be as simple as an HTML table for the Web. We can model, organize, form, and format structured data in ways that are easy for us to manipulate. Much of our current know-how related to data and its management comes from our decades-long experience with structured data.

The second grouping of content is *unstructured data*, mostly consisting of text, which lacks an explicit data model or schema. (But it does comport with the 'structure' of natural languages.) All documents and output from word processors or editors fall into this category, as do transcriptions of talking or speech. For decades, researchers have estimated the amount of information within an enterprise embedded in text documents to approximate 80% of available information; some recent estimates put that contribution at 90% [1]. Whatever the number, the percentage of information in documents represents the preponderance of what might be useful for knowledge purposes within the organization.

The third grouping of content is thus *semi-structured data*, which is of more recent vintage. This category of content does not conform to a formal tabular or structural data model but gets its 'semi-structured' nature by embedding tags or other markers to denote fields within the content. We obtain it from unstructured data via data mining or information extraction. Separate annotations not embedded within the text, as is the case for *metadata*, are also part of this grouping. Markup languages embedded in text are a common form of such sources.

Semi-structured data provides something of a 'middle ground' between structured and unstructured sources. Semi-structured data models are sometimes called 'self-describing' (or schema-less).[3] The first known definition of semi-structured

[3]The earliest known recorded mention of "semi-structured data" occurred in 1992 from N. J.

data dates to 1993 by Peter Schäuble [2]. More current usage also includes the notion of labeled graphs or trees with the data stored at the leaves, with the schema information contained in the edge labels of the graph. Semi-structured representations also lend themselves well to data exchange or the integration of heterogeneous data sources. Another nice aspect of semi-structured formats is that they are readable as text, with a structure that can be understood and assigned by nonprogrammers without dedicated IT staff. Semi-structured data is the preferred form for annotations.

To date, we have good processing engines for specific semi-structured forms, such as rendering HTML in a Web browser or reading XML data sources, but inadequate engines for combining different forms of semi-structured data [3]. Moreover, semi-structured data is the basis for including unstructured text with structured data, but we still have the issues of extracting structure from various formats.

The Formats View

Broad categorizations by content, while useful for generalizing, mask the ways we can express these unstructured, semi-structured, and structured forms in various file formats. Since no 'official' repository for file formats exists, it is impossible to know how many different flavors of formats exist in the wild. The most extensive reference that I have found, kept on Wikipedia, lists nearly 1500 different file formats from AAC (audio coding) to zoo (file compression).[4] Perhaps this sounds worse than it should because these file formats span entire application areas from documents to archives to audio or video or gaming. Further, skewed power-law distributions mean only a fraction of these formats account for most uses. Individual application areas, such as word processing or spreadsheets, have merely scores or hundreds of formats. What we experience in the wild is also the subset of more popular formats, though in a medium as broad as the Web, Google has found tens of thousands of individual schemata [4]. Single enterprises may need to only deal with

---

Belkin and Croft, W. B., "Information filtering and information retrieval: two sides of the same coin?," in *Communications of the ACM: Special Issue on Information Filtering*, vol. 35(12), pp. 29–38. The next two mentions were in 1995 from D. Quass, A. Rajaraman, Y. Sagiv, J. Ullman, and J. Widom, "Querying semistructured heterogeneous information," presented at *Deductive and Object-Oriented Databases (DOOD '95), LNCS*, No. 1013, pp. 319–344, Springer, and M. Tresch, N. Palmer, and A. Luniewski, "Type classification of semi-structured data," in *Proceedings of the International Conference on Very Large Data Bases (VLDB)*. However, the real popularization of the term "semi-structured data" occurred through the seminal 1997 papers from S. Abiteboul, "Querying semi-structured data," in *International Conference on Data Base Theory (ICDT)*, pp. 1–18, Delphi, Greece, 1997 (http://dbpubs.stanford.edu:8090/pub/1996-19), and P. Buneman, "Semistructured data," in *ACM Symposium on Principles of Database Systems (PODS)*, pp. 117–121, Tucson, Arizona, May 1997 (http://db.cis.upenn.edu/DL/97/Tutorial-Peter/tutorial-semi-pods.ps.gz). Of course, semi-structured data had existed before these early references; only it had not been named as such.

[4] As of December 2017, the count was about 1450, with certainly some formats remaining unlisted. See https://en.wikipedia.org/wiki/List_of_file_formats.

a few score common formats, rather than thousands, with perhaps only a few dominant formats in given application areas. Word processors, for example, might be mandated or standardized. Still, even in this area, much readable text in multiple other formats is available.

Structured or semi-structured data formats also have a schema, in addition to the serialization formats used for transmittal. Some markup languages, such as HTML or Markdown, have embedded tags that instruct how to render Web pages or guide the user interface. Other markup languages, such as fielded text, structured text, simple declarative language (SDL), or more recently YAML or its simpler cousin JSON, have become more widely adopted and supported by formal specifications, tools, or APIs. Many prefer JSON, for example, as a form for Web applications. Some formats, like microformats or BibTeX records, rely less on syntax conventions and may use reserved keywords (such as AUTHOR or TITLE) to signal the key for the *key-value pair*.

These various forms, sometimes well specified with APIs and sometimes almost ad hoc as in spreadsheet listings, are what we call 'structs.' *Structs* can all be displayed as text and have, at a minimum, explicit or inferable key-value pairs to convey data relationships and attributes, with data types and values often noted by various white space, delimiter (such as angle brackets), or other text conventions. Some of these simple formats have been more successful than others, though none have achieved market dominance. Few universal principles have emerged as to syntax or format. One positive is that most of these various *struct* forms are easy for casual users to understand and easy for domain experts to write.

The sheer number of file formats one may encounter in the wild (including within the single organization) is such that pairwise translators between forms are not combinatorially possible. The only way to handle the diversity of forms and formats is to establish one or a limited few *canonical* formats and to translate wild forms to those formats. This scalable approach to federation is a central topic of Chap. 9.

The Content View

Refer to Jimmy Johnson by his name, and you might be referring to a former football coach, a NASCAR driver, a former boxing champ, a blues guitarist, or perhaps even a plumber in your hometown. Alternatively, perhaps your Jimmy is none of these individuals. The label 'Jimmy Johnson' is insufficient to establish identity. As another example, let's take the seemingly simple idea of 'cats.' In one source, the focus might be on house cats; in a second, domestic cats; and in a third, cats as pets. Are these ideas the same thing? Now, let's bring in some taxonomic information about the cat family, the Felidae. We have now expanded the idea of 'cats' to include lynxes, tigers, lions, cougars, and many other kinds of cats, domestic and wild (and, also extinct). The 'cat' label used alone clearly fails us miserably here.

As a third example, let's take the concept or idea of the named entity of Great Britain:

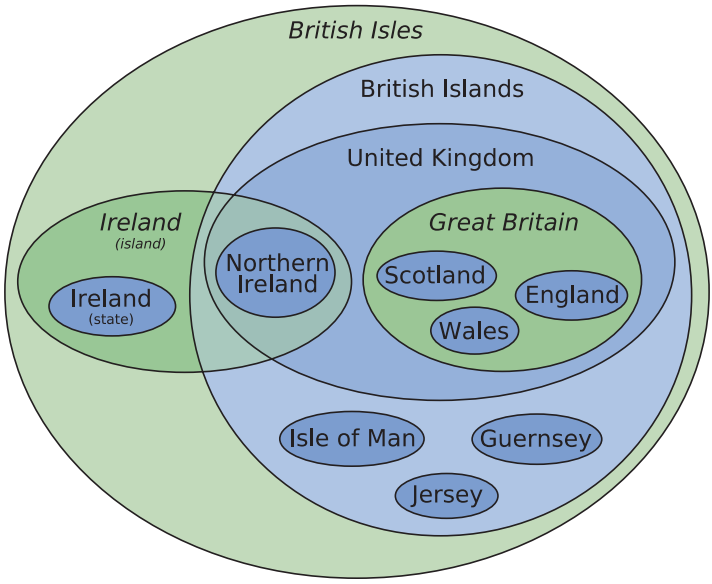


Fig. 5.1 Names can be complicated (courtesy of https://commons.wikimedia.org/wiki/File:British_Isles_Euler_diagram_15.svg)

Depending on usage and context, Great Britain can refer to quite different scopes and things. In one sense, Great Britain is an island. In a political sense, Great Britain can comprise the territory of England, Scotland, and Wales. Even more, precise understandings of that political grouping may include some outlying islands such as the Isle of Wight, Anglesey, the Isles of Scilly, the Hebrides, and the island groups of Orkney and Shetland. Sometimes the Isle of Man and the Channel Islands, which are not part of the United Kingdom, are included in error in that political grouping. Then, in another context, Great Britain may also include Northern Ireland, since the two countries sometimes combine their sports teams. These, plus other confusions, can mean quite different things when referring to 'Great Britain' as the Venn diagram of possibilities shows us in Fig. 5.1.[5]

Even with the same label, items in different information sources can refer to the same thing, but may not be the same thing or may define it with a different scope and content. *Ambiguity* is one source for such error, as our examples show. If we incorrectly identify the object, then connections can get drawn that are in error, which is why *disambiguation* is such a big deal in knowledge systems. In broad terms, these mismatches can be due to structure, domain, data, or language, with many nuances within each type.

Data without context and relationships are meaningless. Logic and consistency almost by definition imply the application of a uniform perspective, a single world-

[5] These associations also vary over time, again well evidenced by the scope of "Great Britain."

view. Multiple authors making contributions without a common frame of reference or viewpoint are unable to bring this consistency of perspective. The `sameAs` approach used to connect items in many current Web systems when they ignore such heterogeneities makes as little sense as talking about the plumber using facts drawn from the blues guitarist. Even if we can overcome the syntactic and format differences already discussed, we still face the hurdle of bridging the semantics of the data federation pyramid shown in Fig. 4.1.

Addressing Semantic Heterogeneity

The *idea of* something—that is, its *meaning*—is conveyed by how we define that something, the context in which we use the various tokens (terms) for that something, and in the variety of words or labels we apply to that thing. The label alone is not enough. We convey the *idea of a parrot* by our understanding of what the name *parrot* means. In languages other than English, the same idea of parrot may be conveyed by the terms Papagei, perroquet, loro, попугай, or オウム, depending on the native language. The idea of the '*United States*,' even just in English, may be conveyed with labels ranging from *America* to *US*, *USA*, *U.S.A.*, *Amerika*, *Uncle Sam*, or even the *Great Satan*. What these examples illustrate is that a single term is more often not the only way to refer to something, and a given token may mean vastly different things depending on the context. The oft-heard phrase, 'things, not strings,' captures this underlying fact [5].

Sources of Semantic Heterogeneity

Our understanding of the patterns in semantic heterogeneities—for which we need to account in the design of our knowledge systems explicitly—is pretty mature. We see confusion potentially arising from multiple terms for a single thing[6]; single terms applying to numerous things; terms whose meaning derives from context; how we characterize things; how we relate things; how we indicate surety or confidence; how we point to things; and how to annotate things.

Pluempitiwiriyawej and Hammer provided one of the first comprehensive schemes for classifying semantic heterogeneities [6]. I have used and added to this schema over many years. By decomposing this space into its various sources of semantic heterogeneities—as well as the work required to provide for such functions as search, disambiguation, mapping, and transformations—we can begin to understand how all of these components can work together to help achieve data interoperability.

Table 5.1 shows more than 40 sources of semantic heterogeneities, structurally organized, each of which is a possible impediment to get data to interoperate across sources:

[6] Though true synonyms are rare, our practical interest is to capture alternate labels for the same thing.

Table 5.1 Sources of semantic heterogeneities

Class	Category	Subcategory	Examples	Type[a] [7]
Language	Encoding	Ingest encoding mismatch	For example, ANSI *v* UTF-8 [8]	Concept
		Ingest encoding lacking	Mis-recognition of tokens because not being parsed with the proper encoding [8]	Concept
		Query encoding mismatch	For example, ANSI *v* UTF-8 in search [8]	Concept
		Query encoding lacking	Mis-recognition of search tokens because not being parsed with the proper encoding [8]	Concept
	Languages	Script mismatch	Variations in how parsers handle, say, stemming, white spaces or hyphens	Concept
		Parsing/ morphological analysis errors (many)	Arabic languages (right-to-left) *v* Romance languages (left-to-right)	Concept
		Syntactical errors (many)	Ambiguous sentence references, such as *I'm glad I'm a man, and so is Lola* (*Lola* by Ray Davies and the Kinks)	Concept
		Semantics errors (many)	River *bank v* money *bank v* billiards *bank* shot	Concept
Conceptual	Naming	Case sensitivity	Uppercase *v* lower case *v* Camel case	Concept
		Synonyms	United States *v* USA *v* America *v* Uncle Sam v Great Satan	Concept
		Acronyms	United States *v* USA *v* US	Concept
		Homonyms	Such as when the same name refers to more than one concept, such as Name referring to a person *v* Name referring to a book	Concept
		Misspellings	As stated	Concept
	Generalization/specialization		When single items in one schema are related to multiple items in another schema or vice versa. For example, one schema may refer to 'phone,' but the other schema has multiple elements such as 'home phone,' 'work phone,' and 'cell phone'	Concept

(continued)

Table 5.1 (continued)

Class	Category	Subcategory	Examples	Type[a] [7]
	Aggregation	Intra-aggregation	When the same population is divided differently (such as Census *v* Federal regions for states, England *v* Great Britain *v* United Kingdom, or full person names *v* first-middle-last)	Concept
		Inter-aggregation	May occur when we include sums or counts as set members	Concept
	Internal path discrepancy		Can arise from different source-target retrieval paths in two different schemas (for example, hierarchical structures where the elements are different levels of remove)	Concept
	Missing item	Content discrepancy	Differences in set enumerations or including items or not (say, US territories) in a listing of US states	Concept
		Missing content	Differences in scope coverage between two or more datasets for the same concept	Concept
		Attribute list discrepancy	Differences in attribute completeness between two or more datasets	Attribute
		Missing attribute	Differences in scope coverage between two or more datasets for the same attribute	Attribute
	Item equivalence		When we assert two types (classes or sets) as being the same when the scope and reference are not (for example, Berlin the city *v* Berlin the official city-state)	Concept
			When we assert two individuals as being the same when they are distinct (for example, John Kennedy the president *v* John Kennedy the aircraft carrier)	Attribute
	Type mismatch		When we characterize the same item by different types, such as a person typed as an animal *v* human being *v* person	Attribute
	Constraint mismatch		When attributes referring to the same thing have different cardinalities or disjointedness assertions	Attribute

(continued)

Table 5.1 (continued)

Class	Category	Subcategory	Examples	Type[a] [7]
Domain	Schematic discrepancy	Element-value to element-label mapping	One of four errors that may occur when attribute names or values may not be completely unambiguous	Attribute
		Attribute-value to element-label mapping		Attribute
		Element-value to attribute-label mapping		Attribute
		Attribute-value to attribute-label mapping		Attribute
	Scale or units	Measurement type	Differences, say, in the metric *v* English measurement systems, or currencies	Attribute
		Units	Differences, say, in meters *v* centimeters *v* millimeters	Attribute
	Precision		For example, a value of 4.1 inches in one dataset *v* 4.106 in another dataset	Attribute
	Data representation	Primitive data type	Confusion often arises in the use of literals *v* URIs *v* object types	Attribute
		Data format	Delimiting decimals by period *v* commas; various date formats; using exponents or aggregate units (such as thousands or millions)	Attribute
Data	Naming	Case sensitivity	Uppercase *v* lower case *v* Camel case	Attribute
		Synonyms	For example, centimeters *v* cm	Attribute
		Acronyms	For example, currency symbols *v* currency names	Attribute
		Homonyms	Such as when the same name refers to more than one attribute, such as Name referring to a person *v* Name referring to a book	Attribute
		Misspellings	As stated	Attribute
	ID mismatch or missing ID		URIs can be a particular problem here, due to actual mismatches but also use of namespaces or not and truncated URIs	Attribute
	Missing data		A common problem, more concerning with closed-world approaches than with open-world ones	Attribute
	Element ordering		Set members can be ordered or unordered, and if ordered the sequences of individual members or values can differ	Attribute

[a]Concept is the shorthand used for the schema or classes or TBox. Attribute is the shorthand used for instance data or entities and their ABox. See Chap. 8 for more on the T Box-ABox split

We have assigned these structural aspects to one of the two types: (a) those that may arise from the *conceptual* differences between sources (mostly from schema differences) and (b) those due to value and *attribute* discrepancies (data). The table also provides examples of what each of these categories of heterogeneities means.

This listing is a reasonably comprehensive view of what is involved in getting things to talk together (*semantic agreement*). Fortunately, via the adoption of standard syntactic protocols and semantic languages, means for managing many of these possible heterogeneities are handled in the background when complying with their rules (axioms) or language constructs. That still leaves us with the heterogeneities associated with human communications and how to measure the attributes of things.

From the conceptual to actual data, then, we see differences in perspective, vocabularies, measures, and conventions. Some of these differences and heterogeneities are intrinsic to the nature of the data at hand. Some of these heterogeneities also arise from the basis and connections asserted between datasets. Only by systematically understanding these sources of heterogeneity—and then explicitly addressing them—can we begin to try to put disparate information on a common footing. Only by reconciling differences can we start to get data to interoperate.

Role of Semantic Technologies

The first advantage of semantic technologies is that all kinds of information are unified. No matter what information you consider, any content type may become a 'first-class citizen.' For really the first time, we can put all kinds of information ranging from traditional databases and spreadsheets (*structured*) to markup, Web pages, XML, and data messages (*semi-structured*), and then on to documents and text (*unstructured*) or multimedia (via *metadata*) on a level playing field. These data, now all treated on an equal footing, can be searched and retrieved by a variety of techniques. These range from SQL, standard text search, or SPARQL, depending on content type. This unique combination enables us to fulfill all of the aspects of findability—find, discover, and navigate. Because of the diversity of search options available, we can vary and optimize search results depending on circumstance and needs. Because all content is represented either as a type of thing, an individual thing, or the relationships between those things, we may use these classifiers for faceting or grouping. Further, the connections put all things in context, useful to ensure that results are relevant and disambiguated.

What works efficiently for transactions and accounting is a poor choice for knowledge problems. Traditional relational databases work best with structured data; are inflexible and fragile when the nature (schema) of the world changes; and thus require constant (and expensive) re-architecting in the face of new

knowledge or new relationships. Conversely, for semantic technologies, we describe things and their relationships based on the 'idea of the thing,' not limited to keywords. Thus, we can describe and find things using alternative terms, synonyms, acronyms, or jargon. We can add on or extend semantic vocabularies without altering what we have already asserted, assuming that the prior assertions still hold true.

We should use semantic technologies instead of conventional information technologies in the areas of knowledge representation (KR) and knowledge management (KM). Semantic technologies are orthogonal to some other current technologies, including cloud computing and big data. Semantic technologies are not limited to open data: they are equivalently useful to private or proprietary data. Semantic technologies do not imply some grand, shared schema for organizing all information, though, at some levels, that is extremely useful. Semantic technologies are not 'one ring to rule them all,' but rather a way to capture the worldviews of particular domains and groups of stakeholders. Semantic technologies appropriately done are not a replacement for existing information technologies, but rather an added layer that can leverage those assets for interoperability and to overcome the semantic barriers between existing information silos. These very same semantic technologies also provide the proper representational basis for symbol-based machine learning and intelligence.

Semantic technologies give us the basis for understanding differences in meaning across sources, specifically geared to address differences in real-world usage and context. These semantic tools are essential for providing common bases for relating structured data across various sources and contexts. These same semantic tools are also the basis by which we can determine what unstructured content 'means,' thus providing the structured data tags that also enable us to relate documents to conventional data sources using semi-structured data. Semantic technologies are therefore the enablers for making information understandable to both humans and machines across sources.

Semantic technologies expressly address these heterogeneities, some more strongly in some areas than others. However, to capture the scope of the heterogeneities listed, we need the technologies to mimic aspects of human language, symbology, and logic. We express ourselves via the equation and the document, not to mention jumping up and down and gesticulating. By accounting explicitly for the relationships between things, we can use semantic technologies to better capture context, essential for navigation and reduction of ambiguity. We can use the richness of relationships to group, classify, filter, or find things. The basic assertion in our semantic languages declares relationships between and for things. These statements, when combined with the objects of some statements being the subjects of others, leads to a graph structure (see Chap. 1). We may apply various logics based on the nature of our declarations to compute over the structure and understand or infer relationships between things. We can use the graph structures for novel traversal mechanisms and network analysis. No other information structure provides these unique advantages.

Semantics and Graph Structures

The graph structures of semantic schema mean that any node can become an entry point to the knowledge space for discovery. The traversal of information relationships occurs from the selection of predicates or properties that we use to create this graph structure in the first place. This richness of characterization and objects also means we can query or traverse this space in multiple languages or via the full spectrum by which we describe or characterize things. Semantic-based knowledge graphs are potentially an explosion of richness in characterization and how those characterizations get made and referred to by any stakeholder.[7] We enable the user community to determine our search structures, rather than some group of designers or information architects. It should not be surprising that search offers one of the quickest and most visible paths to gain the benefits of semantic technologies.

Existing IT assets represent massive sunk costs, legacy knowledge and expertise, and (often) stakeholder consensus. These systems are still mostly stovepiped. Strategies that counsel replacing existing IT systems risk wasting existing assets. We are better served to leverage the value already embodied in these systems while promoting interoperability and integration. The beauty of semantic technologies—adequately designed and deployed in a Web-oriented architecture—is that a thin interoperability layer may be placed over existing IT assets to achieve these aims. We can use the knowledge graph structure to provide the semantic mappings between schema, while we use a Web service framework to convert sources to the canonical data model. Via these approaches, we may preserve prior investments in knowledge, information, and IT assets while enabling interoperability. The existing systems can continue to provide the functionality as initially deployed. Meanwhile, we may expose and integrate the KR-related aspects with other knowledge assets on the physical network. Being able to manage semantic heterogeneity is the kickstarter to this process.

Carving Nature at the Joints

The embracing of semantics and the languages to express them is but the prerequisite. Once we decide the rules of the game, we need to populate our domain. That means we need to capture the concepts, instances, attributes, and relations of our domain. This capturing forms our vocabulary, and how we group, classify, and type that vocabulary should reflect the reality of our domain and how we organize it. Stated in the abstract this sounds like a tall order. However, we help fulfill this order if we seek to organize our domain in the most realistic way possible, what Plato, speaking as Socrates in the dialog with Phaedrus, says[8]:

[7] Robert Hillard notes that he agrees with the importance of semantics and graph structures, but also believes the complexity of real-world information and knowledge graphs are major obstacles to the navigation of content.

[8] Plato, "Phaedrus Dialog (page 265e)," *Perseus Digital Library* available: http://www.perseus.tufts.edu/hopper/text?doc=Perseus%3Atext%3A1999.01.0174%3Atext%3DPhaedrus%3Apage%3D265.

SOCRATES
It seems to me that the discourse was, as a whole, really sportive jest; but in these chance utterances were involved two principles, the essence of which it would be gratifying to learn, if art could teach it.

PHAEDRUS
What principles?

SOCRATES
That of perceiving and bringing together in one idea the scattered particulars, that one may make clear by definition the particular thing which he wishes to explain; just as now, in speaking of Love, we said what he is and defined it, whether well or ill. Certainly by this means the discourse acquired clearness and consistency.

PHAEDRUS
And what is the other principle, Socrates?

SOCRATES
That of dividing things again by classes, where the natural joints are, and not trying to break any part, after the manner of a bad carver. As our two discourses just now assumed one common principle, unreason, and then, just as the body, which is one, is naturally divisible into two, right and left, with parts called by the same names, so our two discourses conceived of madness as naturally one principle within us, and one discourse, cutting off the left-hand part, continued to divide this until it found among its parts a sort of left-handed love, which it very justly reviled, but the other discourse, leading us to the right-hand part of madness, found a love having the same name as the first, but divine, which it held up to view and praised as the author of our greatest blessings.

PHAEDRUS
Very true.

SOCRATES
Now I myself, Phaedrus, am a lover of these processes of division and bringing together, as aids to speech and thought; and if I think any other man is able to see things that can naturally be collected into one and divided into many, him I follow after and 'walk in his footsteps as if he were a god.'

The idea of 'carving nature at the joints' is a mindset we can apply to all of the major divisions in our vocabulary; namely, things, concepts, relations, and attributes.

Forming 'Natural' Classes

As we see, going back at least to Plato and Aristotle, how to properly define and bound categories and concepts has been a topic of much philosophical discussion. If we do not scope the organization of our knowledge and define it consistently, then it is virtually impossible to construct a logical and coherent way to reason over this structure. Aristotle set the foundational basis for understanding what we now call

natural kinds and categories (or 'classes'). The universal desire to understand and describe our world has meant that philosophers have argued these splits and their bases ever since. We can place these philosophical arguments into three broad camps. First, we have *realists*, who believe things have independent order and existence in the natural world, apart from thought. Second, we have *nominalists*, who believe that humans provide the basis for how things are organized in part by how we name them. Third, we have *idealists*, or anti-realists, who believe 'natural' classes are generalized ones that conform to human ideals of how the world is organized but are not independently real [7]. These categories shade into one another, such that these beliefs become strains in various degrees for how any one philosophy might be defined.

The realist strain, also closely tied to the sciences and the scientific method, is what most guides the logical basis of semantic technologies and our view of how to organize the world. Science and technology are producing knowledge in unprecedented amounts, and realism is the best approach for testing the trueness of new assertions. We think realism is the most efficacious approach to knowledge representation designs. Being explicit about the philosophy in how we construct our knowledge representations helps decide sometimes sticky design questions, as we will see multiple times throughout this book.

Aristotle believed that the world fits into categories, that categories were hierarchical in nature, and what defined a particular class or category was its essence or the attributes that uniquely define what a given thing is. A mammal has the essences of being hairy, warm-blooded, and live births. These essences distinguish mammals from other types of animals such as birds or reptiles or fishes or insects. Essential properties are different from accidental or artificial distinctions, such as whether a man has a beard or not or whether he is gray- or red-haired or of a certain age or country. We base a natural classification system on real differences of character and not artificial or single ones. Hierarchies arise from the shared generalizations of such essences among categories or classes. Under the Aristotelian approach, classification is the testing and logical clustering of such essences into more general or more specific categories of shared attributes. Because these essences are inherent to nature, natural clusters are an expression of real relationships in the real world, often hierarchical in structure.

By the age of the Enlightenment, some began to question these long-held philosophies. Descartes famously grounded the perception of the world into innate ideas in the human mind. Descartes' philosophy, built upon that of William of Ockham of Occam's razor fame, maintained that individuals populate the world; no such things as universals exist. In various guises, thinkers from Locke to Hume questioned a solely realistic organization of concepts in the world [9]. While there may be 'natural kinds,' categorization is also an expression of the innate drive by humans to name and organize their world, and was the dominant view of these emerging nominalists.

Charles S. Peirce started a mighty swing back to realism. He was the first, by my reading, who looked at the question of 'natural classes' sufficient to provide design guidance, and which may sometimes be contraposed against what some call 'artificial classes' (we also tend to use the term 'compound' classes). Natural classes were a key underpinning to Peirce's own efforts to provide a uniform classification sys-

tem related to inquiry and the sciences. A *natural class* is a set of members that share the same set of attributes, though with different values (such as differences in age or hair color for humans). Some of those attributes are also more essential to define the *type* of that class (such as humans being warm-blooded with live births and hair and use of symbolic languages). Artificial classes tend only to add one or a few shared attributes and do not reflect the essence of the type [10]. Our use and notion of 'natural classes' hews closely to how Peirce understood the concept:

> So then, a natural class being a family whose members are the sole offspring and vehicles of one idea, from which they derive their peculiar faculty, to classify by abstract definitions is simply a sure means of avoiding a natural classification. I am not decrying definitions. I have a lively sense of their great value in science. I only say that it should not be by means of definitions that one should seek to find natural classes. When the classes have been found, then it is proper to try to define them; and one may even, with great caution and reserve, allow the definitions to lead us to turn back and see whether our classes ought not to have their boundaries differently drawn. After all, boundary lines in some cases can only be artificial, although the classes are natural …. (1902, EP 2:125)

Peirce's ideas of a natural kind appear closely tied to his realism:

> Any class which, in addition to its defining character, has another that is of permanent interest and is common and peculiar to its members, is destined to be conserved in that ultimate conception of the universe at which we aim, and is accordingly to be called 'real.' (1901, CP 6.384)

Another guideline that Peirce provides is that *intension* is also a means for determining a natural classification:

> The descriptive definition of a natural class, according to what I have been saying, is not the essence of it. It is only an enumeration of tests by which the class may be recognized in any one of its members. A description of a natural class must be founded upon samples of it or typical examples. (1902, CP 1.223)

Peirce greatly admired the natural classification systems of Louis Agassiz and used animal lineages in many of his examples. He was a strong proponent of natural classification. Though we have replaced the morphological basis for classifying organisms in Peirce's day with genetic ones, Peirce would surely support this new knowledge, since he grounded his philosophy on a triad of primitive unary, binary, and tertiary relations, bound together in a logical sign process seeking truth.

For example, natural class instances, which are by definition intensional due to the *differentia* that comprises their class, may be declared by assignment to a class type. Once we define a type such as a hairless mammal that walks in a bipedal manner as a *human*, we can after that assign individual people to that class type and thereby infer human properties (or characteristics). We need not specify all possible human properties per individual under a strictly intensional approach nor enumerate all human individuals under a strictly extensional approach. We can let the use of type assignments bridge this divide. We can also see that, depending on the context, we may want to speak about *human* as a class (type) subsuming individual people or to speak about *human* as an instance with particular kinds of properties (attributes). I discuss further this 'punning' metamodeling technique in Chap. 9.

Peirce's concept of 'natural kinds' or 'natural classes' is not limited to things only found in nature. Peirce's semiotics (theory of signs) also recognizes 'natural' distinctions in areas such as social classes, the sciences, and human-made products [10]. These distinctions are important because they affirm essences and realities in the external world. A 'natural' classification is not limited to the animate. 'Natural' classification is premised on reason and subject to testing. Again, the key discriminators are the essences of things that distinguish them from other things, and the degree of sharing of attributes contains the basis for understanding relationships and hierarchies.

Menno Hulswit is one of the scholars who has studied Peirce's concept of 'natural classes' most closely [10]. As he has observed:

> From the natural sciences, Peirce had learned that the forms of chemical substances and biological species are the expression of a particular internal structure. He recognized that it was precisely this internal structure that was the final cause by virtue of which the members of the natural class exist. (p. 759)

> … Peirce's view may be summarized as follows: Things belong to the same natural class on account of a metaphysical essence and a number of class characters. The metaphysical essence is a general principle by virtue of which the members of the class have a tendency to behave in a specific way; this is what Peirce meant by final cause. This finality may be expressed in some sort of microstructure. The class characters which by themselves are neither necessary nor sufficient conditions for membership of a class, are nevertheless concomitant. In the case of a chair, the metaphysical essence is the purpose for which chairs are made, while its having chair-legs is a class character. The fuzziness of boundary lines between natural classes is due to the fuzziness of the class characters. Natural classes, though very real, are not existing entities; their reality is of the nature of possibility, not of actuality. The primary instances of natural classes are the objects of scientific taxonomy, such as elementary particles in physics, gold in chemistry, and species in biology, but also artificial objects and social classes.

> By denying that final causes are static, unchangeable entities, Peirce avoided the problems attached to classical essentialism. On the other hand, by eliminating arbitrariness, Peirce also avoided pluralistic anarchism. Though Peircean natural classes only come into being as a result of the abstractive and selective activities of the people who classify, they reflect objectively real general principles. Thus, there is not the slightest sense in which they are arbitrary: 'there are artificial classifications in profusion, but [there is] only one natural classification.' (1902, CP 1.275) (pp. 765–6)

Though all of this sounds somewhat abstract and philosophical, these distinctions are not merely metaphysical. The ability to organize our representations of the world into natural classes also carries with it the ability to organize that world, reason over it, draw inferences from it, and truth test it. Indeed, as we may discover through knowledge acquisition or the scientific method, this world representation is itself mutable. Our understanding of species relationships, for example, has changed markedly, especially most recently, as the basis for our classifications shifts from morphology to DNA. Einstein's challenges to Newtonian physics similarly changed the 'natural' way by which we need to organize our understanding of the physical world.

A Mindset for Categorization

These points are not academic. The central weakness, for example, that I have noted for Wikipedia over many years has been its category structure. Category inconsistencies are the root source of the problem that Wikipedia cannot presently act as a computable knowledge graph.[9] Categories often do not conform to a natural classification scheme, and many categories are 'artificial' in that they are compound or distinguished by a single attribute. 'Compound' (or artificial) categories (such as `Films directed by Pedro Almodóvar` or `Ambassadors of the United States to Mexico`) are not 'natural' categories, and including them in a logical evaluation only acts to confuse attributes from classification. To be sure, we should decompose such existing categories into their attribute and concept components, and possibly only include the decomposed versions (if then) when constructing a schema of the domain. 'Artificial' categories may be identified in the Wikipedia category structure by both syntactical and heuristic signals. One syntactical rule is to look for the head of a title; one heuristic signal is to select out any category with prepositions. Across all rules, 'compound' categories account for most of what we remove to produce 'cleaned' categories. Including administrative and other problem categories, perhaps half to two-thirds of Wikipedia's categories do *not* meet the definition of natural categories, though Wikipedia's editors continue to make improvements [11]. Independent actors have staged and processed Wikipedia multiple times to overcome these limits to create usable knowledge bases.

Whatever the target for the categorization effort may be, Peirce put forward some general execution steps:

> … introduce the monadic idea of 'first' at the very outset. To get at the idea of a monad, and especially to make it an accurate and clear conception, it is necessary to begin with the idea of a triad and find the monad-idea involved in it. But this is only a scaffolding necessary during the process of constructing the conception. When the conception has been constructed, the scaffolding may be removed, and the monad-idea will be there in all its abstract perfection. According to the path here pursued from monad to triad, from monadic triads to triadic triads, etc., we do not progress by logical involution—we do not say the monad involves a dyad—but we pursue a path of evolution. That is to say, we say that to carry out and perfect the monad, we need next a dyad. This seems to be a vague method when stated in general terms; but in each case, it turns out that deep study of each conception in all its features brings a clear perception that precisely a given next conception is called for. (1896, CP 1.490)

This quote is at the root of Peirce's views concerning the universal categories, the main topic of the next chapter. Triads figure prominently in this thinking. As we weave the various threads in Peirce's philosophy together, we also come to see the logic of how the three components of inquiry work in a similar manner to categorization, itself just a more structured view of what Peirce discussed as a

[9] Some reviewers have suggested that the issue is a matter of scale. While I agree large scale causes its own challenges, I believe the problem is one more of coherence and lack of consistency.

generalization. What we learn from Peirce in this investigation is that categorization, thankfully, is a knowledge representation task, that we can approach logically and systematically. We can adopt a categorical mindset about how to think of the world. The assignments should be defensible, but we should also be ready to change them when faced with better evidence or logic. We learn more about how to think through categorization in Chap. 6.

Connections Create Graphs

When representing knowledge, more things and concepts get drawn into consideration. In turn, the relationships of these things lead to connections between them to capture the inherent interdependence and linkages of the world. As still more things get considered, we make and proliferate more connections. This process naturally leads to a graph structure, with the things in the graphs represented as nodes and the relationships between them represented as connecting edges.[10] More things and more connections lead to more structure. Insofar as this structure and its connections are coherent, the natural structure of the knowledge graph itself can help lead to more knowledge and understanding.

Coherent and logical graphs first require natural groupings or classes of concepts and entity types by which to characterize the domain at hand, situated to one another with testable relations. We characterize entity types with a similar graph of descriptive attributes. Concepts and entity types thus represent the nodes in the graph, with relations being the connecting infrastructure. Relatedness of shared attributes or types of relations can also create ontological structures that enable inference and a host of graph analytics techniques for understanding meaning and connections. For such a structure to be coherent, the nodes (classes) of the structure should also be as natural as possible, applying the same categorization approaches.

Unlike traditional data tables, graphs have some inherent benefits, particularly for knowledge representations. They provide:

- A coherent way to navigate the knowledge space
- Flexible entry points for each user to access that knowledge (since every node or relation is a potential starting point)
- Inferencing and reasoning structures about the space
- Connections to related information
- Ability to connect to any form of information
- Concept mapping, and thus the ability to integrate external content
- A framework to disambiguate concepts based on relations and context
- A common vocabulary to drive content ‘tagging.’

Graphs are the natural structures for knowledge domains if they follow a ‘natural’ classification and we test them for coherence. Once built, graphs offer some

[10] See Fig. 1.3.

analytical capabilities not available through traditional means of information structure. Graph analysis is a rapidly emerging field, but we are already able to gauge some unique measures of knowledge domains, such as influence, relatedness, proximity, centrality, inference, clustering, shortest paths, and diffusion. As science is coming to appreciate, graphs can represent any extant structure or schema. The universal character of graphs makes them an attractive target for many analytic tools.

The essence of knowledge is that it is ever-growing and expandable. New insights bring new relations and new truths. The structures we use to represent this knowledge must themselves adapt and reflect the best of our current, testable understandings. Keeping in mind the need for 'natural' classes—that is, consistent with testable, knowable truth—is a building block in how we should organize our knowledge graphs. Through such guideposts as coherence, inference, and truthfulness, these structural arrangements become testable propositions. As Peirce, I think, would admonish us, failure to meet these tests is grounds for rejiggering our structures and classes. In the end, coherence and computability become the hurdles that our knowledge graphs must clear to become reliable structures.

References

1. C. Anderson, Leveraging data to drive innovation, *Wall Street Journal*, https://www.wsj.com/articles/SB10001424127887323468604578245540627666664
2. P. Schäuble, SPIDER: A multiuser information retrieval system for semistructured and dynamic data, in *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (ACM, 1993), pp. 318–327
3. M. Magnani, D. Montesi, A Unified Approach to Structured, Semistructured and Unstructured Data, *TECHREPORT in Education, Information Processing and Management* **2**, 263–275 (2004)
4. J. Madhavan, S.R. Jeffery, S. Cohen, X. Dong, D. Ko, C. Yu, A. Halevy, Web-scale data integration: You can only afford to pay as you go, in *CIDR* (2007)
5. A. Singhal, Introducing the knowledge graph: Things, not strings, *Official Google Blog* (May 2012)
6. C. Pluempitiwiriyawej, J. Hammer, *A Classification Scheme for Semantic and Schematic Heterogeneities in XML Data Sources*, vol 36 (University of Florida, Gainesville, FL, 2000)
7. P. Steiner, CS Peirce and artificial intelligence: Historical heritage and (new) theoretical stakes, in *Philosophy and Theory of Artificial Intelligence*, ed. by V. C. Müller (Springer, Heidelberg, 2013), pp. 265–276
8. M.K. Bergman, Tutorial: Internet Languages, Character Sets and Encodings, BrightPlanet Corporation Technical Documentation (Mar. 2006)
9. M.R. Ayers, Locke versus Aristotle on natural kinds. J. Philos. **78**, 247–272 (1981)
10. M. Hulswit, Peirce's teleological approach to natural classes. Trans. Charles S. Peirce Soc. **33**, 722–772 (1997)
11. M.K. Bergman, Shaping wikipedia into a computable knowledge base, in *AI3:::Adaptive Information* (Mar. 2015)

Part II
A Grammar for Knowledge Representation

Chapter 6
The Universal Categories

Knowledge representation has the mission to capture human knowledge and then be able to reason over it, in a form understandable to its designers, us humans, and interpretable by a Turing complete computer. The ability to represent natural (human) language and the ability to capture all basic logical premises and arguments are core KR requirements. As in human language, where we split our words into roughly nouns and verbs and modifiers and conjunctions of the same, we need a similar primitive vocabulary and rules for constructing statements. These basic building blocks are known as the *grammar* of the KR language. We then need to embed a well-considered grammar into formal, standardized languages that computers can readily interpret, backed with tools and a user community capable of exercising them to achieve our purposes. The three chapters in this Part II specifically talk to these needs.

Our KR language should represent how we, as humans, think about, organize, and reason about our world. Our KR language needs to address, as discussed in Chap. 4, the significant opportunities for data interoperability and artificial intelligence (machine learning and reasoning). To achieve these purposes, we need to integrate knowledge bases to provide the information pool and the testable bases upon which the KR language operates. To reason over this knowledge, we need a logical foundation that is consistent and coherent, for which we look to Peirce.[1] We begin, in this chapter, with a grounding based on Peirce's universal categories, and then introduce our KR grammar in Chap. 7 and our KR languages and models in Chap. 8.

[1] Some material in this chapter was drawn from the author's prior articles at the *AI3:::Adaptive Information* blog: "Give Me a Sign: What Do Things Mean on the Semantic Web?" (Jan 2012); "A Foundational Mindset: Firstness, Secondness, Thirdness" (Mar 2016); "Re-thinking Knowledge Representation" (Mar 2016); "The Irreducible Truth of Threes" (Sep 2016); "How I Study CS Peirce" (Aug 2017); "Why I Study CS Peirce" (Aug 2017); "How I Interpret CS Peirce" (Sep 2017).

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8_6

A Foundational Mindset

Historically, Peirce is known as the father of *pragmatism* (*pragmaticism*, his preferred term). The ideas behind Peircean pragmatism are how to think about signs and representations (semiosis); logically reason and handle new knowledge (abduction) and probabilities (induction); make economic research choices (pragmatic maxim); categorize; and let the scientific method inform our inquiry. All of these contributions are grounded in Peirce's *universal categories* of *Firstness*, *Secondness,* and *Thirdness*. Herein lays the key to being informed by Peirce when it comes to representing new knowledge, categorization, or problem-solving. It is the mindset of Thirdness and the nature of Firstness and Secondness that provide that guidance.

A Common Grounding in Peirce

The essence of knowledge is that it is ever-growing and expanding. New insights bring new relations and new truths. The structures we use to represent this knowledge must themselves adapt and reflect the best of our current, testable understandings. We want a foundation to the KR language that can capture reality, from cosmology to thought, process, and action. We need a grammar expressed in computer-readable languages that can capture the possibilities and current facts of today, plus new potentials arising from emerging knowledge. We want open and standard computer-readable languages to encourage broader adoption and therefore greater availability of toolsets and expertise. For interoperability, the scaffolding, or knowledge graph at the heart of the system, must have the flexibility to model any knowledge domain, from math and philosophy to lifeforms, society, and technologies. Eventually, we want to express these capabilities in cost-effective, deployable platforms with acceptable maintenance costs and long service lives.

We want to link or integrate existing knowledge bases. That requirement means supporting formats and mapping methods to facilitate the exchange. More importantly, we need a knowledge representation framework and grammar for adapting or growing the knowledge graph, matching objects, attributes, referents, and relations. The framework needs to follow a grammar that enables making and testing logical statements, along with inferencing and other reasoning. We want to construct this entire scaffolding in such a way that we capture all relevant features to provide a rich structure for machine learners.

Studying Peirce is hard. This difficulty is partly the result of Peirce. In his quest for precision in terminology, Peirce has created a unique vocabulary, sometimes jawbreaking, often with multiple terms that change over time for specific concepts. The difficulty partly comes from the cacophony of views about what Peirce did or intended to say. Complications also arise from the fragmentation of his manuscripts, some still unpublished, and sometimes confused chronologies that have, at times,

led to questionable scholarship. I do not doubt that scholars will continue to tease out profound insights from Peirce, likely for centuries to come.[2]

Peirce believed in the *real* as that which is as it is apart from what anyone thinks about it, a refutation of Descartes' view. He believed in *truth*, which the scientific method and social consensus (agreement of signs) can increasingly reveal, but current belief as to what is 'truth' is fallible and can never be realized in the absolute (it is a limit function). Distance and possible different understandings arise in the interplay of the object, its representation, and its interpretation.[3] Better approximations of truth come from questioning using the scientific method (via a triad of logics) and from refining consensus within the community about how (via language signs) we communicate that truth. Peirce termed this overall approach pragmatism, in which he firmly grounded his logics and theory of signs. Through the scientific method and questioning, we may get closer to the truth and to an ability to communicate it to one another, even though absolute 'truth' may require infinite inquiry by an infinite community. At any point, new knowledge may change the basis of our truth-seeking.

The connections of Peirce's sign theory, his threefold logic of *deduction-induction-abduction*, the importance of the scientific method, and his understanding about a community of inquiry have all fed my intuition that Peirce was on to some fundamental insights suitable to knowledge representation. Peirce's writings instruct us that, *firstly*, we need to embrace terminology that is precise for concepts and relations to communicate effectively within our communities. *Secondly*, we need to capture the right particular things of concern in our knowledge domain and connect them using those relations.[4] *Thirdly*, we need to organize our knowledge domain by general types based on logical, shared attributes, and embrace a process for expanding that structure with acceptable effort to deal with new or emergent information.

Truth Is Testable and Fallible

Peirce's time, as is our own, was one of great scientific advance and challenges to conventional understanding. During Peirce's professional lifetime, advances were occurring in the knowledge of waves and fields, chemical periodic table, evolution, electricity, and thermodynamics and gases. Given this ferment, it is clear why Peirce's worldview supported the ideas of the potential fallibility of understood 'truth,' and the fact that truth itself stood upon a gradation of certainty.

Completeness of information and completeness of understanding are each, themselves, ideals. We strive for them, but we never can fully achieve them. While we may reach sufficient certitude to bring about belief, itself an essential motivator in

[2] See Appendix A for further perspectives on Peirce.

[3] But this same logic provides the explanation for the process of categorization, also grounded in Firstness, Secondness, and Thirdness; see Chap. 10.

[4] This approach naturally leads to a knowledge graph structure.

this question, we will never entirely achieve it. 'Truth,' then, is ultimately (as a continuous limit function) unachievable. However, 'belief,' which guides our actions, may be achieved, and thus should be the objective of our inquiries. Any scientist spending much time on Peirce's writings would quickly affirm that, in nature, Peirce is a scientist. His insights and attention are grounded in science. His understandings of measurement and error and precision are those of a scientific practitioner.

Upper Ontologies, Context, and Perspective

Some form of conceptual schema governs every knowledge structure used for knowledge representation (KR). In the semantic Web, such schemata are known as *ontologies*, since they attempt to capture the nature or *being* (Greek ὄντως, or ontós) of the knowledge domain at hand. Because the word 'ontology' is a bit intimidating, a better variant is the *knowledge graph* (because all semantic ontologies take the structural form of a graph). In general knowledge domains, we call such schemata upper ontologies. However, one of the first things we see with existing ontologies is that they are organized around a single, dyadic dimension, even though guided by a diversity of conceptual approaches. For example, in the venerable Cyc knowledge structure, one of the major divisions is between what is tangible and what is intangible. In BFO, the Basic Formal Ontology, the split is between a 'snapshot' view of the world (*continuant*) and its entities versus a 'spanning' view that is explicit about changes in things over time (*occurrent*). Other upper ontologies have dyadic splits such as abstract *v.* physical, perduant *v.* endurant, dependent *v.* independent, particulars *v.* universals, or determinate *v.* indeterminate [1, 2].

Except for Sowa's ontology [2], none of the standard upper ontologies embrace any semblance of Peirce's triadic perspective. Further, even Sowa's ontology only partially applies Peircean principles [3]. Such Cartesian dichotomies become the basis for arguments between their proponents. Moreover, a Cartesian and nominalistic view is precisely what is wrong with these viewpoints. Our states and phenomena are not on and off, but are probable or graded, likely or nuanced, or often shaded. Due to Cartesian thinking, we do not question why we continually apply a dichotomous schema to real-world phenomena. Knowledge, Peirce tells us, is a Thirdness, and therefore has context and perspective, continuity and generality.[5] Peirce was not so much a superhuman of intellect, but more that he rooted out what we need to question in our premises, using sound logic to tease out insight and make questions simpler.

We design ontologies for specific purposes, and the bases for these splits in other ontologies have their rationales and uses. Where the design objective for the ontology is *knowledge representation*, as it is here, we need to model the nature of knowledge explicitly. Knowledge, too, is not black and white, nor is it shades of gray along a single dimension or lacking color. Knowledge is an incredibly rich construct

[5] "Facts" are in Secondness, but knowledge is a function of belief, which is inherently a Thirdness.

intimately related to context and perspective, with various degrees of vibrancy and nuance. The minimum cardinality that can provide such a perspective is three.

Being Attuned to Nature

The fierce realism that Peirce adopted and advocated, strongest in his later years, was premised on a belief of natural evolution and how its tendencies express themselves in nature. He mostly thought and wrote regarding the symbolic world, but he knew that was a continuation of the life and matter that precedes it. Nathan Houser, the profound Peirce researcher and keeper of the flame for many years, astutely observed:

> He [Peirce] had come to believe that attunement to nature was the key to the advancement of knowledge—as it was for life itself—and he thought that: the power to guess nature's ways was one of the great wonders of the cosmos. Just as with animals, whose instinct enables them to 'rise far above the general level of their intelligence' in performing their proper functions, so it is with humans, whose proper function, Peirce insisted, is to embody general ideas in art-creations, in utilities, and above all in theoretical cognition. But if attunement to nature is the key to the advancement of knowledge, it is at most a necessary condition; it puts thought on the scent of truth, which, to attain, must be won by skilled reasoning [4].

The importance of studying Peirce is to tease out those principles, design bases, and mindsets that can apply Peircean thinking to the modern challenge of knowledge representation. This knowledge representation is like Peirce's categorization of science or signs but is broader still in needing to capture the nature of relations and attributes and how they become building blocks to predicates and assertions. In turn, these constructs should be systematized and subjected to logical tests to provide a defensible basis for what is knowledge and truth given current information. Then, all of these representations should be put forward in a manner (symbolic representation) that is machine readable and computable.

Peirce had insights and guidance on every single aspect of these broader KR problems. The objective has been how to take these piece parts and recombine them into a coherent whole that is consistent with Peirce's architectonic [5]. How can Peirce's thinking be decomposed into its most primitive assumptions to build up a new KR representation? Knowledge representation by computers that does not explicitly account for perspective, meaning, and interpretation is doomed as wooden and unable to handle context. We do not all need to agree on the specifics or any single interpretation of what our domains of inquiry may be. However, we do need a framework that can respect and model those differences. One of Peirce's most famous admonitions is 'there follows one corollary which itself deserves to be inscribed upon every wall of the city of philosophy: Do not block the way of inquiry' (1898, CP 1.135). Knowledge representations based on dichotomous choices do just that.

Firstness, Secondness, Thirdness

> A very moderate exercise of this third faculty suffices to show us that the word Category bears substantially the same meaning with all philosophers. For Aristotle, for Kant, and for Hegel, a category is an element of phenomena of the first rank of generality. It naturally follows that the categories are few in number, just as the chemical elements are. The business of phenomenology is to draw up a catalogue of categories and prove its sufficiency and freedom from redundancies, to make out the characteristics of each category, and to show the relations of each to the others. (1903, EP 2:148)

Scholars of Peirce acknowledge how infused his writings on logic, semiosis, philosophy, and knowledge are with the idea of 'threes.' His insights are perhaps most studied regarding his semiosis of signs,[6] with the triad formed by the object, representation, and interpretation. Peirce studied and wrote on what makes 'threes' essential and irreducible. His generalization, or abstraction if you will, he called simply the *'universal categories*,' and, to reflect their fundamental nature, called each separately as *Firstness*, *Secondness,* and *Thirdness*. In his writings over decades, he related or described this trichotomy in dozens of contexts.[7] We have adopted this naming, so also call the triad of the three categories of Firstness, Secondness, and Thirdness the *universal categories*.[8]

Constant Themes of Three

Trichotomies and triads permeate Peirce's theories and writings in logic, realism, categories, cosmology, and metaphysics [7, 8]. He termed this tendency and its application in general as first, second, and third. In Peirce's words:

> The first is that whose being is simply in itself, not referring to anything nor lying behind anything. The second is that which is what it is by force of something to which it is second. The third is that which is what it is owing to things between which it mediates and which it brings into relation to each other. (1897, CP 2.356)

Peirce's fascination with threes is not unique. Scholastic philosophers, ranging from Duns Scotus and the Modists from medieval times to John Locke and Immanuel Kant with his three formulations, and Hegel with his triad, expressed much of their thinking in threes. As Locke wrote in 1690 [9]:

> The ideas that make up our complex ones of corporeal substances are of three sorts. First, the ideas of the primary qualities of things, which are discovered by our senses, and are in them even when we perceive them not; such are the bulk, figure, number, situation, and motion of the parts of bodies which are really in them, whether we take notice of them or no. Secondly, the sensible secondary qualities which, depending on these, are nothing but

[6] See further Chap. 2 and the section *What is Representation?*

[7] See later Table 6.2.

[8] Peirce claimed the term *universal categories* in CP 1.526 (1903). CP 5.41–59 (1903) provides a longer treatment.

> the powers these substances have to produce several ideas in us by our senses; which ideas are not in the things themselves otherwise than as anything is in its cause. Thirdly, the aptness we consider in any substance to give or receive such alteration of primary qualities, as that the substance, so altered should produce in us different ideas from what it did before.

Summary of the Universal Categories

The first hurdle, I think, in attempting to understand Peirce's universal categories is the absolute abstractness of the terms Firstness, Secondness, and Thirdness. In this case, I believe that Peirce's terminology fussiness is proper. Since, ultimately, according to Peirce, all reality, all potential, and all emergence derive from these elements, nothing other than one, two, and three will do. Everything that is, may be, or could surprise us arises from these elements. Nothing further can be decomposed from these elements, yet everything that is and is conceivable is built from these categories.

Across his voluminous writings, summarized across the listings in Table 6.2, I glean this summary understanding of Peirce's three categories from the standpoint of knowledge representation:

- *Firstness* [1ns]—These are *possibilities*, a 'state' of experience wholly in the absolute present, which are basic 'monadic' qualities that may combine in various ways to enable the real things we perceive in the world. They are unexpressed potentialities, the substrate of the real and actual. These are the unrealized essences or attributes or possible juxtapositions; indeed, 'these' and 'they' are misnomers because, once conceived, the elements of Firstness are no longer Firstness.[9] In the sense of categorization, think of Firstness as the universe of ideas or possibilities that might be brought to bear for the new category of inquiry.
- *Secondness* [2ns]—These are the *particular realized things*, *events*, *or concepts* in the world, what we can perceive, point to, and describe (including the idea of Firstness and Thirdness). All *particulars* are in Secondness and may be an entity, event, instance, or individual. In the sense of categorization, we can understand Secondness as the particular instances that may populate the information space for the category, including the ideas of attributes and relations.
- *Thirdness* [3ns]—These are the *laws, habits, thoughts, regularities,* or *continuities* that may be generalized from particulars. All *generals*—what are also known as classes, kinds, or types—belong to this category, as do all regularities,

[9] One of the interesting things I have come to understand about Firstness (1ns) is that it ceases to be 1ns once it is expressed. Firstness does not exist; it is the absolute present, from which we understand that the past is derived. In the absolute present everything is possible; what we remember of that moment is what actually existed. I use the term "reification" to denote when we try to express an unexpressed possibility. One fascinating aspect is how this parallels quantum theory and Schrödinger's cat. Once we try to describe or measure or understand something, we change its character, and never truly grok its essence.

patterns, or logical groupings, or any combinations thereof. Changes in Firstness or Secondness are reasoned over in Thirdness, beginning the process anew. The method of finding and deriving these generalities may also lead to new insights or emergent properties, which, combined with absolute chance, are the source of what Peirce called the 'surprising fact.'

We can summarize Peirce's universal categories like this (Table 6.1):

Table 6.1 Peirce's universal categories[a] [7]

Name	Characterization	What	Quantity	How defined	Valence
Firstness	Quality of feeling	Ideas, chance, possibility	Vagueness, 'some'	Reference to a ground (pure abstraction of a quality)	Monadic
Secondness	Reaction, resistance, relation	Entities, events, brute facts, actuality	Singularity, discreteness, 'this'	Reference to a correlate (by its relate)	Dyadic
Thirdness	Representation, mediation	Signs, habits, laws, necessity	Generality, continuity, 'all'	Reference to an *interpretant*	Triadic

[a]Also called by Peirce the *Ceno-Pythagorean* categories (*c.f.*, CP 2.87, 8.328)

Understanding, inquiry, and knowledge require this irreducible structure; connections, meaning, and communication depend on all three components, standing in relation to one another and subject to interpretation by multiple agents in multiple ways (Peirce's semiosis of signs). Contrast this Peircean view with traditional classification schemes, which have a dyadic or dichotomous nature and do not support such rich views of context and interpretation.[10]

Once the basic structure of the trichotomy and the nature of its primitives were in place, it was logical for Peirce to generalize the design across many other areas of investigation and research. Because of the signs' groundings in logic, Peirce's three main forms of deductive, inductive, and abductive logic also flow from the same approach and mindset. How to think about categorization was another contribution.[11] Using his terminology of the general triad, Peirce writes when the First and Second:

> … are found inadequate, the third is the conception which is then called for. The third is that which bridges over the chasm between the absolute first and last, and brings them into relationship. We are told that every science has its qualitative and its quantitative stage; now its qualitative stage is when dual distinctions—whether a given subject has a given predicate or not—suffice; the quantitative stage comes when, no longer content with such rough

[10] Michael Buckland points out that some systematic library classifications do include means for including contextual and relational aspects, such as facets or geographical, chronological, or other contexts in the Universal Decimal Classification (among others). However, I maintain that these mechanisms appear more as associations or contextual attributes rather than being an integral part of the ontological structure.

[11] See the discussion on *prescission* in Chap. 7.

> distinctions, we require to insert a possible halfway between every two possible conditions of the subject in regard to its possession of the quality indicated by the predicate. Ancient mechanics recognized forces as causes which produced motions as their immediate effects, looking no further than the essentially dual relation of cause and effect. That was why it could make no progress with dynamics. The work of Galileo and his successors lay in showing that forces are accelerations by which [a] state of velocity is gradually brought about. The words 'cause' and 'effect' still linger, but the old conceptions have been dropped from mechanical philosophy; for the fact now known is that in certain relative positions bodies undergo certain accelerations. Now an acceleration, instead of being like a velocity a relation between two successive positions, is a relation between three …. we may go so far as to say that all the great steps in the method of science in every department have consisted in bringing into relation cases previously discrete. (1888, CP 1.359)

Continuity is an aspect of Thirdness, what Peirce called *synechism*, and discovery of new knowledge is itself a process. We may better understand concepts like space and time when we embed them in the idea of continuity. Actions may also express triadic relations, the classic example being 'A gives B to C' (1903, EP 2 170–171). The other classic triadic example is Peirce's sign relation between object, sign, and interpretant. The brilliance of Peirce's mindset is that first, second, and third are a sufficient basis to bootstrap how to represent the world.

The Irreducible Triad

Peirce saw the trichotomous parts of his sign logic as the fewest 'decomposable' needed to model the real world. Robert Burch has called Peirce's ideas of 'indecomposability' the 'reduction thesis' [8]. The thesis is ternary relations suffice to construct any arbitrary relation, but we cannot construct all relations from unary and binary relations alone. Threes are irreducible to capture the basis of knowledge. Peirce did not provide a formal proof for his assertions; there was not yet a complete formalism for predicate calculus at his disposal [10]. Here are some of Peirce's thoughts as to what makes something 'indecomposable':

> I will sketch a proof that the idea of meaning is irreducible to those of quality and reaction. It depends on two main premisses. The first is that every genuine triadic relation involves meaning, as meaning is obviously a triadic relation. The second is that a triadic relation is inexpressible by means of dyadic relations alone …. every triadic relation involves meaning. (1875, CP 1.345)

> And analysis will show that every relation which is tetradic, pentadic, or of any greater number of correlates is nothing but a compound of triadic relations. It is therefore not surprising to find that beyond the three elements of Firstness, Secondness, and Thirdness, there is nothing else to be found in the phenomenon. (1875, CP 1.347)

> It is a priori impossible that there should be an indecomposable element which is what it is relatively to a second, a third, and a fourth. The obvious reason is that that which combines two will by repetition combine any number. Nothing could be simpler; nothing in philosophy is more important. (1905, CP 1.298)

> We find then a priori that there are three categories of undecomposable elements to be expected in the phaneron: those which are simply positive totals, those which involve dependence but not combination, those which involve combination. (1905, CP 1.299)

Peirce thus maintained that we could decompose all higher order relationships (polyadic with more than three terms) to monadic, dyadic, or triadic relations. Further, Peirce maintained that the triadic relation is primary, with monadic and dyadic relations being degenerate forms of it. An interesting aspect of Peirce's Thirdness is how to treat relations between Firstness, Secondness, and Thirdness. Because of the sort of building block nature inherent in a sign, we cannot treat all potential dyadic relations between the three elements equally. According to the 'qualification rule,' 'a First can be qualified only by a first; a Second can be qualified by a First and a Second; and a Third can be qualified by a First, Second, and a Third' [11]. Note that a Third cannot be involved in either a First or Second.[12]

Researchers have now formally proven these assertions by Peirce. Herzberger [12] and then Burch [8] were the first independent researchers to establish the irreducibility of the basic relations of threes in a constrained form, but this was later more broadly proven using Peirce's existential graphs in two different papers by Correira and Pöschel [13] and then Hereth and Pöschel [14].

The Lens of the Universal Categories

Still, the question remains: How can one apply Peirce and his ideas to today's challenges in knowledge representation? What is the essence of trying to approach and solve problems by Peircean means? Is there a lens through which we can think through contemporary problems in domains unheard of in Peirce's time?

One approach taken by scholars is to attempt to complete Peirce's sign classification system. As noted in Chap. 2, Peirce expanded his original three universal categories to 6 (3, plus the degenerate form of Secondness and the 2 degenerate forms of Thirdness[13]); then to 10 in the fuller explication of the sign (see Table 2.1); and then to incomplete 28- and 66-sign versions toward the end of his career. Researchers such as Borges [15], Burch [16], Faria et al. [17], and Jappy [18] have attempted to 'sign-trace' these late, incomplete versions. These are laudable attempts, and often creative and insightful. However, these later Peirce sign systems are incomplete, require filling in the blanks for what Peirce intended, and are not directly relevant to modeling knowledge representation. My first attempts at using Peirce for KR tried to follow this same path, but I abandoned it as being too removed and speculative.

---

[12] See the related discussion under the last section on "Representation" in Chap. 2.

[13] Peirce expanded his original three universal categories to six by adding what he called one "degenerate" form to Secondness and two "degenerate" forms to Thirdness, increasing the original three by an additional three. See further CP 1.365–367 (1887–88). I discuss later sign expansions in the main text.

An Aha! Moment

I was first attracted to Peirce's universal categories because of my interest in representing human language and its meaning. Only through context and perspective—Thirdness—may we hope to capture and understand the nuances of meaning. When I first saw this strength in Peirce's worldview, that (and his writings) led me to look at its applicability elsewhere. My *Aha!* moment, if I can elevate it as such, was when I realized that trying to cram these insights into Peirce's elaborate sign terminology and other literal aspects of his writing was, at least for me, self-defeating. The *Aha!* arose when I chose instead to understand the mindset underlying Peirce's thinking and the triadic nature of his universal categories and semiosis.

I find it amazing and consistent how much Peirce himself relies on the universal categories in his thinking and analysis. His method of thinking through to foundations, *prescission*,[14] is invaluable in deciding edge cases for categorization. I believe he applied this approach, for example, to his later sign expansions. There must be something at the heart of these universal categories that make them such a powerful lodestone. The very generalizations Peirce made around the somewhat amorphous designations of Firstness, Secondness, and Thirdness seemed to affirm that what he was genuinely getting at was a way of thinking, a method of 'decomposing' the world, that had universal applicability irrespective of domain or problem. Thus, to make my *Aha!* moment useful, I needed to understand the essence of what lies behind Peirce's universal categories.

Not only at the most fundamental level but also at almost all levels of understanding and logic, Peirce articulated a worldview built around these universal categories. Peirce uses this triadic structure to describe language, signs, logic, relations, growth, emergence, science, truth, limits, meaning, community, categorization, and consensus-building. Though Peirce acknowledges natural classification systems, such as trees of life and dichotomous keys in taxonomy, in most areas of ideas and concepts and metaphysics, he boils down his arguments into the three universal categories. As noted, he argues that each alone is necessary, each is irreducible, and all three are required to adequately represent any information space, which is, after all, a sign.

Peirce's triadic approach to logic is especially informative. The first leg of the logic triad is *speculative grammar*, in which one strives to capture the signs that most meaningfully and naturally describe the current and potential domain of discourse. The second leg of the logic triad is the means of *logical inference*, be it deductive, inductive, or abductive (hypothesis generating). The third leg is the process or *method of inquiry*, what Peirce most often called the *methodeutic*. The methods of research or science, including the scientific method, result from the application of this logic.[15] The 'pragmatic' part of Peirce's pragmaticism arises from how to

[14] See Chap. 7.

[15] Realize that, by expressing the triad of logic as *speculative grammar*, *critic* (the three modes of inference), and *methodeutic*, we have already progressed through many cycles of semiosis in order to use symbols, embed them in a meta-symbology of language, have those symbols express accepted meanings, and then be applied to the question of categorizing the logic domain. This recursive and accretive nature of the universal categories demonstrates the innate power of Firstness, Secondness, and Thirdness.

select what is essential and economically viable to investigate among multiple hypotheses.

Though scholars widely discuss Peirce's universal categories, most Peircean research focuses on signs, a subset of the categories. Signs are more often the prism by which scholars probe Peirce's philosophy. My approach, instead, has been to broaden my perspective to the universal categories and then to use Peirce's methods to explore them. I have hoped to discern the mindset underlying them, which I could then apply to the contemporary challenges of knowledge representation.

Grokking the Universal Categories

Peirce expressed his notions of the universal categories in many different ways and contexts. Peirce's students have further interpreted these notions. To get at the purpose of the triadic concepts, I thought it useful to research the question in the same way that Peirce recommends. After all, Firstness, Secondness, and Thirdness should themselves be prototypes for what Peirce called the 'natural classes.'[16]

I have assembled from Peirce's writings as many examples of the three members of the universal categories as I could find. This assemblage is 'an enumeration of tests' to use Peirce's phrase. Table 6.2 lists these more than 60 examples of Firstness, Secondness, and Thirdness, the contexts in which they arose, and a citation where to find the supporting material in Peirce's writings. I use initial caps for all assignments to the universal categories to put the listings on a common footing, though Peirce often capitalized his terms. Please do not confuse the three modes of the universal categories with the three entries in an RDF triple (see Chaps. 1 and 8):

Table 6.2 Peirce's universal categories in relation to various topics [6]

Context	Firstness	Secondness	Thirdness	Note[a]
Moods or tones	First	Second	Third	T1
Conceptions of first, second, third	Independent	Relative	Ediating	T2
The categories	Monads	Particulars	Generals	T3
Time	'Present'	'Past'	'Future'	T4
Cognition/ space	Point	Line	Triangle/sphere	T5
Movement	Position	Velocity	Acceleration	T6
Modes of being	Possibility	Existence	Law	T7
Seconds	Internal	External	–	T8
Thirds	Mixtures	Comparisons	Intelligibles	T9

(continued)

[16] See further Chaps. 6 and 12.

Table 6.2 (continued)

Context	Firstness	Secondness	Thirdness	Note[a]
Modality	Possibility	Actuality	Necessity	T10
Phenomena 1	Sensations	Reactions	Generals	T11
Phenomena 2	Qualities of phenomena	Actual facts	Laws (and thoughts)	T12
Phenomena 3	Chance	Existents	Continuity	T13
Active elements	Chance	Law	Habit-taking	T14
Realism	Form	Matter	Entelechy	T15
Existence	Chaos	Regularity	Continuity	T16
Continuity	Feeling	Effort	Habit	T17
Mathematics	Quality	Facts	Laws	T18
Ceno-Pythagorean categories	Originality	Obsistence	Transuasion	T19
Form	Tone	Token	Type	T20
Being	Quality	Relation	Representation	T21
Protoplasm	Sensibility	Motion	Growth	T22
Natural selection	Individual variation	Heritability	Elimination of unfavored characters	T23
Modes of evolution	Absolute chance	Mechanical necessity	Law of love	T24
Doctrines of evolution	Tychasticism	Anancasticism	Agapasticism	T25
Consciousness 1	Feeling	Sense of action/ reaction	Sense of learning	T26
Consciousness 2	Feeling	Altersense	Medisense	T27
Consciousness 3	Immediate feeling	Polar sense	Synthetical consciousness	T28
Thought 1	Abstraction	Suggestion	Association	T29
Thought 2	Possibility	Information	Cognition	T30
Thought 3	Thought-sign	Connected	Interpreted	T31
Synthetical consciousness	Association by contiguity	Association by resemblance	Intelligibility	T32
Mind	Feelings	Reaction-sensations	Conceptions	T33
Logical mind	Ideas	Ideas from prior ideas	Ideas from prior processes	T34
Experiences	Simples	Recurrences	Comprehensions	T35
Universe of experiences	Ideas	Brute activity	Sign	T36
Information	Intensions	Extensions	Comprehensions	T37
Knowledge representation	Attributes	Individuals	Types	T38

(continued)

Table 6.2 (continued)

Context	Firstness	Secondness	Thirdness	Note[a]
Characters or predicates	Internal	External	Conceptual	T39
Relations	Attributes	External relations	Representations	T40
Representation	Representamen	Object	Interpretant	T41
Sign-object	Icon	Index	Symbol	T42
Nature of signs	Qualisign	Sinsign	Legisign	T43
Kinds of characters	Singular characters	Dual characters	Plural characters	T44
Symbols	words (or terms)	propositions	arguments	T45
Sign-interpretant 1	Emotional interpretant	Energetic interpretant	Logical interpretant	T46
Sign-interpretant 2	Rhemes	Dicisigns	Arguments	T47
Signs 1	Possibles	Things	Collections	T48
Signs 2	Abstractives	Concretetives	Collectives	T49
Propositions	Hypothetical	Categorical	Relative	T50
Logical terms	Monads	Dyads	Triads	T51
Separability of ideas	Dissociation	Prescission	Determination	T52
Assertions	Possible modality	Actual modality	Necessary modality	T53
Reasoning	What is possible	What is actual	What is necessary	T54
Logical thinking	Clearness of conceptions	Clearness of distinctions	Clearness of practical implications	T55
Clarity	Doubt	Inquiry	Belief	T56
Logic methods	Abductions	Deductions	Inductions	T57
Logic	Speculative grammar	Logic and classified arguments	Methods of truth-seeking	T58
Sciences of discovery	Mathematics	Philosophy	Special sciences	T59
Philosophy	Phenomenology	Normative science	Metaphysics	T60
Normative science	Esthetics	Ethics	Logic	T61
Concepts of metaphysics	Spontaneity	Dependence	Mediation	T62
Others	Complete in itself, freedom, free, measureless, variety, freshness, multiplicity, manifold of sense, peculiar, idiosyncratic, suchness, one, new, spontaneous, vivid, *sui generis*	Otherness, comparison, action, dichotomies, mutual action, will, volition, involuntary attention, shock, sense of change, here and now (*hinc et nunc*), compulsion, state, occurrence, negation	Idea of composition, intelligence, moderation, comparative, reason, sympathy, intelligence, structure, regularities, conduct, representation, middle, learning, conditional, diffusion	T63

[a]See Footnote 18

The table spans from the potential or abstract, such as 'first' or 'third,' to whole realms of science or logic. This spanning of scope reflects the genius of Peirce's insight wherein semiosis can begin literally at the cusp of Nothingness[17] and then proceed to capture the process of sign-making, language, logic, scientific method, and thought abstraction to embrace the broadest and most complex of topics.[18] I also

[17] The idea of Firstness may range from something like an energetic input that causes chemicals to combine into a new structured form or ordered state to something like a new recognition in the mind occasioned by a flick of the eye or a shifting thought. The representamen is merely a potential sign until it is energized or intrudes on consciousness, wherein the object is now made apparent as interpreted.

[18] Here are the references to the table: T1—CP 1.355; also, Cosmogenic Philosophy, EP 1.297; T2—See CP 6.32–34; T3—This exact categorization was never used directly by Peirce (or so my investigations to date suggest). However, it is clear throughout his writings that he relates monads to Firstness, 'particulars' and 'particularities' to Secondness, and 'generals' or 'generalities' to Thirdness. Further, these terms are understood and used in other categorization schemes, such as those by Aristotle and Kant. We also see, by this chart, that Peirce himself employs many different terms for his universal categories. We have chosen these to be the three main categories in the KBpedia Knowledge Ontology for these reasons. See further CP 1.300–338; T4—CP 2.84–86; see also 2.146; it is NOT 1 → 2 → 3 present v hic et nunc; CP 5.459–463; T5—CP 5.263; T6—CP 1.337; T7—CP 6.343–344; T8—CP 1.365; these are the two degenerate forms; there is no Thirdness; T9—CP 1.366; This is an example of what Peirce called "degenerate" categories of the category. Degenerate means that it is a component of the category, but not sufficient as a concept in the 1o and 2o; T10—CP 5.454; T11—CP 1.418–420; T12—CP 5.121; T13—Per the discussion in *Appendix A*; T14—CP 1.409; T15—NEM 4:295; T16—CP 1.411 and CP 1.175; T17—CP 6.201–202; also called Tritism or Synechism (or "all that there is"); T18—CP 1.417–420; T19—CP 2.87–89; Peirce using his obscure labels in seeking exactitude; T20—CP 4.537; T21—CP 1.555 and CP 2.418; the initial categories were actually bracketed by Being and Substance (five categories total). In CP 4.3 Peirce renamed these labels as *quality*, *reaction*, and *mediation*. However, in that same passage he says, "How the conceptions are named makes, however, little difference." I have chosen to retain his earlier names because they are more commonly referenced and it retains the idea of "representation," more allied with the idea of *knowledge representation*; T22—CP 1.393; T23—CP 1.398; T24—CP 6.302; T25—CP 6.302; T26—CP 1.378; T27—CP 7.551; thought is taken to be as equivalent to medisense; T28—EP 1.260; T29—The analysis of the labels and relations is provided in these two articles: M.K. Bergman, 2017. "KBpedia Relations, Part III: A Three-Relations Model," AI3:::Adaptive Information blog, May 24, 2017; and M.K. Bergman, 2017. "KBpedia Relations, Part IV: The Detailed Relations Hierarchy," AI3:::Adaptive Information blog, June 27, 2017; T30—CP 1.537; T31—CP 5.283–284; T32—EP 1.261; T33—CP 6.18–20; T34—CP 7.348; T35—CP 7.528 cf; T36—CP 6.455; T37—Peirce did not explicitly list these terms, but they can be readily and logically derived from CP 2.419–421. The idea of information being a product of depth (1^o, intensionality) times breadth (2^o, extensionality) is quite insightful; T38—Though "general type" is a common term for Thirdness in Peirce's writings, he rarely used "attribute" and preferred particulars to "individuals." "Attributes" and "individuals" are now in modern usage, and clearly refer to 1ns and 2ns, respectively. We have chosen these two terms for use in the KBpedia Knowledge Ontology for these reasons; T39—Somewhat modified from CP 5.469 cf, with external and conceptual replacements supported by the senses of the accompanying text; T40—Taken from the analysis of Peirce documented in T50; these are the terms chosen for use in terms for use in the KBpedia Knowledge Ontology; T41—CP 1.339; "representation" is also called a "sign"; T42—CP 1.191; can also be called "speculative grammar" or "nature of signs"; in Jappy 2017 this is called "Sign-Object"; Table 1.2 A Synthesis of MSS R478 and R540, 1903; T43—CP 4.537 fn 3; called simply "Sign" in Jappy 2017; Table 1.2 A Synthesis of MSS R478 and R540, 1903; T44—CP 1.370–371; can substitute "facts" for "char-

find this statement by Peirce as another powerful expression of the universal categories: 'The starting-point of the universe, God the Creator, is the Absolute First; the terminus of the universe, God completely revealed, is the Absolute Second; every state of the universe at a measurable point of time is the third' (1888, CP 1.362).

Because I have taken these examples from many contexts, it is important to review this table on a row-by-row basis when investigating the nature of the categories. Review of the columns helps elucidate the 'natural classes' of Firstness, Secondness, and Thirdness. Some items appear in more than one column, reflecting the natural process of semiosis wherein more basic concepts cascade to the next focus of semiotic attention. The last row is a kind of catch-all trying to capture other mentions of the universal categories in Peirce's phenomenology.

It took me a while to realize that Firstness, Secondness, and Thirdness are not a linear sequence, nor one in time. In fact, Peirce likens Firstness to the present, Secondness to the past, and Thirdness to the future (not in a predictive sense, but as probabilities continuing from the past).[19] All possibilities, Firstness, reside in the absolute present, 'for nothing is more occult' (1902, CP 2.85). The instant at which these possibilities act or are acted upon causes them to come into existence, or Secondness. These instances exist in relation or contrast with other instances and events because what is real is past. The continuity of these instances through space and time, the probable future, enables new generalities arising from what we can learn from Secondness and Firstness, as well as to anticipate or plan. Chances or accidents in Firstness may spring 'surprises' in Secondness that trigger new cognition or mediation in Thirdness, which potentially predicates a new basis for categorization, in the sense of knowledge representation, our chosen frame of reference.

My thesis is that studying these assignments for the various contexts shown in Table 6.2 is one way to internalize the mindset of the universal categories. At the most fundamental level, we can see Firstness as the raw, unexpressed possibilities

---

acters"; T45—CP 2.95, also CP 8.337; CSP also expresses "arguments" as inferences or syllogisms; T46—CP 5.475–6; T47—From Jappy 2017; Table 1.2 A Synthesis of MSS R478 and R540, 1903; T48—CP 8.366, with respect to the nature of dynamical objects; T49—CP 8.366, with respect to the nature of dynamical objects; T50—CP 2.325; T51—CP 1.293; T52—CP 1.353; T53—CP 4.57; T54—CP 1.369; T55—CP 3.457; T56—From Max H. Fisch 1986, Ken L. Ketner and Christian W. Cloesel, eds., *Peirce, Semiotic, and Pragmatism: Essays by Max H. Fisch*, Indiana University Press, p. 327; T57—CP 2.98; in an earlier version, I listed "abduction" as a Thirdness, but I was corrected on the Peirce-L mailing list. On the other hand, abduction is at the interface between Thirdness and Firstness, since it is the source of the possibilities that need to be considered for a given category. The dynamic nature of Peirce's semiosis is part of the sign-making and -recognition process; T58—CP 1.191; T59—CP 1.239–242; the "special sciences" include the physical (physics, chemistry, biology, astronomy, geognosy, and whatever may be like these sciences) and the psychical (psychology, linguistics, ethnology, sociology, history, etc.) sciences; T60—CP 1.280–282; T61—CP 1.281; T62—CP 3.422; also, forms of rhemata (singular, dual, or plural); T63—occasional mentions taken from various Peirce writings.

[19] Edwina Taborsky prefers to define Thirdness in this context as "past-future." Thirdness is a continuity of past laws into the future. In her 2006 paper, 'The Nature of the Sign as a WFF—A Well-Formed Formula' (*AIP Conference Proceedings*, vol. 839, pp. 303–313), the three types of time are present, perfect, and progressive, aligned with Firstness, Secondness, and Thirdness.

of the current problem set, the building blocks for the new category, if you will. Chance is the root aspect of Firstness, which means any of these possibilities may express themselves in surprising ways, perhaps causing the need for new categorization. The actual things or events of the new category, as made manifest by their interaction or contact with what also exists in the domain at hand, provide the actual instances of Secondness. The generalities or continuities among these instances, classed into natural types as best we can, provide the Thirdness of this domain. We find much to plumb in Peirce's universal categories.

Applying the Universal Categories

The lens of the universal categories provides a framework for how we may organize and settle upon terminology for existing and emerging knowledge, the first task of a knowledge representation system. Peirce, the logical categorizer, concerned with methods, and interested in pragmatic approaches and solutions, understood that how we categorize our continually emerging world was fundamental.

We see that the categorization effort may arise from one of the three sources. Either we are trying to organize a knowledge domain anew; we are splitting an existing category that has become too crowded and difficult to reason over; or we have found a 'surprising fact,' which is new knowledge that emerges from chance or anomalies observed when attempting to generalize or to form habits. The occasional surprising fact alters what we think we know about reality, which causes us to reinspect and recategorize our world. Abductive reasoning, a Peirce contribution, attempts to probe why the anomaly occurs. The possible hypotheses so formed constitute the Firstness or potentials of the new categorization (identification of particulars and generalization of the phenomena). We scope the category based on the domain and the granularity of the categorization effort.

I think it is evident in Table 6.2, sometimes to multiple levels depending on the context (study some of the supporting material to the table), that Peirce applied this same method. As Peirce instructs, the dynamic universal categories, faced with the unexpected chance arising in Firstness, ripple through our awareness (reality) to cause a new understanding of the state of existence (Secondness). The universal categories give us the primitive elements by which we can generalize our new world, a factor of Thirdness. Peirce's pragmatic maxim helps us decide among many possible alternatives. So the cycle continues. Truth, understood as a limit function, gets continuously exposed as we test and affirm these realities.

Start with any subject domain. We know the things, and therefore the characteristics, of the things that populate this domain. So, we first spend time enumerating and describing the features of the things in this domain. We will call this category of characteristics, Firstness. Then, we try to list and organize the actual things in this domain. These, individually, are the events and entities that we can imagine or specify about this domain. This list of particulars, what we call Secondness, is surely always going to grow, so from an operational viewpoint we want input files for these

items that are easy to update and modify. The items in our domain also have generalities and shared aspects that help place those items into meaningful categories. These groupings, admittedly synthetic in one sense, are also real in another sense when the groupings make logical sense. These generalities are an expression of Thirdness. This categorization into Thirdness is straightforward to do on purely logical grounds but is more difficult when we desire explanatory power. Where questions arise about which universal category to assign something, we look to Peirce and later scholars to see if prior determinations have been postulated and argued. If so, we test those assumptions and adopt or not those assignments, based on our logical assessments. We continue this process as we get deeper and more specific in our categorizations. No matter what the assignment, each should be subject to questioning and testing by the community of users, perhaps altering those assignments as better information or better logic is applied.

This process is the one that we followed in developing the open-source KBpedia Knowledge Ontology (KKO), the knowledge graph of some 200 concepts that provides the upper-level scaffolding for our knowledge representation efforts. KKO is the first knowledge graph to embrace the universal categories explicitly. We will get into specifics about KKO in later chapters.

I earlier mentioned my epiphany from specifics to mindset in Peirce's teachings. This insight has not caused me to suddenly understand everything Peirce was trying to say, nor to come to some new level of consciousness. However, what it has done is to open the door to a new way of how to think about and look at the world. I am now finding via the universal categories that prior, knotty problems of categorization and knowledge representation are becoming (more) tractable, as I discuss in subsequent chapters. Many of these problems, such as how to model events, situations, identity, representation, and continuity or characterization through time, may sound like philosophers' millstones, but they often lie at the heart of the most challenging problems in knowledge modeling and representation. Even the tiniest break in the mental and conceptual logjams around such issues feels like significant progress.

The Categories and Categorization

The area of Secondness is where we surface and describe the particular objects or elements that define this category. Peirce described it thus:

> 'So far Hegel is quite right. But he formulates the general procedure in too narrow a way, making it use no higher method than dilemma, instead of giving it an observational essence. The real formula is this: a conception is framed according to a certain precept, [then] having so obtained it, we proceed to notice features of it which, though necessarily involved in the precept, did not need to be taken into account in order to construct the conception. These features we perceive take radically different shapes; and these shapes, we find, must be particularized, or decided between, before we can gain a more perfect grasp of the original conception. It is thus that thought is urged on in a predestined path. This is the true evolution of thought, of which Hegel's dilemmatic method is only a special character which the evolution is sometimes found to assume.' (1896, CP 1.491)

In Thirdness we are contemplating the category, thinking about it, analyzing it, using and gaining experience with it, such that we can begin to see patterns or laws or 'habits' (as Peirce so famously put it) or new connections and relationships with it. This contemplation or the occasional 'surprising fact' is where new knowledge arises. New knowledge causes us to split and then codify new signs and categories useful to the knowledge space. As domains are investigated to deeper levels or insights expand the branches of the knowledge graph, we tackle each new layer via this threefold investigation. Of course, context sets the perspectives at hand; the multiple listings in Table 6.2 above can help stimulate these thoughts.

Table 6.3 Using the universal categories for categorization

	Firstness	Secondness	Thirdness
Symbols	Idea of; nature of; milieu; 'category potentials'	Reference concepts	Standards
Generality	Cross-products of Firstness	Language (incl. domain); computational	Analysis; representation; continua
Interpreters (human or machine)	What are the ingredients, ideas, and essences of the category?	What are the new things or relationships of the category?	What are the laws, practices, and outputs arising from the category?

Interrelationships adhere to the Peircean Thirdness, and there continues to be growth and additions. Categories thus tend to fill themselves up with more insights and ideas until the scope and diversity compel another categorization. In these ways, categorization is not linear, but accretive and dynamic. Firstness, Secondness, and Thirdness inform how to think about the idea of categorization. I use the kind of mental checklist provided in Table 6.3 when it comes time to split a concept or category into a new categorization.

These Peircean ideas of the universal categories, applied against basic logical principals, and subject to the understanding about fallibility and the limits to truth, provide a basic set of methods of how to think about and categorize the world. When the 'surprising fact' arises that causes us to question premises and regularities, we can apply this same categorization logic to assess the next level of subject specificity. Now, we are in a mediating portion of our information space, likely again requiring new categorization. Peirce's universal categories provide a powerful unifying force for organizing and categorizing knowledge domains.

> Taking any class in whose essential idea the predominant element is Thirdness, or Representation, the self development of that essential idea—which development, let me say, is not to be compassed by any amount of mere 'hard thinking', but only by an elaborate process founded upon experience and reason combined—results in a trichotomy giving rise to three sub-classes, or genera, involving respectively a relatively genuine thirdness, a relatively reactional thirdness or thirdness of the lesser degree of degeneracy, and a relatively qualitative thirdness or thirdness of the last degeneracy. This last may subdivide, and its species may even be governed by the three categories, but it will not subdivide, in the manner which we are considering, by the essential determinations of its conception. The genus

> corresponding to the lesser degree of degeneracy, the reactionally degenerate genus, will subdivide after the manner of the Second category, forming a catena; while the genus of relatively genuine Thirdness will subdivide by Trichotomy just like that from which it resulted. Only as the division proceeds, the subdivisions become harder and harder to discern. (1903, CP 5.72, EP 2:162)

The way I interpret this passage (in part) is that categories in which new ideas or insights have arisen—themselves elements of Thirdness for that category—are targets for new categorization. That new category should focus on the idea or insight gained, such that each new category has a character and scope different from the one that spawned it. Of course, depending on the purpose, some ideas or insights have a more substantial potential effect on the domain, and those should get priority attention. As a practical matter, this means that categories of more potential importance to the sponsor receive the most focus.

Peirce's contributions can make a notable difference in how knowledge representation efforts move forward. I think that it is possible to codify and train others to use this mindset, one purpose of this book. Peirce stood on the shoulders of the giants before him. We can now stand on Peirce's shoulders to mount the next rung on the ladder of knowledge. I believe that Peirce's universal categories and what they imply offer the next adaptive climb upward for knowledge representation.

References

1. N. Guarino, Some organizing principles for a unified top-level ontology, in *AAAI Spring Symposium on Ontological Engineering* (1997), pp. 57–63
2. J.F. Sowa, Signs, processes, and language games: Foundations for ontology, in *Proceedings of the 9th International Conference on Conceptual Structures, ICCS'01* (2001)
3. L. Jansen, Categories: The top-level ontology, in *Applied Ontology: An Introduction*, ed. by K. Munn, B. Smith. (Ontos Verlag, Frankfurt, 2008), pp. 173–196
4. N. Houser, Introduction, *The Essential Peirce: Selected Philosophical Writings, Vol 2 (1893–1913)*, Peirce Edition Project, ed. (Indiana University Press, Bloomington, 1998), pp. xvii–xxxviii
5. A. Atkin, Peirce, Charles Sanders: Architectonic Philosophy, *Internet Encyclopedia of Philosophy*.
6. M.K. Bergman, How I interpret C.S. Peirce, in *AI3:::Adaptive Information* (Sep. 2017)
7. C.S. Peirce, Minute Logic: Chapter I. Intended Characters of this Treatise, *Digital Companion to C.S. Peirce* (1902)
8. R. Burch, *A Peircean Reduction Thesis: The Foundations of Topological Logic* (Texas Tech University Press, Lubbock, TX, 1991)
9. J. Locke, in *An Essay Concerning Human Understanding*, ed. by J. Yolton. (Dutton, New York, 1690)
10. E. Kleinert, On the reducibility of relations: Variations on a theme of Peirce. Trans. Charles S. Peirce Soc. Quart. J. Am. Philos. **43**, 509–520 (2007)
11. D. Savan, An Introduction to C.S. Peirce's Full System of Semeiotic, *Monograph Series of the Toronto Semiotic Circle* (1987)
12. H.G. Herzberger, Peirce's remarkable theorem, in *Pragmatism and Purpose: Essays Presented to Thomas A. Goudge*, ed. by I. W. Sumner, J. G. Slater, F. Wilson (University of Toronto Press, Toronto, Canada, 1981), pp. 41–58

13. J.H. Correia, R. Pöschel, The teridentity and Peircean algebraic logic, in *Conceptual Structures: Inspiration and Application*, ed. by H. Schärfe, P. Hitzler, P. Øhrstrøm (Springer, Aalborg, Denmark, 2006), pp. 229–246
14. J. Hereth, R. Pöschel, Peircean algebraic logic and Peirce's reduction thesis. Semiotica **186**, 141–167 (2011)
15. P. Borges, A Visual Model of Peirce's 66 Classes of Signs Unravels His Late Proposal of Enlarging Semiotic Theory (2010), pp. 221–237
16. R.W. Burch, Peirce's 10, 28, and 66 sign-types: The simplest mathematics. Semiotica **184**, 93–98 (2011)
17. P. Farias, J. Queiroz, On diagrams for Peirce's 10, 28, and 66 classes of signs. Semiotica **147**, 165–184 (2003)
18. T. Jappy, *Peirce's Twenty-Eight Classes of Signs and the Philosophy of Representation: Rhetoric, Interpretation and Hexadic Semiosis* (Academic, Bloomsbury, 2017)

Chapter 7
A KR Terminology

S*peculative grammar*, the theory of the nature and meaning of signs, is the first of the three branches of logic according to Peirce. The basic idea of a speculative grammar is simple when applied to a new concept or domain, such as knowledge representation. What is the terminology—vocabulary and relations—that may be involved in understanding the questions or concepts at hand? What is the 'grammar' for relating this terminology to the logics we need to help increase our understanding of the domain? How shall we split and organize these concepts? That is, how shall we categorize our domain? How do we combine these elements into assertions and statements and then test them for truth and accuracy? Through this grammar, in the KR context, can we maximize the structural features within our focus of inquiry useful to machine learners?

The term 'semiosis' most often brings to mind Peirce's theory of signs.[1] However, for Peirce semiosis was a broader construct still, representing his overall theory of logic and truth-testing. Signs, symbols, and representation are the first part of this theory, the speculative grammar about how to formulate and analyze logic. Though he provides a unique take on it, Peirce's idea of speculative grammar, which he ascribed to Duns Scotus, perhaps should be traced back to the 1300s and the writings of Thomas of Erfurt, one of the so-called Modists of the medieval philosophers [1]. Here is how Peirce placed speculative grammar within his theory of logic:

> All thought being performed by means of signs, logic may be regarded as the science of the general laws of signs. It has three branches: (1) *Speculative Grammar*, or the general theory of the nature and meanings of signs, whether they be icons, indices, or symbols; (2) *Critic*, which classifies arguments and determines the validity and degree of force of each kind; (3) *Methodeutic*, which studies the methods that ought to be pursued in the investigation, in the exposition, and in the application of truth. (1903, CP 1.191, EP 2:260)

[1] Some material in this chapter was drawn from the author's prior articles at the *AI3:::Adaptive Information* blog: "Conceptual and Practical Distinctions in the Attributes Ontology" (Mar. 2015); "KBpedia Relations, Part I: Smarter Knowledge Graphs" (May 2017); "KBpedia Relations, Part II: An Event-Action Model" (May 2017); "KBpedia Relations, Part III: A Three-Relations Model" (May 2017).

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8_7

In terms of the logic triad, speculative grammar is thus a Firstness in Peirce's category structure. Firstness is meant to capture the possibilities of the domain at hand. Secondness is meant to capture the particular facts or real things of the domain at hand, the *critic* in terms of the logic triad. Thirdness is meant to capture methods for discovering the generalities, laws, or new knowledge arising from the domain, the *methodeutic* branch of the triad. We may apply this construct to any topic, from signs to logic and science. The 'surprising fact,' or new insight arising from Firstness or Thirdness, points to potentially new topics that may themselves become new targets for this logic of semiosis.

Without the right concepts, terminology, or bounding—that is, the speculative grammar—it is impossible to understand or compose the *objects* (conceptual and material) within Secondness that populate the domain at hand. Without the right language and concepts to capture the connections and implications of the domain at hand—again, part of its speculative grammar—it is not possible to discover the generalities or the 'surprising fact' or Thirdness of the domain. The speculative grammar is thus needed to provide the right constructs for describing, analyzing, and reasoning over the given domain. Our logic and ability to understand the focus of our inquiry require that we describe and characterize the domain of discourse with proper scope and relationships. How well we bound, characterize, and signify our problem domains—again, the speculative grammar—directly relates to how well we can reason and inquire about that space.

Let's take a couple of examples to illustrate this. First, imagine van Leeuwenhoek first discovering 'animalcules' under his early microscope. Over the ensuing years, new terms and concepts like flagella, cells, and vacuoles were coined and systematized to enable a further understanding of microorganisms, requiring careful inspections and consensual vocabulary. Second, imagine 'action at a distance' phenomena such as magnetic repulsion or static electricity causing hair to stand on end. For centuries these phenomena were assumed to be caused by atomistic particles too small to see or discover. Only when Hertz was able to prove Maxwell's equations of electromagnetism in the mid-1800s were the concepts and vocabulary of waves and fields sufficiently developed to begin to unravel electromagnetic theory in earnest. Progress required the right concepts and terminology.

For Peirce, the triadic nature of the sign—and its relation between the *representamen*, its *object*, and its *interpretant*—was the speculative grammar breakthrough that then allowed him to better describe the process of sign-making and its role in the logic of inquiry and truth-testing (semiosis). Because he recognized it in his work, Peirce understood that a conceptual grammar appropriate to the inquiry at hand is essential to further discovery and validation. As Peirce says in his first paper outlining his early logic of relatives [2]:

> The fundamental principles of formal logic are not properly axioms, but definitions and divisions; and the only facts which it contains relate to the identity of the conceptions resulting from those processes with certain familiar ones. (1870, CP 3.149)

We begin our analysis of a speculative grammar suitable to knowledge representation with the relevant 'things' (nouns) that populate our world and how we organize them in relation to one another.[2] We then expand our discussion of *relations* to include *actions* and perceptions (verbs) between these things, as well as how we talk about or describe those things. Things and relations combined enable us to make *statements* and *assertions*. In the aggregate, multiple statements interact to create many kinds of information structures, some with impressive analytical properties discussed in later chapters. In knowledge representation, terminology can be a tricky business, since different approaches to KR adopt different terms, sometimes overlapping or in conflict with other schemes. I try to point out some of these conflicts for key terms in the three chapters of this Part II on grammar. Throughout this chapter—indeed, this book—we *italicize* the basic KR terminology we have adopted for this book and KBpedia. The *Glossary* consolidates this terminology in one location.

Things of the World

We watch our children first learn the names of things as they begin mastering language. The learning focus is on nouns and building a vocabulary about the things that populate the tangible world. By the time we start putting together our first sentences, typified in books such as Dick and Jane and the dog Spot, our nouns get increasingly numerous and rich, though our verbs remain simple. We acquire terms in our early language more about different kinds of objects than different kinds of actions (though concepts such as 'More' or 'Want' or gesturing to the mouth to signify 'Eat' are learned early!). Our early verbs are fewer in number and much less varied than the differences of form and circumstance we can see from objects. Our knowledge artifacts reflect this imbalance.

Entities, Attributes, and Concepts

Entities and concepts dominate most knowledge graphs. For example, knowledge base constituents of KBpedia, such as Wikidata, Wikipedia, or GeoNames, have millions of concepts or entities within them, but fewer than a few thousand predicates (approx. 2500 useful in Wikidata and 750 or so in DBpedia and schema.org).

Entities are the individual, real things in our domain of interest; they are nameable things or ideas that have an identity, are defined in some manner, can be referenced, and may be related to *types*. Entities are most often the bulk of an overall knowledge

[2] In OWL 2, "thing" is the *root* node, with all other vocabulary items being subsidiary to it. Here, we are using "thing" more in keeping with the common vernacular.

base. An entity is an individual object or *instance,* a Secondness, of a *class*, a Thirdness. When affixed with a proper name or label, we term it a *named entity* (thus, named entities are a subset of all entities). *Attributes* describe and characterize entities. We connect or relate entities to one another through *external relations*. How we refer to, signify, or index these things is what we call *representations*. An entity may be independent or separate or can be part of something else, such as parts of a whole. Entities cannot be topics or types or datatypes.[3]

We look to separate the existence of some things different from other things by the nature of their characteristics, what we can observe and describe for that given thing. So, we describe shapes, sizes, weights, ages, colors, and characteristics of things with increasingly nuanced vocabularies. We note that grasses have linear or simple leaves, oaks have serrated or wavy-shaped leaves, and carrots have branched or compound leaves. We distinguish hair color, eye color, place of birth, current location, and a myriad of factors. Each one of these factors becomes an *attribute* for that object, with the specific values (simple *v* wavy *v* compound) distinguishing instances from one another. We can also assign *values* to attributes, such as having an age of 7 years or a height of 120 cm. Attributes do not exist independently from the things they characterize.[4] For example, 'round' or 'blue' are not things unto themselves but are modifiers or qualifiers or characteristics of particular things. Attributes in Peirce's universal categories are a Firstness. Chen described similar entity-attribute distinctions in his attempt to find common ground across the network, relational, and entity set models in today's commonly used E–R model [3].

A *concept is* something we conceive in the mind, such as an idea or a grouping of like things. When we organize these things according to their shared and natural attributes, a topic we discuss in more detail in Chap. 10, we call them a *type*.[5] Concepts and types are not discrete, tangible things, but are constructs of thought. Topics are a form of a concept, but as used herein are more of a complex of concepts, ideas, and entities. Note this use of topic contrasts to that used in *topic maps*, which subsumes the terms of concepts, entities, and events used herein.[6] Concepts and types are *generals*, a Thirdness in Peirce's universal categories.

Here are some other terms you may encounter in other grammars or knowledge representations for these terms (Table 7.1):

[3] The role for the label "entity" can also refer to what is known as the *root* node in some systems such as SUMO. In the OWL language and RDF data model we use, we know the root node as "thing." Our use of the term "entity" is much different from SUMO and resides at a subsidiary place in the overall TBox hierarchy (see Chap. 8). In this case, and frankly for most semantic matches, equivalences should be judged with care, with context the crucial deciding factor.

[4] Though Peirce, as do we, came to believe that Firstness (and Thirdness, for that matter) was real, for something to exist, it must be actual, which is Secondness.

[5] They are, however, *real*, since their type concept exists independent of our thinking about it. Peirce's insistence that generals may be real helps to situate him among a common split of philosophers into the nominalist, idealist, and realist camps.

[6] See https://en.wikipedia.org/wiki/Topic_map.

Table 7.1 Comparison of common noun terms

KBpedia terminology	Terminology used elsewhere	
Entity	• Object • Instance • Exemplar • Element • Particular	• Member • Record • Individual • Dependent variable • Token
Attribute	• Property • Predicate • Relationship • Feature • Facet	• Dimension • Characteristic • Field • Header • Independent variables
Type	• Concept • Kind • Set	• Collection • Type • Class

The distinctions between entities and concepts are often murky in the real world. For example, let's consider the 'toucan' bird, first introduced in Chap. 1, which we may refer to by word or picture. When we inspect what might be a description of a toucan on Wikipedia, we see that the term more broadly represents the family of *Ramphastidae*, which contains 5 genera and 40 different species. The picture we use to refer to toucan may be, say, that of the keel-billed toucan (*Ramphastos sulfuratus*). However, if we view the images of a list of toucan species, we see just how physically divergent various toucans are from one another. Across all species, average sizes vary by more than a factor of three with great variation in bill sizes, coloration, and range. Further, if I assert that the picture of the toucan is that of my pet keel-billed toucan, *Pretty Bird*, then we can also understand that this representation is for a specific individual bird, and not the physical keel-billed toucan species as a whole. The point is not a lesson on toucans, but an affirmation that distinctions between what we think we may be describing occur over multiple levels. The meaning of what we call a 'toucan' bird is not embodied in its label or even its name, but in the accompanying referential information that places the referent into context. Without such accompanying context, the stand-alone word or picture of 'toucan' may represent either an individual entity or one of perhaps multiple types. I discuss further the importance of context and 'things, not strings' in Chap. 10.

What Is an Event?

Events are like entities, except they have a discrete time beginning and end. Are events entities, and, if not, what are they? Events are part of time, occupy some length of time, and sometimes are so notable as to get names, either as types or named events, such as *germination* or *World War II*. Events are undoubtedly different from tangible objects which occupy some space, have physicality, exist over some length of time, and also get names as types or named instances. Moreover,

both of these are different still than concepts or ideas that are creatures of thought. How to place the notion of events within a consistent worldview is one test for the coherency of a given knowledge representation.

The philosophical question of *What is an event?* is readily traced back to Plato and Aristotle. One place to start is the *Stanford Encyclopedia of Philosophy*, which offers a kind of Cliff Notes version overviewing various views on events [4] (among many other articles in philosophy). At least five or six strains of thought argue the nature of events. The fact we have no real intellectual consensus as to *What is an event?* after 2500 years suggests both that it is a good question and that any answer is unlikely to find consensus. Nonetheless, the question of what is an event provides a good microcosm for understanding Peirce's worldview. For Peirce, 'We perceive objects brought before us; but that which we especially experience—the kind of thing to which the word 'experience' is more particularly applied—is an event. We cannot accurately be said to perceive events' (1897, CP 1.336). He further states that 'If I ask you what the actuality of an event consists in, you will tell me that it consists in its happening *then* and *there*. The specifications *then* and *there* involve all its relations to other existents. The actuality of the event seems to lie in its relations to the universe of existents' (1903, CP 1.24).

Though events are said to occur, to happen, or to take place, entities are said to exist. From Peirce again:

> The event is the existential junction of *states* (that is, of that which in existence corresponds to a *statement* about a given subject in representation) whose combination in one subject would violate the logical law of contradiction. The event, therefore, considered as a junction, is not a subject and does not inhere in a subject. What is it, then? Its mode of being is existential quasi-existence, or that approach to existence where contraries can be united in one subject. Time is that diversity of existence whereby that which is existentially a subject is enabled to receive contrary determinations in existence. (1896, CP 1.494)

As Peirce says, 'individual objects and single events cover all reality ....' (1905, CP 5.429).[7]

The event represents a juxtaposition of states, the comparison of the subject prior and after the event providing the basis for the nature of the event. Each change in state represents a new event, which can trigger new actions and reactions leading to still further events. Simple events represent relatively single changes in state, such as turning off a light switch or a bolt of lightning. More complicated events may involve multiple processes and potentially long durations, such as epoch, ages, or even geological eras. Havel insists that we should distinguish things and events only by differences in time scale: 'In the world of all scales there is no essential difference: things are just long-lasting events and events are just short-lived things, where -long- and -short are relative with respect to our temporal scale perspective' [6].

[7] Here, Peirce uses a different sense for reality than his later belief that the universal categories are real. Also, there are many other useful statements by Peirce regarding events; see [5].

How to characterize events provides a kind of Rorschach test for how one views reality. Events are like the spark that leads us to understand *actions* better and what emerges from them, which in turn helps us better understand predicates and relations. What we learn from Peirce is that *events* are quasi-entities, based on time rather than space, and, like entities, are a Secondness. Like entities, we can name events and intrinsically inspect their attributes. Events may also range from the simple to the triadic and durative.[8] Events are the first portions of activity and process cascades, and can stimulate such seemingly non-energetic actions like thought. Thought, itself, may be a source of further events and action, as may be the expressions of our thought, symbols. Actions always carry with them a reaction, which can itself be the impetus for the next action in the event cascade. Events are the real triggering and causative factors in reality. Entities are a result and manifestation of events. Events, like entities, are Secondness, or what we call *particulars*.

Hierarchies in Knowledge Representation

The human propensity to categorize is an attempt to make sense of the world. We base the act of categorization on how to group things and how to relate those things and groups to one another. The categorization process informs us to characterize or describe the things of the world using what we have termed *attributes* for their similarities.[9] We may also categorize based on the relationships of things to external things.[10] No matter the method, the results of these categorizations are often hierarchical, reflective of what we see in the natural world. We see hierarchies in Nature based on bigger and more complex things built from simpler things, sometimes based on fractals or cellular automata or based on the evolutionary relationships of

[8] Within events, we can also categorize according to the three universal categories. The unexpected flash or shock is a Firstness within events. Peirce's doctrine of tychism places a central emphasis on chance, which he views as the source of processes in nature such as evolution and the "surprising fact" that causes us to reinvestigate our assumptions leading to new knowledge. We more commonly associate an event with action, and that is indeed a major cause of events. (However, chance events or accidents, as an indeterminate group, may trigger events.) An action is a Secondness, however, because it is always paired with a reaction. Reactions may then cause new actions, itself a new event. In this manner activities and processes can come into being, which while combinatorial and compound can also be called events, including those of longer duration. That entire progression of multiple actions represents increasing order, and thus the transition to Thirdness. Peirce makes the interesting insight that thoughts are events, too. "Now the logical comprehension of a thought is usually said to consist of the thoughts contained in it; but thoughts are events, acts of the mind. Two thoughts are two events separated in time, and one cannot literally be contained in the other" (1868, CP 5.288).

[9] The most common analogous terms to attributes are *properties* or *characteristics; in* the OWL language used by KBpedia, we assign attributes to instances (called individuals) via property (relation) declarations.

[10] The act of categorization may thus involve intrinsic factors or external relationships, with the corresponding logics being either *intensional* or *extensional, as* discussed further in Chap. 8.

life-forms. According to Annila and Kuismanen [21], 'various evolutionary processes naturally emerge with hierarchical organization.' Hierarchy—and its intimate connection with categorization and categories—is thus fundamental to the why and how we can represent knowledge for computable means.

Depending on the context, we can establish hierarchical relationships between types, classes, or sets, with instances or individuals, with characteristics of those individuals, and between all of these things. The terminology differs by context, and sometimes the syntax may also carry a formal understanding of how we can process and compute these relationships. Nillson provides a general overview of these kinds of considerations with a useful set of references [7].

Types of Hierarchical Relationships

As early as 1997 Doyle noted in the first comprehensive study of KR languages, 'Hierarchy is an important concept. It allows economy of description, economy of storage and manipulation of descriptions, economy of recognition, efficient planning strategies, and modularity in design'. He also noted, 'hierarchy forms the backbone in many existing representation languages' [8]. The basic idea of a hierarchy is that some item ('thing') is subsidiary to another item. Categorization, expressed both through the categories themselves and the process of how one splits and grows categories, is a constant theme in this book. The idea of hierarchy is central to a category or other such groupings and how we tie those categories or groupings together. A hierarchical relationship is shown diagrammatically in Fig. 7.1 with A or B, the 'things,' shown as *nodes*. All this diagram is saying is that A has some form of superior or superordinate relationship to B (or *vice versa*, that B is subordinate to A). This hierarchical relationship is a direct one, but one of unknown character. Hierarchies can also relate more than two items (Fig. 7.2).

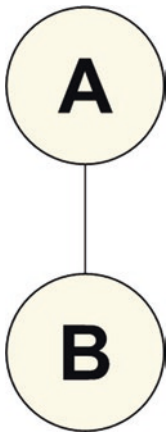


Fig. 7.1 Direct hierarchy

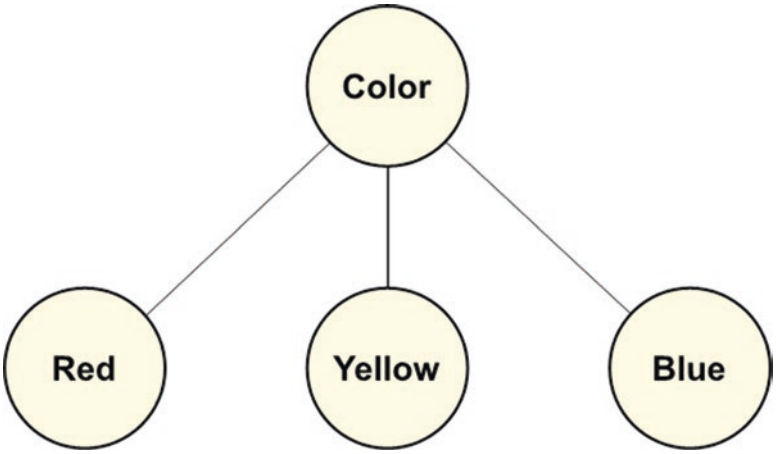


Fig. 7.2 Simple hierarchy

In this case, the labels of the items may seem to indicate the hierarchical relationship, but relying on labels is wrong. For example, let's take this relationship, where we show the mixed nature of primary and secondary colors (Fig. 7.3):[11]

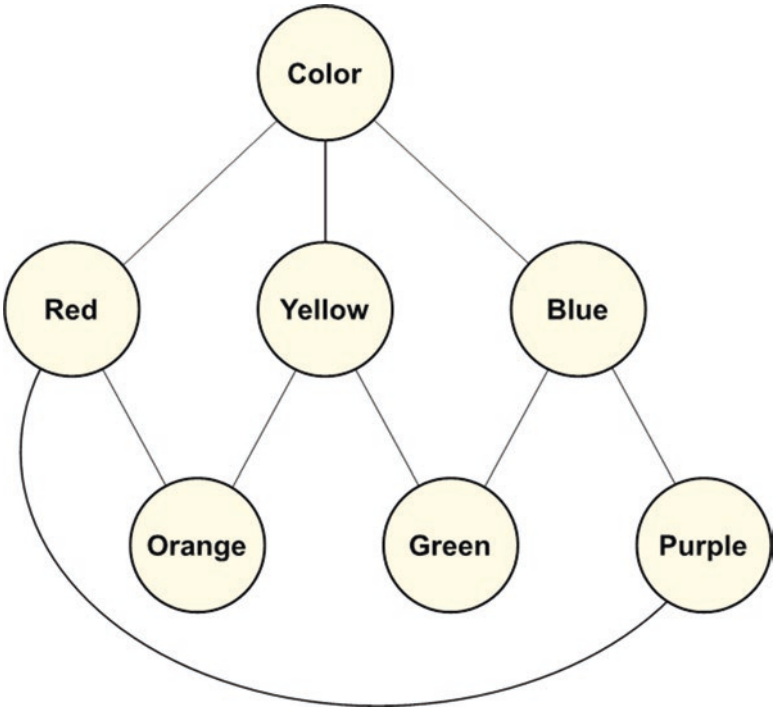


Fig. 7.3 Multiple intensional hierarchy

[11] J W von Goethe (1749–1832) first explicated the standard three-color scheme. What is more commonly used in design is a four-color scheme from Ewald Hering (1834–1918).

Yet perhaps our intent was instead to provide a category for all colors lumped together, as instances of the concept 'color' shown in Fig. 7.4.

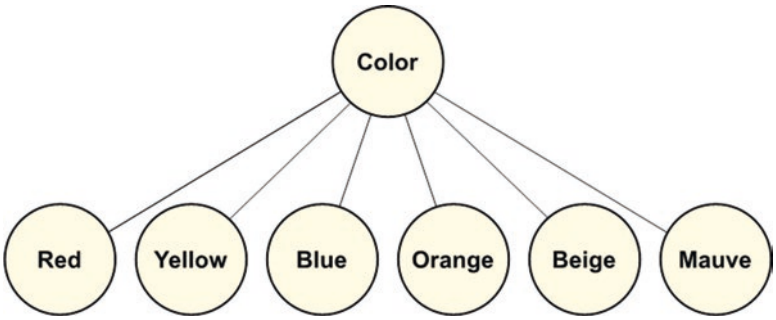


Fig. 7.4 Extensional hierarchy

The point is not to focus on colors—which are, apparently, more complicated to model than first blush—but to understand that hierarchical relations are of many types and what one chooses about a relation carries with it logical implications, the logic determined by the semantics of the representation language used and how we represent it.

Peirce's concept of *prescission* captures the most fundamental expression of a hierarchical relationship, stated as the relation, *prescind*.[12] Here are some quotes of how Peirce described the somewhat tricky method of *prescission*:

> There are three distinct kinds of separation in thought. They correspond to the three categories. Separation of Firstness, or Primal Separation, called *Dissociation*, consists in imagining one of the two separands without the other. It may be complete or incomplete. Separation of Secondness, or Secundal Separation, called *Precission*, consists in supposing a state of things in which one element is present without the other, the one being logically possible without the other. Thus, we cannot imagine a sensuous quality without some degree of vividness. But we usually *suppose* that redness, as it is in red things, has no vividness; and it would certainly be impossible to demonstrate that everything red must have a degree of vividness. Separation of Thirdness, or Tertial Separation, called *discrimination*, consists in representing one of the two separands without representing the other. (1903, EP 2:270)

And,

> But *prescission*, if accurately analyzed, will be found not to be an affair of attention. We cannot prescind, but can only distinguish, color from figure. But we can prescind the geometrical figure from color; and the operation consists in imagining it to be so illuminated that its hue cannot be made out (which we easily can imagine, by an exaggeration of the familiar experience of the indistinctness of hues in the dusk of twilight). In general, prescission is always accomplished by imagining ourselves in situations in which certain elements of fact cannot be ascertained. This is a different and more complicated operation than merely attending to one element and neglecting the rest. (1903, CP 2.248)

[12] Peirce also termed this concept *precission*, *prescisive abstraction*, *prescision*, or *precisive abstraction* (1902, CP 4.235). It comes from the same root as precision in measurements but has a different meaning as described in the text, which is one reason we prefer the spelling of *prescission* to distinguish it as much as possible.

Prescission is an asymmetrical separation of two elements objectively considered; it is a logical operation that does not make any ontological or epistemological assumptions about the two elements being considered [9]. The process works by examining the two elements in isolation and then asking whether one might be possible or exist without the other. We can *dissociate* red from blue or a triangle from a square, but we cannot *prescind* different colors or shapes. However, we *can* prescind color from shape because color is not possible without having a spatial aspect, but we *cannot* prescind shape from color because a shape may exist without color. These distinctions are not grounded in experience, nor are they subjective, important to Peirce in finding a realistic logic. Prescission carries no connotation of meaning.

We can apply this same process to the ideas of generals and particulars. The general type of 'man' cannot be prescinded from a single, individual 'man' because we cannot conceive of a general type of 'man' without conceiving of individual 'men.' On the other hand, I can prescind the individual 'man' from the general type of 'man' since the idea of the general 'man' does not depend on the existence of any individual 'man.' Peirce uses the same process of prescission to prescind the concepts of First, Second, and Third. Then, through a method he termed *hypostatic abstraction* (*e.g.*, CP 4.235), which is how to turn sign predicates into subjects (such as turning the predicate 'collect' into a general, singular of 'collection'), Peirce names the universal categories of Firstness, Secondness, and Thirdness.[13]

Table 7.2 Example hierarchical relationships

A	Subsumes	B
A	Prescinds	B
A	Is more basic than	B
A	Is a superClassOf	B
A	Is more fundamental than	B
A	Is broader than	B
A	Includes	B
A	Is more general	B
B	Is-a	A
A	Is parent of	B
A	Has member	B
A	Has an instance of	B
A	Has attribute	B
A	Has part	B

In use, we may see a variety of hierarchical relationships. Table 7.2 shows some (vernacular) examples one might encounter. Again, though we have now labeled the relationships, which in a graph representation are the *edges* between the *nodes*, it is still unclear the populations to which these relations may apply and what their exact semantic relationships may be.

[13] "Prescind" is often more clearly stated as "prescinded from." Roughly equivalent phrases are to "leave out of consideration," "separate from something," or "withdraw attention from."

Table 7.3 shows some hierarchical relations that one might want to model, and whether the item resides in the universal categories of Firstness, Secondness, or Thirdness.

Table 7.3 Possible pairwise (——) hierarchical relationships

Firstness		Secondness		Thirdness
Attribute	——	Token (instance)		
		Sibling		
		|		
		Sibling		
		Child	——	Parent
		|		
		Parent		
		Token	——	Type
		Part	——	Whole
		|		
		Whole		
		Sub	——	Super
				|
				Sub

Note that, depending on the context, some of the items may reside in either Secondness or Thirdness (depending on whether the *referent* is a particular instance or a general). Also note the familial relationships shown: child-parent-grandparent and child-child relationships occur in actual families and as a way of talking about inheritance or relatedness relations. The idea of type or is-a is another prominent one in ontologies and knowledge graphs. Natural classes or kinds, for example, fall into the type-token relationship. Also note that mereological relationships, such as part-whole, may also leave open ambiguities. We also see that specific pairs, such a sub-super, child-parent, or part-whole, need context to resolve the universal category relation.

Reliance on item labels alone for the edges and nodes, even for something as seemingly straightforward as color or pairwise relationships, does not give us sufficient information to determine how to evaluate the relationship nor how to organize properly. We thus see in knowledge representation that we need to express our relationships explicitly. Labels are merely assigned names that, alone, do not specify the logic mode, what populations are affected, or even the exact nature of the relationship. Without these basics, our knowledge graphs cannot be computable. Well over 95% of the assignments in contemporary knowledge bases have this item-item character. We need interpretable relationships to describe the things that populate our domains of inquiry to categorize that world into bite-sized chunks.

Salthe categorizes hierarchies into two types: compositional hierarchies and subsumption hierarchies [22]. A subsumptive hierarchy ‘subsumes’ its children, and a

compositional hierarchy is 'composed' of its children. Mereological and part-whole hierarchies are compositional, as are entity-attribute. Subsumption hierarchies are ones of broader than, familial, or evolutionary. Cottam et al. believe that hierarchies are so basic as to propose a model abstraction over all hierarchical types, including levels of abstraction [23].[14]

These discussions of structure and organization are helpful to understand the epistemological bases underlying various kinds of hierarchy. We should also not neglect recursive hierarchies, such as fractals or cellular automata, which are also simple, repeated structures commonly found in nature. Fortunately, Peirce's universal categories provide a powerful and consistent basis for us to characterize these variations, and his notion of prescission also helps adjudicate logical hierarchical relationships. When paired with logic and the KR languages discussed in Chap. 8, and with 'cutting Nature at its joints' in Chap. 5, we end up with an expressive grammar for capturing all kinds of internal and external relations to other things.

So far we have learned that most relationships in contemporary knowledge bases are of a noun-noun or noun-adjective nature, which I have loosely lumped together as hierarchical relationships. These relationships span from attributes to instances (individuals) and classes[15] or types, with and between one another. We have learned Peirce's logical concept of how to prescind a superordinate concept from a subordinate one. We have further seen that labels either for the subjects (nodes) or their relationships (edges) are an insufficient basis for computers (or us!) to reason over. Mostly, we have come to see that we need to ground our relationships in specific semantics and logics for reasoning machines to process our representations without ambiguity.

Structures Arising from Hierarchies

Structure is a tangible part of thinking about a new KR installation since we may apply many analytic choices against the knowledge artifact. Different kinds of structure are best for various tools or kinds of analysis. The type of relations chosen for the artifact affects its structural aspects. These structures can be as small and straightforward as a few members in a list, to the entire *knowledge graph* fully linked to its internal and external knowledge sources. Knowledge structures arise from the various hierarchical relationships just discussed. Here are some of the prominent types of structures that may arise from connectedness and characterization hierarchies:

[14] There is a very helpful 25-page listing of references dealing with "hierarchy" at the conclusion of Salthe's 2012 paper, Hierarchical Structures.

[15] In the OWL 2 language used by KBpedia, a *class* is any arbitrary collection of objects. A class may contain any number of *instances* (called *individuals*), or a class may be a subclass of another. Instances and subclasses may belong to none, one, or more classes. Both extension and intension may be used to assign instances to classes.

- *Lists*—unordered members or instances, with or without gaps or duplicates, useful for bulk assignment purposes. Lists occur through a direct relation assignment (*e.g.*, `rdf:Bag`).
- *Neural networks* (*graphs*)—graph designs based on connections modeled on biological neurons, still in the earliest stages for relations and KR formalisms [12].
- *Ontologies* (*knowledge graphs*)—sometimes ontologies are treated as synonymous with knowledge graphs[16], but more often as a superset that may allow more control and semantic representation.[17] Ontologies are a central design feature of KBpedia [13].
- *Parts of speech*—a properly designed ontology has the potential to organize the vocabulary of the KR language itself into corresponding parts of speech, which aids some methods of natural language processing.
- *Sequences*—ordered members or instances, with or without gaps or duplicates, useful for bulk assignment purposes. Sequences occur through a direct relation assignment (*e.g.*, `rdf:Seq`).
- *Taxonomies* (*trees*)—trees are subsumption hierarchies with single or multiple class inheritance for instances; most knowledge graphs allow multiple inheritances.
- *Typologies*—are essentially multi-inheritance taxonomies, with the hierarchical organization of types as natural as possible. Natural types (classes or kinds) enable us to make the largest number of disjoint assertions, leading to efficient processing and modular design. Typologies are a central design feature of KBpedia; see Chap. 10.

Typically KR formalisms and their internal ontologies (taxonomy or graph structures) have a starting node or *root*, often called 'thing,' 'entity,' or the like. Close inspection of the choice of the root may offer important insights into different KR language philosophies. 'Entity' as a root, for example, is not compatible with a Peircean interpretation, since all entities are within Secondness, one of the three subsidiary branches in our main KR structure.

KBpedia's foundational structure is the subsumption hierarchy shown in the KBpedia Knowledge Ontology (KKO)—that is, KBpedia's upper ontology—and its nodes derive from the universal categories. The terminal, or *leaf*, nodes in KKO each tie into typologies. *Types* are the constituents of a typology. Types, which are generals along with typologies, are the classification of natural kinds of instances as determined by shared attributes (though not necessarily the same values for those attributes). Most of the types in KBpedia are composed of entities, but attributes and relations also have aggregations of types. In turn, a *typology* is a hierarchical

[16] In the semantic Web space, "ontology" was the original term because of the interest to capture the nature or being (Greek ὄντως, or ontós) of the knowledge domain at hand. Because the word 'ontology' is a bit intimidating, a better variant has proven to be the knowledge graph (because all semantic ontologies take the structural form of a graph). In this book, I tend to use the terms ontology and knowledge graph interchangeably.

[17] RDF graphs are more akin to the first sense; OWL 2 graphs more to the latter; see next chapter.

classification of related types as determined by the essence or characteristics of its root. Subsequent chapters discuss these items in some detail; Appendix B describes KKO.

Of course, choice of a KR formalism and what structures it allows must serve many purposes. We desire uses of the KR formalism and the knowledge graph to include knowledge extension and maintenance, record design, querying, reasoning, graph analysis, logic and consistency tests, planning, hypothesis generation, question and answering, and subset selections for external analysis. We have often supported other tasks such as machine learning, natural language processing, data wrangling, statistical and probabilistic analysis, search indexes, and other data- and algorithm-intensive applications using dedicated external applications. We have as a goal to build structures into the KR installation to support these kinds of uses, or to export data suitable to external applications. Chapter 12 expands on these platform considerations.

A Three-Relation Model

If hierarchy provides the basis for the scaffolding in a knowledge graph, then *actions* offer the means to make a knowledge system dynamic. Moving beyond static knowledge representations is the way for these systems to support active learning, respond to sensors, plan, hypothesize, and solve problems. Peirce's universal categories and these hierarchical perspectives dovetail nicely into a three-relation model that captures all aspects of knowledge representation to support the full slate of anticipated artificial intelligence applications. Relations are the way we describe connections among things, including attributes which we only express for subjects.

Guarino, in some of the earliest (1992) writings leading to semantic technologies, had posited knowledge bases split into concepts, attributes, and relations [14]. This split was close to my thinking and provided comfort since it arose in the earliest days of the semantic Web.[18] Some of the impressive work by Sekine [15] extending the concept of entity types influenced me greatly. Still, I was confused by the mixing of attributes and entities; indeed, most practitioners do not appreciate or employ the purposeful separation of attributes from other relations, let alone entities. It was only after the study of Peirce that I realized he had a way to untangle the knot of attributes, events, relations, actions, perceptions, thoughts, and belief. His '*architectonic*' began providing answers to epistemological questions across the board. It still does.

[18] I raise the early work by Guarino for a reason. We, the community of KR practitioners, have not gotten our basic grammar right about how we think about these problems. Most everyone still gets bollixed up trying to handle concepts like relations (for me, split into the three categories of attributes, external relations, and representations), events, generals (types or classes), and particulars (individuals or instances). Peircean principles give us logical and defensible ways to think about these problems. That approach strikes me as superior to heated assertions that often lack logical underpinnings.

Besides Peirce, I studied thinkers across history who may have tackled fundamental concepts in knowledge organization. Aristotle's categories were influential, and have mostly stood the test of time and figured prominently in my thinking, as they did for Peirce. Peirce was a student of Kant and Hegel (as well, in contrast, Descartes), and the logicians DeMorgan, Boole, and Venn, but he created a unique synthesis. I also reviewed efforts such as Sarbo's to apply Peirce to knowledge bases [16], as well as most other approaches discussing Peirce with some correspondence to KBs [10, 11, 17].

Our resulting three-relation model is consistent with Peirce's thinking, even though he never had today's concepts of digital knowledge representation as an objective. For example, he labeled one of his sections 'The Conceptions of Quality, Relation and Representation, Applied to this Subject' ('Upon Logical Comprehension and Extension'; 1867, CP 2.418). Thirty-five years later, Peirce still held to this split, '… there are but three elementary forms of predication or signification, which as I originally named them (but with bracketed additions now made to render the terms more intelligible) were qualities (of feeling), (dyadic) relations, and (predications of) representations' (1903, EP 2:424; CP 1.561).

Thirdness is the sauce that gives meaning to what is different in Peirce's architectonic over standard knowledge representations. Too many pivotal problems exist that we cannot address with dichotomous worldviews. Disambiguation is made difficult without context. The world is probabilistic. Chance happens. New information is a barrage, we continuously seek knowledge, and beliefs evolve and change. Though we may partially describe context with nouns related to perception, situations, states, and roles, we ultimately require an understanding of events, actions, and relations. Until these latter factors are better captured and understood, our ability to establish context remains limited. Peirce elaborates:

> Now every simple idea is composed of one of three classes; and a compound idea is in most cases predominantly of one of those classes. Namely, it may, in the first place, be a quality of feeling, which is positively such as it is, and is indescribable; which attaches to one object regardless of every other; and which is *sui generis* and incapable, in its own being, of comparison with any other feeling [a*ttributes*], because in comparisons it is representations of feelings and not the very feelings themselves that are compared. Or, in the second place, the idea may be that of a single happening or fact, which is attached at once to two objects, as an experience, for example, is attached to the experiencer and to the object experienced [e*xternal relations*]. Or, in the third place, it is the idea of a sign or communication conveyed by one person to another (or to himself at a later time) in regard to a certain object well known to both [r*epresentations*].' (1905, CP 5.7) (labeling brackets added)

We now have the basis to define the three modes of relations within KBpedia. The first of these is the grouping of a*ttributes*, the relationship of a subject with its intrinsic qualities or characteristics, which are a Firstness within Peirce's universal categories. The second of these modes is *external relations*, which are all of the ways a *particular* or *general* may relate to another *particular* or *general*. These include hierarchical relations other than attributes (which are monadic).[19] Relations of

[19] However, we can type attributes, so it is possible to organize and reason over them.

a*ction* (one thing affecting another) or *perception* (one thing experiencing an external change) are external relations, which are a Secondness within the universal categories. The third mode of relations we call r*epresentations* since these are the ways we describe, point to, or otherwise indicate the thing at hand. These relations give our subjects perspective and meaning, though we cannot easily reason over these relations. They are a Thirdness within Peirce's universal categories. These constructs are central to our approach to knowledge representation.

Attributes, the Firstness of Relations

Attributes are the intensional characteristics of an object, event, entity, type (when viewed as an instance), or concept. The relationship is between the individual instance (or particular) and its attributes and characteristics, in the form of *A:A*. Attributes may be intrinsic characteristics or essences of single particulars, such as colors, shapes, sizes, or other descriptive characteristics. Attributes may be adjunctual or accidental happenings to the particular, such as birth or death. Attributes may be contextual for placing the particular within time or space or external circumstances, absent having a direct relationship (in that case it is an *external relation*).

Attributes are specific to the individual, and only include events that are notable for the individual. They are a Firstness, and in totality try to capture the complete characteristics of the individual particular, which is a Secondness. Since attributes are the properties of an entity, we can better interoperate entity data by concentrating on those aspects that let us match data in one set of records to similar data in different records. In the next chapter, we will discuss building a new vocabulary and structure upon RDF to provide more sophisticated handling of 'properties' than RDF or OWL alone can offer in their native forms.

Calling out attributes for such attention is not new. The attributes-relation split has not been an uncommon one in the KB literature [10, 18], though it is not an accepted canon and is infrequent in other knowledge representations. Philosophers draw distinctions about intrinsic *v* extrinsic properties [19] or intensionality *v* extensionality.[20] For conceptual models with specific reference to ontologies, Wand et al. [20] in 1999 were making the distinction between *intrinsic properties* (akin to what we term attributes herein) and *mutual properties* between things (what we term external relations). Unfortunately, at that time, the conventions of RDF had not yet become prevalent, and the idea of annotation properties had not yet emerged (from OWL). These later distinctions are important, but the Wand et al. discussion still is helpful to elucidate the same pragmatic and theoretical considerations.

[20] At least for Carnap, he thought "… the full meaning of a concept is constituted by two aspects, its intension and its extension. The first part comprises the embedding of a concept in the world of concepts as a whole, i.e., the totality of all relations to other concepts. The second part establishes the referential meaning of the concept, i.e., its counterpart in the real or in a possible world."

With all of this discussion of attributes the attentive reader might be confused: Are attributes not nouns or adjectives that seem similar to objects as we discussed for hierarchies? Alternatively, are attributes a more verb-like relation? The answer, naturally, is that it depends. When we think about an attribute as some quality of something, we reify it as a noun and make it its object. Considered in this manner, the 'idea' of an attribute makes it a real thing, and a Secondness in that reified state (which, of course, is not the same as the underlying thing). Without that relation to the something, it does not exist, which makes it only an ephemeral quality, a Firstness. We can both describe and relate attributes, depending on the context. It is this kind of contextual lens that makes Peirce's universal categories so powerful.

External Relations, the Secondness of Relations

External relations are actions or assertions between an event, entity, type, or concept and another particular or general. An external relationship has the form of *A:B*. External relations may be simple ones of a direct relationship between two different instances. External relations may be copulative by combining objects or asserting membership, quantity, action, or circumstance. External relations may be mediative to provide meaning, context, relevance, generalizations, or other explanations of the subject vis-à-vis the external world. External relations are extensional.

All actions are external relations. Actions may be reactions to perceptions or stimuli. Actions may be energetic, arising from the subject and affecting the external environment in some way. We may understand some actions as a basis of thought, which results in new actions or modified concepts or thought. External relations are by definition a Secondness. Notice how these three groupings of external relations are themselves an example of the universal categories. It is in this manner that bigger, more abstract notions may be broken down into more manageable pieces by employing the universal categories.

Representations, the Thirdness of Relations

The third category in our model of relations is the least used and, perhaps, the most confused regarding how other KR systems treat their scope. *Representations* are signs (1905, CP 8.191) and the means by which we point to, draw attention to, or designate, denote, or describe a particular object, entity, event, type, or general. A representational relationship has the form of *re:A* (about *A*). Representations can be designative of the subject, that is, be icons or symbols (including images, labels, definitions, and descriptions). Representations may be indexes that more or less help situate or provide a traceable reference to the subject. Representations may be associations, resemblances, and likelihoods about the subject, more often of indeterminate character (such as a probability assignment). Representations are the

Thirdness of relations. Representations cannot be deductively reasoned over, but some characteristics may be derived or analyzed through inductive or abductive inferential means.

For example, *annotations* are representations. Annotations capture the circumstances or conditions or contexts or observations for the thing at hand. Where did we discover or find it? When did we find or elaborate on it? By whom or when was it found or elaborated? What is our commentary about it? While these are all external elaborations of the thing at hand, and not intrinsic to the nature of the thing, they are all characterizations about a given thing. In these regards, annotations have as their focus a given object, similar to what is valid for attributes. We cannot deductively reason over annotations, though annotations play pivotal roles. Annotations are an essential means for tagging, matching, and slicing and dicing the information space. *Metadata* is a similar concept, more oriented to provenance and description.

Labels, which are also representations as are definitions, are the means to broaden the correspondence of real-world reference to match the true referents or objects in the knowledge base.[21] Broader reference enables us not to limit referents to any given label or string. In best practice, labels should reflect all of the various ways a given object may be identified (synonyms, acronyms, slang, jargon, all by language type). These considerations improve the means for tagging, matching, and slicing and dicing, even if we cannot reason over the annotations.

The Basic Statement

We now have a starting grammar to talk about the things of the world, and the relations that place them into context with the external world. We have our subjects and objects (*nodes*) and our model of how to relate them (*edges*). The combination of these parts gives us the basis for making basic statements about the world, what we assert as statements of *fact*.[22] Practitioners call this primary construct a *triple*. Triples are statements in the RDF language that relate a *subject* and *object* through a connecting *property* (or *predicate*). Triples take the form of *s – p - o*, with the subject and property (and object optionally) referenced by an *IRI* (Web link). I expand on the construct of triples in the next chapter; see also the discussion related to Fig. 1.2.

A proposition captures a relation, an assertion about the subject. ‘Any portion of a proposition expressing ideas but requiring something to be attached to it in order to complete the sense, is in a general way relational. But it is only a *relative* in case the attachment of indexical signs will suffice to make it a proposition, or, at least, a complete general name’ (1897, CP 3.463). ‘But the Logic of Relations has now reduced logic to order, and it is seen that a proposition may have any number of

[21] See the discussion of *semsets* in Chap. 10.

[22] If validated, they are indeed *fact* assertions. However, as discussed elsewhere, facts are subject to question and have some degree of *fallibility*; acceptance of an assertion as fact is a matter of *belief*.

subjects but can have but one predicate which is invariably general' (1903, CP 5.151).

We now have a much clearer way for how to build up the assertions in our knowledge representations. We now know that attributes are a Firstness in the universal categories; that Secondness captures all events, entities, and relations; and that Thirdness provides the context, meaning, and ways to indicate what we refer to in the world. We see how context is operative: relations as a construct, for example, are in Secondness, but within relations the mode of representations is in Thirdness. We now have a framework of triadic relations in attributes, external relations, and representations for how to describe things and relate them to one another. Peirce and his architectonic provide the richest, most expressive basis for capturing human language and conducting logical reasoning, both for individuals (particulars) and for concepts (generals). This starting grammar sets the foundation for us to compute and reason over human language for modern KR purposes.[23]

References

1. A. Isnenghi, A Semiótica de CS Peirce e a Gramática Especulativa de Modistae (or, "C.S. Peirce's Semiotic And Modistae's Grammatica Speculativa"), Cognitio-Estudos: Revista Eletrônica de Filosofia, vol. 1809 (2008)
2. C.S. Peirce, Description of a notation for the logic of relatives, resulting from an amplification of the conceptions of Boole's calculus of logic. Mem. Am. Acad. Arts Sci. **9**, 317–378 (1870)
3. P.P.-S. Chen, The entity-relationship model—Toward a unified view of data. ACM Trans. Database Syst. **1**, 9–36 (1976)
4. R. Casati, A. Varzi, Events, *The Stanford Encyclopedia of Philosophy*, https://plato.stanford.edu/archives/win2015/entries/events/
5. M.K. Bergman, KBpedia relations, Part II: An event-action model, in *AI3:::Adaptive Information* (May 2017)
6. I.M. Havel, Scale dimensions in nature. Int. J. Gen. Syst. **24**, 295–324 (1996)
7. J.F. Nilsson, Ontological constitutions for classes and properties, in *Conceptual Structures: Inspiration and Application*, H. Schärfe, P. Hitzler, and P. Øhrstrøm, eds. (Springer, Aalborg, 2006), pp. 35–53
8. J. Doyle, *Hierarchy in Knowledge Representations* (MIT Artificial Intelligence Laboratory, 1977)
9. N. Houser, Peirce, phenomenology, and semiotics, in *The Routledge Companion to Semiotics*, P. Cobley, ed. (Routledge, London, 2010), pp. 89–100
10. F. Lehmann, R. Wille, *A Triadic Approach to Formal Concept Analysis* (Springer, Heidelberg, 1995)
11. J.I. Farkas, A Semiotically Oriented Cognitive Model of Knowledge Representation [Sl: sn] (2008)
12. Santoro, A., Raposo, D., Barrett, D. G. T., Malinowski, M., Pascanu, R., Battaglia, P., and Lillicrap, T., A simple neural network module for relational reasoning, *arXiv: 1706.01427 [cs]* (June 2017)
13. M.K. Bergman, KBpedia relations, Part III: a three-relations model, in *AI3:::Adaptive Information* (2017)

[23] Additional Peirce quotes may be found in my initial article [13].

14. N. Guarino, Concepts, attributes and arbitrary relations: some linguistic and ontological criteria for structuring knowledge bases. Data Knowl. Eng. **8**, 249–261 (1992)
15. S. Sekine, Extended named entity ontology with attribute information, in *Proceedings of the Sixth International Language* (2008), pp. 52–57
16. A.J.J.V. Breemen, J.J. Sarbo, The machine in the ghost: the syntax of mind. Signs Int. J. Semiotics **3**, 135–184 (2009)
17. J.F. Sowa, Top-level ontological categories. Int. J. Human–Computer Stud. **43**, 669–685 (1995)
18. Y. Lin, Z. Liu, M. Sun, Knowledge representation learning with entities, attributes and relations. Ethnicity **1**, 41–52 (2016)
19. B. Weatherson D. Marshall, Intrinsic vs. extrinsic properties, *The Stanford Encyclopedia of Philosophy*, https://plato.stanford.edu/archives/fall2017/entries/intrinsic-extrinsic/
20. Y. Wand, V.C. Storey, R. Weber, An ontological analysis of the relationship construct in conceptual modeling. ACM Trans. Database Syst. **24**, 494–528 (1999)
21. A. Annila, E. Kuismanen, Natural hierarchy emerges from energy dispersal. Biosystems **95**, 227–233 (2009)
22. S. Salthe, Hierarchical structures. Axiomathes **22**, 355–383 (2012)
23. R. Cottam, W. Ranson, R. Vounckx, Hierarchy and the Nature of Information. Information **7**(1), 1 (2016)

Chapter 8
KR Vocabulary and Languages

We have now armed ourselves with basic terminology and a framework around which to express a starting vocabulary for knowledge representation. However, we have one question to answer before we can adopt the languages (or symbol systems) we need to convey that vocabulary: What current languages can capture Peirce's theory of logic while being consistent, coherent, and practical for our needs in knowledge representation? To resolve that question, we need to delve deeper into Peirce's logic and options provided by current language choices. The practical choices resulting from these intersecting forces will then enable us to specify our starting KR vocabulary into a working language suitable for computers.

A vocabulary, in the sense of knowledge systems or ontologies, may and should be expandable, but it is also a controlled vocabulary.[1] That is, we declare new terms and relations to the system and define them at levels required by the formalism to achieve the vocabulary's purpose. Terminology is a social process, driven by user needs and the occasional 'surprising fact,' such as the emergence of the Internet or smartphones, that requires our terms to adapt and our knowledge to grow. Our vocabularies also serve other purposes, such as providing consistent labels to user interfaces or helping interoperate with other knowledge sources.

Early in my exposure to semantic technologies, I encountered the phrase 'ontological commitment.[1]' This phrase was common in the early literature, and, for some reason, I found the idea off-putting. It seemed to me it conveyed buying into one ontology versus another, and I did not like the idea of boxing myself in. There is a significant plurality within the semantic technology community that does not

[1] Some material in this chapter was drawn from the author's prior articles at the *AI3:::Adaptive Information* blog: "Thinking 'Inside the Box' with Description Logics" (Nov 2008); "'Structs': Naïve Data Formats and the ABox" (Jan 2009); "Advantages and Myths of RDF" (Apr 2009); "Uses and Control of Inferencing in Knowledge Graphs" (Mar 2017); "KBpedia Relations, Part V: The Updated KBpedia Grammar" (Jul 2017); "Why I Study CS Peirce" (Aug 2017).

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8_8

like the idea of a governing schema. what is sometimes pejoratively called 'one ring to rule them all' [2].

I have come to embrace a very different view when it comes to the idea of representing knowledge representation. If one believes in reality and truth, and that the purpose of knowledge is to further the understanding of truth, then a knowledge representation system must be based on logics and formalisms that can capture knowledge of every sort and can provide coherent and testable means for discovering and testing new knowledge.

As we will see in the context of the logics and the languages we have chosen for this task, the idea is not to decide what is true and what is false. Our primary objective is not to pass judgment on such topics as fake news. Instead, the design is to test new information we introduce to the system—a system which we deem already to be correct and true within the limits of our assertions—as to whether that new information is consistent and coherent. We first test for consistency in syntax and grammar, and for minimally acceptable completeness. If the new information meets the threshold and does not violate the knowledge graph already in place—that is, we deem it coherent—we accept that new information. If the new information meets the threshold but violates what we already have, we either reject it as incoherent or revise the existing assertions to reflect this new information. In this way, new information may cause us to change the knowledge graph in the system. True, we could extend this basis to test for the falsity of news or other information. However, the more technical point is that we premise our starting basis on 'open' assumptions consistent with the nature of knowledge, as I discuss in the next chapter.

These absolute groundings are further essential to provide a consistent and logical basis for computers to test and analyze current and new assertions. We want the representations available (that is, *features*) for machine learning to reflect and adapt to truth as we test it. While I dislike the phrase 'ontological commitment,' its very scariness helps cement the importance of inspecting (and *re*-inspecting) the formalisms upon which we base our knowledge representation systems.

In creating a feature-rich logic machine for AI, we, of course, want a system that is *sound*, *consistent*, *coherent*, and relatively *complete*. *Sound* is an evaluative criterion where all provable statements are valid in all models. *Consistent* is where all axioms[2] in the knowledge base (domain), subject to deductive reasoning, are true (or, at least, exhibit no contradictions). *Coherent* is where the knowledge base (domain) is consistent and has a high degree of conjunction for nondeductive assertions.[3] C*omplete* is an evaluative criterion where all statements that are true in the model are provable and meet minimum standards.

[2] An *axiom* is a premise or starting point of reasoning. In an *ontology*, each *statement* (assertion) is an axiom.

[3] *Coherence* has a long history within epistemic logic [3], with more recent trends toward accommodating probabilistic measures, though specific methods lacking broad consensus. Peirce's views would tend to align coherence with *abduction* and *pragmatism*. Researchers that embrace aspects of Peirce's approach include Roche [4] and Douvan and Meijs [5].

Our interpreters are both artificial agents and humans. We need us, as humans, to scope the domain and provide the vocabulary, then to construct and oversee the knowledge graph, then to maintain and extend the system, and lastly review tests and tentative assignments before we agree to commit to the knowledge base. These aspects must be suitable for direct translation into any human language. Humans will also have many non-AI uses for the knowledge system, reinforcing the need for understandability and usability. To complete the speculative grammar leg of Peirce's theory of logic, we also need to capture factors relevant to the Secondness of *critic*, the methods of logic, and the Thirdness of the *methodeutic*, the application of these methods to practical problems addressed with practical means.

Logical Considerations

KR practitioners know that the choice of a formalism (*i.e.*, syntax and language) for knowledge representation involves a trade-off in expressivity and practicality [6]. Knowledge graphs and knowledge bases should be scoped based on their anticipated domain of use and populated with 'vivid' knowledge [7]. We cannot feasibly specify all aspects of all items while remaining computationally tractable. We need to *infer*[4] connections, properties, and relationships without explicitly stating all of them. When we do make assignments, we need to know what those statements *entail* based on prior statements.[5] We base inference and entailment on the underlying KR language, as applied through its defined and understood vocabulary and semantics [8].

We could derive the logic that governs our semantics from multiple logic families. One option is propositional logic, but unfortunately this logic evaluates statements such as 'Aristotle is a philosopher' and 'Plato is a philosopher' as unrelated. Another option is set theory, with its basis in sets and members and strong mathematical background. However, set theory lacks predication and ideas of identity and can be problematic for the largest of sets. Advances have continued in set theory, such that the basis for a complete KR language may be at hand [9]. But these limitations led mathematicians and logicians in the nineteenth century, noted previously, to work out the new logic of relations. We now call this field predicate calculus or first-order logic (FOL). Paternal rights to FOL are granted to both Peirce [6] and Frege, though both apparently worked without the other's knowledge.[6]

---

[4] *Inference* is the act or process of deriving logical conclusions from premises known or assumed as true. The logic within and between *statements* in an *ontology* is the basis for inferring new conclusions from it, using software applications known as inference engines or *reasoners*.

[5] *Entailment* is a consequence arising from a statement deemed true based on some underlying logic. The logical consequence is said to be *necessary* and *formal*; necessary, because of the rules of the logic (the conclusion is the consequent of the premises); and formal because the logical form of the statements and arguments hold true without regard to the specific *instance* or content.

[6] Considerable literature deals with this area (search repositories with "set theory" and "knowledge representation"), but set theory is not as developed as FOL and does not have the relationship to Peirce.

Peirce attempted to explicate the basis, applicability, and interpretation of deductive, inductive, and abductive logic, the latter of which he introduced to modern logic. Peirce's primacy of logic proceeds as follows: Decompose every statement into its fundamental premises. Conduct all logical tests, including the implications resulting from inference. Single out anomalies or 'surprising facts' for special attention subject to the pragmatic maxim. Every bit of Peirce's logical work has applicability to knowledge representation.

First-Order Logic and Inferencing

First-order logic is superior to propositional logic in that it allows variables and quantifiers. FOL enables us to say 'x is a philosopher,' where we treat the subject as a variable and turn 'is a philosopher' into a predicate. We can establish relations between these predicates with logical connectors, such as AND, OR, NOT, and IF-THEN statements. We may base class membership on *intensional*[7] or *extensional*[8] grounds. We can apply quantifiers to statements using universal ('for every') and existential ('there exists') quantifiers, as well as use negation. FOL is *sound* (all provable statements are true in all models) and *complete* (all statements which are true in all models are provable). Moreover, Peirce based his 'beta' version of existential graphs on FOL plus identity [24]. Because of these efforts, we view Peirce as one of the founders of first-order logic.

FOL is a powerful and expressive formalism for knowledge representation.[9] Gödel's completeness theorem proved that deduction in FOL is sound and complete. Still, Alonzo Church and Alan Turing proved independently, in 1936 and 1937, that FOL under certain conditions, notably the halting problem or quantification over infinite sets, was undecidable. What this means is that a computer may never calculate some problems to a final result. Various options that reduce the expressivity of FOL have been formulated to overcome the problem of reliably computing to completion. We will speak of one of them, descriptive logics, shortly.

One of the KR formalisms, conceptual graphs, is a complete expression of FOL and is patterned on Peirce's existential graphs [10]. Some proponents call for KR formalisms that can handle higher order logics, such as predicates of predicates. Recent attempts to bridge description logics to category theory using underlying ideas are intriguing and redolent of Peirce [11]. KR and its logical and formal underpinnings, I believe, are set to undergo a renaissance.

---

[7] The *intension* of a class comprises what characteristics its instances have. Intension is most akin to the *attributes* or characteristics of the instances in a set defining its class membership.

[8] The *extension* of a class comprises the enumeration of its instances, consisting of the things to which it subsumes. Most hierarchical relationships in contemporary knowledge bases are explicit assignments to class or type, by definition extensional. Extensional enumeration alone is not feasible for large knowledge bases.

[9] "Knowledge Representation and Reasoning," *Wikipedia*, Oct. 2017.

Inferencing is the drawing of new facts, probabilities, or conclusions based on reasoning over existing evidence. Inferencing is a common term heard in association with semantic technologies. Inference engines (also known as *reasoners*, semantic reasoners, reasoning engines, or rules engines) are the application components. Peirce classed inferencing into three modes: deductive reasoning, inductive reasoning, and abductive reasoning. Deductive reasoning extends from premises known as true and clear to infer new facts. Inductive reasoning looks at the preponderance of evidence to infer what is probably true. Abductive reasoning poses possible explanations or hypotheses based on available evidence, often winnowing through the possibilities based on the total weight of evidence at hand or what is the most practical explanation. Though we may apply all three reasoning modes to knowledge graphs, the standard and most used form is deductive reasoning. Knowledge base completion, a new field, sometimes uses inductive reasoning. Abductive reasoning, to my knowledge, has not yet been applied to knowledge graphs. Inductive and abductive logics offer much additional leverage for a knowledge system, and I expect to see their use increase given their potential usefulness.

We can use inferencing to broaden and contextualize search, retrieval, and analysis. We can create inference tables in advance and layer them over existing data stores for speedier use and automatic invoking of inferencing. More complicated inferencing means that models can also perform as complete conceptual views of the world or knowledge bases. Quite complicated systems are emerging in such areas as common sense and biological systems, as two examples.

We can apply inference engines at the time of graph building or extension to test the consistency and logic of the new additions. Additionally, we may apply semantic reasoners to a current graph to expand queries for semantic search or other reasoning purposes. As noted, as inductive and abductive reasoners become available, they will expand this list of capabilities to include question answering, hypothesis generation and testing, forecasting, decision-making, and real-time systems in robotics. The contributions these types of tools will make are dependent upon the method of logic inference. Based on their syllogistic form, here is a comparison of the three methods [12] (Table 8.1):

Table 8.1 Syllogistic forms for inference methods

Deduction	Induction	Hypothesis/abduction
All M is P (Rule)	S is M (Case)	S is P (Result)
S is M (Case)	S is P (Result)	All M is P (Rule)
S is P (Result)	So, All M is P (Rule)	So, S is M (Case)

In his later years Peirce revised his views about abduction; see Chap. 15

Peirce explicated deductive and inductive reasoning in the clearest of ways and corrected erroneous views of what constituted inductive reasoning. He most importantly recognized that, just as many problems are distributive in nature, so are many of the logical questions. For this, Peirce decomposed the basic syllogism of the Greek philosophers to articulate a third kind of inference, abductive reasoning.

'Deduction proves that something *must* be, Induction shows that something *actually is* operative, Abduction merely suggests that something *may be'* (1903, EP 2:216).

Deductive Logic

Deduction is the 'tracing out the consequences that would ensue upon the truth or falsity of that hypothesis' (nd, MS [R] S64). 'By Deduction, or mathematical reasoning, I mean any reasoning which will render its conclusion as certain as its Premisses, however certain these may be' (1911, MS [R] 856:2).

Deduction is the most common logic in knowledge representations in their current form. We use deductive logic to infer hierarchical relationships, create forward and backward chains, check if *domains* and *ranges* are consistent for assertions, assemble attributes applicable to classes based on member attributes, conform with transitivity and cardinality assertions, and check virtually all statements of fact within a knowledge base. In backward chaining, we conduct the reasoning tests 'backward' from a current consequent or 'fact' to determine what antecedents can support that conclusion, based on the rules used to construct the graph. ('What reasons bring us to this fact?') In forward chaining the opposite occurs; namely, we state a goal or series of goals, and then existing facts (as rules) are checked to see which ones can lead to the goal ('A goal X may be possible because of?'). The reasoner iterates the process until the goal is reached or not; if reached, we may add new knowledge using heretofore unstated connections to the knowledge base. We base consistency tests solely on deductive logic. Either an asserted statement *satisfies* specifications, or it fails.

Like so much Peirce did, he continued to refine his understanding of things through inspection and categorization. Here is one of his later formulations for deduction:

> A *Deduction* is an argument whose Interpretant represents that it belongs to a general class of possible arguments precisely analogous which are such that in the long run of experience the greater part of those whose premisses are true will have true conclusions. Deductions are either *Necessary* or *Probable*. Necessary Deductions are those which have nothing to do with any ratio of frequency, but profess (or their interpretants profess for them) that from true premisses they must invariably produce true conclusions. A Necessary Deduction is a method of producing Dicent Symbols by the study of a diagram. It is either *Corollarial* or *Theorematic*. A Corollarial Deduction is one which represents the conditions of the conclusion in a diagram and finds from the observation of this diagram, as it is, the truth of the conclusion. A Theorematic Deduction is one which, having represented the conditions of the conclusion in a diagram, performs an ingenious experiment upon the diagram, and by the observation of the diagram, so modified, ascertains the truth of the conclusion. Probable Deductions, or more accurately, Deductions of Probability, are Deductions whose Interpretants represent them to be concerned with ratios of frequency. They are either *Statistical Deductions* or *Probable Deductions Proper*. A Statistical Deduction is a Deduction whose Interpretant represents it to reason concerning ratios of frequency, but to reason concerning them with absolute certainty. A Probable Deduction proper is a Deduction whose Interpretant does not represent that its conclusion is certain, but that precisely analogous reasonings would from true premisses produce true conclusions in the majority of cases, in the long run of experience.' (1903, EP 2:297–298; CP 2.267–268)

At present, most current knowledge approaches only use what Peirce calls the 'necessary' deduction. Peirce placed deductive reasoning in Secondness, though he did consider other placements early in his career.[10] The placement in Secondness, however, does make sense because it is the logic of actualness, and whether actual things conform to the premises asserted for them.

Inductive Logic

Induction is 'any reasoning from a *sample* to the whole sampled' (1911, NEM 3:178), with the sample taken at random. Induction is the probabilistic form of Peirce's reasoning triad. Peirce placed the inductive form of reasoning in Thirdness, consistent with its nature of potential.

Peirce wrote much on induction. One succinct summary is that 'Induction consists in starting from a theory, deducing from it predictions of phenomena, and observing those phenomena to see how nearly they agree with the theory' (1903, EP 2:216). And, '... observe that neither Deduction nor Induction contributes the smallest positive item to the final conclusion of the inquiry. They render the indefinite definite; Deduction Explicates; Induction evaluates: that is all' (1908, CP 6.475). Some of his longer passages from his later career expound further on this nature:

> The validity of Induction consists in the fact that it proceeds according to a method which though it may give provisional results that are incorrect will yet, if steadily pursued, eventually correct any such error. The two propositions that all Induction possesses this kind of validity, and that no Induction possesses any other kind that is more than a further determination of this kind, are both susceptible of demonstration by necessary reasoning. (1906, NEM 4:319)

And:

> The true guarantee of the validity of induction is that it is a method of reaching conclusions which, if it be persisted in long enough, will assuredly correct any error concerning future experience into which it may temporarily lead us. This it will do not by virtue of any deductive necessity (since it never uses all the facts of experience, even of the past), but because it is manifestly adequate, with the aid of retroduction and of deductions from retroductive suggestions, to discovering any *regularity* there may be among experiences, while *utter irregularity is not surpassed in regularity by any other relation of parts to whole*, and is thus readily discovered by induction to exist where it does exist, and the amount of departure therefrom to be mathematically determinable from observation where it is imperfect. (1908, CP 2.769)

Consistent with the universal categories, Peirce also saw three subdivisions, or types, within inductive reasoning, with the first being *crude induction*:

> The first and weakest kind of inductive reasoning is that which goes on the presumption that future experience as to the matter in hand will not be utterly at variance with all past experi-

[10] Staat [13] concurs that the placement of the three types of inferential logic into Firstness, Secondness and Thirdness is the order of abduction, deduction and induction, though, when considered in the order of inquiry, it is abduction, induction, deduction. This has been a matter of some confusion to scholars.

> ence. Example: 'No instance of a genuine power of clairvoyance has ever been established: So I presume there is no such thing.' I promise to call such reasoning *crude induction*.... Crude induction is the only kind of induction that is capable of inferring the truth of what, in logic, is termed a universal proposition. (1908, CP 2.756–7)

The second type of induction is the strongest of the three, what Peirce called *quantitative induction*:

> This [type] investigates the interrogative suggestion of retroduction, 'What is the 'real probability' that an individual member of a certain experiential class, say the S's, will have a certain character, say that of being P?' This it does by first collecting, on scientific principles, a 'fair sample' of the S's, taking due account, in doing so, of the intention of using its proportion of members that possess the predesignate character of being P. This sample will contain none of those S's on which the retroduction was founded. The induction then presumes that the value of the proportion, among the S's of the sample, of those that are P, probably approximates, within a certain limit of approximation, to the value of the real probability in question. I propose to term such reasoning *Quantitative Induction.* (1908, CP 2.758)

Lastly, the third type, intermediate between the prior two:

> The remaining kind of induction, which I shall call *Qualitative Induction*, is of more general utility than either of the others, while it is intermediate between them, alike in respect to security and to the scientific value of its conclusions. In both these respects it is well separated from each of the other kinds. It consists of those inductions which are neither founded upon experience in one mass, as Crude Induction is, nor upon a collection of numerable instances of equal evidential values, but upon a stream of experience in which the relative evidential values of different parts of it have to be estimated according to our sense of the impressions they make upon us. (1908, CP 2.759)

In the first type, *crude induction*, we may only detect falsity if we persist the inference long enough. In the strongest second type, *quantitative induction*, the sample is a sub-collection of a population of units; its inductive strength arises from being able to apply the theory of errors. The third type, *qualitative induction*, does not have the advantage of definite populations, but, as we enlarge the sample, the inferential evidence gets stronger (1904, EP 2:302).

Inductive logic is only at the beginning phases of application to knowledge systems, with a leading computational approach for general purposes being inductive logic programming (ILP) [14]. Induction has been used for question answering and to expand search [15] and in areas like knowledge base completion, learning [16], and schema induction [17]. These thrusts deserve more attention, particularly in light of the Peircean bases emphasized throughout this book. Most machine learning involving knowledge bases is a form of inductive reasoning.

Abductive Logic

One of Peirce's signal achievements was to bring the idea of abduction to modern logic. Peirce wrote and revised his views on abduction[11] over his entire working life. A consistent thread in his characterization was that abduction is a kind of infer-

[11] Alternate terms used by Peirce for *abduction* included retroduction, hypothesis, and presumption.

ence that originates a hypothesis by concluding in an explanation, though an indeterminant one for a given observation, often of a curious or surprising nature. Peirce studied abduction because of his belief in its essential role in the scientific method, as well as the unique inferential and logical possibilities it allowed. Peirce went so far as to state that pragmatism is the 'logic of abduction' (1903, CP 1.595 *ff.*). He also called the combination of abduction with induction an 'analogy' (1896, CP 1.65). In 1903 he offered the following syllogistic form for abduction (CP 5.189, EP 2:231):

> The surprising fact, *C*, is observed;
> But if *A* were true, *C* would be a matter of course,
> Hence, there is reason to suspect that *A* is true.

This part of the inference chain begins with the 'surprising fact' or an event or a question. By 1911, however, Peirce wrote, 'I do not, at present, feel quite convinced that any logical form can be assigned that will cover all 'Retroductions' [abductions][12]. For what I mean by a Retroduction is simply a conjecture which arises in the mind' (NEM 3:203–4). However, he also claimed that abduction is the 'only kind of reasoning that opens new ground' (NEM 3:206). Though the syllogistic form still works, Peirce came to believe that there was a qualitative and 'guessing' or 'instinctual' aspect to some abductions, which we need in any case to subject to the pragmatic test. In that same year of 1911 he more broadly stated:

> By Retroduction [abduction] I mean that kind of reasoning by which, upon finding ourselves confronted by a state of things that, taken by itself, seems almost or quite incomprehensible, or extremely complicated if not very irregular, or at least surprising; we are led to suppose that perhaps there is, in fact, another definite state of things, because, though we do not perceive any unequivocal evidence of it, nor even of a part of it, (or independently of such evidence if it does exist,) we yet perceive that this supposed state of things would shed a light of reason upon that state of facts with which we are confronted, rendering it comprehensible, likely (if not certain,) or comparatively simple and natural. (1911, MS [R] 856:3–4)

One way to understand Peirce's insight on abduction, though unclear whether this was his actual method, is to split the idea of hypothesis generation and testing into two parts and rethink their roles. In abduction, the conscious and unconscious mind when faced with a choice rapidly screens and mentally evaluates possible explanations for possible outcomes to test. Multiple possible pathways may explain the diverse potential results. Since the actual testing of a hypothesis using inductive logic incurs time and expense, we try to weigh, in our minds, the potential importance of the hypothesis and its likelihood of results against the time and cost to generate them. A careful weighing in our mind of potentials and costs invokes other signals and perceptions, some perhaps unconscious, such that we may often express our selections as a 'guess.' As part of his belief in the continuity of nature, however, Peirce also noted how often guesses are correct compared to random likelihood.

> Abduction and induction have, to be sure, this common feature, that both lead to the acceptance of a hypothesis because observed facts are such as would necessarily or probably result as consequences of that hypothesis. But for all that, they are the opposite poles of reason, the one the most ineffective, the other the most effective of arguments. The method

> of either is the very reverse of the other's. Abduction makes its start from the facts, without, at the outset, having any particular theory in view, though it is motived by the feeling that a theory is needed to explain the surprising facts. Induction makes its start from a hypothesis which seems to recommend itself, without at the outset having any particular facts in view, though it feels the need of facts to support the theory. Abduction seeks a theory. Induction seeks for facts. In abduction, the consideration of the facts suggests the hypothesis. In induction, the study of the hypothesis suggests the experiments which bring to light the very facts to which the hypothesis had pointed. The mode of suggestion by which, in abduction, the facts suggest the hypothesis is by resemblance,—the resemblance of the facts to the consequences of the hypothesis. The mode of suggestion by which in induction the hypothesis suggests the facts is by contiguity,—familiar knowledge that the conditions of the hypothesis can be realized in certain experimental ways. (1901, CP 7.218)

In succinct terms, and Peirce's definition (1908) for retroduction, it is "the passage of thought from experiencing something, E, to predicating a concept of the mind's creating; the subject of the predication being a specified class to which E belongs, or an indefinite part of such class" (MS [R] 842: 29–30). In more modern terms, we can define abduction (or abductive reasoning) as a mode of symbolic inference that involves the screening and selection from a domain D of the possible explanation paths to an outcome O, possibly involving any element E of D, with the selection of candidate paths for inductive testing based on plausibility, economy, and potential impact. Abduction does not produce probable results, only qualified candidates (most often called hypotheses).

Redux: The Nature of Knowledge

Having discussed the three types of inferential logic, let's now turn our attention to the logical *context* for knowledge, which was a third of the emphasis in Chap. 2. Knowledge, after all, is not merely counting peas or tallying results but is the discovery and verification of 'facts' about the world sufficient to generate belief. A useful framework for evaluating this context goes under the ideas of *closed* or *open* worlds.

The *closed-world assumption*, or *CWA*, is the presumption that what is not currently known as true is false. CWA also has a logical formalization. CWA is the most common logic applied to relational database systems and is particularly useful for transaction-type systems. In knowledge management, for which OWA is most often the best choice, we may use the closed-world assumption in two situations: (1) when the knowledge base is known as complete (*e.g.*, a corporate database containing records for every employee) or (2) when the knowledge base is known as incomplete, but we must derive a 'best' definite answer from incomplete information.

The *open-world assumption*, or *OWA*, is a formal logic assumption that the truth value of a statement is independent of whether or not any single observer or agent knows it. OWA directly conforms to Peirce's view of reality, knowledge, and truth. Missing values are expected and do not falsify what is there. A corollary assumption is that we will always be adding more information to the system, and the design

should promote that fact. OWA is used in knowledge representation to codify the informal notion that in general no single agent or observer has complete knowledge, and therefore cannot make the *closed-world assumption*. The OWA limits the kinds of inference and deductions an agent can make to those that follow from statements that are known to the agent as true (or probably true).

OWA is useful when we represent knowledge within a system as we discover it, and where we cannot guarantee that we have discovered or will discover complete information. Of course, this is the very essence of knowledge. In OWA, we may consider statements about knowledge that are not explicitly stated or inferred as unknown, rather than wrong or false.

Besides this contextual perspective, logic constructs may bring other expressive properties. Here are some of the more important ones that warrant consideration for a KR language:

- *Cardinality*—is where the number of members in a class or type is set or limited, such as hasBiologicalParent set to a cardinality of two.
- *Disjoint*—is where membership in one class excludes membership in another; this is a useful property in that it allows us to 'slice and dice' large, well-designed knowledge bases for more effective processing or analysis.
- *Domain* (property)—a statement that declares the classes or types from which to draw the subject of the assertion.
- *Function*—is any algebraic or logical expression allowable by the semantics and primitives used in the KR language where an input is related to an output.
- *Inverse*—is when a property, say, `hasParent`, can be defined as the inverse property of `hasChild`.
- *Negation*—is a unary operation that produces a value of *true* when its operand is false and a value of *false* when its operand is true.
- *Range* (property)—a statement that declares the classes or data types from which to draw the object data or types of an assertion.
- *Reflexivity*—is when every element of X is related to itself, and every class is its own subclass, such as every person is a person.
- *Rules*—we may supplement the underlying logic with rules engines (if-then, exclusions, inclusions) that may add further to the specifications allowed.
- *Symmetric*—is when A relates to B exactly if it relates B with A.
- *Transitivity*—is when item A is related to item B, and item B is related to item C, then A is also related to C; this is the critical property for establishing inheritance chains.

The use or not of these constructs both may affect how reasoners operate in a knowledge base and may add to the feature pool available to machine learners. An optimal KR language would provide all of these capabilities.

In operating KR and knowledge management systems, closed-world applications can interface with the open-world graph of the KR system via agreed, canonical data transfer models. Proper design can readily integrate simulation models, search engines, forecasting software, language processors, or transaction systems with the knowledge representation, enabling all parts to contribute to their strengths.

Given the importance of context to knowledge representation, we devote much of Chap. 9 to the open world topic. We discuss architectures and platform designs that enable integrating closed and open systems in Chap. 13.

Particulars, Generals, and Description Logics

We began our logic discussion centered on first-order logic, and its suitability to our KR needs, save for its lack of decidability. Early researchers in knowledge representation developed description logics specifically to overcome this lack, as well as other pragmatic considerations around KR [18]. Description logics are one of the underpinnings to the semantic Web. They grew out of earlier frame-based logic systems from Marvin Minsky and also semantic networks. Description logics (DLs) as a term and discipline was first defined in the 1980s by Ron Brachman, among many others [18]. DLs or fragments thereof are quite akin to FOL, but slightly less expressive, lacking negation or unique name assumption, as examples. DLs can (usually) be made decidable, that is, able to resolve all mathematical expressions in the language, while FOL is not. Description logics firmly embrace the open-world assumption, a central aspect of knowledge systems, as we continue discussing in the next chapter.

One aspect of description logics and their semantics is that they traditionally split concepts and their relationships from the different treatment of instances and their attributes and roles. This split corresponds nicely to the split between generals and particulars, respectively, that we have adopted from Peirce. In description logics, we know the concept split as the TBox (for terminological knowledge, the basis for T in TBox) and the instance split as the ABox (for assertions, the basis for A in ABox). A TBox is a conceptualization associated with a set of facts. TBox statements describe this conceptualization through a set of concepts and relationships between them. In its entirety, a TBox specifies the schema for the conceptualization, that is, an *ontology*. All generals, from a Peircean perspective, belong to the TBox.

The ABox is the complement that describes the instances (or instance *records*) and their attributes that populate that conceptualization. In these regards, extensional relationships dominate in the TBox, intensional ones in the ABox.[12] Though no formal or actionable difference exists between the ABox and TBox in description logics, keeping them separate is often a practical design choice.

Of course, the choice of KR logic and formalism must consider how to handle other types of relations and the whole panoply of trade-offs incurred during actual implementation, including importantly usability, toolsets, and maintenance. None

[12] One confusing aspect is that some computer science database textbooks use the term ‘intension’ to refer to the schema of a database, and ‘extension’ to refer to particular instances of a database [19], an unfortunate use also by one of the major textbooks in description logics [18]. Peirce noted similar confusions long ago (*c.f.*, CP 2.393), and as a result tended to use the term *comprehension* over intension.

of these logical options prevents, in and of themselves, making inconsistent assignments or perhaps introducing cycles or other errors into our knowledge bases. Whatever logic or formalism we choose, it is essential to test for internal consistency and coherence. Keeping proven reasoners at the ready while developing is but one example of best practices when building or maintaining knowledge bases. I discuss these and related best practices in Chap. 14.

Pragmatic Model and Language Choices

The preceding discussion on logic has informed us about how to select a desirable knowledge representation language. We want a language that can model and capture intensional and extensional relations, one that potentially embraces all three kinds of inferential logic; one that is decidable; one that is compatible with a design reflective of particulars and generals; and one that is open world in keeping with the nature of knowledge. We want this KR language, or languages, to accommodate Peirce's guidance, especially that related to practicality and various use and adoption criteria. In short, we seek pragmatic choices that balance the trade-off in expressivity and tractability.

Many, especially in the semantic Web community, have chosen topic maps or the Resource Description Framework (RDF) as their sole modeling basis. At the more expressive end of the spectrum, others have advocated conceptual graphs and the more powerful constructs of first-order logic. We have chosen more of a middle path: we use RDF as our data model language while using the Web Ontology Language, OWL 2, as our language for the knowledge graph, and the basis for mapping to external information sources. Both RDF and OWL 2 conform to description logics, and both are open, standardized efforts from the World Wide Web Consortium. Via these choices, we also gain access to many other standards and tools from the W3C, as the remainder of this section describes.

RDF: A Universal Solvent

RDF (Resource Description Framework) is a family of World Wide Web Consortium (W3C) specifications originally designed as a *metadata* model. In practice, RDF has become a general method for modeling information through a variety of syntax formats. In RDF, we make *statements* about *resources* in the form of *subject-predicate-object* expressions, called *triples*.

A *triple* may sound fancy, but substitute verb for *predicate* and noun for *subject* and *object*. In other words: *Dick sees Jane* or the *ball is round*. It may sound like a kindergarten reader, but it is how we can easily represent data and build it up into more complex vocabularies and structures. We combine multiple statements to flesh out our understanding of individual things. Since *subjects* or *objects* may act as

'nodes' to one another (the *predicate*s act as connectors or 'edges'), we may create hierarchical and relationship structures as we add statements (see Fig. 1.2). As we aggregate these node-edge-node triple statements, a network structure emerges, known as the RDF *graph*.

The referenced 'resources' in RDF triples have unique identifiers, IRIs, that are Web compatible and Web scalable, such as http://mkbergman.com/me/about.rdf. These identifiers can point to precise definitions of predicates or refer to specific concepts or objects, leading to less ambiguity and clearer meaning or semantics.

We can apply RDF triples equally to *unstructured* (say, text), *semi-structured* (say, HTML documents), or *structured data* sources (say, standard databases). This flexibility makes RDF almost a 'universal solvent' for representing data structure.[13] By defining new types and predicates, we can create more expressive vocabularies within RDF. This expressiveness enables RDF to define controlled vocabularies with exact semantics.[14] These features make RDF a powerful data model and language for data federation and interoperability across disparate datasets.

We represent instance data simply as k*ey-value pairs (*also known as a *name–value pairs* or *attribute–value pairs*), where the *subject* is the instance (particular) itself, the *predicate* is the attribute, and the *object* is the value. We may express all or part of the data model as a collection of tuples `<attribute name,value>` where each element is a key-value pair. The key is the defined *attribute,* and the value may be a reference to another object or a literal string or value. In the base form of the RDF data model, useful in describing static things or basic facts, we keep it simple: no range or domain constraints; no existence or cardinality constraints; and no transitive, inverse, or symmetrical properties. A combination of these for the same subject forms an instance *record*, part of the ABox as noted above. A *dataset* is a combination of one or more *records*, transmitted as a single unit (though we may break it into parts due to size), including simple text files.

Because of RDF's universality and open standards, a vibrant ecosystem exists of translators to alternate syntaxes, languages, and serializations, with JSON and straight text (through comma-separated value and RDF formats) being the most popular. Because of its diversity of serializations and its simple data model, it is also easy to create new converters using RDF. Generalized conversion languages such as GRDDL provide framework-specific conversions, such as for microformats. Once in a standard RDF representation, it is straightforward to incorporate new datasets or new attributes, and to aggregate disparate data sources as if they came from a single source. This universality enables meaningful composition of data from different applications regardless of format or serialization.

RDFS (RDF Schema) is the next layer in the RDF stack designed to overcome some of the baseline limitations. RDFS introduces new predicates and classes that bound these semantics. Importantly, RDFS establishes the basic constructs necessary to create new vocabularies, principally through adding the *class* and *subClass* declarations and adding *domain* and *range* to *properties* (the RDF term for *predi-*

[13] See Chap. 10.

[14] RDF Semantics, World Wide Web Consortium, 2004.

cates). RDFS supplies the basic *data types* used in the vocabulary, which are predefined ways that attribute values may be expressed, including various literals and strings (by language), URIs, Booleans, numbers, and date-times.[15] Many useful RDFS vocabularies exist, and it is possible to apply limited reasoning and inference support against them. We can also use this intermediate canonical form, now with a bit of added schema, to communicate queries, context selections, and labels and forms for user interfaces. The RDFS structure and label properties allow us to populate context-relevant drop-down lists and auto-complete entries in user interfaces solely from the input data and structure. This ability is generalizable using a reasonably straightforward input schema.

Thus, RDF is a framework for modeling all forms of data, for describing that data through vocabularies, and for interoperating that data through shared conceptualizations and schema. We can represent, describe, combine, extend, and adapt data and their organizational schema flexibly and at will using the HTTP protocol. Importantly, via existing or easily constructed converters, we can do this without the need to change what already exists. We can augment our existing relational data stores, and transfer and represent our current information as we always have. The RDF data model provides an abstract, conceptual framework for defining and using metadata and metadata vocabularies, as well as for our primary purpose of representing a message or data in a readily consumable form. In our design, RDFS is the language mostly focused on the ABox.

OWL 2: The Knowledge Graph Language

We need something more expressive and powerful for the conceptual and reasoning aspects of the TBox. Our choice, again a W3C standard, is *OWL 2,* the Web Ontology Language designed for defining and instantiating formal Web *ontologies*, or *knowledge graphs*. An OWL ontology may include descriptions of *classes*, along with their related *properties* and *instances*. A variety of OWL dialects may be employed, specialized to process more quickly for different specific needs, such as rule testing or querying. OWL 2 is the primary formalism used in KBpedia.[16] OWL 2 provides nearly complete capabilities from description logics and offers some tricks of its own in metamodeling. An inspection of OWL's standardized direct semantics provides further detail in these regards [20].

Before OWL 2, the initial version of OWL was more challenging to ensure decidability. The earlier version also did not allow users to treat classes as instances depending on context. Fortunately, OWL 2 added a metamodeling technique called *punning*. When used for *ontologies*, it means to treat a thing as both a *class* and an *instance*, with use depending on context.

[15] See, for example, XSD (XML Schema Definition) for more information.

[16] References herein to OWL refer to OWL 2 unless otherwise noted.

While we are using OWL 2 as our standard KBpedia language, we are not relying on OWL's distinction of object and datatype properties for external relations and attributes, respectively. External relations, it is true, by definition are object properties, since both subject and object are identifiable things. However, attributes, in some cases such as rating systems or specific designators, may also refer to controlled vocabularies, which can (and, under best practice, should) be object properties. So, while most attributes use datatype properties, not all may. Relations and attributes are a better cleaving since we can use relations as patterns for fact extractions and the organization of attributes gives us a cross-cutting means to understand the characteristics of things independent of the entity type. So while the splits most often conform to object properties for external relations and datatype properties for attributes, relying on this split is not dependable.[17] In any case, all of these assignments become valuable potential features for machine learning, in addition to the standard text structure.

OWL provides sufficient expressive richness to describe the relationships and structure of entire worldviews, or the so-called terminological (TBox) construct in description logics. Thus, we see that the complete structural spectrum of description logics can be satisfied with RDF and its schematic progeny, with a bit of an escape hatch for combining poorly defined or structural pieces using undecidable OWL fragments.

W3C: Source for Other Standards

We can use many other W3C standards in the system for graphics standards, rules, selected vocabularies, translators and validators, and the like.[18] I will mention only three of the prominent ones we use. The first capability is the RDF query language, SPARQL [21], which provides querying of either the ABox or the TBox or driving reports and templated data displays. Utilizing RDF's simple triple structure, SPARQL can also be used to query a dataset without knowing anything in advance about the data, which is a useful discovery mode. The second contributor is the Simplified Knowledge Organizational System (SKOS) vocabulary, which we use as a concept classification language [22]; see further discussion of SKOS below. The last notable contribution is linked data, which is a set of recommended techniques and guidelines for exposing RDF resources to the Web. It is the right technique to use if open sharing. The Linking Open Data movement that is promoting this pattern has become highly successful, with billions of useful RDF statements now available for use and consumption online.

[17] For interoperability with external datasets where our logic model and vocabulary provides, we allow assignment of either object or datatype properties, which we reconcile at the schema level.

[18] Consult the *World Wide Web Consortium*'s W3C Web site at https://www.w3.org/.

The KBpedia Vocabulary

In Chap. 7 we discussed the main terminological aspects to our approach to knowledge representation, grounded in Peirce's universal categories of Firstness, Secondness, and Thirdness, the topic of Chap. 6. We then added commentary about logical and inferencing needs, with pragmatic choices being made for the W3C languages to implement these design considerations. We are now in a position to provide a working introduction to the KBpedia vocabulary and the upper structure of its knowledge graph, the KBpedia Knowledge Ontology, or KKO. We supplement these materials with online resources and further KBpedia details; see Appendix B.

By design, this introduction is only a summary. KBpedia is under active use and development as of the time of this writing, and we expect details to change, perhaps in material respects. As a result, I try to keep the summary and explanations general enough to retain some longevity. Again, for the current specifications, see the online resources.[19]

Structured on the Universal Categories

In keeping with the universal categories, we organize KBpedia under the standard RDF *root* of 'Thing'[20] into the three main and sole branches of *Monads* (Firstness, 1ns or 1), *Particulars* (Secondness, 2ns or 2), or *Generals* (Thirdness, also called SuperTypes, 3ns or 3). We also tend to categorize the upper structure of KBpedia, formally known as the KBpedia Knowledge Ontology, or KKO, in a triadic manner. This triad follows the senses of possible building blocks (1ns), actual things in the category (2ns), and generalizations about the category (3ns). Applicable categories in the upper structure may be prefixed with 1, 2, or 3 to keep track of these splits. We list the features available for machine learning in Appendix C.

The *Monads* branch (1ns) captures the qualities, constituents, characteristics, or attributes of the actual things or general realities that comprise human knowledge. We can talk about these things, but, once we do, we instantiate the quality, so that our actual statements and assertions about these things occur in the other two branches. Nonetheless, from a modeling standpoint, it is still possible to relate statements about monads to their placements in the knowledge graph, enabling some reasoning, if desired, by proxy, using this branch. The *Particulars* branch (2ns) represents all individual, real things across which knowledge may pertain. Entities and events are the two main sub-branches of the particulars, with the third sub-

[19] KBpedia and its documentation are available for free under open source licensing; for current specifications and downloads see http://kbpedia.org.

[20] 'Thing' is the same as 'resource' in RDF and is the existential starting node for an OWL knowledge graph.

branch being the instantiation of monads. The *Generals* branch (3ns), the third of the three, comprises all concepts, types, and generalizations we may make about the things to which knowledge may refer, as well as the concepts and generalizations that apply to knowledge itself and how it is represented and communicated. Its three main sub-branches represent constituents of reality, relations (predications), and manifestations, including matter, life, and symbols. KBpedia's upper triadic structure is the domain of discourse for knowledge representation and its potential scope. Naturally, since it is absurd to capture all instances or all generalities related to knowledge, KBpedia is not a complete representation. Instead, it is a scaffolding of the more pivotal joints in the knowledge skeleton, which provide reference tie-ins for specific knowledge domains to expand coverage using similar construction methodologies.

By comparison, please note that most existing KR graphs or ontologies correspond to the *Generals* branch in our design, with the data or ABox corresponding to the *Particulars* branch in KBpedia and KKO. However, none of these other existing systems are triadic, and none are explicit about modeling meaning or context.

Three Main Hierarchies

We embed three hierarchical backbones within the KKO structure. One is for instances (particulars) that correspond to the ABox. One is for relations. The third is for classes, types, and generalities, corresponding to the TBox. I overview these three backbones under the instances, relations, and generals vocabulary sections below.

The Instances Vocabulary

More than 95% of the knowledge items in KBpedia are instances, either entities or events. The constituent knowledge bases for these include Wikipedia and Wikidata, described in Chap. 11. All instances in KBpedia belong to one or more types, which are the subject of the *Generals* branch. However, we may use different characteristics to describe and compare instances to one another, as shown in Table 8.2 below. These descriptions relate to how we characterize the instances, not where they occur in the general conceptual schema. These items may be discovered and inspected using the online KBpedia browser.[21] Hopefully, most of these items are pretty clear. You may obtain full definitions and other contextual specifications from the open-source KKO artifact.

The *Monoidal Dyads* sub-branch captures the items in the *Monad* main KKO branch, previously noted, as reified as actual instances. Its triadic splits follow the general form. *Events* were discussed at length in Chap. 7, and capture the span from Peirce's absolute chance (*tychism*, or spontaneity) to his Thirdness of *synechism*.

[21] See http://kbpedia.org/knowledge-graph/; links for how to use are provided on that same page.

Table 8.2 Full, upper hierarchy of the KBpedia particulars[a]

- 2-Particulars
  - 1-Monadic Dyads
    - 1-Monoidal Dyad
    - 2-Essential Dyad
    - 3-Inherential Dyad
  - 2-Events
    - 1-Spontaneous
    - 2-Action
      - 1-Exertion
      - 2-Perception
      - 3-Thought
    - 3-Continuous
      - 1-Triadic Action
      - 2-Activities
      - 3-Processes
  - 3-Entities
    - 1-Single Entities
      - 1-Phenomenal
      - 2-States
        - Situations
      - 3-Continuants
        - Time
          - 1-Instants
          - 2-Intervals
          - 3-Eternal
        - Space
          - Points
          - Areas
            - 2D-Dimensions
          - Space/Regions
            - 3D-Dimensions
    - 2-Part Of Entities
      - 1-Members
      - 2-Parts
      - 3-Functional Components
    - 3-Complex Entities
      - 1-Collective Stuff
      - 2-Mixed Stuff
      - 3-Compound Entities

[a]Various downloads of KKO and KBpedia may be obtained from https://github.com/Cognonto/kko. To view the KKO artifact, you will need an ontology editor, such as the open source Protégé ontology development environment

Actions are the actual instances of action and may arise from perception, exertion, or thought, as previously discussed. *Entities* span from single ones, according to the universal categories, to parts of entities and then combinations of entities, again in correspondence with the categories. *Complex entities* may span from simple collections to mixtures to compound ones, the latter best exemplified by the constituent entities comprising the whole universe.

To my knowledge, no existing knowledge graph or ontology other than KBpedia provides a similar classification scheme for the nature of instances, likely because none of the other systems are modeled using Peircean perspectives.

The Relations Vocabulary

I provided a fairly detailed introduction to the *Relations* vocabulary in Chap. 7. We model these relations as abstract possibilities under the relational monads (2ns and 3ns) in the *Monads* main branch. We model these relations as concepts used in knowledge representation according to the *Predications* branch of 2ns in Table 10.2. Note that KKO represents the concepts of these relations, in addition to the relational expressions themselves. In KKO, we provide the specific relations as object or data type properties (or both), depending. We include the separate listing of relations as classes so that we may talk about and reason over them as *concepts*. Using them in actual triples requires the properties.

Since we already introduced the top level of the relations in Chap. 7, let's move on to the next two levels. The next Table 8.3 shows the second level of the relation hierarchy, with the following Table 8.4 showing the third level. I will highlight some aspects of these tables where they may not be entirely evident, according to the 1ns, 2ns, and 3ns relation sub-branches.

Table 8.3 The first two levels of relations

						Definition	Concepts
Inference	**Relations—assertions—facts**	**Attributes**	A:A	1°	**Intrinsic**	Innate characteristics or essences of single entities or events (particulars)	Oneness, qualities, feelings, inherent, negation, is, has, intensional, naturalness, internal, innateness
				2°	**Adjunctual**	Events that may occur to single entities or events (particulars) that help characterize it	Birth, death, marriage, events, accidents, surprises, happenings, extrinsic, adjunctual
				3°	**Contextual**	Circumstances or placements of single entities or events (particulars) that help characterize it	Space, time, continuity, contiguous, smooth, otherness, ratings, level, situational (w/ respect to A), sensible, contiguous, all placements thereto, derivative, genres, classificatory, rankings
		External relations	A:B	1°	**Simple**	A simple, direct relationship (no intermediaries) between two different objects (entities, events, or their types, considered as instances)	Is a, simple without parts, part of, members in types or classes, genealogical roles (parent, child, brother), identity, extensional
				2°	**Copulative**	Relationships of combination, membership, quantity, action, or circumstance	Accidental, real, place, time, situation, quantity, facets, aspects, conjunctive, lists, one-to-many, many-to-one, sum of, contextual, verbs
			re A	3°	**Mediative**	True, triadic external relations, such as 'A gives B to C'; relationships of relevance, meaning, or explanation—namely, Thirdness—about subjects and types	Concepts, generalities, similarity, genres, aspects, comparison, performance, thought, triadic, agreement/difference, placement in space/time (contiguity), conditional, reasoning, classification
		Representations		1°	**Denotatives**	Icons or symbols that name or describe the subject	Names, labels, images, descriptions, definitions, denotations, icons, designations, proper nouns
				2°	**Indexes**	Indirect references or pointers that help situate or draw attention to the subject	URIs, identifiers, keys, indices, references, semes, propositions (w/o objects), codes, selections, directional, citations, pronouns
				3°	**Associatives**	A situational and contextual assertion of proximity, affiliation, or adjacency of the subject with regard to any contiguity	See also, lists, links (incoming + outgoing), associations, likenesses, resemblances

Table 8.4 The first three levels of relations

						Definition
Attributes	A:A	1°	Intrinsics	1°	**Qualities**	An internal characteristic or aspect of an object; collectively these define intensionally what kind of thing to which the object belongs, though that relationship is not intrinsic
				2°	**Components**	A contributing part of or integral input or aspect that adds to the understanding about the subject (A)
				3°	**Forms** (configurations)	Forms or arrangements that are of the nature or perceivable of the subject (A)
		2°	Adjunctual	1°	**Quantities**	A characteristic of a subject (A) that is expressed as a number quantity
				2°	**Eventuals**	Chance, accidental, or planned occurrences that directly involve subject (A)
				3°	**Extrinsics**	External events or circumstances that directly involve subject (A) or help define the nature or reality of subject (A)
		3°	Contextual	1°	**Situants**	Attributes or characteristics that help situate, or place in a locational context, the subject (A)
				2°	**Ratings**	An assigned value or characterization that orders subject (A) in relation to other subjects for a given attribute
				3°	**Classifications**	A characterization of subject (A) that involves evaluating subject (A) and providing a multifactor typing, coding, or value in relation to a given attribute or set of attributes
External relations	A:B	1°	Simple	1°	**Equivalences**	A simple relationship between a subject (A) and an object (B) that asserts equalness or sameness
				2°	**Parts**	A simple relationship where the object (B) is a part of or component of subjext (A), including the idea of 'whole'
				3°	**Descendants**	A simple relationship where object (B) is a direct child or parent or subsumption (hyponym) or supersumption (hypernym) to subject (A)
		2°	Copulative	1°	**Identities** (is B)	This simple relation is for all of the is-a relations to types (B) for subject (A)
				2°	**Actions**	Simple relations of energetics, perception, or thought of subject (A) to some other object (B)
				3°	**Conjoins**	Relations that involve the joining of a subject (A) to an object (B) via an intermediate object
		3°	Mediative	1°	**Comparisons**	Relations that compare, contrast, or size up similarities or differences or overlaps between subject (A) and object (B)
				2°	**Performances**	Relations of quantity or rank for how subject (A) performed in relation to object (B)
				3°	**Circumstances**	Relations of subject (A) to external circumstances, situations, or contexts

						Definition
Representations	re:A	1°	Denotatives	1°	**Media**	Iconic images or sounds that invoke the identification with a given object or representation. Media in this sense draws attention to the object
				2°	**Labels**	Symbolic text strings that help to name or draw attention to a particular object
				3°	**Descriptions**	Text strings that may be longer than labels and provide additional or contextual information or specify attributes about the object, beyond drawing attention
		2°	Indexes	1°	**Pointers**	Physical or symbolic indicators of a given thing and which draw attention to it
				2°	**Identifiers**	Generally (unique) symbols or strings that provide a key to the given subject, often within some conventional scheme for generating and recognizing the token assigned
				3°	**Codes**	An assigned symbolic token or string that groups the object with similar items; the generation and interpretation of the token are (often) done in relation to an understood schema
		3°	Associatives	1°	**Lists**	An aggregation, either ordered or unordered, of objects similar to one another with respect to given characters or types
				2°	Relateds (*see also*)	An indicator of some nature to other objects similar or related to the given object; the criteria and degree or strength of relationship between the items are
				3°	Augments	An indicator to an external factor in relation to the object, which factor itself leads to still further explanations

Attributes Relations (1ns)

Attributes are the intensional characteristics of an object, event, entity, type (when viewed as an instance), or concept. We split attributes into three categories. The *intrinsic* relations are innate characteristics or essences of single entities or events (particulars). Example concepts include oneness, qualities, feelings, inherent, negation, is, has, intensional, naturalness, internal, and innateness. *Qualities*, one intrinsic sub-branch, are an internal characteristic or aspect of an object; collectively these define intensionally what types to which the object belongs, though that relationship is not intrinsic. *Elementals* are a contributing part of or integral input or aspect that adds to the understanding of the subject. *Configurations*, or forms or arrangements, are of the nature or perceivable of the subject. The *adjunctual* are occurrences that may occur to single entities or events (particulars) that help characterize it. Example concepts include birth, death, marriage, events for the individual, accidents, surprises, happenings, extrinsic, and adjunctual. Though Peirce used 'accidental' much, he applied it in most cases to 'accidental actuals'; thus, 'adjunct' better captures potentiality. Within adjunctuals we have *quantities*, characteristics of a subject that we express as a numerical quantity; *eventuals*, chance, accidental, or planned occurrences that directly involve a subject; or *extrinsics*, which are external events or circumstances that directly involve the subject or help define the nature or reality of it.

The *contextual* relations are circumstances or placements of single entities or events (particulars) that help characterize it. Example concepts include space, time, continuity, and classificatory. These relations include anything that has gradation over space and time, including ideas and concepts that also shade. The three sub-branches of the contextual relations are *situants* (1ns), which are attributes or characteristics that help situate, or place the subject in a locational or time context; *ratings (2ns)*, which are an assigned value or characterization that orders the subject in relation to other subjects; or *classifications (3ns)*, which are characterizations of the subject in regard to multifactor typing, coding, or value in relation to a given attribute or set of attributes.

External Relations (2ns)

External relations are assertions between an object, event, entity, type, or concept and another particular or general. *External relations* also have three subcategories, with the first (1ns) being *direct*, which are simple relationships (no intermediaries) between two different objects considered as instances. Example concepts include *is a*, simple without parts, part of, members in types or classes, or genealogical roles (parent, child, brother). Direct relations, in turn, have three sub-branches, including *equivalences*, a simple relationship that asserts equality or sameness; *parts*, a simple relationship where the object is a part or component of the subject; or *descendants*, which are a simple relationship where the object has a genealogical, subsumption, or supersumption relationship to the subject.

The *Copulative* relations are the 2ns sub-branch of the *External Relations*. They convey combination, membership, quantity, action, or joins.[22] The three sub-branches include *typings*, all of the *is-a* relations to types; *actions*; and *conjoins*, relations that involve the joining of a subject to an object via an intermediate object. We should note that the two sub-branches of *external relations* to this point, the *direct* and *copulative*, represent the *simple* relations according to Peirce's logic of relatives [23].

The last sub-branch in Thirdness of the *External Relations* is the *Mediative* relations, which are the true, triadic external relations, such as 'A gives B to C.' These are the relationships of relevance, meaning, explanation, or cognition. Sub-branches of the mediative relations are *comparisons*; *performances*, which are relations of quantity or rank for how a subject performed in relation to an object; or *cognitives*, which relate to thinking, knowing, or representing. While we might consider thoughts as something that occurs internally, thoughts are not innate and are internal representations of the external world.

Representation Relations (3ns)

The Thirdness branch of relations is the *Representations*, which are signs (1905, CP 8.191) and the means by which we point to, draw, or direct attention to, or designate, denote, or describe a particular object, entity, event, type, or general. The first Representation sub-branch is the *denotatives*, icons or symbols that name or describe the subject. Its three sub-branches are *media*, iconic images or sounds that invoke the identification with a given object or representation; *labels*, symbolic text strings that help name or draw attention to a particular object; or *descriptions*, text strings that may be longer than labels and provide additional or contextual information or specify attributes about the object, beyond drawing attention.

The second branch of the *Representations* is the *indexes*, indirect references or pointers that help draw attention to the subject. *Indexes* are references or attention-directors to a subject.[23] The three sub-branches of *indexes* are *pointers*, physical or symbolic indicators of a given thing and which draw attention to it; *identifiers*, such as URIs, which are generally (unique) symbols or strings that provide a key to a given subject, often within some conventional scheme for generating and recognizing the token assigned; and *codes*, an assigned symbolic token or string that groups the object with similar items.

---

[22] To gain a feel for this relation, see (in English), https://en.wikipedia.org/wiki/List_of_English_copulae.

[23] Here is a supporting quote from Peirce on the notion of *index*: "Indices may be distinguished from other signs, or representations, by three characteristic marks: first, that they have no significant resemblance to their objects; second, that they refer to individuals, single units, single collections of units, or single continua; third, that they direct the attention to their objects by blind compulsion. But it would be difficult if not impossible, to instance an absolutely pure index, or to find any sign absolutely devoid of the indexical quality. Psychologically, the action of indices depends upon association by contiguity, and not upon association by resemblance or upon intellectual operations" (1901, CP 2.306).

The last branch (3ns) of the *Representations* is the *associatives*, contextual assertions of proximity, affiliation, or adjacency of the subject to any contiguity.[24] The three sub-branches are *lists*, either ordered or unordered aggregations of objects similar to one another with respect to given characters or types; *relateds* (*see also*), which are indicators of some nature to other objects similar or related to it; or *augments*, which are an external indicator that leads to still further explanations.

Current practice rarely incorporates any of these vocabulary aspects—discussed in the sections above on *monads*, *particulars*, and *relations*—in knowledge graphs and ontologies. The Peirce-inspired design of these first and second branches of KKO demonstrate a logical, recursive approach to organizing knowledge domains. These aspects provide a deeper, more abstract pool of features of possible use to machine learners. Reasoning tasks should also see a jump-step-up in capabilities.

The Generals (KR Domain) Vocabulary

The core of KBpedia, as it is for most current knowledge graphs, is the TBox, or the conceptual schema for the domain. This KKO branch is populated entirely with generals and is where most current reasoning with knowledge graphs occurs. This central schema is also the point at which it is best to link external knowledge bases into the system.[25] Because of this mapping role, we term the nodes in the *Generals* branch as *reference concepts* (or *RefConcepts* or *RC*s). All RCs are OWL *classes*. More than 95% of the 55,000 current *base concepts* in KBpedia are RCs. These RCs provide a rich pool of tie-in points for enabling integration with external sources.

The RCs are organized into natural hierarchies of related kinds or *types*, what we term *typologies*.[26] Typologies are multi-instance hierarchies; each one has a top-level node called a *SuperType*. The distribution of typologies in KBpedia covers the scope of substantive human knowledge, and all of the SuperTypes, by design, are part of the upper KKO knowledge graph. Also, we design the typologies as *disjoint* (nonoverlapping) with one another where possible, which promotes efficiency in reasoning and other analyses. Typologies are explained further in Chap. 10.

---

[24] "Association is the only force which exists within the intellect, and whatever power of controlling the thoughts there may be can be exercised only by utilizing these forces; indeed, the power, and even the wish, to control ourselves can come about only by the action of the same principles. Still, the force of association in its native strength and wildness is seen best in persons whose understandings are so little developed that they can hardly be said to reason at all. Believing one thing puts it into their heads to believe in another thing; but they know not how they come by their beliefs, and can exercise no control over the inferential process. These unconscious and uncontrolled reasonings hardly merit that name; although they are very often truer than if they were regulated by an imperfect logic, showing in this the usual superiority of instinct over reason, and of practice over theory" (1886, CP 7.453).

[25] While instance mappings are possible, it is more effective to define relationships at the class level, since member instances can then be inherited without direct assignment.

[26] These are a critical design component of our approach, which we discuss at length in Chap. 11.

We provide a complete view of the upper *Generals* branch in Table 10.1, in the chapter on *typologies*. The structure of the *Generals* branch follows our understanding of Peirce's universal categories. Note that the structure enables us to organize and reason over predicates and attributes, as well as the more standard classes of things that encompass the knowledge domain. Further, via ties to the other two main KBpedia branches, *Monads* and *Particulars*, we can also significantly expand the abstract characterization and reasoning of all things within that domain.

Other Vocabulary Considerations

Before we close out discussion of the KBpedia vocabulary, we need to touch upon two further considerations: the vocabulary terms provided by W3C standards and the vocabulary for mapping external sources to KBpedia. As Table 8.5 shows, we rely much on the SKOS vocabulary in KBpedia for various annotation labels and some conceptual relationships. We use RDFS for SKOS, property range and domain declarations, and property hierarchies. OWL is used to declare classes and to split our properties into annotations, object properties, and data type properties.

SKOS, or the Simple Knowledge Organization System [22], is a formal language and schema designed to represent such structured information domains as thesauri, classification schemes, taxonomies, subject-heading systems, controlled vocabularies, or others, in short, most of the 'loosely defined' ontology approaches discussed herein. It is a W3C initiative more fully defined in its SKOS Core Guide. As an RDF Schema, SKOS adds some language and defined relationships to the RDF baseline. SKOS also has a rich set of annotation and labeling properties to enhance the human readability of schema developed in it.

Table 8.5 External mapping and annotation properties

RDFS	rdfs:domain rdfs:range rdfs:subClassOf rdfs:subPropertyOf
OWL	owl:AnnotationProperty owl:Class owl:DatatypeProperty owl:disjointWith owl:equivalentClass
SKOS-Preferred	skos:altLabel skos:broaderTransitive skos:definitionskos:hiddenLabel skos:narrowerTransitive skos:prefLabel
SKOS-Optional	skos:broader skos:changeNote skos:editorialNote skos:example skos:historyNote skos:narrower skos:note skos:relatedskos:scopeNote

As noted, the *Generals* branch is the target for mapping to external sources. The design approach is to define the classes in KBpedia broadly and to consider external mappings of the `subClassOf` nature. What this means is that the parental concept in KBpedia tends to subsume the concepts in the contributing external sources, and to, therefore, inherit the instances brought in by external classes.[27] However, not all mappings represent class-to-class relationships. Further, some mappings may be more of the nature of intersections or partial overlaps, rather than complete inheritance. As a result, KBpedia has adopted multiple mapping predicates, some approximate, as shown in Table 8.6:

Table 8.6 Mapping and alignment relations

correspondsTo	The property *correspondsTo* is used to assert a close correspondence between an external class, named entity, individual, and instance with a Reference Concept class. correspondsTo relates the external class, named entity, individual, or instance to the class by both its subject matter and intended scope. This predicate should be used where the correspondence between the two entities is felt to be nearly equivalent to a *sameAs* assertion and is reflexive, but without the full entailments of intensional class memberships. In these cases, both entities are understood to have the same type and intended scope, but without asserting a full class-level or *sameAs* individual relationship. This predicate is for aligning two different ontologies or knowledge bases based on node-level correspondences, but without entailing the actual ontological relationships and structure of the object source. For example, the *correspondsTo* predicate may be used to assert close correspondence between Reference Concepts and Wikipedia categories or pages, yet without entailing the actual Wikipedia category structure. This property asserts a different and stronger relationship than *isAbout*
isAbout	The property *isAbout* is used to assert the relation between an external named entity, individual, and instance with a Reference Concept class. *isAbout* relates the external named entity, individual, or instance to the class by its subject matter. The relation acknowledges that the scope of the class cannot be determined solely by the aggregation or extent of its associated individual entity members and that the nature of the Reference Concept class may not alone bound or define the individual entity. This property is therefore used to create a topical assertion between an individual and a Reference Concept
isRelatedTo	Check the definition of *isAbout* for the definition of this property; *isRelatedTo* is the inverse property of *isAbout*

(continued)

[27] Due to OWL 2 and punning (*c.f.*, Chap. 9), depending on context, we can talk about the classes in the *Generals* branch as instances and characterize them, while they can still act as classes for mapping and logical inheritance purposes.

Table 8.6 (continued)

relatesToXXX	The various properties designated by *relatesToXXX* are used to assert a relationship between an external instance (object) and a particular (*XXX*) SuperType. There may be as many *relatesToXXX* properties as there are numbers of SuperTypes. The assertion of this property does not entail class membership with the asserted SuperType. Rather, the assertion may be based on particular attributes or characteristics of the object at hand. For example, a British person might have a *relatesToXXX* asserted relation to the SuperType of the geopolitical entity of Britain, though the actual thing at hand (person) is a member of the Person class SuperType. This predicate is used for filtering or clustering, often within user interfaces. Multiple *relatesToXXX* assertions may be made for the same instance
isLike	The property *isLike* is used to assert an associative link between similar individuals who may or may not be identical, but are believed to be so. This property is not a general expression of similarity, but rather the likely but uncertain same identity of the two resourcesThis property is an alternative to *sameAs* where there is not a certainty of sameness, and when it is desirable to assert a degree of overlap of sameness via the *hasMapping* reification predicate. This property can and should be changed if the certainty of the sameness of identity is subsequently determined *isLike* has the semantics of likely identity, but where there is some uncertainty that the two resources indeed refer to the same individual with the same identity. Such uncertainty can arise when, for example, we use common names for different individuals (*e.g.,* John Smith). It is appropriate to use this property when there is strong belief that the two resources refer to the same individual with the same identity, but that association cannot be made at present with full certitude
hasMapping	The *hasMapping* property is used to reify *isAbout*, *isRelatedTo*, or an *isLike* property assertion with a statement as to its degree of mapping or relationship between subject and object. The *hasMapping* property may be expressed as a mapping percentage value, some quantitative metric value, or a qualitative descriptor characterizing the linkage degree or overlap between the two classes, predicates, individuals, or data types. This value might be calculated from some external utility, may be free form, or may be based on some defined listing of mapping values expressed as literals
hasCharacteristic	The property *hasCharacteristic* is used to assert the relation between a Reference Concept, or any other classes, and external properties that may be used in external ontologies to characterize, describe, or provide attributes for data records associated with that concept or that class. It is via this property or its inverse, *isCharacteristicOf,* that external data characterizations may be incorporated and modeled within a domain ontology based on the KBpedia vocabulary
isCharacteristicOf	The property *isCharacteristicOf* is used to assert the relation between a property and a Reference Concept (or its punned individual), or any other classes, to which it applies. Such properties may be used in external ontologies to characterize, describe, or provide attributes for data records associated with that concept or that class. It is via this property or its inverse, *hasCharacteristic*, that external data characterizations may be incorporated and modeled within a domain ontology

We cover the general topic of mapping in some detail in Chap. 13.

References

1. R. Davis, H. Shrobe, P. Szolovits, What is a knowledge representation? AI Mag. **14**, 17 (1993)
2. M.K. Bergman, The rationale for semantic technologies, in *AI3:::Adaptive Information* (July 2012)
3. F. Huber, Formal representations of belief, *Stanford Encyclopedia of Philosophy* (Oct. 2008)
4. W. Roche, Coherence and probability: A probabilistic account of coherence, in *Coherence: Insights from Philosophy, Jurisprudence and Artificial Intelligence* (Springer, Dordrecht, 2013), pp. 59–91
5. I. Douven, W. Meijs, Measuring coherence. Synthese **156**, 405–425 (2007)
6. R.J. Brachman, H.J. Levesque, *Knowledge Representation and Reasoning* (Morgan Kaufmann, Los Altos, 2004)
7. H.J. Levesque, Making believers out of computers. Artif. Intell. **30**, 81–108 (1986)
8. H.J. Levesque, R.J. Brachman, Expressiveness and tractability in knowledge representation and reasoning. Comput. Intell. **3**, 78–93 (1987)
9. Y. Zhou, A Set Theoretic Approach for Knowledge Representation: the Representation Part, *arXiv*, vol. 1603 (Mar. 2016)
10. J.F. Sowa, Conceptual Graph Summary. http://www.jfsowa.com/cg/cgif.htm
11. E. Patterson, Knowledge Representation in Bicategories of Relations, *arXiv*, vol. 1706 (June 2017)
12. C. Sdrolia, Signifying Nature: Semeiosis as the Foundation of Post-Critical Cosmology in Charles S. Peirce (Goldsmiths, University of London, London, 2014)
13. W. Staat, On abduction, deduction, induction and the categories. Trans. Charles S. Peirce Soc. **29**, 225–237 (1993)
14. S. Muggleton, Inductive logic programming. New Generation Comput. **8**, 295–318 (1991)
15. C. d'Amato, N. Fanizzi, B. Fazzinga, G. Gottlob, T. Lukasiewicz, Combining semantic web search with the power of inductive reasoning, in *International Conference on Scalable Uncertainty Management* (Springer, Berlin, 2010), pp. 137–150
16. L. De Raedt, *Logical and Relational Learning* (Springer Science & Business Media, Heidelberg, 2008)
17. J. Völker, M. Niepert, Statistical Schema Induction, in *The Semantic Web: Research and Applications* (2011), pp. 124–138
18. F. Baader, D. Calvanese, D. McGuiness, D. Nardi, P. Patel-Schneider, *The Description Logic Handbook: Theory, Implementation and Applications* (Cambridge University Press, Cambridge, 2003)
19. See https://en.wikipedia.org/wiki/Extension_(semantics), retrieved December 6, 2017
20. B. Motik, P.F. Patel-Schneider, B.C. Grau, *OWL 2 Web Ontology Language Direct Semantics (Second Edition)*, World Wide Web Consortium (2012)
21. L. Feigenbaum, E. Prud'hommeaux SPARQL by Example, *Cambridge Semantics* (May 2013)
22. *SKOS Simple Knowledge Organization System Reference: W3C Recommendation*, World Wide Web Consortium (2009)
23. C.S. Peirce, The Logic of Relatives, *The Monist* (1897), pp. 161–217
24. J.J. Zeman, The Graphical Logic of CS Peirce, Ph.D., University of Chicago (1964)

Part III
Components of Knowledge Representation

Chapter 9
Keeping the Design Open

At the time of my high school years, Alfred Wegener's theory of continental drift was still a question mark for many mainstream scientists. In my college years, a young American biologist, Lynn Margulis, postulated and was ridiculed for the theory of endosymbiosis; that is, that certain cell organelles originated from initially free-living bacteria. In 1980 the Alvarez's hypothesized that the age of dinosaurs was ended by an asteroid strike near the Yucatan at the end of the Cretaceous. In the 1990s we were just starting to get a glimmer that the *Helicobacter* bacteria had been the cause of misdiagnosed peptic ulcers for decades. Today, we widely accept all of these then-revolutionary hypotheses as scientific truth.[1]

We now see continental drift as a major explanation for the geographic dispersal of plant and animal families across the globe. Margulis' theory is understood to embrace cell organelles from mitochondria to chloroplasts, informing us that the fundamental unit of all organisms—the cell—is itself an amalgam of archaic symbionts and bacteria-like life forms. We now correlate asteroid strikes to historical extinction events through geologic time. Though the native human genome has some 23,000 genes, researchers estimate that more than three million genes arise from bacterial fellow travelers in our gut and skin 'microbiomes.' We know that our ecosystem of bacteria is involved in nutrition and digestion, contributing perhaps as much as 15% of the energy value we get from food. Besides ulcers, researchers have implicated symbiotic bacteria in heart disease, type II diabetes, obesity, malnutrition, multiple sclerosis, other autoimmune diseases, asthma, eczema, liver disease, bowel cancer, and autism, among others. Within my professional life, major aspects

[1] Some material in this chapter was drawn from the author's prior articles at the *AI3:::Adaptive Information* blog: "Open Source Business Models" (Aug 2005); "Open Source and the 'Business Ecosystem'" (Aug 2005); "Climbing the Data Federation Pyramid" (May 2006); "The Open World Assumption: Elephant in the Room" (Dec 2009); "Listening to the Enterprise: Total Open Solutions, Part 1" (May 2010); "Metamodeling in Domain Ontologies" (Sep 2010); "What is a Reference Concept?" (Dec 2010); "Declining IT Innovation in the Enterprise" (Jan 2011); "We Are an Open World" (Sep 2012); "The Era of Openness" (Jan 2015); "The Importance of Semsets in Knowledge Graph Design" (Mar 2017).

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8_9

of science, geology, and biology have undergone massive and fundamental shifts in understanding. Concomitant changes have swept through society. Such is the nature of knowledge, with the seeming rapidity of advances steadily increasing.

This chapter begins our Part III. All three chapters cover the components of knowledge representation design responsive to such fast-moving changes. In this chapter, we discuss the importance of open design to capture rapid changes in knowledge, indeed to capture the broad trends toward openness across all aspects of human informational and economic activity. These imperatives help inform the structural considerations that go into how to federate and interoperate data from multiple sources in multiple formats. In the following Chap. 10, we discuss our typology design, the basis by which we can adapt our overall design to new domains or expand the knowledge we capture for any given domain. In Chap. 11, we explain how these open components naturally also lead to a design founded on knowledge bases and graphs, as the proper structural expressions of this open and connected nature. Think of Part III, combined with the three earlier chapters of Part II, as describing all of the design and building block inputs needed for a responsive knowledge representation *system,* the topic of Part IV that follows.

The Context of Openness

Since ancient times, an exemplar being the Library of Alexandria, humans have used libraries to collate documents and to provide access to knowledge. Repositories and books sometimes threaten authoritarian regimes or close-minded orthodoxies, as does information and knowledge in general. Book burnings and ransacking of libraries are some of the saddest events of human history.

Fortunately, a notable and profound transition is under way. This transition is not something we can tie to a single year or event. It is also something that is quite complex in that it is a matrix of forces, some causative and some derivative, all of which tend to reinforce one another to perpetuate the trend. The trend that I am referring to is *openness*, and it is a force that is both creative and destructive, and one that in retrospect is also inevitable given the forces and changes underlying it. It is hard to gauge exactly when the blossoming of openness began, but by my lights the timing corresponds to the emergence of open-source software and the Internet. Over the past quarter century, the written use of the term ‘open’ has increased more than 40% in frequency in comparison to terms such as ‘near’ or ‘close,’ a pretty remarkable change in usage for a more or less common term.[2]

[2] The data is from Google book trends data based on this query (https://books.google.com/ngrams/graph?content=open%2Cclose%2Cnear%2Copenness&case_insensitive=on&year_start=1980&year_end=2008); the years 2009 to 2014 were projected based on prior actuals to 1980; percentage term occurrences were converted to term frequencies by 1/n.

An Era of Openness

Though the term of 'openness' is less common than 'open,' its change in written use has been even more spectacular, with its frequency more than doubling (112%) over the past 25 years. The change in growth slope appears to coincide with the mid-1980s [1], consistent with my thesis of being linked to open-source software and the Internet. Because 'openness' is more of a mindset or force—a point of view, if you will—it is not itself a discrete thing, but an idea or a concept [2]. In contemplating this world of openness, we can see quite a few separate, yet sometimes related, strands that provide the weave of the 'openness' definition:

- Open source—refers to a computer program in which the source code is available to the general public for use or modification from its original design. Open-source code is typically a collaborative effort where programmers improve upon the source code and share the changes within the community so that other members can help improve it further.
- Open standards—are standards and protocols, some informal or put forward by individuals, that are fully defined and available for use without royalties or restrictions; stakeholders often suggest and modify these open standards in public collaboration, with adoption subject to some open governance procedures.
- Open content—is a creative work, generally based on text, that others can copy or modify; open-access publications are a particular form of open content that provides unrestricted online access to peer-reviewed scholarly research.
- Open data—is the idea that specific data should be freely available to everyone to use and republish as they wish, without restrictions from copyright, patents, or other mechanisms of control; open data is a special form of open content.
- Open knowledge—is what open data becomes when it is useful, usable, and used; according to the Open Knowledge Foundation, the key features of openness are availability and access wherein the data must be available as a whole and at no more than a reasonable reproduction cost, preferably by downloading over the Internet.
- Open knowledge bases—are open knowledge packaged in knowledge-base form.
- Open access to communications—is nondiscriminatory access to communications networks, allowing new models such as crowdsourcing (obtaining content, services, or ideas from a large group of people), citizen science, or crowdfunding (raising funds from a large group of people) to arise.
- Open rights—are an umbrella term to cover the ability to obtain content or data without copyright restrictions and gaining use and access to software or intellectual property via open licenses.
- Open logics—are the use of logical constructs, such as the open-world assumption, which enable us to add data and information to existing systems without the need to re-architect the underlying data schema; such logics are essential to knowledge management and the continuous addition of new information.

- Open architectures—are means to access existing software and platforms via such means as open APIs (application programming interfaces), open formats (published specifications for digital data), or open Web services.
- Open government—is a governing doctrine that holds that citizens have the right to access the documents and proceedings of the government to allow for effective public oversight; online access to government data and information is one goal.
- Open education—is an institutional practice or programmatic initiative that broadens access to the learning and training traditionally offered through formal education systems, generally via educational materials, curricula, or course notes at low or no cost without copyright limitations.
- Open design—is the development of physical products, machines, and systems through the use of publicly shared design information, often via online collaboration.
- Open research—makes the methodology and results of research freely available via the Internet, and often invites online collaboration; we refer to it as open science if the research is scientific in nature.
- Open innovation—is the use and combination of open and public sources of ideas and innovations with those internal to the organization.

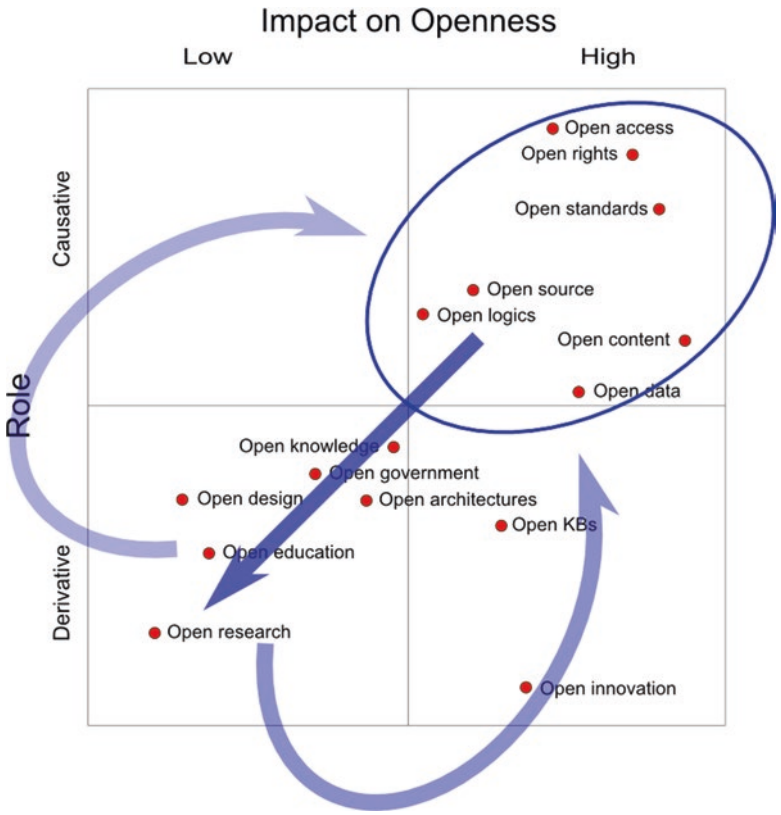


Fig. 9.1 Openness begets more openness

In looking at the factors above, we can ask two formative questions. First, is the given item above primarily a causative factor for 'openness' or is it a derivative due to a more 'open' environment? Second, does the factor have an overall high or low impact on the question of openness? Figure 9.1 plots these factors and dimensions

Early expressions of 'openness' helped cause the conditions that lead to openness in other areas. As those areas also become more open, positive reinforcement is passed back to earlier open factors, all leading to a virtuous circle of increased openness. Though perhaps not strictly 'open,' other various and related factors such as the democratization of knowledge, broader access to goods and services, more competition, 'long tail' access and phenomenon, and, in genuinely open environments, more diversity and more participation also could be plotted on this matrix.

Once viewed through the lens of 'openness,' it starts to become clear that all of these various 'open' aspects are remaking information technology and human interaction and commerce. The impacts on social norms and power and governance are just as profound. Though many innovations have uniquely shaped the course of human history—from literacy to mobility to communication to electrification or computerization—none appear to have matched the speed of penetration nor the impact of 'openness.' So, what is driving this phenomenon? Where did the concept of 'openness' arise?

The matrix in Fig. 9.1 helps us hypothesize one foundational story. Look at the question of what is causative and what might be its source. One conclusion is the Internet—specifically the Web, as reinforced and enabled by open-source software—is a primary causative factor. Relatively open access to an environment of connectivity guided by standard ways to connect and contribute began to fuel still further connections and contributions. The positive values of access and connectivity via standard means, in turn, reinforced the understood value of 'openness,' leading to still further connections and engagement. More openness is like the dropped sand grain that causes the entire dune to shift. The Web with its open access and standards has become the magnet for open content and data, all working to promote derivative and reinforcing factors in open knowledge, education, and government.

The fruits of 'openness' tend to reinforce the causative factors that created 'openness' in the first place. More knowledge and open aspects of collaboration lead to still further content and standards that lead to further open derivatives. In this manner, 'openness' becomes a kind of engine that promotes further openness and innovation. A kind of open logic (premised mainly on the open-world assumption, see next section) lies at the heart of this engine. Since new connections and new items are continually arising and fueling the openness engine, we bolt on new understandings to original starting understandings. This accretive model of growth and development is similar to the deposited layers of pearls or the growth of crystals. The structures grow according to the factors governing the network effect, and the nature of the connected growth structures may be represented and modeled as graphs. In general, the greater the degree of expressed structure, the higher its potential contribution to interoperability.

'Openness,' like the dynamism of capitalism, is both creative and destructive.[3] The effects are creative—actually transformative—because of the new means of collaboration that arise based on the new connections between new understandings or facts. 'Open' graphs create entirely new understandings as well as provide a scaffolding for still further insights. The fire created from new understandings pulls in new understandings and contributions, all sucking in still more oxygen to keep the innovation cycle burning. However, the creative fire of openness is also destructive. Proprietary software, excessive software rents, silo-ed and stovepiped information stores, and much else are being consumed and destroyed in the wake of openness. Older business models—indeed, existing suppliers—are in the path of this open conflagration. Openness is sweeping private and 'closed' solutions into the firestorm. The massive storehouse of legacy kindling appears likely to fuel the openness flames for some time to come.

'Openness' becomes a form of adaptive life, changing the nature, value, and dynamics of information and who has access to it. Though much of the old economy is—and will be—swept away in this destructive fire, new and more fecund growth is replacing it. From the viewpoint of the practitioner on the ground, I have not seen a more fertile innovation environment in information technology. Once the proper conditions for 'openness' were in place, it now seems inevitable that today's open circumstances would unfold. The Internet, with its (generally) open access and standards, was a natural magnet to attract and promote open-source software and content. A hands-off, unregulated environment has allowed the Internet to innovate, grow, and adapt at an unbelievable rate.

Of course, coercive state regimes can control the Internet to varying degrees and can limit innovation. Cybersleuths and hackers may access our information stores and private data, unknown or undetected by us. Any change in the Internet from 'open' to more 'closed' may also act over time to starve the openness fire. Examples of such means to slow openness include imposing Internet regulation, walled gardens like Facebook, limiting access (technically, economically, or by fiat), moving away from open standards, or limiting access to content. Any of these steps would starve the innovation fire of oxygen. Access to information wins out over risks, though we do need to self-impose restrictions to guard privacy. Openness reduces the ability of authoritative regimes or close-mindedness to threaten our knowledge.

The forces impelling openness are strong. Still, these observations are no proof for cause and effect. The correspondence of 'openness' to the Internet and open source may be a coincidence. However, my sense suggests a more causative role. In all of these regards 'openness' is a woven cord of forces changing the very nature and scope of information available to humanity. 'Openness,' which has heretofore largely lurked in the background as some unseen force, now emerges as a criterion by which to judge the wisdom of various choices. 'Open' appears to contribute

[3] "Creative destruction" is a term from the economist Joseph Schumpeter that describes the process of industrial change from within whereby old processes are incessantly destroyed and replaced by new ones, leading to a constant change of economic firms that are winners and losers.

more and be better aligned with current forces. Business models based on proprietary methods or closed information appear, at least for today's circumstances, on the losing side of history.

The Open-World Assumption

The *open-world assumption* (OWA) is a different logic premise for most organizations. Relational database systems, for example, embrace the alternate *closed-world assumption* (CWA). OWA is a formal logic assumption that the truth value of a *statement* is independent of whether or not it is known as true by any single observer or agent. OWA is used in *knowledge representation* to codify the informal notion that in general no single agent or observer has complete knowledge, and therefore cannot make the closed-world assumption. The OWA limits the kinds of *inference* and deductions an agent can make to those that follow from statements known to the agent as true. OWA is useful when we represent knowledge within a system as we discover it, and where we cannot guarantee that we have discovered or will discover complete information. In the OWA, statements about knowledge that are not included in or inferred from the knowledge explicitly recorded in the system may be considered unknown, rather than wrong or false. Semantic technology languages such as *OWL* and *RDF* make the open-world assumption. In contrast to the closed-world approach of transaction systems, IT systems based on the logical premise of the open-world assumption (OWA) mean the following:

- Lack of a given assertion does not imply whether it is true or false; it merely is not known.
- A lack of knowledge does not imply falsity.
- Everything is permitted until it is prohibited.
- Schema can be incremental without re-architecting prior schema ('extensible').
- Information at various levels of incompleteness can be combined.

Some enterprise circumstances—say a complete enumeration of customers or products or even controlled engineering or design environments—may warrant a closed-world approach. CWA is the presumption that what is not currently known as true is false. Engineering an oil drilling platform or launching a rocket, in fact, demands that. A closed-world assumption performs well for transaction operations with easier data validation. The number of negative facts about a given domain is typically larger than positive ones. So, in many bounded applications, the number of negative facts is so large that their explicit representation can become practically impossible. In such cases, it is simpler and shorter to state known 'true' statements than to enumerate all 'false' conditions.

On the other hand, the relational model is a paradigm where the information must be complete, and a single schema must describe it. Traditional databases require we agree on a schema before data can be stored and queried. The relational model assumes that only explicitly represented objects and relationships exist in the

domain. It assumes that names are unique, and it is how we identify objects in the domain. The result of these assumptions is that relational systems have a *single* (canonical) model where objects and relationships are in a one-to-one correspondence with the data in the database [1].

It is natural to take a successful approach and try to extend it to other areas. However, beginning with data warehouses in the 1980s, business intelligence (BI) systems in the 1990s, and the general issue of most enterprise information being bound up in documents for decades, the application of the relational model to these areas has been disappointing. CWA and its related assumptions are a poor choice when we attempt to combine information from multiple sources, to deal with uncertainty or incompleteness in the world, or to try to integrate internal, proprietary information with external data. Irregularity and incompleteness are toxic to relational model design. In the open semantic Web, we can share data that is structured differently via RDF triple statements (*subject-predicate-object*). For example, OWA allows storing information about suppliers without cities and names alongside suppliers with that information. Information can be combined with similar objects or individuals even though they have different or nonoverlapping attributes. We now check duplicates based on the logic of the system and not unique name evaluations. Data validation in OWA systems can become more complicated (via testing against restriction statements) or partially easier (via inference).

It is interesting to note that the theoretical underpinnings of CWA by Reiter arose about the same time (1978) that data federation and knowledge representation (KR) activities also started to come to the fore [9]. CWA and later work on (for example) default reasoning appeared to have informed early work in description logics and its alternative OWA approach. However, the initial path toward KM work based on the relational model also seems to have been set in this timeframe.

We are still reaping the whirlwind from this unfortunate early choice of the relational model and CWA for knowledge representation, knowledge management, and business intelligence purposes. Moreover, while much theoretical and logical discussion exists for alternative OWA and CWA data models, surprisingly few discussions occur for the implications of these models. We may couple the data models behind these approaches (Datalog or non-monotonic logic in the case of CWA; monotonic in the case of OWA; OWA is also firmly grounded in *description logics*) with other assumptions. From a theoretical standpoint, I have found the treatment of Patel-Schneider and Horrocks useful in comparing these approaches [1]. However, the *Description Logics Handbook* and some other varied sources are also helpful [3].[4]

[4] Model theory is a formal semantic theory which relates expressions to interpretations. A "model" refers to a given logical "interpretation" or "world" (see, for example, the discussion of interpretation in Patrick Hayes, ed., 2004. *RDF Semantics—W3C Recommendation*, 10 February 2004). The logic or inference system of classical model theory is monotonic. That is, it has the behavior that if S entails E then (S + T) entails E. In other words, adding information to some prior conditions or assertions cannot invalidate a valid entailment. The basic intuition of model-theoretic semantics is that asserting a statement makes a claim about the world: it is another way of saying that the world is, in fact, so arranged as to be an interpretation which makes the statement true. An assertion

I think it is fair to assert that the closed-world assumption and its prevalent mindset in traditional database systems have hindered the ability of organizations and the vendors that support them to adopt incremental, low-risk means to knowledge systems and management. CWA, in turn, has led to overengineered schema, too-complicated architectures, and massive specification efforts that have led to high deployment costs, blown schedules, and brittleness.

In limited cases, the relational model can embrace the open-world assumption, such as the null in SQL. Similarly, semantic Web approaches can be closed world, such as frame languages or Prolog or other special considerations. We can also use relational systems for managing our instance data, while we rely on open-world systems for the knowledge graph.

In most real-world circumstances, much we do not know, and we interact in complex and external environments. Knowledge management inherently occupies this space. Ultimately, data interoperability implies a global context. Open world is the proper logic premise for these circumstances. Via the OWA framework, we can readily change and grow our conceptual understanding and coverage of the world, including the incorporation of external *ontologies* and data. Since this can comfortably coexist with underlying closed-world data, a design based on OWA can readily bridge both worlds. Open-world frameworks provide some incredible benefits where open-world conditions apply:

- Domains can be analyzed and inspected incrementally.
- Schema can be incomplete and developed and refined gradually.
- The data and the structures within these open-world frameworks can be used and expressed in a piecemeal or incomplete manner.
- We can readily combine data with partial characterizations with other data having complete characterizations.
- Systems built with open-world frameworks are flexible and robust; as we gain new information or structure, we can incorporate without negating the information already resident.
- Open-world systems can readily bridge or embrace closed-world subsystems.

Open world does not necessarily mean open data, and it does not necessarily mean open source. OWA technologies are neutral to the question of open or public sources. We can apply the techniques equivalently to internal, closed, proprietary data and structures. Moreover, we can use the same technologies as a basis for bringing external information into the organization. Open world is a way to think

---

amounts to stating a constraint on the possible ways the world might be. In comparison, a non-monotonic logic system may include default reasoning, where one assumes a 'normal' general truth unless it is contradicted by more particular information (birds normally fly, but penguins don't fly); negation by failure, commonly assumed in logic programming systems, where one concludes, from a failure to prove a proposition, that the proposition is false; and implicit closed-world assumptions, often assumed in database applications, where one concludes from a lack of information about an entity in some corpus that the information is false (e.g., if someone is not listed in an employee database, he or she is not an employee). See further Non-monotonic Logic from the *Stanford Encyclopedia of Philosophy*.

about the information we have and how we act on it. An open-world assumption accepts that we never have all necessary information and lacking that information does not itself lead to any conclusions.

In the past, there have been questions about performance and scalability with open semantic technologies. Progress on these fronts has been rapid, with billion triple systems now common and improvements steady. Fortunately, the incremental approach that we advocate herein dovetails well with these rapid developments. There should be no arguing of the benefits of a successful incremental project in a smaller domain, perhaps repeated across multiple domains, in comparison to previous, large, costly initiatives that never produce (even though their underlying technologies are performing). Architecture considerations are also inherent in these OWA designs, which we discuss in *Web-oriented architectures* in Chap. 12.

It is perhaps not surprising that one of the fields most aggressive in embracing ontologies and semantic technologies is the life sciences. Biologists and doctors experience daily the explosion in new knowledge and understandings. Knowledge workers in other fields would be well advised to follow the lead of the life sciences in rethinking their foundations for knowledge representation and management. It is good to remember that if your world is not open, then your understanding of it is closed.

Open Standards

Open standards provide a different kind of openness. The rationale for open standards is not the logic or nature of knowledge, but rather the desire to adopt languages and systems that have the highest likelihood of being shared with others. We employ open standards and best practices in KBpedia to (1) obtain the most accurate results, and (2) facilitate interoperability with external data and systems [4]. We mostly base our open standards on those from the World Wide Web Consortium (W3C), which established the standards for the original Web and the design of Web pages and Web protocols. Specific W3C standards used by KBpedia include *RDF*, *RDFS*, *OWL 2*, *SKOS*, *SPARQL*, and *SWRL,* introduced in the prior chapter.

Other standards, such as HTML, are also used where appropriate. *De facto* standards may contribute, arising from the effort of individuals or projects. We also may employ open-source standard libraries and tools. For KBpedia, these include the ontology IDE, Protégé, the OWL API, and the search engine Lucene. In the use of these standards, we apply best practices, many of which we have developed through our client work.[5] Some of these include the use of *semsets* to capture the multiple labels applied to a given thing; how to construct and manage ontologies (also known as knowledge graphs); ensuring multilingual capabilities; and build and management workflows. We discuss these in the following sections and chapters. We have written most supporting KBpedia code in Clojure, a modern language based on the

[5] See Chap. 13.

original AI language Lisp, in part due to its ability to run on the Java virtual machine. This ability means we may concurrently use any existing Java application with our various KBpedia build, testing, analysis, and management routines.

Open standards, like open source, provide positive feedback across the entire development ecosystem. Developers most often write open-source software with open standards and languages. Tooling written in open standards has a broader base of adoption. Developers and knowledge workers prefer to work with open standards because they desire transferable job skills and experience. Like other aspects of the 'openness' phenomenon, open standards are a positive contributor to innovation and still more openness.

Information Management Concepts

Openness means we also need to accommodate some additional concepts in our design. The first of these considerations relates to how we refer to and name things. Not all of us use the same words for things, and we should be explicit ('open') about this fact in our vocabularies. The second consideration is that we need to provide relatively balanced and equal-weighted concepts in our reference structures. In the case of KBpedia, with its use as a general-purpose reference structure, this means we need to capture a set of concepts that capture relatively well the entire knowledge domain. However, the same principles apply to restricted domains and how to define their overall conceptual structure. The third consideration is that, depending on context, we may also use the same term to refer to either an instance or a general class. Again, we should be explicit about these referential differences, with logic and design suitable to them. For lack of a better phrase, I collectively term these three considerations as information management concepts that we need to embrace in our designs.

We intricately associate our vocabularies with how we see and understand the world. We all know the apocryphal claim of how Eskimos have many more words for snow, but the idea likely applies to multiple perspectives in multiple domains. My own first experience is when I was an undergraduate learning plant taxonomy. We had to learn hundreds of strange terms such as *glabrous* or *hirsute* or *pinnate*, all terms of art for how to describe leaves, their shapes, their hairiness, fruits and flowers, and such. What happens, though, when one learns the terminology of a domain is that one's eyes are opened to see and distinguish more. What had previously been for me a field of view composed of multiple shades of green made up of shrubs and trees began to emerge as distinct species of plants and individual variants that I could discern and identify. As I learned nuanced distinctions, I began to see with greater clarity. In knowledge representation systems, where so much of the knowledge is bound up in text and natural language, training oneself to see the individual leaves and trees from the forest is a critical step in capturing the domain. In part, this attention leads to a richer domain vocabulary.

Things, Not Strings

One of the strongest contributions that semantic technologies make to knowledge-based artificial intelligence (KBAI) is to focus on what things mean, as opposed to how they are labeled. The phrase that captures this focus on underlying meaning is 'things not strings.' The *idea of* something—that is, its *meaning*—is conveyed by how we define that something, the context for how we use the various tokens (terms) for that something, and the variety of names or labels we apply to that thing. In Chap. 5, I provided the examples of *parrots* and the *United States* to illustrate this concept, among other semantic heterogeneities.

We should not view knowledge graphs, properly understood, as being comprised of labels, but of concepts, entities, and relationships between those things. If we construct our knowledge graphs using single labels for individual nodes and relations, we will not be able to capture the nuances of context and varieties of reference. A knowledge graph useful to a range of actors must reflect the languages and labels meaningful to those actors. To distinguish the accurate references of individual terms we need the multiple senses of words to each be associated with its related concepts and then to use the graph relationships for those concepts to help disambiguate the intended meaning of the term based on its context of use.

According to WordNet, a synset (short for *synonym set*) is 'defined as a set of one or more synonyms that are interchangeable in some context without changing the truth value of the proposition in which they are embedded' [5]. In our view, the concept of a synset is helpful but still does not go far enough. Any name or label that draws attention to a given thing can provide the same referential power as a synonym. If two parties use two different terms to refer to the same thing, we need not go so far as to try to enforce a truth criterion. We can include in this category abbreviations, acronyms, aliases, argot, buzzwords, cognomens, derogatives, diminutives, epithets, hypocorisms, idioms, jargon, lingo, metonyms, misspellings, nicknames, nonstandard terms (see Twitter), pejoratives, pen names, pseudonyms, redirects, slang, sobriquets and stage names, as well as, of course, synonyms. Collectively, we call all of the terms that may refer to a given concept or entity a *semset*. In all cases, these terms are mere pointers to the actual something at hand.

In the KBpedia knowledge graph, these terms are defined either as `skos:prefLabel` (the preferred term), `skos:altLabel` (all other semset variants), or `skos:hiddenLabel` (misspellings). *Preferred label* (or *prefLabels* or *title*) is the readable string (name) for each *object* in KBpedia.[6] We provide labels as a convenience; the actual definition of the object comes from the totality of its description, prefLabel, altLabels, and connections (placement) within the *knowledge graph*. Labels of all kinds are *representations* and reside in Thirdness.

[6] Other label types may be added to this roster, such as *short-* and *long-labels* that might be the reference for user interface labels where alternatives to *prefLabel* are desired. All labels may also be expressed in any of the standard ISO human languages.

You can inspect for yourself how this concept of semset works in KBpedia. You can go to the standard online KBpedia search page and enter a query, for example, 'mammal.'[7] By changing between 'Preferred Label' and 'All content' on the drop-down list under 'Search Concepts,' you can get a tenfold range of results. Naturally, as one would expect, increasing the number of terms something might be known by acts to increase the possible matches within the knowledge graph. Semsets give us a way to narrow or broaden queries, as well as in combination with linked concepts to disambiguate the context of specific terms. We can apply these same considerations to SPARQL queries or programmatically when working with the KBpedia knowledge graph (or any other graph constructed to KBpedia's standards).

Charles Peirce held strong views about precision in naming things, best expressed by his article on *The Ethics of Terminology*.[8] His beliefs often led him to use obscure or coined terms to avoid poor understanding of common terms. He also proposed a variety of defining terms throughout the life of many of his concepts in his quest for precision. He also understood that terms (*symbols*) could be interpreted in different ways (*interpretants*) by various agents (interpreters). With inquiry, truth-seeking, and consensus of the community of users, we can reference our desired objects with more precision. That is our ideal. Peirce would concur that many ways refer to the same thing in the real world. The idea of *semset* is expressly designed to capture that insight.

The Idea and Role of Reference Concepts

Interoperability comes down to the nature of things and how we describe those things or quite similar things from different sources. Given the robust nature of semantic heterogeneities in diverse sources and datasets on the Web (or anywhere else, for that matter!), how do we bring similar or related things into alignment? Then, how can we describe the nature or basis of that alignment?

Of course, classifiers since Aristotle and librarians for time immemorial have been putting forward various classification schemes, controlled vocabularies, and subject headings. When one wants to find related books, it is convenient to go to a central location where we may find books about the same or similar topics. If we can categorize the book in more than one way—as all are—then something like a card catalog is helpful to find additional cross-references. Every domain of human endeavor makes similar attempts to categorize things. On the Web we have none of the limitations of physical books and physical libraries; locations are virtual, and copies can be replicated or split apart endlessly because of the virtually zero cost of another electron. However, we still need to find things, and we still want to gather related things together. As stated by Elaine Svenonius, 'Organizing information if it

[7] See http://kbpedia.org/knowledge-graph/search/?query=mammal&index=rcs.

[8] See further CP 2.219-226 (1903). Also an earlier article that helps provide Peirce's views on communications is "How to Make Our Ideas Clear."

means nothing else means bringing all the same information together' [6]. This sentiment and need remain unchanged whether we are talking about books, Web documents, chemical elements, or our information stores.

Like words or terms in human language that help us communicate about things, how we organize things needs to have an understood and definite meaning, hopefully, bounded with some degree of precision, that enables us to have some confidence we are communicating about the same something with one another. However, when applied to the Web and machine communications, we need further precision in characterizations and definitions.

The notion of a Web basis organizing things is both easier and harder than traditional approaches to classification. It is easier because everything is digital: we can apply multiple classification schemas and can change them at will. We are not locked into legacy structures like huge subject areas reserved for arcane or now historically less relevant topics, such as the Boer Wars or phrenology (though we still accommodate access). We need not move physical books around on shelves to accommodate new or expanded classification schemes. We can add new branches to our classification of, say, nanotechnology as rapidly as the science advances. The notion is harder because we can no longer rely on the understanding of human language as a basis for naming and classifying things. Actually, of course, language has always been ambiguous, but it can be manifestly more so when put through the grinder of machine processing and understanding. Machine processing of related information adds the new hurdles of no longer being able to rely on text labels ('names') alone as the identifier of things and requires we be more explicit about our concept relationships and connections. Fortunately, here, too, much has been done in helping to organize human language through such lexical frameworks as WordNet and similar. We have learned much while grappling with these questions of how to organize and describe information to aid interoperability in an Internet context.

One formalized approach has been put forward by the FRSAD (Functional Requirements for Subject Authority Data) working group [7], a community of librarians and information specialists, dealing with subject authority data. Subject authority data is the type of classificatory information that deals with the subjects of various works, such as their concepts, objects, events, or places. As the group stated, the scope of this effort pertains to the 'aboutness' of various conceptual works. The frameworks for this effort, as with the broader FRBR effort, are new standards and approaches appropriate to classifying electronic bibliographic records. The FRSAD approach distinguishes the *idea of something* (which it calls a *thema*, or entity used as the subject of a work) from the name or label of something (which it calls *nomen*). For many in the logic community, steeped in the Peirce triad of *sign–object–interpretant*,[9] this distinction seems rather obvious and straightforward. However, in

[9] C.S. Peirce's sign relations are covered in the Representations section of Chap. 2. In the context of this discussion, the *sign* corresponds to any of the labels or identifiers associated with the (reference concept) *object*, the meaning of which is provided by its *interpretant*. See also John Sowa, 2000. "Ontology, Metadata, and Semiotics," presented at ICCS'2000 in Darmstadt, Germany, on August 14, 2000; see http://www.jfsowa.com/ontology/ontometa.htm.

library science, labels have been used interchangeably as identifiers, and making this distinction clean is a real contribution. The FRSAD effort does not discuss how the thema is found or organized.

The notion that we use for a *reference concept* contains elements of this approach. A *reference concept* (RC) is *the idea of something* or a *thema* in the FRSAD sense. However, as we use it, an RC is also a reference-linking point for external sources or expanded vocabularies. If properly constructed and used, a reference concept becomes a fixed point in an information space. Think of an RC as a fixed starting point for navigating, relating, or mapping content. It is a guiding star in a constellation of information, or, to use a different analogy, a defined, fixed survey marker as used by surveyors to measure new mapping points. As one or more external sources link to these fixed points, it is then possible to gather similar content together and to begin to organize the information space, in the sense of Svenonius. Further, if the RC is itself part of a coherent structure, then additional value can be derived from these assignments, such as inference, consistency testing, and alignments. If the right factors are present, it should be possible to relate and interoperate multiple datasets and knowledge representations.

We have six requirements for a reference concept, some provided by RDF or OWL:

1. *Persistent IRI*—By definition, a Web-based reference concept should adhere to linked data principles and should have an IRI as its address and identifier. Also, by definition as a 'reference,' the vocabulary or ontology in which the concept is a member should be given a permanent and persistent address. Steps should be taken to ensure 24 × 7 access to the RC's IRIs since external sources will be depending on them. As a general rule, the concepts should also be stated as single nouns and use CamelCase notation (that is, class names should start with a capital letter and not contain any spaces, such as MyNewConcept).
2. *Preferred label*—It provides a preferred label annotation property that is used for human-readable purposes and in user interfaces. For this purpose, a construct such as the SKOS property of `skos:prefLabel` works well. Note that this label is *not* the basis for deciding and making linkages, but it is essential for mouseovers, tooltips, interface labels, and other human use factors.
3. *Definition*—It gives all RCs a reasonable definition, since that and linkages are what gives an ontology its semantics. Remember not to confuse the label for a concept with its meaning. For this purpose, a property such as `skos:definition` works well, though others such as `rdfs:comment` or `dc:description` are also commonly used. The definition and linkages to other concepts are the two most critical sources for the concept's meaning. Adequate text and content also aid semantic alignment or matching tasks.
4. *Semset*—This includes explicit consideration for the idea of a '*semset*' as described above, which means a series of alternate labels and terms to describe the concept.
5. *Language independence*—It keeps the identifier separate from its labels, and qualifies entries for definition, preferred label, and alternative labels with language tags. Though an additional step (for example, assigning the RDF

xml:lang='en' tag for English), adhering to this practice gives language independence to reference concepts. Sources such as Wikipedia or Wikidata, with their richness of concepts and multiple language versions, can then be a basis for creating alternative language versions.

6. *Part of a coherent structure*—It tests for consistency and coherence when modifying the knowledge structure. A cohesive structure provides the benefits of reliable inferencing, discovery, navigation, and analysis. Adequately constructed RDFS and SKOS data models and OWL ontologies can deliver these benefits.

To this basic set of reference concepts, it is also necessary to add the mapping predicates that relate the RCs to external sources. The mapping predicates have their own set of design guidelines:

1. Provide the same *completeness of specification* as RCs.
2. Capture a spectrum of *mapping alignments* from exact or sameAs to approximate to represent the real correspondence between items.
3. *Range and domain*—use domains and ranges, as provided for by RDFS, to assist testing, disambiguation, and external concept alignments. Domains apply to the subject (the left-hand side of a triple) and ranges to the object (the right-hand side of the triple).

In part, many current vocabularies meet these guidelines to some extent. However, few vocabularies provide complete coverage, and across a broad swath of domain needs, gaps remain. This unfortunate observation applies to upper level ontologies, reference vocabularies, and domain ontologies alike.

KBpedia is a knowledge graph of approximately 55,000 reference concepts designed according to these design guidelines. We organize its reference concepts into about 80 modular and distinct (mostly disjoint) *typologies*, which I discuss in some detail in the next Chap. 10. The RCs that represent the top-level nodes of these typologies we also term *SuperTypes* (also Super Types), which are collections of (mostly) similar reference concepts. We design most of the SuperType disjoint from the other SuperType classes. Each typology we use in KBpedia thus has its corresponding top-level SuperType node.[10] SuperTypes, and the typologies they represent, thus provide a higher level of clustering and organization of the reference concepts. The KBpedia Knowledge Ontology (KKO) only contains the highest level RCs and SuperTypes. This design enables a higher level view of KBpedia with only a couple hundred RCs and makes clear these SuperType typology tie-in points.

For specific domain purposes, you may use KBpedia or portions thereof as the initial grounding structure. You may expand it into new domain areas following similar design considerations. Potentially, you may turn nearly any of the existing 55,000 RCs in KBpedia into a SuperType, providing a new tie-in point to the new RCs reflecting the expanded domain.

[10] In KBpedia, disjoint SuperTypes are termed "core," and other SuperTypes used mostly for organizational purposes are termed "extended." KBpedia has a total of about 80 SuperTypes, with 30 or so deemed as "core." See further Appendix B.

Punning for Instances and Classes

In ontologies, we may want to treat our concepts as both classes and instances of a class. *Punning*, in computer science, refers to a programming technique that subverts or circumvents the type system of a programming language, by allowing us to treat a value of a particular type as a value of a different type. When used for ontologies, it means to treat a thing as both a class and an instance, with the use depending on context. To better understand why we should pun, let's look at a couple of examples, both of which combine organizing categories of things and then describing or characterizing those things. This dual need is common to most domains.

For the first example, let's take a categorization of apes as a kind of mammal, which is then a kind of animal. In these cases, ape is a class (general), which relates to other classes, and apes may also have members, be they particular kinds of apes or individual apes. At the same time, we want to assert some characteristics of apes, such as being hairy, two legs and two arms, no tails, capable of walking bipedally, with grasping hands, and with some being endangered species. These characteristics apply to the notion of apes as an instance. As another example, we may have the category of trucks, which we further split into truck types, brands of trucks, type of engine, and so forth. Again, we may want to characterize that a truck is designed primarily for the transport of cargo (as opposed to automobiles for people transport, both of which are vehicles), or that trucks may have different driver's license requirements or different license fees than autos. These descriptive properties refer to trucks as an instance. These mixed cases combine both the organization of concepts and their relations and set members with the description and characterization of these concepts as things unto themselves. This dual treatment is a natural and common way to refer to things for most any domain of interest.

Prior practice has been to represent these mixed uses in RDFS or OWL Full, which makes them easy to write and create since most 'anything goes' (a loose way of saying that the structures are not decidable).[11] OWL 2 has been designed to fix

[11] A good explanation of this can be found in Rinke J. Hoekstra, 2009. *Ontology Representation: Design Patterns and Ontologies that Make Sense*, thesis for Faculty of Law, University of Amsterdam, *SIKS Dissertation Series No. 2009-15*, 9/18/2009. 241 pp. See http://dare.uva.nl/document/144859. In that, Hoekstra states (pp. 49–50): "RDFS has a non-fixed meta modelling architecture; it can have an infinite number of class layers because rdfs:Resource is both an instance and a super class of rdfs:Class, which makes rdfs:Resource a member of its own subset (Nejdl et al., 2000). All classes (including rdfs:Class itself) are instances of rdfs:Class, and every class is the set of its instances. There is no restriction on defining sub classes of rdfs:Class itself, nor on defining sub classes of instances of instances of rdfs:Class and so on. This is problematic as it leaves the door open to class definitions that lead to Russell's paradox (Pan and Horrocks, 2002). The Russell paradox follows from a comprehension principle built in early versions of set theory (Horrocks et al., 2003). This principle stated that a set can be constructed of the things that satisfy a formula with one free variable. In fact, it introduces the possibility of a set of all things that do not belong to itself …. In RDFS, the reserved properties rdfs:subClassOf, rdf:type, rdfs:domain and rdfs:range are used to define both the other RDFS modelling primitives themselves and the models expressed using these primitives. In other words, there is no distinction between the meta-level and the domain."

this by adding punning, which is to evaluate the object as either a class or an individual based on contextual use; the IRI is shared, but we may view its referent as either a class or an instance based on context. Thus, we allow the use of objects both as concepts (classes) and individuals (instances), the knowledge graph is decidable, and we may use standard OWL 2 reasoners against them [8].

We can diagrammatically show this instance-class dual-use of punning as follows (Fig. 9.2):

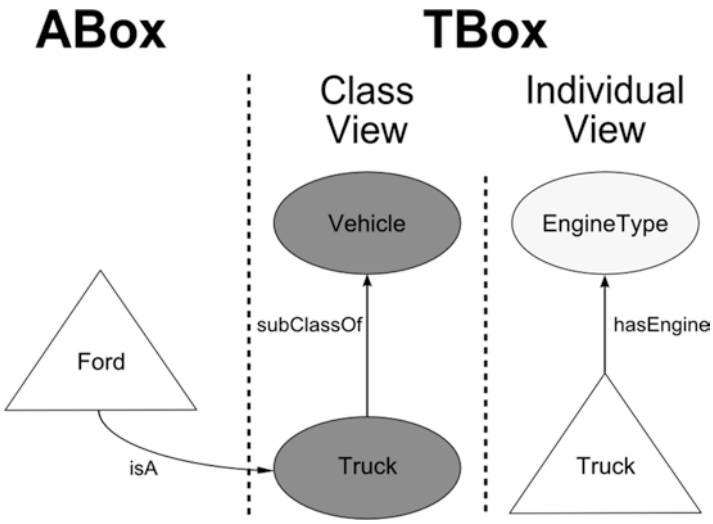


Fig. 9.2 Example of punning 'truck'

Taming a Bestiary of Data Structs

The real world is one of heterogeneous datasets, multiple schema, and differing viewpoints. Even within single organizations—and those which formerly expressed little need or interest to interoperate with the broader world—data integration and interoperability have been a real challenge, as we discussed in Chap. 4. We should view simple instance record assertions and representations—the essence of data exchange—separately from schema representations. Data values and attributes pose similar problems to that of concepts when trying to get datasets to interoperate. Like dictionaries for human languages, or stars and constellations for navigators, or agreed standards in measurement, or the Greenwich meridian for timekeepers, we need fixed references to orient and 'ground' each new dataset over which we attempt to integrate. For data values, symbol grounding means that when we refer to an object or a number—say, the number 4.1—we are also referring to the same metric. 4.1 inches is not the same as 4.1 centimeters or 4.1 on the Richter scale, and object names for set member types also have the same challenges of ambiguous semantics as do all other things referred to by language. Without such fixities of reference,

everything floats concerning other things, the cursed 'rubber ruler' phenomenon. In Chap. 5 we discussed the wide variety of formats and data *structs* in the wild, noting that specific design approaches might be embraced to help. We address how to tame this diversity in this section, at the same time putting our data onto a common framework.

Rationale for a Canonical Model

In the context of data interoperability, a critical premise is that a single, canonical data model is highly desirable. Why? Because of $2N$ v 2^N. That is, a single reference ('canon') structure means that fewer tool variants and converters need be developed to talk to the myriad of data formats in the wild. With a canonical data model, talking to external sources and formats (N) requires only converters to and from the canonical form ($2N$). Without a canonical model, the exponential explosion of needed format converters becomes 2^N, meaning that every format needs to have a converter to and from all of the other formats.[12] For example, without a canonical data model, ten different formats would require 1024 converters, with a canonical format, 20 (assuming bidirectional converters).

A canonical data model merely represents the agreed-upon internal representation. It need not affect data transfer formats. Indeed, in many cases, we may employ different internal data models from what we use for data exchange. Many data systems, in fact, have two or three favored flavors of data exchange such as XML, JSON, or the like. In most enterprises and organizations, the relational data model with its supporting RDBMs is the canonical one. In some notable Web enterprises—say, Google—the exact details of their internal canonical data models are hidden from view, with APIs and data exchange standards being the only portions visible to outside consumers. Generally speaking, a canonical, internal data standard should meet a few criteria:

- Be expressive enough to capture the structure and semantics of any contributing dataset
- Have a schema itself which is extensible
- Be performing
- Have a model to which it is relatively easy to develop converters for different formats and models
- Have published and proven standards
- Have sufficient acceptance to have many existing tools and documentation.

[12] The canonical data model is especially prevalent in enterprise application integration. An entertaining animated visualization of the canonical data model may be found at http://soa-eda.blogspot.com/2008/03/canonical-data-model-visualized.html.

Other desired characteristics might be free or open-source tools, suitable for much analytic work, efficient in storage, and easy for users to read and maintain.

The RDF Canonical Data Model

Many wild data forms are patently inadequate for modeling and interoperability purposes. That is why many of these simpler forms might be called 'naïve': they achieve their immediate objective of simple relationships and communication, but require understood or explicit context to meaningfully (semantically) relate to other forms or data. However, besides naïve forms, two common formats with many variants should also be explicitly considered: the entity-attribute-value (EAV) model and RDBM systems.

EAV is a data model to describe entities where the number of *attributes* (properties, parameters) that can be used to describe them is potentially vast, but the number that may apply to a given entity is relatively modest. In the EAV data model, each attribute-value pair is a fact describing an entity. EAV systems trade off simplicity in the physical and logical structure of the data for complexity in their metadata, which, among other things, plays the role that database constraints and referential integrity do in standard database designs.

On the other hand, RDBMSs use the relational model and store their data in a tabular form, with rows corresponding to the individual data records and columns representing the properties or attributes. RDF can be modeled relationally as a single table with three columns corresponding to the *subject-predicate-object* triple. Conversely, a relational table can be modeled in RDF with the *subject* IRI derived from the primary key or a blank node; the *predicate* from the column identifier; and the *object* from the cell value. Because of these affinities, it is also possible to store RDF data models in existing relational databases. (In fact, many RDF 'triple stores' are RDBM systems with a tweak, sometimes as 'quad stores' where the fourth tuple is the *graph*.) Moreover, these affinities also mean that RDF stored in this manner can also take advantage of the historical experience gained from RDBMS performance and SQL query optimizations.

RDF (Resource Description Framework) might be called a superset of these two forms and is exquisitely suited to accommodate them. In fact, because of its flexible data structure ranging from implied EAV through both of these forms and including schema as well, RDF is a kind of 'universal solvent' that can readily model most any known data form.[13] When we match this flexible format representation with the ability to handle semantic differences through ontologies at the OWL 2 level, it is clear why RDF provides a competent data model around which to build an interoperable framework. Moreover, because we give all of the information unique Web identifiers (IRIs), and the whole system resides on the Web accessible via the HTTP

[13] Minor exceptions, such as modeling recursion in ASN.1 (Abstract Syntax Notation), are so rarely encountered as to be dismissed.

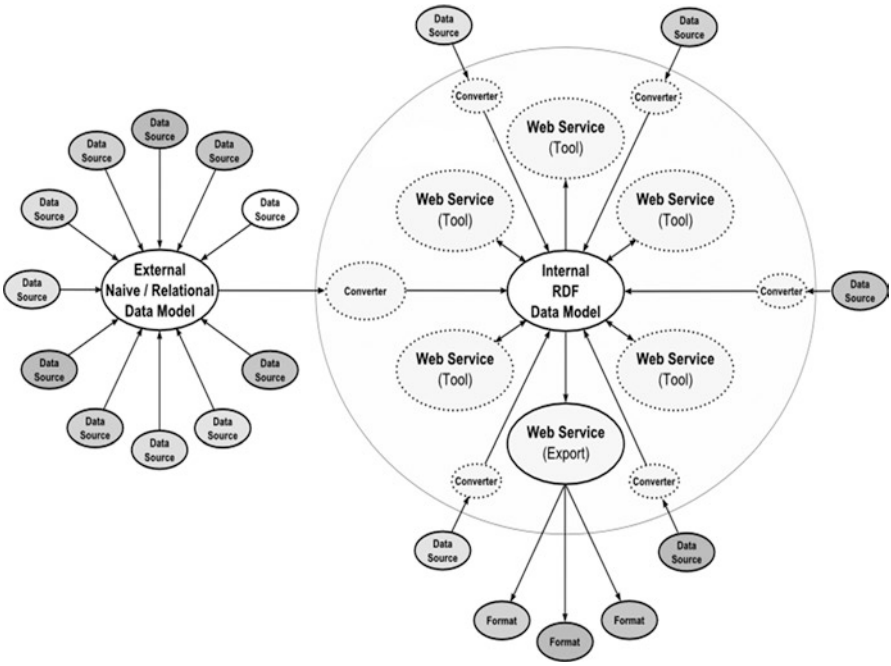


Fig. 9.3 Interoperability and RDF as the canonical data model

protocol, our information may reside anywhere the Internet connects. These are the reasons why we have chosen RDF as our canonical data model.

Figure 9.3 shows how we approach data interoperability. The input bubbles at the left of the diagram represent the different data formats that the system must address. We process each through a converter to RDF, which is the internal form used on the right. We map schema information associated with each left-side external source to this internal RDF (and OWL 2) data model in advance, before ingesting content.[14] Note that the converter responsible for the external content ingest may (should!) be a callable Web service so that external sources may call for or schedule ingests according to security standards.

Converters (also known as translators or RDFizers)[15] are an essential bridge to this external world, which we should design for reuse and extensibility. While some may be one-off converters (sometimes off-the-shelf RDFizers), and often devoted to large-volume external data sources, it is also helpful to emphasize one or more 'standard' naïve external formats. A 'standard' external format allows for a more

[14] Depending on the nature of the new external content, it may also be necessary to update the *knowledge graph* at this point.

[15] As of the writing of this book, my census found hundreds of converters of various record and data structure types to RDF. These converters—also sometimes known as translators or 'RDFizers'—take some input data records with varying formats or serializations and convert them to a form of RDF serialization (such as RDF/XML or N3), often with some ontology matching or characterizations.

sophisticated converter and enables specific tools more easily justified around the standard naïve format. In today's environment, that 'standard' may be JSON or a derivative, or new standards as they arise. Other common 'naïve' formats could be SQL from relational databases or other formats familiar to the community at hand.

In many ways, because we emphasize the *ABox* and instance records and assertions in data exchange, the actual format and serialization are pretty much immaterial. Emphasizing one or a few naïve external formats is the cost-effective approach to tools and services. Even though the format(s) chosen for this external standard may lack the expressiveness of RDF (because the burden is principally related to data exchange), we can readily optimize this layer for the deployment at hand.

Other Benefits from a Canonical Model

As we can see in Fig. 9.3, converters may themselves be *bona fide* Web services. Besides import converters, it is also useful to have export services for the more broadly used naïve external formats. Exporters allow us to share data and schema with external applications, up to the full expressiveness of RDFS, SKOS, or OWL 2. We may devote other services to data cleanup or attribute (property) or object reconciliation (including disambiguation). In this manner, we can improve the authority and trustworthiness of installations while promoting favored external data standards. Another common service is to give naïve data unique IRI identifiers and to make it Web accessible, thus turning it into linked data.

Such generic services are possible because the canonical RDF model is the 'highest common denominator' for the system. Because RDF is the consistent basis for tools and services, once a converter is available, and we have mapped the external information schema to the internal structure, we can reuse all existing tools and services. Moreover, we are now ready to share this system and its datasets with other instances, within the organization and beyond.

References

1. P.F. Patel-Schneider, I. Horrocks, Position paper: a comparison of two modelling paradigms in the semantic web, in *Proceedings of the 15th international conference on World Wide Web* (ACM, New York, 2006), pp. 12–12
2. K.O. Stanley, J. Lehman, L. Soros, Open-Endedness: The Last Grand Challenge You've Never Heard Of, *O'Reilly Media,* https://www.oreilly.com/ideas/open-endedness-the-last-grand-challenge-youve-never-heard-of
3. F. Baader, D. Calvanese, D. McGuiness, D. Nardi, P. Patel-Schneider, *The Description Logic Handbook: Theory, Implementation and Applications* (Cambridge University Press, Cambridge, 2003)
4. Anon, KBpedia—Open Standards, *KBpedia,* http://kbpedia.com/standards/
5. Princeton University, Wngloss(7wn) Manual Page, *WordNet 3.0 Reference Manual.* https://wordnet.princeton.edu/wordnet/man/wngloss.7WN.html

6. E. Svenonius, *The Intellectual Foundation of Information Organization* (MIT Press, Cambridge, 2000)
7. M.L. Zeng, M. Žumer, A. Salaba, *Functional Requirements for Subject Authority Data (FRSAD): A Conceptual Model* (Walter de Gruyter, Berlin, 2011)
8. P. Hitzler, M. Krötzsch, B. Parsia, P.F. Patel-Schneider, S. Rudolph, *OWL 2 Web Ontology Language Primer* (2012)
9. R. Reiter, On closed world data bases, *Logic and Data Bases* (H. Gallaire and J. Minker, eds., Plenum Press, New York, 1978), pp 55-76

Chapter 10
Modular, Expandable Typologies

Typology is not a typical term within semantic technology circles, though it is used extensively in such fields as archaeology, urban planning, theology, linguistics, sociology, statistics, psychology, anthropology, and others. Based on etymology, 'typology' is the study of types.[1] However, as used in the fields noted, a 'typology' is the result of the classification of things according to their characteristics. As stated by Merriam Webster, a 'typology' is 'a system used for putting things into groups according to how they are similar.' Though some have attempted to make academic distinctions between typologies and similar notions such as classifications or taxonomies [1], we think this idea of grouping by similarity is the best way to think of a typology. In this classification, each of our SuperTypes, as was introduced in the prior chapter, gets its own typology. The idea of a SuperType, in fact, is exactly equivalent to the root node of a typology, wherein we relate multiple entity types with similar essences and characteristics to one another via a natural classification.

A *typology* is a systematic classification of types according to their common or shared characteristics. A typology could be composed of all living things; all animals; all product types; or something as narrow as supervised machine learning algorithms. While *types* are classifications of *instances* according to their shared characteristics, a *typology* is a classification of *types* on a similar basis. Types enable us to 'carve Nature at the joints,' while typologies give us the organizational framework for coherently subsuming types under a general type. We have complete flexibility to define our general root type for a given typology as narrowly or broadly as we wish, flexibility of immense design importance.

[1] Some material in this chapter was drawn from the author's prior articles at the AI3:::Adaptive Information blog: "Climbing the Data Federation Pyramid" (May 2006); "'Structs': Naïve Data Formats and the ABox" (Jan 2009); "Advantages and Myths of RDF" (Apr 2009); "structWSF: A Framework for Data Mixing" (Jun 2009); "Big Structure and Data Interoperability" (Aug 2014); "Logical Implications of Interoperability" (Jun 2015); "How Fine Grained Can Entity Types Get?" (Mar 2016); "Rationales for Typology Designs in Knowledge Bases" (Jun 2016); "Threes All of the Way Down to Typologies" (Oct 2016).

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8_10

In this chapter, we discuss the use of types as our general classification structure, and then typologies as modular ways to further organize those types. We give particular attention to the benefits that accrue from a typology design. We conclude our chapter with an overview of the typologies used in the KBpedia knowledge structure, and how its design may act as a prototype for other domain applications.

Types as Organizing Constructs

Typologies assume the structural form of a subsumption hierarchy, most often in the form of a tree. Each node in that tree is a type, which by definition is itself a Thirdness or general using Peirce's universal categories. Since types are the basic building blocks of a typology, let's take a few minutes to understand them.[2]

The Type-Token Distinction

As used in knowledge representation and philosophy, *types* are the classification of natural kinds. Besides kinds, we often equate types to sets or laws (the Peircean view, using his common terms) [2]. Peirce is clear that a type is a general, or collective (1908, CP 8.367). Also, in the base semantic technology language of RDF, a 'type' is what is used to declare an instance of a given class. This use is in keeping with the sense of an *instance* as a member of a type.

Toward the end of his career, Peirce proposed *mark*, *token*, and *type* as the trichotomous forms of symbols corresponding, respectively, to his universal categories of Firstness, Secondness, and Thirdness (1908, CP 8.364). The examples he posed to illustrate these ideas related to language symbols. His quote on these distinctions is:

> A common mode of estimating the amount of matter in a MS [manuscript]. or printed book is to count the number of words. There will ordinarily be about twenty *the's* on a page, and of course they count as twenty words. In another sense of the word 'word,' however, there is but one word 'the' in the English language; and it is impossible that this word should lie visibly on a page or be heard in any voice, for the reason that it is not a Single thing or Single event. It does not exist; it only determines things that do exist. Such a definitely significant Form, I propose to term a *Type*. A Single event which happens once and whose identity is limited to that one happening or a Single object or thing which is in some single place at any one instant of time, such event or thing being significant only as occurring just when and where it does, such as this or that word on a single line of a single page of a single copy of a book, I will venture to call a *Token*. An indefinite significant character such as a tone of voice can neither be called a Type nor a Token. I propose to call such a Sign a *Tone* [later *mark*]; In order that a Type may be used, it has to be embodied in a Token which shall be a sign of the Type, and thereby of the object the Type signifies. I propose to call such a

[2] From a vocabulary standpoint, we describe the role of nouns and relations in Chap. 7; I also speculate on parts of speech in relation to Peircean terminology in Chap. 16 (esp. Table 16.2).

> Token of a Type an *Instance* of the Type. Thus, there may be twenty Instances of the Type 'the' on a page.—(1906, CP 4.537)

The quote nicely illustrates the distinctions. However, in my view, a too literal reading of this passage by some linguists and semioticians has unfortunately led to a couple of misinterpretations. The first is that tokens are simply occurrences. They are not; they are instances, which may include entities and events (1903, CP 2.245). The second misinterpretation is that the distinction applies to only the symbols on the page, as opposed to the complete sign that those symbols represent. In other words, the *type-token* distinction is not one solely of the written page, a too limited interpretation, but is one of representation by symbols of any type to the particular-general distinction. Tokens and types are not limited to words, but to any instance and general distinction that we may represent symbolically. In Peirce's 10-classification schema for signs, he defined *sinsign*, another term for token:

> A *Sinsign* (where the syllable *sin* is taken as meaning 'being only once,' as in *single*, *simple*, Latin *semel*, etc.) is an actual existent thing or event which is a sign. It can only be so through its qualities; so that it involves a qualisign, or rather, several qualisigns. But these qualisigns are of a peculiar kind and only form a sign through being actually embodied.—(1903, CP 2.245)

Further, he equated the idea of *type* with the *legisign*:

> A *Legisign* is a law that is a Sign. This law is usually established by men. Every conventional sign is a legisign [but not conversely]. It is not a single object, but a general type which, it has been agreed, shall be significant. Every legisign signifies through an instance of its application, which may be termed a *Replica* of it. Thus, the word 'the' will usually occur from fifteen to twenty-five times on a page. It is in all these occurrences one and the same word, the same legisign. Each single instance of it is a Replica. The Replica is a Sinsign. Thus, every Legisign requires Sinsigns.—(1903, CP 2.246)

That Peirce saw the 'collective' type as a general, and not limited solely to language matters, also comes from his referring to his three methods of reasoning—deductive, inductive, and abductive—as types (1913, CP 8.385).

OK, so types and tokens are not limited to the written word and can apply to any general-particular distinction, respectively.[3] Types have an identity and are real, but are not an existent as defined as something with a material instantiation. 'Every conventional sign is a legisign' (CP 2.246), or type, it is not an individual thing, and every type requires instances, or sinsigns (CP 2.246).

[3] Aspects of Peirce's definition of types have some interesting parallels to type theory (https://en.wikipedia.org/wiki/Type_theory), especially homotopy type theory (https://en.wikipedia.org/wiki/Homotopy_type_theory), that we do not have time to pursue further here. In type theory, well-founded types are ones where we can define objects by primitive recursion and prove properties by induction (see Thompson, S., *Type Theory and Functional Programming*, Addison Wesley, 1991). Primitive recursion over Boolean properties (which is why dichotomous keys for classification are so useful) is an interesting link to type theory, as are type families and creating new types. Further, some proposed resolutions to improve the representation of subsets in type theories involve representing propositions distinct from types or as types.

Types and Natural Classes

In Chap. 5 we discussed the importance of natural classes, which is related to a realistic view of the world.[4] Realism means we believe what we perceive in the world is real—it is not just a consequence of what we understand and can be aware of in our minds—and that there are forces and relationships in the world independent of us as selves. Realism is a long-standing tradition in philosophy that extends back to Aristotle and embraces, for example, the natural classification systems of living things as espoused by taxonomists such as Agassiz and Linnaeus. Adhering to realism and a natural classification is the best way to create and organize our types.

Peirce embraced this realistic philosophy but also embedded it in a belief that our understanding of the world is fallible and that we needed to test our perceptions via logic (the scientific method) and shared consensus within the community. As we have noted, his overall approach is known as pragmatism and is firmly grounded in his views of logic and his theory of signs (called semiotics or semeiotics). While absolute truth is real, it acts more as a limit, to which our seeking of additional knowledge and clarity of communication with language continuously approximates. Through the scientific method and questioning, we get closer and closer to the truth and to an ability to communicate it to one another. Still, new knowledge may change those understandings, which in any case will always remain proximate.

An intensional understanding of attributes is key to the classification of entities into categories (that is, 'types'). Further, Peirce was expansive in his recognition of what kinds of objects could be classified, specifically including ideas, with application to areas such as social classes, human-made objects, sciences, chemical elements, and living organisms [3]. Again, here are some of Peirce's own words on the classification of entities:

> All classification, whether artificial or natural, is the arrangement of objects according to ideas. A natural classification is the arrangement of them according to those ideas from which their existence results.—(1902, CP 1.231)

> The natural classification of science must be based on the study of the history of science; and it is upon this same foundation that the alcove-classification of a library must be based.—(1903, CP 1.268)

> All natural classification is then essentially, we may almost say, an attempt to find out the true genesis of the objects classified. But by genesis must be understood, not the efficient action which produces the whole by producing the parts, but the final action which produces the parts because they are needed to make the whole. Genesis is production from ideas. It may be difficult to understand how this is true in the biological world, though there is proof enough that it is so. But in regard to science it is a proposition easily enough intelligible. A science is defined by its problem; and its problem is clearly formulated on the basis of abstracter science.—(1902, CP 1.227)

[4] Philosophers often contrast realism to idealism, nominalism, or conceptualism, wherein how the world exists is a function of how we think about or name things. Descartes, for example, summarized his conceptualist view with his aphorism "I think, therefore I am."

A natural classification system is one, then, that logically organizes entities with shared attributes into a hierarchy of types, with each type inheriting attributes from its parents, distinguished by what Peirce calls its *final cause*, or purpose. This hierarchy of types is thus naturally termed a *typology*.

An individual that is a member of a natural class has the same kinds of attributes as other members, all of which share this essence of the final cause or purpose. We look to Peirce for the guidance in this area because his method of classification is testable, based on discernable attributes, and grounded in logic. Further, that logic is itself grounded in his theory of signs, which ties these understandings ultimately to natural language. Peirce's own words can better illustrate his perspective, some of which I have discussed elsewhere under his idea of 'natural classes' (see Chap. 5):

> Thought is not necessarily connected with a brain. It appears in the work of bees, of crystals, and throughout the purely physical world; and one can no more deny that it is really there, than that the colors, the shapes, etc., of objects are really there.—(1906, CP 4.551)

> What if we try taking the term 'natural,' or 'real, class' to mean a class of which all the members owe their existence as members of the class to a common final cause? This is somewhat vague; but it is better to allow a term like this to remain vague, until we see our way to rational precision.—(1902, CP 1.204)

> … it may be quite impossible to draw a sharp line of demarcation between two classes, although they are real and natural classes in strictest truth. Namely, this will happen when the form about which the individuals of one class cluster is not so unlike the form about which individuals of another class cluster but that variations from each middling form may precisely agree… When one can lay one's finger upon the purpose to which a class of things owes its origin, then indeed abstract definition may formulate that purpose. But when one cannot do that, but one can trace the genesis of a class and ascertain how several have been derived by different lines of descent from one less specialized form, this is the best route toward an understanding of what the natural classes are.—(1902, CP 1.208).

'Natural classes' thus are a testable means to organize the real objects in the world, which include both Secondness and Thirdness. Secondness consists of all extant things, namely, entities and events. We include Thirdness because generals are real. What makes these items real and classifiable into types is because they have (1) *identity*, which means we may refer to them via symbolic names; (2) *context* related to other objects; (3) characteristic *attributes*, with some expressing the essence of what type of object it is; and (4) *realness*, since the general is not a fiction of our minds but a type recognized by others.

Natural classifications may apply to truly 'natural' things, like organisms and matter, but also to human-made objects and social movements and ideas. The key argument is that shared attributes, including a defining kind of 'essence' (Aristotle) or 'final cause' (Peirce), help define the specific class or type to which an object may belong. For Peirce, what science has to tell us, or what social consensus settles upon, holds sway. If accomplished well, natural classification systems lend themselves to hierarchical structures that may be reasoned over. Further, if we make natural splits between typologies, then it is also possible to establish nonoverlapping ('*disjoint*') relationships between typologies that provide powerful restriction

and selection capabilities across the knowledge structure. We believe that KBpedia already achieves these objectives, though we continue to refine the structure based on our mappings to other external systems and other logical tests.

Very-Fine-Grained Entity Types

Entity recognition or extraction is a key task in natural language processing (NLP) and one of the most common uses for knowledge bases. Entities are the unique, individual things in the world. Entities typically account for 90% or so of the items in a knowledge base that we may type, though we may also type events, attributes, relations, ideas, and concepts. Context plays an essential role in entity recognition. We have come to define types of finer and finer bases over time.

The 'official' practice of named entity recognition used within NLP began with the Message Understanding Conferences, especially MUC-6 and MUC-7, in 1995 and 1997. These conferences began competitions for finding '*named entities*' within candidate texts as well as the practice of in-line tagging [4]. Many named entities signal their status via capitalization, such as *Rome* or *John F. Kennedy*. Sometimes named entities are also written in lower case, with examples such as rocks ('gneiss') or common animals or plants ('daisy') or chemicals ('ozone') or minerals ('mica') or drugs ('aspirin') or foods ('sushi'). We give some deference to Kripke's idea of 'rigid designators' for how to identify entities; rigid designators include proper names as well as natural kinds of terms like biological species and substances. Because of these blurrings, the nomenclature of 'named entities' began to fade away, though some practitioners still use the term. Much has changed in the 20 years since the seminal MUC conferences regarding entity recognition and characterization. We are learning to adopt a very-fine-grained approach to entity types. What we see evolve with fine-grained entity types has led us, in part, to the logic of our typology design.

The original MUC conferences only recognized three initial entity types: person, organization, and location names. However, it did not take long for various groups and researchers to want more entity types, more distinctions. BBN categories, proposed in 2002, were used for question answering and consisted of 29 types and 64 subtypes [5]. Sekine put forward and refined over many years his extended entity types, which grew to about 200 types [6]. These ideas of extended entity types helped inform a variety of tagging services over the past decade, notably including OpenCalais, Zemanta, AlchemyAPI, and OpenAmplify, among others. Moreover the research community also expanded its efforts into more and more entity types, or what we now term fine-grained entities.[5] Ling and Weld proposed 112 entity types in 2012 [7]. Another one, from Gillick et al. in 2014, proposed 86 entity types

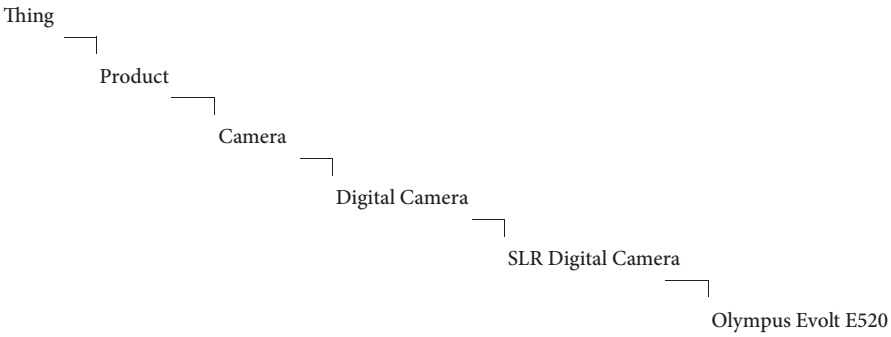
[5] For example, try this query, https://scholar.google.com/scholar?q="fine-grained+entity," also without quotes.

[8], organized, in part, according to the same person, organization, and location types from the earliest MUC conferences. A report in 2017 pointed to 1941 entity types drawn from both Wikipedia and Wordnet [9]. In KBpedia, at the time of this writing, we provide mappings to a large number of entity types in external knowledge bases, including the DBpedia ontology (738 entity types), schema.org (636 types) and GeoNames (654 types).

This growth in entity types comes from wanting to describe and organize things with more precision. Tagging and extracting structured information from text are obviously key drivers. For a given enterprise, what is of interest—and at what depth—for a particular task varies widely. The desired depth, or degree of fine graininess, increases for entity types within our domains of inquiry. For example, let's take a general thing such as a *camera*. A photographer may want finer grained distinctions such as *SLR cameras* or further subtypes like *digital SLR cameras* or even specific models like the *Canon EOS 7D Mark II*, or even the name of the photographer's favorite camera, such as 'Shutter Sue.' Capitalized names (common for named entity recognition) often signal we are dealing with an actual individual entity, but again, depending on context, a named automobile such as Chevy Malibu may refer to a specific car or the entire class of Malibu cars. If our domain of interest is transportation in general, treating the Chevy Malibu as an instance of a Chevy may be sufficient; but if our domain of inquiry includes Chevrolet automobiles, we probably want details including specific years and models of Malibus. This kind of hierarchical organization provides paths for inferencing, as well as user interface benefits (see Chap. 11). We can visualize this hierarchy of types something like what we show in Table 10.1.

Recent efforts in fine-grained entity recognition are notable because machine learners have been trained to recognize all of the various types indicated. Which entity types to include, the different conceptions of the world, and how to organize entity types vary broadly across these references.

Table 10.1 Hierarchical nature of typologies



Perhaps 40,000 entity types are included in the baseline KBpedia knowledge structure to accommodate such fine-grained entity recognition. Over the past two decades we see logarithmic growth in recognition of entity types (Fig. 10.1).

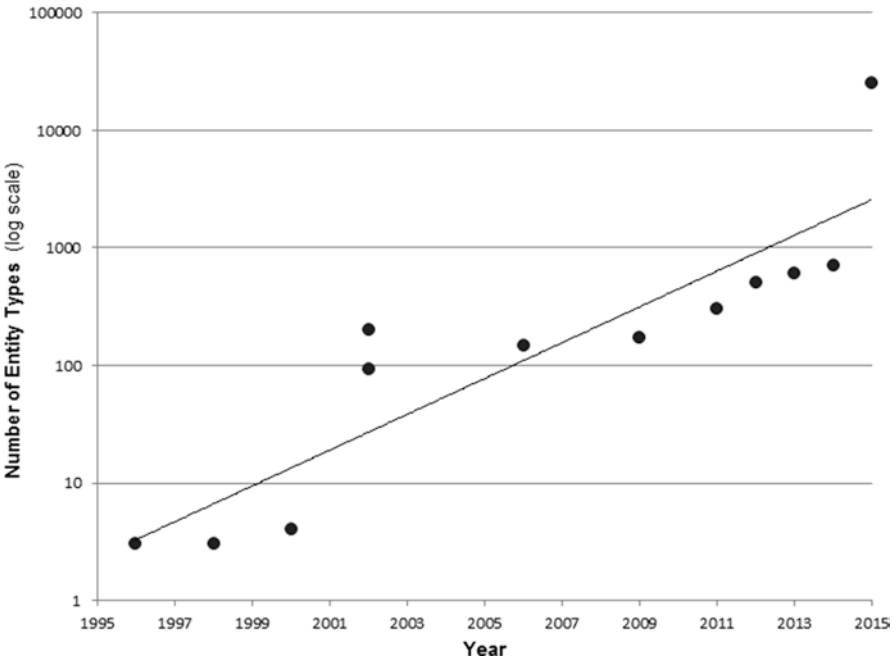


Fig. 10.1 Growth in recognition of entity types

Each type has a basis—ranging from attributes and characteristics to hierarchical placement and relationship to other types—that can inform computability and logic tests, potentially including neighbor concepts. We base supervised machine learners on these features. Linking to knowledge bases helps provide the instance data to drive these learners. Ensuring that type placements are accurate and meet these tests means that we may use the now-placed types and their attributes to test the placement and logic of subsequent candidates. The candidates need not be only internal typology types, but may also be used against external sources for classification, tagging, or mapping.

It is no longer sufficient to satisfy our domain queries with entity types classified at the level of person, organization, and location names. We want more precision; more detail; and more relevance. The fact that knowledge bases, such as Wikipedia, but also our own data stores and domain-specific knowledge bases as well, can provide entity-level instance information for literally thousands of entity types means that rich information is now available for driving the finest of fine-grained entity extractors. It is essential to have a grounded understanding of what an entity is, how to organize them into logical types, and an intensional understanding of the attributes and characteristics that allow us to conduct inferencing over these types. These understandings, in turn, point to the features that are useful to machine learners for artificial intelligence. These understandings can also inform a flexible design for accommodating entity types from coarse to fine grained, with variable depth depending on the domain of interest.

A Flexible Typology Design

The only sane way to tackle knowledge bases for knowledge representation and management is to provide consistent design patterns that are easier to test, maintain, and update. Open-world systems must embrace repeatable and mostly automated workflow processes, plus a commitment to timely updates, to deal with the constant change in knowledge. We also want natural workflows for the knowledge workers who use them; else quality checks and frequent updates will suffer. We thus seek semiautomatic methods for constant knowledge updates.

The *typology* structure is not only a natural organization of natural classes, but it also enables flexible interaction points with inferencing and mapping across its design. The typology design is the result of the classification of things according to their shared attributes and *essences*. The idea is that we divide the world into real, discontinuous, and immutable 'kinds.' If using statistical terminology, a typology is a composite measure that involves the classification of observations using attributes treated as variables.

Our typology design has arisen from the intersection of (1) our efforts with SuperTypes, and creating a computable structure that uses powerful disjoint assertions; (2) an appreciation of the importance of entity types as a focus of knowledge base terminology; and (3) our efforts to segregate entities from other constructs of knowledge bases, including attributes, relations, and annotations. Though these insights may have resulted from serendipity and practical use, they have brought a new understanding of how best to organize knowledge bases for artificial intelligence uses. It just so happens that these splits are in complete accordance with Peirce's writings.

The simple bounding and structure of the typology design make each typology understandable merely by inspecting its structure. The typologies can also be read into programs such as Protégé to examine or check complete specifications and relationships. Because each typology attempts to have a coherent, modular, and consistent design, new concepts or structures may be related to any part of its hierarchical design. The organization of entity types also has a different structure than the more graph-like organization of higher level conceptual schema or knowledge graphs. In the cases of broad knowledge bases, such as Wikipedia, where 70% or more of the overall schema is related to entity types, more attention can now be devoted to the remaining 30%, to extend insights into type placements based on relationships and attributes. The combination of logical coherence with a flexible, accordion-like structure gives typologies a unique set of design benefits.

Construction of the Hierarchical Typologies

We develop the initial typology by first gathering the relevant types (concepts) and automatically evaluating them for orphans (unconnected concepts) and fragments (connected portions missing intermediary parental concepts). We allow no instances

in the typology, only types.[6] We typically see multiple roots, multiple fragments, and numerous orphans in the first builds. We test and refine until we fix these problems, resulting in a single root to connect all concepts in the typology. We query source knowledge bases for missing concepts and evaluate again in a recursive manner. We then write candidate placements to CSV files and evaluate them with various utilities, including crucially manual inspection and vetting. (Because the system bootstraps what is already known and structured in the system, it is mandatory to build the structure with coherent concepts and relations.)

We should include all types related to a given typology as a sub-module. This design means we may maintain and inspect each typology separately. We may share some types across typologies (due to multiple inheritances), and when many or all typologies are present the entire knowledge system assumes the form of an interconnected graph. External structures, especially those based on SKOS,[7] are well suited for direct incorporation as typologies.

Once we complete the overall candidate structure, we then analyze it against prior assignments in the knowledge base. We create CSV files that we may view and evaluate with various utilities for such tasks as SuperType disjoint analysis, coherent inferencing, and logical placement tests. Again, however, to retain the integrity of the structure, we manually vet final assignments. The objective of the build process is a connected typology that passes all coherency, consistency, completeness, and logic tests. If we discover subsequent errors, we rerun the build process with updates to the processing scripts. Upon acceptance, we should ensure that each new type added to a typology is complete by including a definition, *semset*, guideline annotations, and connections. We write out the completed typology in both RDF and CSV formats.

Build, testing and maintenance routines, scripts, and documentation must be integral to the design. Knowledge bases are inherently open world, which means that the entities and their relationships and characteristics are continually growing and changing due to new knowledge underlying the domain at hand. Such continuous processing and keeping track of the tests, learnings, and workflow steps place a real premium on literate programming. We discuss the build process in Chap. 14.

Typologies Are Modular

Since each typology has a single root, it is readily plugged into or removed from the broader structure. Each typology is rather simple in scope and structure, given its hierarchical nature. We can readily build, test, and maintain each typology.

[6] However, like the Chevy Malibu case described earlier, items that appear as instances in the putative typology may be expanded to become an eventual class (type), with its own instances, akin to the *punning* discussion in the prior chapter.

[7] For best interoperability with KBpedia, the SKOS reference should include the SKOS DL version; see M.K. Bergman, "SKOS Now Interoperates with OWL 2," *AI3:::Adaptive Information* blog, February 10, 2011.

Typologies pose relatively small ontological commitments. This isolated design means that the scale and scope of the overall system may be easily adjusted, and we may use the existing structure as a source for extensions (see next). Unlike more interconnected knowledge graphs (which can have many network linkages), typologies are organized strictly along these lines of shared attributes, which is both simpler and also provides an orthogonal means for investigating type-class membership.

Our learning path to a typology design began with our early experience with SuperTypes in UMBEL.[8] We started to explore typologies (though we did not call them that at the time) because we observed that about 90% of the concepts in UMBEL were *disjoint* from one another. Disjoint assertions are computationally efficient and help better organize a knowledge graph. Besides computational efficiency and its potential for logical operations, we also observed that these SuperTypes could also aid our ability to drive display widgets (such as being able to display geolocational attributes for geolocational types on maps). As we looked over the tens of thousands of concepts in UMBEL, we began to see that we could organize them into a tractable number of SuperTypes. The SuperType tagging and possible segregation of STs into individual modules led us to review other separations and tags. Given that the SuperTypes were all geared to entities and entity types—and further represented about 90% of all concepts at hand—we began to look at entities as a category with more care and attention. This analysis took us back to the beginnings of entity recognition and tagging in natural language processing. We saw the progression of understanding from named entities and just a few entity types, to the more recent efforts in the so-called fine-grained entity recognition, as we reviewed above.

What was blatantly obvious, but which other researchers and we had previously overlooked, was that most knowledge graphs (or upper ontologies) were themselves made up of mostly entity types. In retrospect, this should not have been surprising. Most knowledge graphs deal with real things in the world, which are most often entities. Entities are the observable, often nameable, things in the world around us. How we organize and refer to those entities—that is, the entity types—constitutes the bulk of the vocabulary for a knowledge graph. The trick to the transition is moving from the idea of discrete numbers of entity types to a system and design that support continuous interoperability through a generalized, modular typology structure.

The '*type*-orientation' of a typology is also attractive because it offers a construct that we can apply to all other (nonentity) parts of the knowledge base. We can also type actions, attributes, roles, events, and relations. A mindset around natural kinds and types helps define the *speculative grammar* we need to do *knowledge-based artificial intelligence*. Because the essential attributes or characteristics across

[8] UMBEL was a precursor to KBpedia first begun in 2006 and first released in 2008. KBpedia has now supplanted its earlier design.

typologies in an entire domain can differ broadly—such as entities *v* attributes, living *v* inanimate things, natural things *v* human-made things, and ideas *v* physical objects—it is possible to make disjointedness assertions between entire groupings of natural classes. Disjoint assertions, combined with logical organization and inference, further provide a typology design that lends itself to reasoning and tractability. The internal process to create these typologies also has the beneficial effect of testing placements in the knowledge graph and identifying gaps in the structure, as informed by fragments and orphans. This computability of the structure is its foundational benefit since it determines the accuracy of the typology itself and drives all other uses and services.

Typologies Are Expandable

A typology design for organizing entities can thus be visualized as a kind of accordion or squeezebox, expandable when detail requires, or collapsible to more coarse-grained when relating to broader views. Each class (type) within the typology can become a tie-in point for external information, providing a collapsible or an expandable scaffolding (the 'accordion' design). Via inferencing, multiple external sources may be related to the same typology, even though at different levels of specificity. Further, we may accommodate very detailed class structures in this design for domain-specific purposes. Moreover, because of the single tie-in point for each typology at its root, it is also possible to swap out entire typology structures at once, should design needs require this flexibility.

The idea of nested, hierarchical types organized into broad branches of different entity typologies also provides a very flexible design for interoperating with a diversity of worldviews and degrees of specificity. A typology design, logically organized and placed into a consistent grounding of attributes, can readily interoperate with these different worldviews. The photographer, as discussed above, is interested in different camera types and even how specific cameras can relate to a detailed entity typing structure. Another party more interested in products across the board may have a view to greater breadth, but lesser depth, about cameras and related equipment. A typology design, logically organized and placed into a consistent grounding of attributes, can readily interoperate with these different worldviews. Typologies for attributes and relations, as we have implemented in KBpedia, also extend this basis to include full data interoperability of attribute:value pairs.

KBpedia's Typologies

So, to understand this typology design in action, it is worth inspecting the KBpedia knowledge structure. I provide the general vocabulary for KBpedia in Chap. 8. *Appendix B* is a broad overview of KBpedia. Further, the KBpedia Web site offers access to KBpedia's overall upper structure (KKO),[9] plus the approximately 70 typologies with formal-type listings. You may inspect the KKO files and the typologies in an ontology editor to glean additional details.[10] As noted, nearly 90% of the classification structure of KBpedia resides in the *Generals* branch of KKO, which is also the location for all KBpedia types and typologies.

Full Listing of Typologies

Unlike the KKO upper structure, we do not necessarily organize each typology according to Peirce's triadic logic. That is because we are dealing with objects of a more or less uniform character (such as *animals* or *products* or *atomic elements*). About 85 such typologies exist in the KBpedia structure, 70 of which with formal typologies ('◇'), and about 30 of which are deemed 'core' ('□'), meaning they capture the bulk of the classificatory system.

The best perspective to see the full listing of the typologies in KBpedia is to inspect the *Generals* branch of the KKO knowledge graph, which also includes *Predications* types and contains about 85 SuperTypes. Table 10.2 provides this *Generals* branch organization, as first organized around the Peircean universal categories of Firstness (1), Secondness (2), and Thirdness (3). Also, recall that the *Generals* branch is itself the Thirdness (3) branch of the broader KBpedia Knowledge Ontology (KKO)[11]:

[9] See http://kbpedia.org.

[10] For example, using the open-source Protégé ontology development environment (https://protege.stanford.edu/).

[11] The remaining portions of the upper KKO are shown in Chap. 8.

Table 10.2 Full, upper hierarchy of KBpedia generals

3-Generals (SuperTypes)
1 Constituents ◊
1 Natural phenomena □
2 Time types ◊
Times □
Event types □
3 Space types ◊
1 Shapes □
2 Places □
Area region □
Location place □
3 Forms □
2 Predications ◊
1 Attribute types ◊
1 Intrinsic attributes ◊
1 Qualities
2 Components
3 Forms (configurations)
2 Adjunctual attributes ◊
1 Quantities
2 Eventuals
3 Extrinsics
3 Contextual attributes ◊
1 Situants
2 Ratings
3 Classifications
2 External relation types ◊
1 Direct relations ◊
1 Equivalences
2 Parts
3 Descendants
2 Copulative relations ◊
1 Identities
2 Action types ◊
3 Conjoins
3 Mediative relations ◊
1 Comparisons
2 Situation types □
3 Cognitives
3 Representation types ◊
1 Denotatives ◊
2 Indexes ◊
3 Associatives ◊

(continued)

Table 10.2 (continued)

3-Generals (SuperTypes)				
3 Manifestations ◊				
	1 Natural matter ◊			
		1 Atom elements □		
		2 Natural substances □		
		3 Chemistry □		
	2 Organic matter ◊			
		1 Organic chemistry □		
			Biological processes □	
		2 Living things ◊		
			1 Prokaryotes □	
			2 Eukaryotes ◊	
				1 Protist fungus □
				2 Plants □
				3 Animals □
			3 Diseases □	
		3 Agents ◊		
			1 Persons □	
			2 Organizations □	
			3 Geopolitical □	
	3 Symbolic ◊			
		1 Information ◊		
			1 AV info ◊	
				Audio info □
				Visual info □
			2 Written info □	
			3 Structured info □	
		2 Artifacts ◊		
			FoodDrink □	
			Drugs □	
			Products □	
			Facilities □	
		3 Systems ◊		
			1 Conceptual systems ◊	
				1 Concepts ◊
				2 Topic categories ◊
				3 Learning processes ◊
			2 Social systems ◊	
				Society □
				Economic systems □
			3 Methodeutic ◊	
				1 Inquiry methods ◊
				2 Knowledge domains ◊
				3 Emergent knowledge ◊

In Table 10.2 the mark '◊' indicates a formal typology for the entry in KBpedia, which means that a corresponding file for inspecting it exists. The mark '□' indicates that the formal typology is also one of the 'core' KBpedia typologies, meaning that it contains a more substantial number of types with possible disjointedness assertions to other typologies.[12] If time is limited, those typologies are the most fruitful to inspect. The largest files, of course, are the ones with the largest number of types.

'Core' Typologies

So, let's take a bit deeper look at these 30 'core' ('□') typologies. Here are those 30, with a definition of the type coverage for each:

Table 10.3 'Core' KBpedia typologies

Constituents	Natural phenomena	This typology includes natural phenomena and natural processes such as weather, weathering, erosion, fires, lightning, earthquakes, and tectonics. We explicitly include clouds and weather processes. Also, it covers climate cycles and general natural events (such as hurricanes) that are not specifically named. Biochemical processes and pathways are expressly excluded, occurring under a different typology.
	Area or region	The AreaRegion typology includes all nameable or definable areas or regions that we may find within 'space.' Though the distinction is not sharp, this typology is meant to be distinct from specific points of interest (POIs) that may be mapped (often displayed as a thumbtack). We may show areas or regions on a map as a polygon (area) or path (polyline).
	Location or place	The LocationPlace typology is for bounded and defined points in 'space,' which can be positioned via some form of coordinate system and we often show as points of interest (POIs) on a map. This typology is distinguished by areas or locations, which are often best displayed as polygons or polylines on a map.

(continued)

[12] Jack Park has questioned why chemistry appears in this schema, while physics and quantum phenomena do not. I agree those topics are worthy, likely under the Natural Matter node at the interface between Firstness and Secondness. Peirce does address these ideas a bit, and even posited something like the Big Bang (1888, CP 1.411–2). These fundamental perspectives on matter are an active area of research, though there are not many crumbs from Peirce on these topics. Still, as we learn more, I can readily see including such topics in the schema. As for the inclusion of chemistry and organic chemistry, we understand them better at present and they importantly demark the transition from natural matter to life. Chemistry is the laws or "habits" (Peirce's term) for how matter interacts and what products (compounds) may result, so is a natural Thirdness with respect to matter. Organic chemistry provides the building blocks or possible compounds or substrates to life, so is equivalent to a Firstness regarding organic matter and life.

Table 10.3 (continued)

	Shapes	The shapes typology captures all 1D, 2D, and 3D shapes, regular or irregular. Most shapes are geometrically describable things. Shapes have only a minor disjointedness role, with more than half of KKO reference concepts having some aspect of a shapes specification.
	Forms	This typology category includes all aspects of the shapes that objects take in space; forms is thus closely related to shapes. The forms typology is also the collection of natural cartographic features that occur on the surface of the earth or other planetary bodies, as well as the form shapes that naturally occurring matter may assume. Positive examples include mountain, ocean, and mesa. We exclude artificial features such as canals. Most instances of these natural features have a fixed location in space.
Time related	Events	These are nameable occasions, games, sports events, conferences, natural phenomena, natural disasters, wars, incidents, anniversaries, holidays, or notable moments or periods of time. Events have a finite duration, with a beginning and end. Individual events (such as wars, disasters, newsworthy occasions) may also have names.
	Times	This typology is for specific time or date or period (such as eras, or days, weeks, months type intervals) references in various formats.
Natural matter	Atoms and elements	The atoms and elements typology contains all known chemical elements and the constituents of atoms.
	Natural substances	The natural substances typology is minerals, compounds, chemicals, or physical objects that are not living matter, not the outcome of purposeful human effort, but are found naturally occurring. We also place other natural objects (such as rock and fossil) in this typology. Chemicals can be natural substances, but only if they are naturally occurring, such as limestone or salt.
	Chemistry	This typology covers chemical bonds, chemical composition groups, and the like. It excludes natural substances or living thing (organic) substances. Organic chemistry and biological processes are, by definition, separate typologies. The chemistry typology thus includes inorganic chemistry, physical chemistry, analytical chemistry, materials chemistry, nuclear chemistry, and theoretical chemistry.
Organic matter	Organic chemistry	The organic chemistry typology is for all chemistry involving carbon, including the biochemistry of living organisms and the materials chemistry (and polymers) of organic compounds such as fossil fuels.
	Biochemical processes	The biochemical processes typology is for all sequences of reactions and chemical pathways associated with living things.
Living things	Prokaryotes	The prokaryotes include all prokaryotic organisms, including the Monera, Archaebacteria, bacteria, and blue-green algae. Also included in this typology are viruses and prions.
	Protists and fungus	This typology is for the remaining cluster of eukaryotic organisms, explicitly including the fungus and the protista (protozoans and slime molds).

(continued)

Table 10.3 (continued)

	Plants	This typology includes all plant types and flora, including flowering plants, algae, nonflowering plants, gymnosperms, cycads, and plant parts and body types. Note that we also include all plant parts.
	Animals	This large typology includes all animal types, including specific animal types and vertebrates, invertebrates, insects, crustaceans, fish, reptiles, amphibia, birds, mammals, and animal body parts. We also include all animal parts. Also, we cover the groupings of such animals (such as herds, flocks). We include humans, as an animal, but exclude individual persons. We specifically exclude diseases. Animals have many of the similar overlaps to plants. However, there are more terms for animal groups, animal parts, and animal secretions, among others. Also, animals can include some human traits (posture, dead animal).
	Diseases	Diseases are atypical or unusual or unhealthy conditions for (mostly human) living things, generally known as conditions, disorders, infections, diseases, or syndromes. Diseases only affect living things and sometimes are caused by living things. This typology also includes impairments, disease vectors, wounds and injuries, and poisoning.
Agents	Persons	The appropriate typology for all named, individual human beings. This typology also includes the assignment of formal, honorific, or cultural titles given to specific human individuals. It further contains names given to humans who conduct particular jobs or activities (we know the latter as an avocation). Examples include steelworker, waitress, lawyer, plumber, and artisan. We specifically include ethnic groups. Note that we include persons as living animals under the animals typology.
	Organizations	Organization is a broad typology and includes formal collections of humans, sometimes by legal means, charter, agreement, or some mode of formal understanding. Examples of these organizations include geopolitical entities such as nations, municipalities, or countries, or companies, institutes, governments, universities, militaries, political parties, game groups, international organizations, trade associations, etc. All institutions, for example, are organizations. Also included are informal collections of humans. Informal or less defined groupings of humans may result from ethnicity or tribes or nationality or shared interests (such as social networks or mailing lists) or expertise (‘communities of practice’). This dimension also includes the notion of identifiable human groups with set members at any given point in time. Examples include music groups, cast members of a play, directors on a corporate board, TV show members, gangs, teams, mobs, juries, generations, minorities, etc.
	Geopolitical	Named places that have some informal or formal political (authorized) component. Notable subcollections include country, IndependentCountry, State_Geopolitical, City, and province.

(continued)

Table 10.3 (continued)

Artifacts	Products	The products typology includes any instance offered for sale or barter or performed as a commercial service. A product is often a physical object made by humans that is not a conceptual work or a facility (which have their own typologies), such as vehicles, cars, trains, aircraft, spaceships, ships, foods, beverages, clothes, drugs, and weapons.
	Food or drink	This typology is any edible substance grown, made or harvested by humans. The category also includes the concept of cuisines explicitly.
	Drugs	This typology is a drug, medication, or addictive substance, or a toxin or poison.
	Facilities	Facilities are physical places or buildings constructed by humans, such as schools, public institutions, markets, museums, amusement parks, worship places, stations, airports, ports, car stops, lines, railroads, roads, waterways, tunnels, bridges, parks, sports facilities, and monuments. All can be geospatially located. Facilities also include animal pens and enclosures and general human 'activity' areas (golf course, archeology sites, etc.). Importantly facilities include infrastructure systems such as roadways and physical networks. Facilities also include the components that go into making them (such as foundations, doors, windows, roofs). Facilities can also include natural structures that have been converted or used for human activities, such as occupied caves or agricultural facilities. Finally, facilities also include workplaces. Workplaces are areas of human activities, ranging from single-person workstations to large aggregations of people (but which are not formal political entities).
Information	Audio info	This typology is for any audio-only human work. Examples include live music performances, record albums, or radio shows or individual radio broadcasts.
	Visual info	The visual info typology is for any still image or picture or streaming video human work, with or without audio. Examples include graphics, pictures, movies, TV shows, individual shows from a TV show, etc.
	Written info	This typology includes any general material written by humans including books, blogs, articles, manuscripts, but any written information conveyed via text.
	Structured info	This information typology is for all kinds of structured information and datasets, including computer programs, databases, files, web pages, and structured data that can be presented in tabular form.
Social	Finance and economy	This typology pertains to all things financial and concerning the economy, including chartable company performance, stock index entities, money, local currencies, taxes, incomes, accounts and accounting, mortgages, and property.
	Society	This category includes concepts related to political systems, laws, rules, or cultural mores governing societal or community behavior, or doctrinal, faith, or religious bases or entities (such as gods, angels, totems) governing human spiritual matters. We include culture, issues, beliefs, and various activisms (most-isms).

Because Table 10.3 does not show all of the typologies, we collapse some of the hierarchical aggregations a bit. Note that the typologies that are not part of this 'core' listing also have complete descriptions within the online ontology files, as well as, of course, other specifications related to their roles in the knowledge graph.

Tailoring Your Own Typologies

The open-source nature of KBpedia is such that you may use as little or as much of this structure as you would like to build your own domain knowledge representations. The basic KKO structure, plus expansions or constrictions of existing KBpedia typologies, provides consistent scaffolding, with some promise of interoperability with external systems, for your knowledge efforts.

The quickest way to leverage KBpedia is to create and add your domain typologies. As needed, these may be large expansions of new detail and scope. Some areas may only require sporadic extensions or attention to the types already in KBpedia. I noted earlier the importance of addressing orphans and fragments as you build these typologies. You may need to create some new branches, including perhaps major ones, to capture the new domain scope. Once you are done revising KKO and its relevant typologies, you should turn your attention to integrating relevant instance data from local data stores or knowledge bases appropriate to the domain. Once fueled by instance data, including attributes and descriptive text, your knowledge system will be a valuable basis for knowledge supervision in machine learning. The outcomes of such learners can usefully aid many knowledge management tasks, importantly including tagging and categorization, and continued growth of your knowledge structures.

References

1. A. Marradi, Classification, typology, taxonomy. Qual. Quant. **24**(2), 129–157 (1990)
2. L. Wetzel, Types and tokens, in *The Stanford Encyclopedia of Philosophy*, ed. by E.N. Zalta (2014)
3. M. Hulswit, Natural classes and causation, in *The Online Digital Encyclopedia of, Charles S. Peirce* (2000)
4. N. Chinchor, Overview of MUC-7, in *MUC-7 Proceedings* (1997)
5. A. Brunstein, *Annotation Guidelines for Answer Types*. Linguistic Data Consortium (2002)
6. S. Sekine, Extended named entity ontology with attribute information, in *Proceedings of the Sixth International Language* (2008), pp. 52–57
7. X. Ling, D.S. Weld, Fine-grained entity recognition, in *Proceedings of the 26th AAAI Conference on Artificial Intelligence* (2012)
8. D. Gillick, N. Lazic, K. Ganchev, J. Kirchner, D. Huynh, Context-dependent fine-grained entity type tagging. *arXiv:1412.1820 [cs]* (2014)
9. S. Murty, P. Verga, L. Vilnis, A. McCallum, Finer grained entity typing with TypeNet. *arXiv:1711.05795 [cs]* (2017)

Chapter 11
Knowledge Graphs and Bases

Virtually everywhere one looks we are in the midst of a transition for how we organize and manage information, indeed even relationships. Social networks and online communities are changing how we live and interact. NoSQL and graph databases—married to their near cousin 'big data'—are changing how we organize and store information and data. Semantic technologies, backed by their ontologies and RDF data model, are showing the way for how we can connect and interoperate disparate information in ways only dreamed about a decade ago. Moreover, we are building all of this upon the infrastructure of the Internet and the Web, a global, distributed network of devices and information that is undoubtedly one of the most significant technological developments in human history.

The graph is a shared structure across all of these developments.[1] Graphs are the new universal paradigm for how we organize and manage information. Graphs have an inherently expandable nature and one which can also capture any existing structure. So, as we see all of the networks, connections, relationships, and links—both physical and informational—grow around us, it is useful to step back a bit and contemplate the universal graph structure at the core of these developments. Some form

[1] Some material in this chapter was drawn from the author's prior articles at the *AI3:::Adaptive Information* blog: "OWL Ontologies: When Machine Readable is Not Good Enough" (Mar 2006); "An Intrepid Guide to Ontologies" (May 2007); "Ontologies as the 'Engine' for Data-Driven Applications" (Jun 2009); "Confronting Misconceptions with Adaptive Ontologies" (Aug 2009); "Ontology-driven Applications Using Adaptive Ontologies" (Nov 2009); "When Linked Data Rules Fail" (Nov 2009); "An Executive Intro to Ontologies" (Aug 2010); "The Nature of Connectedness on the Web" (Nov 2010); "Ontology-Driven Apps Using Generic Applications" (Mar 2011); "The Age of the Graph" (Aug 2012); "Big Structure: At The Nexus of Knowledge Bases, the Semantic Web and Artificial Intelligence" (Jul 2014); "What is Big Structure?" (Aug 2014); "The Value of Connecting Things—Part I: A Foundation Based on the Network Effect" (Sep 2014); "The Value of Connecting Things—Part II: The Viking Algorithm" (Sep 2014); "The Value of Connecting Things—Part III: Ten Benefits from Big Structure" (Sep 2014); "'Deep Graphs': A New Framework for Network Analysis" (Apr 2016); "Uses and Control of Inferencing in Knowledge Graphs" (Mar 2017); "KBpedia Relations, Part I: Smarter Knowledge Graphs" (May 2017).

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8_11

of conceptual schema governs every knowledge structure used for knowledge representation (KR) or knowledge-based artificial intelligence (KBAI). In the semantic Web space, we call such schema 'ontologies.' Because the word ontology is a bit intimidating, a better variant is the *knowledge graph* (because all semantic ontologies take the structural form of a graph). In our knowledge representation efforts, we tend to use the terms *ontology* and *knowledge graph* interchangeably.

What an ontology—or knowledge graph—means is dependent on the context and purpose. In the case of an *upper ontology* and *typologies*, we see the conceptual scaffolding. In the relation of *attributes* to *instances*, we see the intensional aspects of the graph and the basis for data records. Relations between nodes, different than those of a hierarchical or subsumptive nature, provide still different structural connections across the knowledge graph. Indeed, one can and should organize the types of *types* in a knowledge graph to better modularize it and segregate similar purposes and functions. We design some ontologies to capture the scope of particular knowledge domains, while others we may use for administrative purposes or in support of user interfaces. We discuss all of these aspects in this chapter, plus what is desirable in knowledge bases and how to use them to populate these knowledge structures.

Graphs and Connectivity

Graphs, as conceptual or analytical structures, are relatively new. The explication of graph theory only began about 300 years ago. The use of graphs for expressing logic structures only began about 100 years ago, with its intellectual roots, in fact, arising from Charles Peirce and his existential graphs. Though likely trade routes and primitive transportation or nomadic infrastructures were perhaps the first expressions of physical networks, the emergence and then prevalence of networks are also fairly recent phenomena. Transportation, communications, and electrical grid were the first purpose-built physical networks. The Internet and the Web are surely the catalyzing development that has brought graphs and networks to the forefront.

In mathematics, a graph is an abstract representation of a set of objects where pairs of the objects are connected. We term these objects *nodes* or *vertices*; we call the connections between the objects *edges*. Typically, we depict a graph in diagrammatic form as a set of dots or bubbles for the nodes, joined by lines or curves for the edges. If we define a logical relationship between connected nodes we call the graph 'directed.' We can express various structures or topologies through this conceptual graph framework. Graphs are one of the focuses of study in discrete mathematics.[2] The word 'graph' was first used in a mathematical sense by J.J. Sylvester in 1878 [1].

[2] Topics in discrete mathematics, which are all applicable to graphing techniques and theory, include theoretical computer science, information theory, logic, set theory, combinatorics, probability, number theory, algebra, geometry, topology, discrete calculus or discrete analysis, operations research, game theory, decision theory, utility theory, social choice theory, and all discrete analogues of continuous mathematics.

Graphs are modular and can be both readily combined and broken apart. From a computational standpoint, this can lend itself to parallelized information processing (and, therefore, scalability). If we represent the graph in RDF, graph extractions are themselves valid models. Graphs have some unique strengths for search and pattern matching. Besides options like finding paths between two nodes, depth-first search, breadth-first search, or finding shortest paths, emerging graph and pattern-matching approaches may offer entirely new paradigms for search. Graphs also provide new methods for visualization and navigation, useful for both seeing relationships and framing information from the local to global contexts. The interconnectedness of the graph allows us to explore data via contextual facets, which is revolutionizing data understanding in a way similar to how the basic hyperlink between documents on the Web changed the contours of our information spaces.

Graph algorithms are a significant field of interest within mathematics, computer science, and social sciences. Via approaches such as network theory or scale-free networks, we can analyze and model topics such as relatedness, centrality, importance, influence, 'hubs' and 'domains,' link analysis, spread, diffusion, and other dynamics. Many would argue, as do I, that graphs are the most 'natural' data structure for capturing the relationships of the real world. If so, we should continue to see new algorithms and approaches emerge based on graphs to help us better understand our information. RDF is a natural data model for such purposes.

Graph Theory

Graph theory is the manipulation and analysis of graph structures. The first paper in that field is the *Seven Bridges of Königsberg*, written by Leonhard Euler in 1736. The objective of the article was to find a walking path through the city that would cross each bridge once and only once. Euler proved that the problem had no solution.[3] Later, Cayley broadened the approach to study tree structures, which have many implications in theoretical chemistry. By the twentieth century, the fusion of ideas coming from mathematics with those coming from chemistry formed the origin of much of the standard terminology of graph theory.

Graph theory forms the core of network science, the applied study of graph structures and networks. Besides graph theory, the field draws on methods including statistical mechanics from physics, data mining and information visualization from computer science, inferential modeling from statistics, and social structure from sociology. Classical problems embraced by this realm include the four-color problem of maps, the traveling salesman problem, and the six degrees of Kevin Bacon. Graph theory and network science are the suitable disciplines for a variety of information structures and many additional classes of problems. Graphs are among the

[3] The generalized understanding is that in any connected graph, only zero or two nodes may have odd numbers of connections to traverse the entire graph only once per path (edge); the Königsberg example has four nodes with odd numbers, and thus fails Euler's test.

most ubiquitous models of both natural and human-made structures. They can be used to model many types of relations and process dynamics in physical, biological, and social systems. Graphs can represent many problems of practical interest. This breadth of applicability makes network science and graph theory two of the most critical analytical areas for study and breakthroughs for the foreseeable future.

Graphs and graph theory also have broad applicability to natural systems. For instance, researchers use graph theory extensively to study molecular structures in chemistry and physics. A graph makes a natural model for a molecule, where vertices represent atoms and edges bonds. Similarly, in biology or ecology, researchers employ graphs to express such systems as species networks, ecological relationships, migration paths, or spread of diseases. Graphs are also proper structures for modeling biological and chemical pathways. Some of the exemplary natural systems that lend themselves to graph structures include:

- Chemical reaction networks
- Gene regulatory networks
- Spin networks
- Neural networks
- Ecological networks
- Petri nets (chemistry).

The growth of social networks has paralleled the growth of the Internet and Web. Social network analysis (SNA) has arguably been the most critical driver for advances in graph theory and analysis algorithms in recent years. We are now elucidating new and interesting problems and challenges—from influence to communities to conflicts—through techniques pioneered for SNA. The suitability of the graph structure to capture relationships has been a real boon to a better understanding of social and community dynamics. SNA has introduced many new concepts, including such things as influence, diversity, centrality, and cliques. Particular areas of social interaction that lend themselves to SNA include:

- Social networks
- Military conflicts and terrorism
- Value networks
- Project networks
- Workflows
- Business ecosystems.

We have unearthed entirely new insights using SNA including finding terrorist leaders, analyzing prestige, or identifying keystone vendors or suppliers in business ecosystems. Real networks, in comparison to random networks, are both modular and hierarchical, distributed over a sparse topology [2].

What these examples show is the nearly universal applicability of graphs, from the abstract to the physical and gradations from the small to the large. We also see how to build upon basic graph structures and concepts with more structure. This breadth points to the many synergies and innovations that may be transferred from diverse fields to advance the usefulness of graph theories. Still, despite the advances

that have occurred in graph theory, and the increased attention from social network analysis, many graph problems remain some of the hardest in computation. Optimizations, partitioning, mapping, inferencing, traversing, and graph structure comparisons remain challenging. Some of these challenges are only growing due to the growth in the size of networks and graphs.

Given the ubiquity of graphs as representations of real systems and networks, it is not surprising to see their use in computer science as a means for information representation. It is notable that we may represent virtually any data structure as a graph, but the paradigm has even broader applicability. The critical breakthroughs have come through using the graph as a basis for data models and logic models. These, in turn, provide the bases for crafting entire graph-based vocabularies and languages. Once we embrace such structures, it is also natural to extend the mindset to graph databases as well.

The Value of Connecting Information

The hackneyed phrase of 'connect the dots' reflects our basic intuition of the value in making connections among relevant data. However, what is this value? How might we quantify it? The reason it is useful to try to quantify the value of connected information is that such an estimate helps to define what effort or cost we can justify building our connected knowledge structures. For most big data projects, for example, we already know that 50–80% of the costs in assembling relevant datasets are due to data wrangling—the effort to extract, transform, and clean the input data.[4] Nowhere, however, do we know what it is worth to go to the next step of working to connect those data.

The 'network effect' was first realized in the early days of telephone networks, where the value of the system increased as a function of more users.[5] We have also long recognized a similar effect in connecting information and the breaking down of information or 'data silos.' This emergence of structure is particularly evident in physical networks, such as the growth of a telecommunications network. Two telephones can make only one connection, 5 can make 10 connections, 12 can make 66 connections, etc. It is this very multiplier effect that has led to most of the thinking of how to quantify the network effect.

[4] "Data scientists, according to interviews and expert estimates, spend from 50 to 80% of their time mired in this more mundane labor of collecting and preparing unruly digital data, before it can be explored for useful nuggets," is a quote from Steve Lohr, 2014, "For Big-Data Scientists, 'Janitor Work' Is Key Hurdle to Insights," August 17, 2014, New York Times, see http://www.nytimes.com/2014/08/18/technology/for-big-data-scientists-hurdle-to-insights-is-janitor-work.html. Also, as another example of the common 80% estimate for data preparation costs, see http://radar.oreilly.com/2013/09/data-analysis-just-one-component-of-the-data-science-workflow.html

[5] These benefits of the network effect were reportedly a major driver of Theodore Newton Vail's efforts to consolidate the thousands of initial telephone networks in the United States under the banner of the American Telephone & Telegraph (Ma Bell) company.

The earliest effort to estimate the value of physical networks was Sarnoff's law, developed by David Sarnoff, for many years the leader of the Radio Corporation of America (RCA). He posited that the value of a broadcast network was directly proportional to its number of viewers (n). However, the problem with this formulation is that a broadcast network is only one way, from broadcaster to user. What of networks with interactions or two-way linkages? The benefits of such networks must surely be more than linear.

Once we get into interaction effects, we get into multipliers. The nature of those multipliers comes from the extent of real interactions, as well as perhaps the nature of the network itself. Metcalfe's law was the first direct derivation from the telecommunications model. Robert Metcalfe formulated it about 1980 in relation to Ethernet and fax machines. The 'law' was then named for Metcalfe and popularized by George Gilder in 1993 [3]. The actual algorithm proposed by Metcalfe calculated the number of unique connections in a network with n nodes as $n(n-1)/2$. This formulation makes Metcalfe's law a quadratic growth equation. We may simplify the law[6] to state that the value of a telecommunications network is proportional to the square of the number of users of the system (n^2). Gilder's popularization and the early growth of the Internet made estimating the benefits of network effects a very timely topic. As a value measure, we can use the network effect to estimate the benefits for increasing numbers of users. Some have even blamed Metcalfe's law for contributing to the creation (and then bursting) of the 'dot-com bubble' of the late 1990s [4].

However, the Metcalfe formulation is not universally accepted, and others have proposed different estimates. From the perspective of social groups, Reed came up with the largest multiplier formulation premised on arbitrary-sized groups forming among any and all participants (nodes) [5]. On the other hand, under the provocative title, "Metcalfe's Law is Wrong," Briscoe, Odlyzko, and Tilly (BOT) challenged both the Metcalfe and Reed approaches in 2006 [6]. Using the proxy of Internet valuation, the authors were able to show how absurd the implications of either approach were at scale. Like the bet of rice (or wheat) doubling each of the 64 squares on a chessboard bankrupting the kingdom, we can see the exponential implications of these two 'laws' to (eventually) violate common sense. The fundamental fallacy claimed by the authors for both the Metcalfe and Reed approaches is that all potential links are of equal value. There must be some law of diminishing returns to slow the unsustainable rates of exponential or (to a lesser extent) quadratic growth. After much hand waving, the authors chose Zipf's law[7] as their basis for this diminishing return. To approximate this distribution, they (BOT) offered the

[6] For a well-connected network, every node (n) connects to every other node ($n-1$), which gives us $n\times(n-1)$ or ($n2-n$). Working this out, two nodes have two connections ($2\times2-2$), three nodes have six connections ($3\times3-3$), and the expression converges on the square of 'n' for larger values of 'n,' e.g., ($100\times100-100$) is 99% of (100×100). This convergence at larger number is the basis for the exponential simplification, $2n$. Most of the other 'laws' stated herein are simplifications in a similar manner.

[7] See, for example, https://www.princeton.edu/~achaney/tmve/wiki100k/docs/Zipf_s_law.html

simple *n log (n)* formulation of Zipf's law. This approximation is reasonable, but one that is never related directly to the real nature of graphs or networks.

Yaakov Stein, a network and signal processing researcher of the first rank, used his experience when joining LinkedIn to help understand and quantify connections in real networks [7]. He began without a LinkedIn account and documented his experience as he joined and expanded his network of contacts on the service. He charted direct links, and then meticulously looked at and recorded secondary and tertiary links. His formulation recognized that the value to an individual user equaled raising the access to the entire network (*1*) for that user plus the diminishing benefit represented by the participating graph's other participants as measured by the average degree of separation (*D*). *D* is an inherent measure of the graph type.

Though his context was a social network, the insight is that relations diminish by distance within a graph, with average link distance (directly related to the degree of separation) a relevant metric. Connected 'facts' or 'friends' is essentially the same thing. It is all about what we share among graph nodes. Stein's approach grounds the multiplier effect in an inherent characteristic of the graph: its average degree of separation. Like Zipf's law, the degree of separation is a distance measure, but one now based on the real nature of graphs. Here is the Stein formulation:

$$V = n^{\left(1+\frac{1}{D}\right)}$$

where *V* is the potential value, *n* is the number of graph nodes, and *D* is the graph's average degree of separation. Thus, a graph with a degree of separation of 4 would exhibit a network-wide power factor of 5/4 (4/4 plus 1/4).

I modified Stein's approach to calculate the value of knowledge graph formulation, or the VKG (Viking) algorithm, using this expression:

$$V = F * n^{\left(1+\frac{1}{D}\right)}$$

where *V* is the potential value, *F* is the average assertion accuracy, *n* is the number of graph nodes, and *D* is the graph's average degree of separation. *F* is analogous to F-measure, the combined precision and recall statistic for information retrieval and NLP tasks (see further Chap. 14). *F* in the case of the Viking algorithm is also a combined statistic that represents the 'accuracy' (verifiable truthfulness) of statements asserted in the graph.

F is essentially an estimated value for one minus the residual falsity for the average statement in a graph, after removal of all assertions that do not meet the existing coherency, consistency, or completeness tests. Sampling statements across the graph determine *F* and manually testing for truthfulness (or in a logical sense, validity for the existing statements in the graph). An *F* of 1 signifies complete truthfulness (accuracy); an *F* of 0 represents absolute falsity.

Now corrected with our assumed *F* factor, we can begin to tease out the value benefits of connecting 'facts' versus the unconnected 'facts.' As with any logarith-

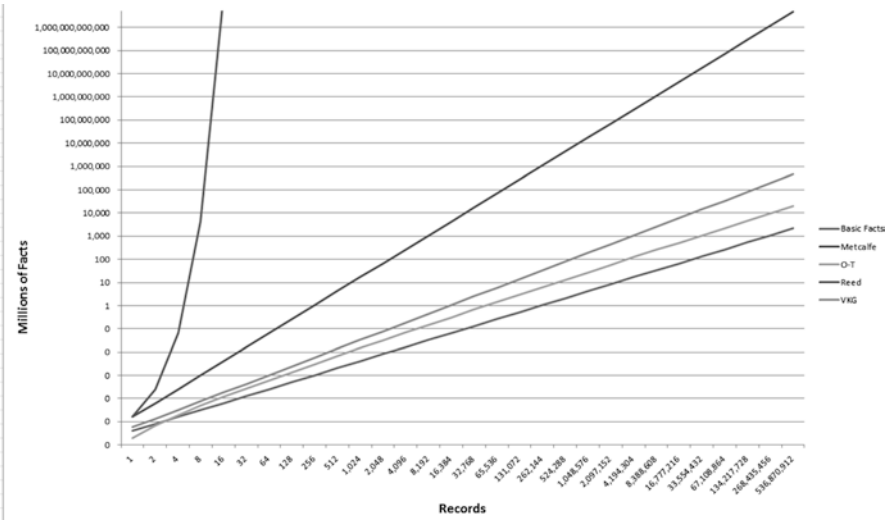


Fig. 11.1 Comparison of network multipliers

mic function, we see that the benefit value from connections increases in a growing manner at larger scales. For example, at a level of 1000 records, the benefits from connections are 7× greater than unconnected data. By the time the scale grows to 1 million or 500 million records, the benefit of connections increases to 44× to 215×, respectively. However, the potential value of connectedness is also a function of the general degree of information separation for the given domain.

I consolidate these various estimates of connected value in Fig. 11.1. At the nominal scales of 100,000 and 1,000,000 records, the value of data connections in comparison to the unconnected 'facts' case can show huge increases. Based on empirical experience to date, I think we can say that the benefits of connecting previously unconnected data may fall somewhere within the limits of Fig. 11.1. Even at rather low scales and more loosely connected domains, the value improvements in making connections with data are manifold. At larger scales for tighter networks, the multipliers can become astounding.

We are still in the early phases of gathering statistics for such things, but, in general, most any knowledge graph would have a *D* factor ranging from 2 to 8, as I document in Table 11.1. We also should assume that network effects are not linear. We should expect a leveling or flattening of the curve; the benefits of connections are not limitless. The shape of the curve likely varies by domain and the nature of the network. It is a topic worth studying.

Table 11.1 Increased value for connecting nodes, various networks

Domain	100,000 Records	1,000,000 Records
Food webs	203×	611×
Genetic differences	38×	84×
Twitter	23×	46×
Facebook	17×	33×
Potential research collaborators	14×	26×
UMBEL	8×	12×
Social networks (general)	5×	8×
Mobile ad hoc networks	3×	5×

Another implication the Viking algorithm allows us to test is the benefit of adding structure to our datasets. Actually, 'adding structure' is not strictly correct; it is 'structurizing' the data via characterizations, attributes, and categorizations. Of course, mere connections for structure's sake are silly. Not all structure is created equal. From a KR perspective, typing is the most important, and individual instance annotations are the least. Assigning or classifying our records into types, for example, applies to all records across the datasets and provides powerful cross-record linkages. Adding annotations or metadata to single records provides much lower benefits. Each across-dataset structure characterization adds about 25–30% value per structure. Adding four structural characterizations, for example, more than doubles the 'facts' assertion value (~140%) to the datasets. The good thing is that we can add such structure as a slight increase over standard data-wrangling efforts, and with more impact than standard wrangling.

The graph structures, preferably guided by domain ontologies, provide the logic means to test for subsequent structure additions. Not only does adding structure get easier with a foundation of existing structure, but it also increases the value of the information by orders of magnitude. At this stage, what the Viking algorithm gives us is a defensible means for assessing the value of adding structure (through connections) to our datasets. We see potentially huge multiplier effects that compound further benefits with scale (subject to the leveling curve caveat). We also see that the most developed forms of structure—namely, ontologies—bring further benefits in inference and testable coherence.[8]

While our current proxy for value—namely, asserted 'facts'—is useful, a perhaps more useful one would be 'fact' assertions with a monetary value. Such estimates will show, again, that not all 'facts' are created equal, and some have more monetary value than others. Transitioning our estimates of value to a monetary basis will help set parameters for the cost-benefit analysis of data collection and structuring that is the ultimate basis for planning such KR initiatives.

[8] Ontologies also provide the means for tagging (providing structure) to unstructured documents, which also brings multiplier benefits from structure. On the retrieval side, such structure also aids faceting and filtered "slicing and dicing" of underlying datasets, thereby improving retrieval efficacy.

As we look at Table 11.1 and play with some parameters, we can see that some guidelines emerge. First, more structure always provides benefits—adding structure provides a multiplier effect in value. Second, more connections are more valuable as a multiplier effect than adding more data, which has an additive effect. Third, the benefits of structure increase with increasing dataset sizes (scale). Fourth, particular kinds of structure, such as types or categorizations, enable cross-dataset selections and comparisons that are inherently more valuable than record-specific annotations. Fifth, by adding correct and coherent connections, it may be possible to move the entire graph to a lower average degree of separation (*D*), with further multiplier benefits. Sixth, structure can be added incrementally and appears cumulative to some level. Seventh, we should not view data wrangling as an overall 'cost' to the effort but as a means for achieving the multiplier benefits arising from structure and connections.

Graphs as Knowledge Representations

Graphs are an iconic and intuitive way to visualize and express connections. Graphs, expressed in mathematical or logical form, are a rich substrate for analysis and reasoning. Graphs appear to be the natural structure for capturing real relationships in the material world and the conceptual realm.

One key aspect of graphs is their inherent extensibility. Once we understand graphs as an excellent way to represent both logic and data structures, their usefulness to knowledge representations becomes clear. Graph-theoretic methods are particularly useful in linguistics since natural language has a discrete, connected structure. Not only can graphs represent the syntactic and compositional structure, but they can also capture the interrelationships of terms and concepts within those languages (that is, the *semantics*). We see the usefulness of graph theory to linguistics by the various knowledge bases such as WordNet (in multiple languages) and VerbNet. Domain ontologies emphasize conceptual relationships over lexicographic ones for a given knowledge domain. Semantic networks and neural networks are similar knowledge representations.

The main reasoning in the knowledge graph relies on its hierarchical, hyponymous relations and instance types. These establish the parent-child lineages and enable us to relate individuals (or instances, which might be entities or events) to their natural kinds, or types. Entities belong to types that share specific defining essences and shared descriptive attributes. For effective inferencing, it is wise to try to classify entities into the most natural kinds possible. Clean classing into appropriate types is one way to realize the benefits from related search and related querying. Types may also have parental types in a hyponymous relation. This 'accordion-like' design, discussed in the prior Chap. 10, is an important aspect that enables us to tie external schema to multiple points in KBpedia.

Disjointedness assertions, where two classes are logically distinct, and other relatedness options provide other powerful bases for winnowing potential candidates in a graph and testing placements and assignments. Each of these factors

also may be used in SPARQL queries. These constructs of semantic Web standards, combined with a properly constructed knowledge graph and the use of synonymous and related vocabularies in *semsets*, provide potent mechanisms for how to query a knowledge base. By using these techniques, we may dial-in or broaden our queries, much in the same way that we choose different types of sprays for our garden-watering hose. We can focus our queries to the particular need at hand.

Once a completed graph passes its logic tests during construction, perhaps importantly after being expanded for the given domain coverage, its principal use is as a read-only knowledge structure for making subset selections or querying. The standard SPARQL query language, which we occasionally supplement with rule-based queries using SWRL or for bulk actions using the OWL API, is the means by which we access the knowledge graph in real time. In many instances, such as for the KBpedia knowledge graph, these are patterned queries. In such cases, we substitute variables in the queries and pass those from the HTML to query templates. When doing machine learning, we often retrieve slices via query and then stage them for the learner. We may generate entity lists for things like training recognizers and taggers. Some of the actions may also do graph traversals to retrieve the appropriate subset. However, the primary real-time use of the knowledge structure is search.

Among many other options, SPARQL also gives us the ability to query specific property paths [8]. We can invoke these options either in our query templates or programmatically. We may programmatically broaden or narrow our searches of the graph, depending on the relation chosen (`subClassOf` is one example) and the length of the specified property path. Switching inferencing on or off also acts to broaden or narrow the search swath considerably. Besides all of the standard query options provided by the SPARQL standard, we may also remove duplicates, identify negated items, and search inverses, select named graphs, or select graph patterns. Beyond SPARQL and now using SWRL, we may also apply abductive reasoning and hypothesis generation to our graphs, as well as mimic the action of expert systems in AI through if-then rule constructs based on any structure within the knowledge graph. A helpful online tutorial with examples helps highlight some of the possibilities in combining OWL 2 with SWRL [9].

Upper, Domain, and Administrative Ontologies

The root of the *ontology* term is the Greek *ontos*, or *being* or the *nature of things*. Classical philosophers used the term ontology for the study of the nature of being or the world, the nature of existence. Tom Gruber, among others, made the term popular in computer science and artificial intelligence about 15 years ago when he defined ontology as a "formal specification of a conceptualization." Since then, I have continued to find ontology one of the harder concepts to communicate to clients and quite a muddled mess even as used by some practitioners. I have concluded that this problem is not because I have failed to grasp some ephemeral nuance, but because the 'ontology' term as used in practice is indeed fuzzy and imprecise.

A Lay Introduction to Ontologies

Ontologies are the structural frameworks for organizing information on the semantic Web and within semantic organizations.[9] Ontologies have the structural form of a graph; we often use *knowledge graph* synonymously. Ontologies provide unique benefits in discovery, flexible access, and information integration due to their inherent connectedness. We can layer ontologies on top of existing information assets, which means they are an enhancement and not a displacement for prior investments. Moreover, ontologies may be developed and matured incrementally, which means their adoption may be cost effective as benefits become evident.

Ontologies provide an organizing context for relating disparate information together and for making meaningful inferences. The framework itself is a function of the worldview, context, and domain scope at hand. Flexibility here is not weakness; it is the power to capture the meaningful vocabulary and discourse for entire domains of knowledge. The trick to designing a proper ontology is to maintain internal coherence and self-consistency while capturing the vocabulary and discourse of its stakeholders and users. When done, it is then possible to relate disparate information and data to other data and to make intelligent business inferences. So, the use of an ontology does not limit freedom. It does set the context for making connections and setting relations. As long as it is coherent, the 'correct' ontology is the one that best captures the scope and domain at hand, and is one that is continually responding to the open nature of knowledge and its community of users.

When I refer to the idea of 'worldview' as synonymous with an ontology, I do not mean that as cosmic, but how we may convey a given domain or problem area. One group might choose to describe and organize, say, automobiles, by color; another might choose body styles such as pickups or sedans; or, still, another might use brands such as Honda and Ford. None of these views is inherently 'right' (indeed multiples might be combined in a given ontology), but each represents a particular way—a 'worldview'—of looking at the domain. So long as all ascertainable 'facts' in an ontology may be confirmed and its logic kept consistent, different 'worldviews' are perfectly acceptable.

Understanding, using, and manipulating ontologies can bring practical benefits:

- Ontologies help make explicit the scope, definition, language, and meaning (*semantics*) of a given knowledge domain or worldview.
- Ontologies may represent any form of *unstructured* (documents or text), *semi-structured* (XML or Web pages) or *structured* (database) data.
- Ontologies provide a coherent navigation and search mechanism for moving through disparate information spaces, with any *node* or *edge* providing a possible entry point.
- Ontologies, if hierarchically structured in part, enable the power of inheritance, reasoning, and inference.

[9] I personally prefer an embracing understanding of the term, consistent with Deborah McGuinness's 2003 paper, *Ontologies Come of* Age [10].

- Ontologies may provide the power to generalize and hypothesize (*abductive reasoning*) about their domains.
- Ontologies provide guidance on how to correctly 'place' information in that domain, useful for external concept matching and mapping.
- Ontologies can provide a more effective basis for information extraction or content clustering.
- Ontologies may be queried and filtered to provide pre-qualified corpora and training sets, useful for *unsupervised* and *supervised machine learning*, respectively.
- Ontologies may be a source of structure and controlled vocabularies helpful to disambiguate context and to inform domain 'lexicons.'
- Ontologies can help relate and 'place' other ontologies or worldviews to one another; in other words, ontologies can help organize ontologies.

The most prevalent use of ontologies at present is in semantic search. Semantic search has benefits over conventional search by being able to make inferences and matches not available to standard keyword retrieval. Perhaps a pinnacle application for ontologies is to help map and integrate other structures and information, both within and without the organization. Furthermore, if we populate a knowledge graph sufficiently with accurate instance data, often from various knowledge bases, then ontologies can also be the guiding structures for efficient machine learning and artificial intelligence.

Ontologies Are a Family of Graphs

If you pose the query 'ontology filetype:owl' to Google, you will see more than 10,000 results. According to Ontolog Forum, a community of ontology practitioners, we can classify ontologies by some key measures. *Expressiveness* is the extent and ease by which an ontology can describe domain semantics. *Structure* they define as the degree of organization or hierarchical extent of the ontology. They further define *granularity* as the level of detail in the ontology. By these notions, we may include the concepts of *folksonomies* and *topic maps* in the definition. The Forum also defines other dimensions of use, logical basis, and purposes for ontologies [11]. One of these dimensions is to characterize ontologies by 'levels,' specifically *upper*, *middle*, and *lower* levels. These are useful distinctions, but we prefer to classify them into *upper*, *domain*, and *administrative* ontologies.

Upper ontologies provide the top-level conceptual structure and schema, which often function as the reference structure for specific domain ontologies. Examples of upper-level ontologies include the Suggested Upper Merged Ontology (SUMO), the Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE), PROTON, Cyc, and BFO (Basic Formal Ontology). Most of the content in these upper levels is akin to broad, abstract relations or concepts. Most all of them have both a hierarchical and networked structure, though their actual subject structure

relating to concrete things is pretty weak. KBpedia's Knowledge Ontology (KKO) is an example of an upper ontology.

Domain (or content) *ontologies* embody more of the traditional ontology functions such as information interoperability, inferencing, reasoning, and conceptual and knowledge capture of the applicable domain. We can broadly or narrowly define these domains; specific instantiations may cover multiple or a diverse range of subject matter. In KBpedia's design, we compose the domain ontology of multiple typologies, including for relations and attributes.

Administrative ontologies govern internal application use and user interfaces. These areas might relate to providing metadata as a result of workflow steps and general workflow management, as well as driving visualization or display widgets or informing user interfaces. Possible user interface aids provided by administrative ontologies may include attribute labels and tooltips; navigation and browsing structures and trees; menu structures; auto-completion of entered data; contextual drop-down list choices; spell checkers; and online help systems. Administrative ontologies may also support internal applications such as workflow systems, access control, archive management, and the like.

Incipient Potentials

For over 20 years, some researchers such as Nicola Guarino (1998) [12] and Michael Uschold (2008) [13] have argued that we could rely upon ontologies for even more central aspects of overall applications, what Uschold termed 'ontology-driven information systems.' I agree. Here are five areas of (largely) untapped potential:

1. Lack of a well-known *relations ontology*: Structurally, we may use OWL to reason over actions and relations in a similar means as we reason over entities and types, but our common ontologies have yet to do so. Creating such schema is within grasp since we have language structures such as VerbNet and other resources we could put to the task. KBpedia has its own relations typologies that attempt to capture these aspects.
2. Lack of a well-known *attributes ontology*: The lack of a schema and organized presentation of attributes means it is challenging to do *ABox*-level integration and interoperability. This gap is largely due to the primary focus on concepts and entities in the early stages of semantic technologies. As the KBpedia knowledge graph shows, it is possible to formulate logical and reusable schema for instance attributes as well.
3. A *quantity units ontology* is the next step beyond attributes, as we attempt to bring data values for quantities (as well as the units and labeling used) into alignment. The QUDT ontologies (quantities, units, and data types), or something similar, may provide such a template.
4. A *statistics and probabilities ontology* is also appropriate given the idea of continua (Thirdness) from Peirce and capturing the idea of *fallibility*. Probabilistic

reasoning is still a young field in ontology. Some early possibilities include Costa [14] and the PR-OWL ontology using Multi-Entity Bayesian Networks [15], probabilistic first-order logic that goes beyond Peirce's classic deterministic logic, as well as fuzzy logic applied to ontologies [16].

5. *ODapps* ('ontology-driven applications') are generic software packages driven by ontology specifications for specific applications. They may enable us to (1) import or export datasets; (2) create, update, delete (CRUD), or otherwise manage data records; (3) search records with full-text and faceted search; (4) manage access control at the interacting levels of users, datasets, tools, and CRUD rights; (5) browse or view existing records or record sets, based on simple to possible complex selection or filtering criteria; or (6) process result sets through workflows of various natures, involving specialized analysis, information extraction, or other functions.

Realizing these potentials will enable our knowledge management (KM) efforts to shift to the description, nature, and relationships of the information environment. Under this broadened understanding, we now give explicit focus to the actual concepts, terminology, and relations that comprise coherent ontologies, subject to the direct control and refinement by their users, the knowledge workers, and subject matter experts.

Good Ontology Design and Construction

While Chap. 14 focuses on best practices and includes a section on ontologies, it is worthwhile here to reiterate three design considerations that should go into the construction of an ontology. These three factors are *coherence*, *completeness*, and *scope*, introduced in prior chapters.

Coherence is a state of logical, consistent connections, a logical framework for intelligently integrating diverse elements. In the sense of a knowledge graph, this means that we have drawn the right connections (edges or predicates) between the object nodes in the graph. Structure without coherence is where we have not drawn correct or complete connections. The nature of the content graph lacks logic. The hip bone is not connected to the thigh bone, but perhaps to something wrong or ludicrous, like the arm or cheekbone.

Completeness is to conform to some minimum standard of characterization. For KBpedia, we have set that minimum as a preferred label, robust set of alternative labels (*semset*), a definition, language characterization, and one or more types or parents. If we have information on attributes, we should include that as well. However, it is not necessary to discover and document all attributes, though we should add new ones as we encounter them. See further what we discussed for completeness for reference concepts (RCs) in Chap. 9.

S*cope* means we answer a series of questions in the positive for the ontology:

- Does the ontology provide balanced coverage of the subject domain?[10] This question gets at the issue of properly scoping and bounding the subject coverage such that the breadth and depth are roughly equivalent.
- Does the ontology embed its domain coverage into a proper context? Reusing existing and well-accepted vocabularies and including concepts in the subject ontology that aid such connections is good practice.
- Are the relationships in the ontology coherent, per our earlier condition?
- Has the ontology been constructed according to good practice?

If we can answer these questions affirmatively—including importantly the use of testing scripts throughout construction—then we deem the ontology ready for production-grade use.

The skills needed to create these ontologies are logic, coherent thinking, and domain knowledge. That is, any subject matter expert or knowledge worker likely has the skills required to contribute to useful ontology development and refinement. Ontology development is a trainable skill.

KBpedia's Knowledge Bases

We want knowledge sources, putatively *knowledge bases*, to contribute the actual instance data to populate our ontology graph structures. Matching with knowledge bases can also point out gaps and oversights in our knowledge graphs that we should augment to provide better domain coverage. Sufficient instance data is an absolute essential if we are to use our knowledge structures for *supervised* or *unsupervised machine learning*, or what we call herein *KBAI*.

We want knowledge bases to define and populate attributes for their instances. This kind of information is what we see in a data record. The best knowledge bases have large data stores, all consistently characterized. We prefer large sources because we can spread the effort of mapping and conversion across more records.

We prefer knowledge bases that provide identity and information for disambiguation. Identity works in that we can point to authoritative references (with associated Web identifiers) for all of the individual things and properties in our relevant domain. We can use these identities to decide the 'canonical' form, which also gives us a common reference for referring to the same things across information sources or datasets. We also want richness in how to describe those things.

Besides our earlier criteria of consistency, coherence, and completeness, our desiderata for what we find useful in a knowledge base include the following:

[10] The sense of 'balance' is from the perspective of the sponsor, roughly bounded by the topic domain at hand. Work is always required to bring the knowledge graph up to this level of coverage. This sense is different for a library, where 'balance' is from the perspective of the patrons.

- *Comprehensive*—Does the knowledge base support the domain scope at hand? Smaller, focused knowledge bases may be quite valuable if the overlap is good.
- *Referenceable*—Is the knowledge source authoritative? Does it use IRIs or URIs for referencing its objects?
- *Open Standards*—Does it meet open standards? It is often easier to interoperate with open standards with more tooling available.
- *Computable*—Does the KB support reasoning, inference, set selection, relations, attributes, data types, and retrieval? If so, incorporation is easier.
- *Multilingual*—If not already multilingual, does it have a structure (such as ID *v* label based) that supports multiple languages? Support for multiple languages increases usefulness and applicability.

The idea that we can purposefully craft knowledge bases to support knowledge-based artificial intelligence, or KBAI, flows from these kinds of realizations. We begin to see that we can tease out different aspects of a knowledge base, each with its logic and relation to the other aspects.

KBpedia KBs

As of 2018, about 20 different knowledge bases contribute the instance data and some key mappings to KBpedia. Six of these are primary ones, defined as adding large numbers of instances but scope coverage to KBpedia as well. We have selected the secondary KBs based on their common usage or their ability to contribute more limited concepts and structure to the overall KBpedia.

Primary KBs

Wikipedia, the primary source for structure, concepts, and definitions, and Wikidata, the primary source for millions of instance data and a rich system of attributes, are the two most significant contributors to KBpedia. We use DBpedia as a source for direct machine-readable Wikipedia data. While we first root the conceptual schema of KBpedia in Peirce's universal categories, we use the OpenCyc and UMBEL knowledge bases to inform the construction of KBpedia's typologies. We extend KBpedia's geographical and geopolitical reach using the GeoNames knowledge base. Here is a bit longer description of each source, current as of mid-2018:

- *Wikipedia* is a crowdsourced, free-access and free-content knowledge base of human knowledge. It has more than five million articles in its English version. Nearly 35 million articles exist across all Wikipedias in about 280 different languages. Though not universal, most all recent AI advances leveraging knowledge bases have utilized Wikipedia in one way or another, due to its scope, quality, and open-access structure. Wikipedia is a common denominator in question

answering and commercial natural language applications that leverage artificial intelligence, and witness Siri, Watson, Cortana, and Google Now, among others. Even Freebase, the core of Google's Knowledge Graph, did not blossom as a separate data crowdsourcing concern until its former owner, Metaweb, decided to bring Wikipedia into its system. More than 1000 research papers leverage Wikipedia for AI and NLP purposes [17]. Many other knowledge bases are derivatives or enhancements to Wikipedia in one way or another.[11] One is hard-pressed to identify any large-scale knowledge base, available in electronic form, that is being exploited as much for AI or semantic technology purposes.[12]

- *Wikidata* is a crowdsourced, open *knowledge base* of (currently) about 55 million structured *entity records*. Each record consists of *attributes* and values with robust cross-links to multiple languages. Wikidata is a crucial entities source.
- *Cyc* is a common-sense *knowledge base* developed over 20 years involving about 1000 person-years of effort. The smaller open-source OpenCyc version is the one we use in KBpedia; an OWL version was available until that project ended in 2017. A ResearchCyc version of the entire system is still available to researchers. The Cyc platform contains a dedicated logic language, CycL, and has many built-in functions in areas such as *natural language processing*, search, inferencing, and the like. *UMBEL* is based on a subset of OpenCyc.
- *DBpedia* is a project that extracts structured content from *Wikipedia* and then makes that data available as *linked data*. Millions of entities are characterized by DBpedia in this way. As such, DBpedia is one of the largest—and most central—hubs for *linked data* on the Web.
- *GeoNames* integrates geographical data such as names of places in various languages, elevation, population, and others from multiple sources. We obtain nearly 800 feature descriptors from GeoNames for organizing geographic and geopolitical information, as well as millions of well-characterized and -defined place names and regions.
- *UMBEL*, short for Upper Mapping and Binding Exchange Layer, is an *upper ontology* of about 35,000 reference concepts, designed to provide universal mapping points for relating different ontologies or schema to one another, and a vocabulary for aiding that mapping.

The combination of these sources, organized by Peirce's triadic universal categories and typologies in the KKO, makes KBpedia a singularly unique knowledge resource.

[11] Though a bit dated, an 82-page technical report by Olena Medelyan et al. from the University of Waikato in New Zealand, *Mining Meaning from Wikipedia* [18], describes the unique structural and content reasons why Wikipedia is an absolutely irreplaceable source for notable entities, and semantic Web and natural language research.

[12] An exception is the biomedical community through its Open Biological and Biomedical Ontologies (OBO) initiative.

Secondary KBs

We have mapped about 15 leading external vocabularies and ontologies to KBpedia, with the first three playing a more prominent role. This listing of mappings is as follows (Table 11.2):

Table 11.2 Secondary knowledge bases for KBpedia

schema.org	This extendable vocabulary describes common things, businesses, and events on the Web. Major search engines, including Google, sponsor it. There are more than 700 types in the vocabulary. Millions of Web documents are marked up with this vocabulary.
DBpedia Ontology	This ontology, an extension of the base DBpedia knowledge base, is meant to be an organizational framework for the information in Wikipedia infoboxes. There are more than 700 types in this ontology.
Dublin Core	Dublin Core, and its metadata extensions, is a generalized vocabulary for describing conceptual works, developed by the library community. It is a widely used core vocabulary across many domains.
Bibliography Ontology	This generalized bibliographic ontology is used to describe books and periodicals; it is the most widely used bibliographic schema.
Description of a Project (DOAP)	A general vocabulary for describing projects.
Friend of a Friend	FOAF is a project devoted to linking people and information using the Web based on social networks, representational networks, and information networks.
FRBR	This vocabulary for the functional requirements for bibliographic records is a recommendation of the International Federation of Library Associations and Institutions (IFLA) for how to structure catalog databases to reflect the conceptual structure of information resources.
Geo	Geo is a vocabulary for representing latitude, longitude, and altitude information in the WGS84 geodetic reference schema.
Music Ontology	MO is a vocabulary for describing music-related topics (i.e., artists, albums, and tracks).
Open Organizations	OO is a vocabulary that provides supplementary terms for organizations wishing to publish open data about themselves.
Organization Ontology	The organization ontology is a core ontology for organizational structures, aimed at supporting linked data publishing of organizational information across some domains.
Programs Ontology	The programs ontology is a simple vocabulary for describing media programs. It covers brands, series (seasons), episodes, broadcast events, broadcast services, etc.
SIOC	The SIOC initiative (semantically interlinked online communities) is a vocabulary for the integration of online community information.
Time Ontology	The OWL-time ontology is a vocabulary of temporal concepts, for describing the temporal properties of resources in the world or described in Web pages.
TRANSIT	TRANSIT is a vocabulary for describing transit systems, routes, stops, and schedules.
US PTO	The US Patent and Trademark Office provide links to millions of organizations and brands that have sought or received trademark protection from the US Government.

The base KBpedia also includes entity mappings (organizations only) to Freebase (though abandoned, prior users have transferred much to Wikidata) and the US Patent and Trademark Office (USPTO) databases. Since these are not full mappings, we do not include them in the statistics for the base KBpedia.

Candidate KBs for Expansion

For specific domains, multiple and rich sources may exist to expand KBpedia to accommodate that scope. Chapter 13 develops the topics of finding and screening such sources, using some of the acceptance criteria above. For now, let me note that, in varying degrees, vocabularies, thesauri, taxonomies, and, even, tables of content may be useful starting points for domain concepts and scope expansions. One may find local instance data from internal relational data stores and spreadsheets. Sometimes you may find useful domain data and structure from academic publications, trade organizations, or various sector studies.

As for KBpedia, some new areas that we are contemplating include country-specific economic and demographic data, more online databases, brand and product data, expanded corporate and ownership data, sustainability metrics associated with significant economic pathways, or lexical databases, such as WordNet or VerbNet. As a sponsor of the open-source project, we will be responsive to multilingual versions and will work to catalyze more mappings, more linkages, and more extensions.

References

1. J.J. Sylvester, Chemistry and algebra. Nature **17**, 284 (1878)
2. L.A. Bunimovich, D.C. Smith, B.Z. Webb, Specialization models of network growth, *arXiv:1712.01788 [physics]* (2017)
3. G. Gilder, *Metcalfe's Law and Legacy* (Forbes ASAP, 1993), p. 158
4. S.F. Peralta, *Moore's Law, Metcalfe's Law, and the Dot Com Bubble* (2011)
5. D.P. Reed, That sneaky exponential—Beyond Metcalfe's law to the power of community building, in *Services, in XVth International Symposium on Services and Local Access* (Edinburgh, 1999)
6. B. Briscoe, A. Odlyzko, B. Tilly, Metcalfe's law is wrong, in *IEEE Spectrum* (2006)
7. Y.J. Stein, *The value of being linked in.* http://www.dspcsp.com/pubs/linkedin.pdf
8. S. Harris, A. Seaborne, *SPARQL 1.1 Query Language*, World Wide Web Consortium (2013)
9. M. Kuba, *OWL 2 and SWRL tutorial*. https://dior.ics.muni.cz/~makub/owl/
10. D.L. McGuinness, Ontologies come of age, in *Spinning the Semantic Web: Bringing the World Wide Web to its Full Potential*, ed. by D. Fensel, J. Hendler, H. Lieberman, W. Wahlster (MIT Press, Cambridge, 2003), pp. 171–194
11. O. Bodenreider, F. Olken, Ontology Summit 2007 Communique, in *Ontology Summit 2007* (Ontolog Forum, Gaithersburg, MD, 2007)
12. N. Guarino, Formal ontology and information systems, in *Proceedings of FOIS'98* (IOS Press, Trento, 1998), pp. 3–15

13. M. Uschold, Ontology-driven information systems: past, present and future, in *Proceedings of the Fifth International Conference on Formal Ontology in Information Systems (FOIS 2008)*, ed. by C. Eschenbach, M. Grüninger (IOS Press, Amsterdam, 2008), pp. 3–20
14. P.C. Costa, *Bayesian Semantics for the Semantic Web*, Ph.D., George Mason University (2005)
15. K.B. Laskey, MEBN: a language for first-order Bayesian knowledge Bases. Artif. Intell. **172**, 140–178 (2008)
16. F. Bobillo, U. Straccia, Fuzzy ontology representation using OWL 2. Int. J. Approx. Reason. **52**, 1073–1094 (2011)
17. F.M. Suchanek, G. Weikum, Knowledge bases in the age of big data analytics, in *Proceedings of the VLDB Endowment* (2014), pp. 1713–1714
18. O. Medelyan, D. Milne, C. Legg, I.H. Witten, Mining meaning from Wikipedia. Int. J. Hum. Comput. Stud. **67**, 716–754 (2009)

Part IV
Building KR Systems

Chapter 12
Platforms and Knowledge Management

Having discussed terminology and components in previous chapters, now let us turn our attention in this *Part IV* to building an actual knowledge representation *system*. The major theme of the effort is to obtain maximum value from the work of converting and integrating data not only to achieve the aims of *data interoperability* and *knowledge-based artificial intelligence* (KBAI) but to leverage maximum benefits from *knowledge management* as well. We follow this chapter on platforms with two additional chapters in *Part IV* on how to build out and tailor a system for your own domain needs and on testing and best practices.

The material in these three chapters draws on our experience in building semantic technology platforms for a variety of clients and applications over the prior decade.[1] In various guises and tailorings, we have created stand-alone and Drupal-based platforms using PHP, and have created stand-alone systems using the Clojure language. Though we have released portions of these efforts as open source—Clojure components related to KBpedia, and PHP and Drupal frameworks for the Open Semantic Framework (OSF)—we are not prescriptive in this chapter or elsewhere in the book about how to build a KR/KM platform. Rather, we emphasize guidelines and lessons learned versus any specific design or language. Platforms will continue to emerge and evolve, and what we should seek from those platforms

[1] Some material in this chapter was drawn from the author's prior articles at the *AI3:::Adaptive Information* blog: "'and Now You Know the REST of the Story . . .'" (Feb 2007); "WOA: A New Enterprise Partner for Linked Data" (Oct 2008); "WOA! So RESTful it is UMBELievable!" (Oct 2008); "A General Web-oriented Architecture (WOA) for Structured Data" (May 2009); "The Fundamental Importance of Keeping an ABox and TBox Split" (May 2009); "Ontology-driven Applications Using Adaptive Ontologies" (Nov 2009); "The Open World Assumption: Elephant in the Room" (Dec 2009); "I Have Yet to Metadata I Didn't Like" (Aug 2010); "An Ontologies Architecture for Ontology-driven Apps" (Dec 2011); "Architecting Semantic Technologies for the Enterprise" (Jan 2013); "Enterprise-scale Semantic Systems" (Jan 2013); "Semantic Technology Access Control Using Datasets" (Feb 2013); "Five Fundamental Distinctions of Enterprise Software" (Jan 2014); "Logical Implications of Interoperability" (Jun 2015); "Creating a Platform for Knowledge-based Machine Intelligence" (Sep 2015); "A Foundational Mindset: Firstness, Secondness, Thirdness" (Mar 2016); "Why I Study CS Peirce" (Aug 2017).

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8_12

regarding design and architecture is of more guiding importance than any specific instantiation.

We begin this chapter by critically reviewing the work objectives of a platform. These functional understandings are related to the earlier *TBox* and *ABox* splits we discussed for *description logics* in Chap. 8. We also discuss the importance of content and general workflows. From this basis, we then proceed to look at platform considerations. As noted in Chap. 4, the platform should support three main opportunities in general *knowledge management*, *data interoperability*, and *knowledge-based artificial intelligence* (KBAI). We also discuss access control and governance, and other enterprise considerations. The last section of this chapter deals with the overall *Web-oriented architecture*, emphasizing the importance of Web connectivity and the use of modular Web services for scalability and flexibility. The entirety of these considerations helps us set the overall guidelines for the design and architecture for a responsive knowledge representation and management platform.

Uses and Work Splits

To contemplate what a knowledge representation platform should look like, we first need to define what kinds of work we anticipate the platform to do. These work requirements are related to the purposes we have for the platform, as well as the existing state of tooling and applications available to support them. (Chapters 15 and 16 offer additional use cases.) Workflows are also intimately tied to these questions.

The State of Tooling

I have been tracking and documenting the state of semantic technology, graphics visualization, and knowledge management tooling for nearly two decades. For many years I maintained *Sweet Tools*, a searchable and faceted compendium of semantic technologies that grew to a listing exceeding 1000 tools, the most comprehensive available.[2] In our platform work, we have used and integrated some of the leading tools available from this listing. We have also extended and created many of our tools and ontologies that we have contributed back to the community as open source.[3]

[2] *Sweet Tools* is still online with its searchable tool listing at http://www.mkbergman.com/sweet-tools/ (for statistics on the latest release, see http://www.mkbergman.com/991/the-state-of-tooling-for-semantic-technologies/). However, it has not been updated since the beginning of 2012 and is substantially out of date. There is no alternative survey to my knowledge.

[3] Prior tools that we have released under various open-source licenses include BibJSON, Citizen Dan, conStruct, irON, Open Semantic Framework, OSF for Drupal, scones information tagger, structWSF, and structXML. Ontologies and vocabularies that we have released under various open-use licenses include BIBO (Bibliographic Ontology), the MUNI Ontology, the Music Ontology, and UMBEL.

We now have much tooling and demo experience to draw upon since the seminal article on the semantic Web appeared in the *Scientific American* in 2001 [1]. The primary sources for supporting the semantic Web are the European Union, mostly for academics, and the US Government, mainly for intelligence and biomedical purposes to academics and businesses alike.

In the early years, ontology standards and languages were still in flux, and the tool basis was similarly immature. Frame logic, description logics, common logic, and many others were competing at that time for primacy and visibility. Practitioners based most ontology tools at that time such as Protégé [2], OntoEdit [3], or OilEd [4] on F-logic or the predecessor to OWL, DAML+Oil. The emergence of OWL and then OWL 2 by the W3C helped solidify matters. The University of Manchester introduced the OWL API [5], which now supports OWL 2 [6]. Protégé, in version 5x, is now solely based on OWL 2 and has become a popular open-source system, with many visualization and OWL-related plug-ins. A leading commercial editor is TopBraid Composer, which uses the Eclipse IDE platform and Jena API.[4] The OWL API is now a standard used by Protégé and leading reasoners (Pellet, HermiT, FaCT++, RacerPro). It supports a solid ontology management and annotation framework, and validators for various OWL 2 profiles (RL, EL, and QL).

RDF data management systems, or 'triple stores,' such as OpenLink's Virtuoso, Ontotext's GraphDB, and Franz's AllegroGraph, are now mature offerings. One may also apply modifications of existing data stores by Oracle, MarkLogic, and a variety of NoSQL databases to the design ideas presented herein. Developers presently have multiple open-source and commercial options to choose from, including cloud options such as Amazon's Neptune, for hosting RDF and OWL databases. The more comprehensive frameworks have opted to become ontology-engineering environments and to provide all capabilities in one box via plug-ins.

Java is the language of choice for about half of the semantic technologies, though existing toolsets use more than a score of different languages. Academic tools are often the most innovative, but the degree of completeness is often frustrating and most academic and grant-supported tools have limited or no support. Many, after a single experimental release, are abandoned or see no further development. Newer academic releases (often) are more strategically oriented and parts of broader programmatic emphases. Programs like AKSW from the University of Leipzig or the Freie Universität Berlin or Finland's Semantic Computing Research Group (SeCo), among many others, are exemplars of this trend. Promising projects and tools are now much more likely to be spun off as potential ventures, with accompanying better packaging, documentation, and business models.

Full-text search is weak in RDF triple stores, and many leading approaches now match a text engine with the semantic portions. Some excellent components exist, but not yet packaged into single-stop solutions as RedHat did with Linux. The ontology tooling is especially difficult for standard knowledge workers to use, and the coupling of tools into current, actual workflows is lacking. Our experience is

[4] Jena is fundamentally an RDF API. Jena's ontology support is limited to ontology formalisms built on top of RDF. Specifically this means RDFS, the varieties of OWL, and the now-obsolete DAML+OIL. See http://jena.apache.org/.

that most potential components are incompletely tested, and lack many basic expectations suitable for enterprise environments. Much scripting is necessary to glue together existing parts. However, some of the design guidance provided herein, especially about the use of canonical data forms, Web services, and suitable modular architectures, can help overcome many of these problems. It is possible to create a proper enterprise knowledge management environment at acceptable cost using available open-source components and solid architectural design. The Apache Software Foundation is doing an especially good job of picking, incubating, and supporting a diversity of open-source tools useful to semantic technologies. These tools include Ant, Hadoop, HTTP server, Jackrabbit, Jena, Mahout, Marmotta, Maven, OpenNLP, Singa, Stanbol, SystemML, Tika, Tomcat, UIMA, ZooKeeper, and the Lucene and Solr search engines and Nutch crawler. Additional tooling that would make this task easier still includes the following:

- Vocabulary managers—We lack easy inspection and editing environments for concepts and predicates. Though standard editors allow direct ontology language edits (OWL or RDFS), these are not presently navigable or editable by non-ontologists. Intuitive browsing structures with more 'infobox'-like editing environments could be helpful here.
- Graph API—It would be wonderful to have a graph API (including analysis options) that could communicate with the OWL API. As a second option, it would be helpful to have a graph API that communicates well with RDF and ontologies.
- Large-graph visualizer—While I have earlier reviewed large-scale graph visualization software [7], with Gephi and Cytoscape being my two preferred alternatives, they are neither easy to set up nor use. I would like more easily to select layout options with quick zooms and scaling options.
- Graphical editor—Some browsers or editors provide nice graph-based displays of ontologies and their properties and annotations. However, the better design we advocate here is to edit the ontology graph directly in its deployment environment.
- Component services—We recommend piecing out ontology and knowledge management functions into individual components that we can integrate directly into existing workflows with minimal training.

TBox, ABox, and Work Splits

To better understand what kinds of functions we require and how they may relate to existing tools or applications, recall the discussion of *description logics* in Chap. 8. Description logics and their semantics traditionally split *concepts* and their relationships from the different treatment of *individuals* and their attributes and roles, expressed as fact assertions. The concept split is known as the *TBox* (for *terminological* knowledge, the basis for *T* in *TBox*) and represents the schema or taxonomy

of the domain at hand, what we also call the *knowledge graph*. The TBox is the structural and extensional component of conceptual relationships. The second split of individuals is known as the *ABox* (for *assertions*, the basis for *A* in *ABox*) and describes the attributes of individuals, the roles between individuals, and other assertions about individuals regarding their class membership with the TBox concepts. The ABox is the repository for data records and can be a light layer over existing data stores. Both the TBox and ABox are consistent with set-theoretic principles.

TBox and ABox logic operations differ, and their purposes vary. TBox operations are based more on inferencing and tracing or verifying class memberships in the hierarchy (that is, the structural placement or relation of objects in the structure). ABox operations are more rule based and govern fact checking, instance checking, consistency checking, and the like. ABox reasoning is often more complicated and at a larger scale than that for the TBox. However, even with these TBox and ABox splits, we can also see that some work done by a knowledge management system falls outside of the specific purview of instances and concepts (Fig. 12.1):

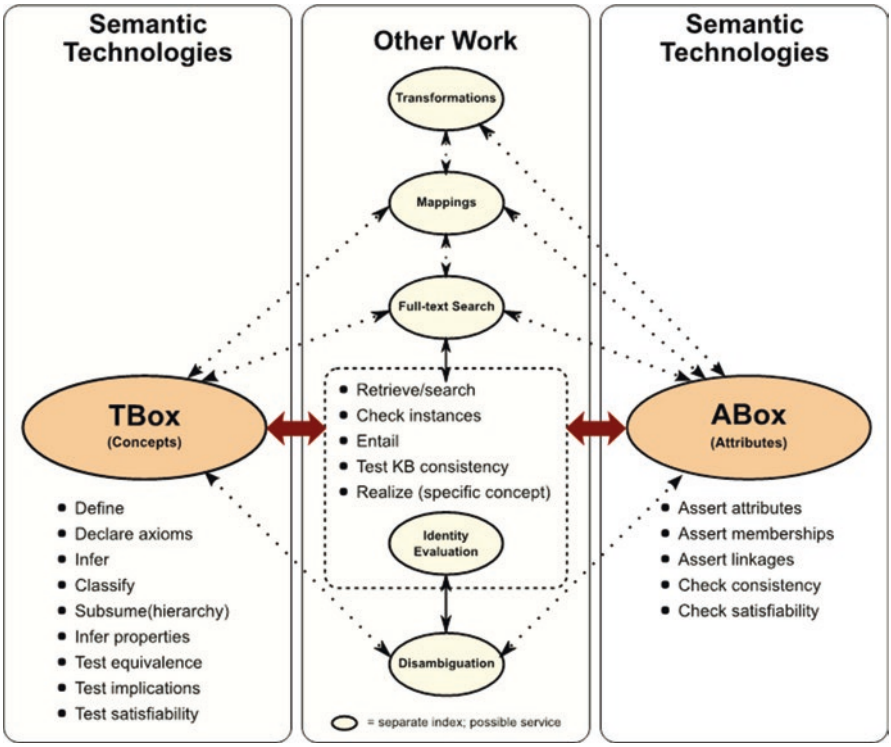


Fig. 12.1 Possible work splits in a KM platform

Searching across the entire database or conducting machine learning, as examples, are such functions that work against the whole knowledge structure, or which pose work requirements orthogonal to the TBox-ABox splits. Table 12.1 summarizes how we may segregate these significant work areas against the TBox, the ABox, or possibly separate to them.

Table 12.1 Possible work activities in a knowledge management platform

TBox	Possibly separate work tasks	ABox
• *Definitions* of the *concepts* and *properties* (relationships) of the controlled vocabulary. • *Declarations* of *concept axioms* or *roles.* • *Inferencing* of relationships, be they transitive, symmetric, functional, or inverse to another property. • *Equivalence testing* as to whether two classes or properties are equivalent to one another. • *Subsumption*, which is checking whether one concept is more general than another. • *Satisfiability*, which is the problem of checking whether a concept has been defined (is not an empty concept). • *Classification*, which places a new concept in the proper place in a taxonomic hierarchy of concepts. • *Logical implication*, which is whether a generic relationship is a logical consequence of the declarations in the TBox. • *Infer property assertions* implicit through the transitive property .	• *Mappings* are the core of interoperability in that concepts, and attributes get matched across schema and datasets. • *Transformations* are the means to bring disparate data into common grounds, the second leg of interoperability. • *Entailments*, which are whether the stated condition implies other propositions. • *Instance checking*, which verifies whether a given individual is an instance of (belongs to) a specified concept. • *Knowledge base consistency*, which is to verify whether all concepts admit at least one individual. • *Realization*, which is to find the most specific concept for an individual object. • *Retrieval*, which is to find the individuals that are instances of a given concept. • *Identity relations*, which is to determine the equivalence or relatedness of instances in different datasets. • *Disambiguation*, which is resolving references to the proper instance. • *Machine learning* based on entities and features in the knowledge base.	• *Membership assertions,* either as *concepts* or as *roles.* • *Attributes assertions.* • *Linkages assertions* that capture the above but also assert the external sources for these assignments. • *Consistency checking* of instances. • *Satisfiability checks*, which are meeting the conditions of instance membership.

The TBox should be a coherent structural description of the domain, which expresses itself as a knowledge graph with meaningful and consistent connections across its concepts. Somewhat irrespective of the number of instances (the ABox) in the knowledge base, the TBox is relatively constant in size given the desired level of descriptive scope for the domain. (In other words, the logical model of the domain is mostly independent of the number of instances in the domain.) As its name suggests, the TBox is where we define terminology for the vocabulary of the domain, the predicates used, and the relationships of those concepts to one another via the predicates available. A key aspect of the TBox functionality is classification through subsumption hierarchies, from which we set much of the logic and inferencing capabilities of the structure. The TBox also requires checks during its building and maintenance to ensure that we have provided complete definitions (*satisfiability*) and consistency and logic tests to make sure that our placements within the knowledge graph remain *consistent* and *coherent*.

The ABox of instances consists of the specific individual things in the KB that are relevant to the domain. Instances can be many or few, as in the millions within KBpedia, accounting for 90% or more of the total number of objects in the knowledge base. We characterize instances by various types of structured data, provided as attribute-value pairs, and which we describe with long or short texts and with multiple aliases and synonyms, and we relate to other instances via type or kind or other relations, possibly in multiple languages.

We can perhaps better illustrate this work split with showing the interactions of all of these contributing parts.

Whether a single database or the federation across many, we have data records (instances in the ABox) and a logical schema (*ontology* of concepts and relationships in the TBox) by which we try to relate this information. As Table 12.1 and Fig. 12.1 show, the TBox is where the reasoning work occurs; the ABox is where assertions and data integrity occur. This meaningful work broadly relates to the growth and maintenance of the knowledge base itself. For instance, all aspects of ontology editing relate to these components, as do logic, consistency, coherency, and satisfiability checks. These portions are essential to the integrity of the knowledge structure via its editing and maintenance but represent very little of the desired work we want to extract from the knowledge structure. These work tasks are separate from the needs of the TBox and ABox themselves.

The middle column of Table 12.1 and Fig. 12.1 list some of those work tasks that reside outside of the knowledge graph and knowledge base build and maintenance tasks. Some of these tasks may apply across the entire knowledge structure, such as search or retrieval. Other tasks are specialized ones that may involve subsets of the structure or dedicated extractions of one form or another.

What Fig. 12.1 readily shows is that platforms with only semantic technologies lack the major work functions desired. It is this gap to bring in and facilitate dataset

exchanges to external applications that most requires tailored scripting for specific installations (along with the need to create the domain knowledge graph and ingest data, of course). It is why stand-alone semantic technology platforms have not been, generally, commercially successful. Not shown in the figure is the further general weakness of semantic technology platforms; namely, they are hard to learn and use. We need more visual frameworks with well-segregated tasks, such as what we are beginning to see in such tools as the SKOS-based PoolParty.

Providers have increasingly embraced platforms that integrate conventional text search engines, such as Solr, for generalized retrieval, plus use in instance and consistency checks. However, critical areas such as mappings, transformations, and identity evaluation remain weak. *Mappings* refer to the suite of aids that suggest matching correspondences between objects in the domain knowledge base with external sources, with choices often manually vetted. *Transformation* is the ability to convert subsets of the knowledge graph to the dataset format required by various external applications. These include machine learning, AI, or specialized natural language processing (NLP) like parsing into parts of speech or transforming external sources into new records or updating the knowledge base. *Identity evaluation* means to contextualize a possible entity reference to its disambiguated actual subject. Maintaining identity relations and disambiguation as separate components also has the advantage of enabling us to swap out different methodologies or algorithms as better methods become available. We could apply a low-fidelity service, for example, for quick or free uses, while we reserve more rigorous methods for paid or batch-mode analysis. We may deploy any of these mapping, transformation, or identification activities as a Web service, preferably using an internal canonical data transfer form, discussed further toward the end of this chapter.

Breaking our description logics design into the TBox and ABox, and then enumerating the work tasks we wish to do against these structures, helps us to think through the modularity and architecture we want to see in our actual deployments. The practical aspects of our work tasks and where and how they should occur become clearer. We know that we can architect a framework that is amenable to swapping in and out different analysis methods, and that can be modular to use or not different work tasks and applications. Here are some general principles that should apply to most domain installations:

- We want to handle our concepts, and their definitions and relationships (TBox) separate from our instance data, and subject to rigorous testing, vetting, and updating since this is the controlling logical structure of our knowledge management system.
- The task of knowledge graph creation and maintenance should be the responsibility of knowledge workers and their management, not the IT department.
- We want to handle our instance data (ABox) separately and directly, using comparatively constant and readily understandable attribute-value pairs.

- We can reuse these instance records in varied and multiple worldviews in relation to different TBoxes or external applications; we can support these different perspectives without affecting instance data in the slightest.
- We should approach architectural decisions from the standpoint of the *work* to be done, leaving open unique analysis or tasks like disambiguation or full-text search as functions, which may be added or not at another time.
- Ontologies should be modular, scoped according to appropriate user groups, and kept as simple and easy to understand as possible; this is a significant rationale for the *typology* design discussed in Chap. 10. We should assert inter-ontology relationships via a rather simple upper ontology, such as what is provided by the KBpedia Knowledge Ontology.
- We may base mapping on suggestions from TBox (extensional) relationships or ABox (intensional) relationships, and is a particularly weak yet important part of tooling.
- We can treat logic and consistency testing as external applications, and conduct them on schedule or on demand via services using canonical formats.
- We should evaluate instances separately from concepts, which also via triangulation may aid such tasks as disambiguation or entity identification.
- We should include access control and governance (missing from Fig. 12.1) in most enterprise settings or where we use proprietary or private data.
- We can often keep instance records in situ, especially useful when incorporating the massive amounts of data in existing relational databases.
- We may add to instance stores incrementally, via in situ or staged, following these same design principles, and given the discussion in Chap. 9.
- We should premise the entire system on continuous change given the nature of knowledge and its openness.

Content Workflows

Two of these critical work splits are thus to (1) keep knowledge updated and (2) directly involve knowledge workers and subject matter experts. These requirements go hand and hand since the source of new knowledge comes from these workers and their accumulated content in the first place. More simply put, to capture knowledge, the systems to do so must be in the hands of the knowledge workers themselves, and must integrate cleanly into their existing content workflows. It is inefficient not to leverage existing workflows. Users will likely ignore new knowledge graph maintenance and use tasks unless they are dead simple to implement. We best achieve adoption through an incremental series of nonthreatening tasks. Figure 12.2 sketches out broad steps and interactions that one might want to see in a content workflow.

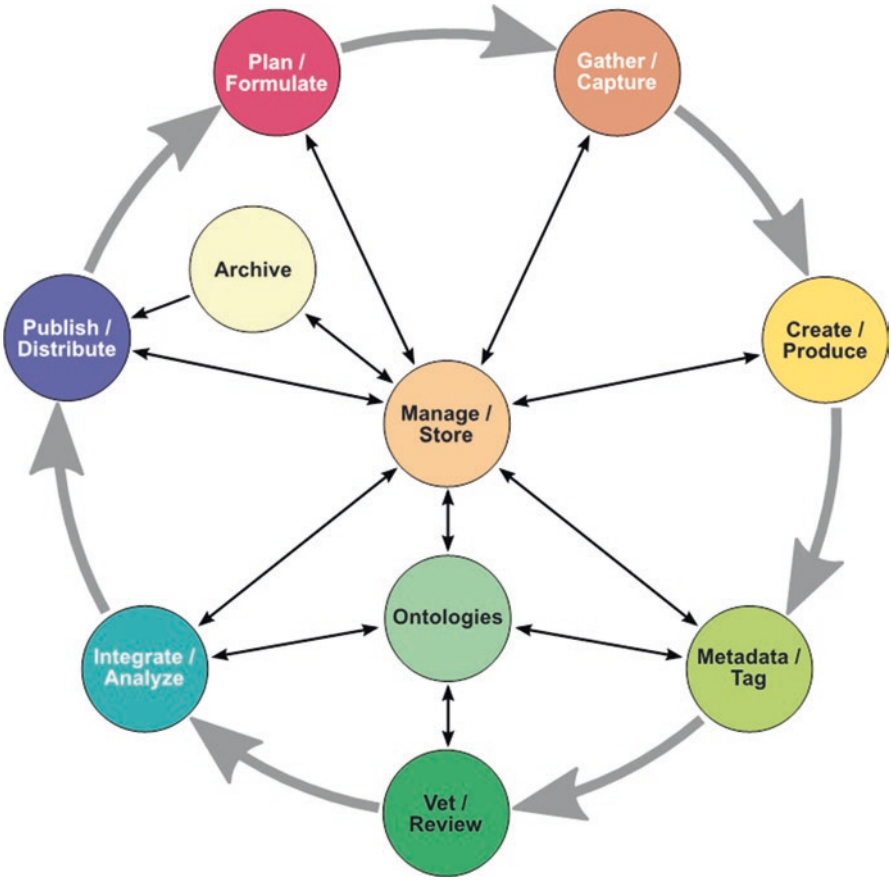


Fig. 12.2 Content workflows in a KM platform

Respect for workflows is the first principle when setting boundaries around functional requirements. We express this respect in two different ways. The first is that we cannot unduly disrupt existing workflows when introducing interoperability improvements. While workflows can be improved or streamlined, new tools and practices must fit with existing ways of doing tasks to see adoption. Users mostly resist jarring changes to existing work practices.

The second way is that we should explicitly model and codify the workflows of how we do tasks. This codification becomes the 'language' of our work and helps define the tooling points or points of interaction as we merge activities from multiple disciplines in our domain. These workflow understandings also help us identify useful points for APIs in our overall interoperability architecture. An excellent use of an administrative ontology is to codify and model the workflow and approval steps associated with informal and formal content workflows in the organization.

Some steps within Fig. 12.2 may not be active within an organization, such as tagging or assigning metadata. Cases like this probably need to identify tasks in the associated content creation and review where we can link metadata additions into current workflows. These kinds of incremental additions to existing workflows continue to suggest the wisdom of breaking apart the individual steps in ontology creation and maintenance to more atomic parts, such as flagging a new concept or adding to a *semset* label for an existing one. We may then slipstream these additional steps into separate ontology suggestions that authorized editors review and vet before final acceptance. These steps, of course, and how we refer to them, may vary across circumstances and organizations. Nonetheless, we may apply the general ideas of work steps, approval types, and users to any formal or informal workflow that presently exists.

These considerations provide the rationale for assigning metadata that characterizes our information objects and structure. We should base this metadata on controlled vocabularies and relationships in domain and administrative ontologies, as determined by their users (knowledge workers). The vocabularies and the tagging of information objects with them are a first principle for ensuring how we can find and transition states of information. These vocabularies need not be elaborate, but they should be constant and consistent across the entire content life cycle. Backbone aspects of these vocabularies should capture the overall information workflow, as well as concrete steps for individual tasks. As a complement to such administrative ontologies, domain ontologies provide the context and meaning (semantics) for our information.

This common grounding of data model and semantics means we can connect our sources of information. The properties that define the relationships between things determine the structure of our knowledge graph. Seeking commonalities for how our information sources relate to one another helps provide a coherent graph for drawing inferences. How we describe our entities with attributes provides a second type of property. Attribute profiles are also a good signal for testing entity relatedness. Properties—either relations or attributes—give another filter to draw insight from available information.

If the above sounds like a dynamic and fluid environment, you would be right. Ultimately, knowledge is a challenge in a technology environment that is rapidly changing. New facts, perspectives, devices, and circumstances are continually arising. For these very reasons a knowledge management framework must embrace the *open world assumption* (see Chap. 9), wherein we can grow and extend the underlying logic structure and its vocabulary and data at will.

Though perhaps not quite at the level of a first principle, I also think KM improvements should be easy to use, easy to share, and easy to learn. I imply tooling in this, but also it is important we be able to develop a language and framing for what constitutes our knowledge domain. We should pursue the question of interoperability to discover insights and gain efficiencies. The thing about interoperability is that it extends over all aspects of the information life cycle, from capturing and creating information to characterizing and vetting it, to analyzing it, or publishing or distributing it. Eventually, information and content already developed become input to

new plans or requirements. These aspects extend across multiple individuals and departments and even organizations, with portions of the life cycle governed (or not) by their own set of tools and practices.

Today, overall, we only embrace pieces of this cycle in most daily workflows. Editorial review and approvals, or database administration and management, or citation gathering or reference checking, or data cleaning, or ontology creation and management, or ETL activities, or hundreds of other specific tasks sit astride this general backbone. Besides showing that interoperability is a systemic activity for any organization (or should be), we can also derive a couple of other insights from Fig. 12.2.

First, we can see that some form of canonical representation and management is central to interoperability. The form need not be a central storage system, but can be distributed using Web identifiers (IRIs) and protocols (HTTP). Second, we characterize and tag our information objects using ontologies, both from structural and administrative viewpoints, but also by domain and meaning. We can combine and analyze our information when we characterize it with a common semantics.

A third insight is that a global schema (from the standpoint of the enterprise) specific to workflows and our content is a key for linking and combining activities at any point within the cycle. A common vocabulary for stages and interoperability tasks, included as a best practice for our standard tagging efforts, provides the conventions for how batons can get passed between activities at any stage in this cycle. The challenge of making this insight operational is one more of practice and governance than of technology. It should be a purposeful activity in its own right, backed with appropriate management attention and incentives.

An enabling mindset for the knowledge workers involved is to pay explicit attention to content workflows and common vocabularies for those flows and the information objects they govern. This focus becomes the scaffolding for an administrative ontology and a basis for investigating tooling and automation in processing information. We can already put in place chains of tooling and workflows to achieve a degree of interoperability. We do not need to provide global answers or scope at the inception. We can start piecemeal, and expand as we benefit. The biggest gaps remain codification of workflows for the overall information life cycle, and the application of taggers to provide the workflow and structure metadata at each stage in the cycle. Again, these are not matters so much of technology or tooling, but willingness, and policy and information governance.

Platform Considerations

Semantic technologies have not yet reached the point of fulfilling their prophecy nor of being sufficiently buzz-worthy to fuel their demand.[5] Enterprise customers are intrigued with the idea of semantic solutions but remain skeptical. Better search is often the crucial leverage point in the sale. Enterprises do not seem interested in

[5] See Chaps. 15 and 16.

linked data alone (if at all), though some like the idea of possibly contributing linked data back to others. On the other hand, all enterprises competing in the current environment understand that knowledge, and their use and management of it, is perhaps the pivotal factor in their relevance and survival.

I have had the good fortune to work with some cutting-edge, reference enterprise deployments of semantic technologies. These efforts in enterprise-scale systems have been eye-opening. We have opened one eye for how semantic technologies need to integrate and adapt to existing enterprise practices and deployments. We have opened the other eye to see how semantic technologies should be presented and sold to internal enterprise stakeholders.

We have a working example in the Open Semantic Framework that shows the way for how a few common representations and conventions can work to distribute both schema and information (data) across a potentially distributed network. Further, by not stopping at the water's edge of data interoperability, we can also embrace further, structural characterizations of our content. Adding this wrinkle enables us to support a variety of venues for content consumption simultaneously and efficiently, as well as to broaden our leverage of the knowledge asset through cheaper, more streamlined machine learning and artificial intelligence. What I set out in the next section are the multiple purposes and the ontology-driven aspects of a general knowledge representation and management platform to support enterprise aims.

Supporting Multiple Purposes

Our avowed purposes in data interoperability and KBAI, supported by general KM (knowledge management) uses, set the overall application scope for our platform. At the same time, we understand that particular uses of the platform will vary by domain, desired application emphases, and actual instance data. We further assume that initial demands and scope may warp and grow as we experience platform results, and external demands dictate. All of these considerations demand a platform design that is open, modular, and extensible, capable of supporting multiple purposes (and, thus, cost justifications). We need to put forward reasonable projected benefits that greatly exceed development costs, and then to continue to justify such assertions to sustain a healthy, dynamic knowledge management system. Specific domain applications are surely the instrumental justification for an initial installation, but an adaptive KM platform should also meet the two core requirements of search and knowledge management.

Search

Enterprises, familiar with structured query language (SQL), have understood for quite some time that queries and search are more than text searches to search engines. Semantic technologies have their structured query approach, SPARQL. State-of-the-art semantic search has found a way to combine these various underlying retrieval

engines with the descriptive power of the graph and semantic technologies to provide a universal search mechanism across all types of information stores. The simplest way to understand semantic search is to deconstruct the basic RDF triple down to its fundamentals. This first observation is that the RDF data model can represent anything, that is, an object or idea. Moreover, we can represent that object in virtually any way that any viewer would care to describe it, in any language. In semantic search, we may derive facets from not only what types of things exist in the search space, but also what kinds of attributes or relations connect them. Gratifyingly, this all comes for free. Unlike conventional faceting, no one needs to decide what are the important 'dimensions' or any such. With semantic search, the very basis of describing the domain at hand creates an organization of all things in the space.

In semantic search, every property represents a different pathway, and every node is an entry point. SPARQL enables us to pose queries, including with variables, which can navigate and slice and dice the information space into usable result subsets at will. We do not need to state all of the relationships and types of things in our information space; we can infer them from the assertions already made. We can use these broad understandings of our content to do better targeting, tagging, highlighting, or relating concepts to one another. The fact that semantic search is a foundation for semantic publishing is noteworthy.

We first adopted Solr (and then Lucene) because traditional text search of RDF triple stores was not sufficiently performant and made it difficult to retrieve logical (user) labels in place of the IRIs used in semantic technologies. In our design, the triple store is the data orchestrator. The RDF data model and its triple store are used to populate the Solr schema index. The structural specifications (schema) in the triple store guide the development of facets and dynamic fields within Solr. These fields and facets in Solr give us the ability to gain Solr advantages such as aggregates, auto-completion, filtering, spell checkers, and the like. We can also capture the full text if the item is a document, enabling us to combine standard text search with the structural aspects orchestrated from the RDF. On the RDF side, we can also leverage the schema of the underlying ontologies to do inferencing (via forward chaining). We have been able to (more or less) seamlessly embrace geo-locational based search, time-based search, use of multiple search profiles, and switchable ranking and scoring approaches based on context (using Solr's powerful extended disMax *edismax* parser).[6] This combination gives us an optimal search platform to do full-text search, aggregates, and filtering.

Knowledge Management

Our earlier Fig. 12.1 showed the two bracketing left- and right-work areas in semantic technologies. These are the very same knowledge graph (TBox) and instance data (ABox) areas that form the knowledge base that our KM system must manage. Here are some of the tasks we need to manage: (1) insert and update concepts in the upper ontology; (2) update and manage attributes and track specific entities as new sources

[6] Similar capabilities may be implemented with Lucene or ElasticSearch.

of data are entered into the system; (3) establish coherent linkages and relations between things; (4) ensure that these updates and changes are done wholly and consistently while satisfying the logic already in place; (5) update how we name and refer to things as we encounter them; (6) understand and tag our content workflows such that we can determine provenance and authority and track our content; and (7) do these tasks using knowledge workers, who already have current tasks and activities.

These actions should be continuous, and established procedures with annotations and logging should govern them. The entire premise of a knowledge management system is to keep current and up to date. This need for currency means that use and updates of the semantic technologies portion, which is the organizing basis for the knowledge in the first place, must be part of daily routines and work tasking, subject to management and incentives. Responsive, tailored tooling linked to current workflows is the technical requirement. Management procedures and training need to complement the technology to ensure that the human factors are also in place.

An Ontology-Based Design

We have seen that an upper ontology governs the overall knowledge graph, with typologies and domain ontologies tailoring the scope and providing instance coverage. We have also seen, in the case of the content life cycle, where we can capture content workflows and approvals into metadata that tracks content across the system and provides provenance information using an administrative ontology. The platform should also provide a standard set of access and retrieval services including browse, full-text search, CRUD, direct record retrievals, and the like. We may embed these within an access and permission service, also governed by an administrative ontology, that acts at the level of registered datasets (see next section). We should also design our queries and requests to the platform to include a parameter for getting result sets in particular formats such as XML or JSON or RDF (various flavors), or others of domain importance. Administrative ontologies can also guide how HTML pages and forms are dynamically populated, often contextually, based on standard SPARQL queries. For specific purposes, we can also return these result sets as pre-staged, properly formatted result streams (usually in the form of SPARQL queries) for driving particular applications. We only need to add a basic converter to the platform's Web service stack to 'drive' a new application in a specific format.

As explained in the concluding section of this chapter, we recommend packaging these platform capabilities as Web services that we can interact with and drive via standard HTTP requests using standard *application programming interfaces* (APIs). Alternatively, we can issue these requests from simple to comprehensive Web apps that create the API queries based on user interface choices such as selections from drop-down lists or clicking on various listed options. The platform thus acts as a single, uniform Web interface to all of the capabilities of the structured data system organized by the adaptive ontologies. Further, we may ingest virtually any data structure and convert it via an import service made part of the underlying canonical structure. Lastly, the dataset nature of the framework, and its neutrality to underlying data

stores or content management systems, also makes the platform an excellent framework for one or many nodes to share information and collaborate across the Web.

'Ontology-driven apps' through this platform design thus provide two profound benefits. First, once we write the templates, we can drive the entire system via simple Web form selections or interactions without the need for any programming or technical expertise. Second, we can power entirely new applications through the addition of new, minor output converters. These potentials arise from the native power of the design basis for ontology-driven apps. Conceptually, the design is simplicity itself. Operationally, the system is extremely flexible and robust. Strategically, it means our development and specification efforts may now move from coding and programmers to the subject matter users who define ontologies and depend on them.

Enterprise Considerations

Security is an additional enterprise requirement that warrants particular attention. Whether profit or nonprofit, all enterprises are unique, with potential proprietary information both internally and externally (with the public or possible competitors). Though individual consumers also have requirements for privacy and confidentiality, these information flows are strictly between the individual and outside entities. In an enterprise, access may occur and be among many internal individuals and all of their external contacts. *Access control* is the protection of resources against unauthorized access. It is a process by which use of resources is regulated according to a security policy and is permitted by only authorized system entities according to that policy.[7]

We may provide access control, like many other enterprise considerations, through a third-party application, by an administrative ontology linked to other features tagged in the knowledge store, or both. As one example, we have provided access control in some installations of the Open Semantic Framework using a three-dimensional matrix of datasets, users/groups, and CRUD rights to tools/endpoints. A dataset refers to a named grouping of records, best designed as similar in record types and intended access rights (though technically a dataset is any named grouping of records). We need to first grant access for given user/group to a particular Web service, and specify whether that user has CRUD (*create–read–update–delete*) rights in whole or part to interact with specified datasets within the knowledge base. It is in the nexus of user type, a tool (API), and dataset that we may establish access control for the semantic system.

In an enterprise context, a given individual (user) may have different access rights depending on the circumstance. A worker in a department may be able to see and do different things for departmental information than for enterprise information. A manager may be able to view budget information that is not readable by support personnel. A visitor to a different Web site or portal may see different infor-

[7] See further RFC 2828, "Internet Security Glossary," May 2000, *The Internet Society*, provided by the Internet Engineering Task Force (IETF) (see http://www.ietf.org/rfc/rfc2828.txt).

mation than visitors to other Web sites. Supervisors might be able to see and modify salary data for individual employees that is not viewable by others. The user role or persona thus becomes the access identifier to the system. As system managers, we define what information and what tools users might use for the datasets for which they have access.

The combination of datasets x tools x roles can lead to many access permutations. With, say, 20 tools with five different roles and just ten different datasets, we already have about 1000 permutations. As portals and dataset numbers grow, this combinatorial explosion gets even worse. Of course, not all combinations of datasets, tools, and roles make sense. In fact, only a relatively few number of patterns cover 95% or more of all likely access options. Because access rights are highly patterned, these theoretical combinations can, in fact, be boiled down to a small number of practical templates—which we call profiles—to which we may assign a newly registered dataset or user. (Of course, the enterprise could also tweak any of the standard profiles to meet any of the combinatorial options for a specific, unusual individual, such as for a tax auditor.)

Another enterprise consideration relates to training. Inter-team communications must be grounded in shared vocabulary and concepts. Even then, it is still necessary to continuously describe and explicate the benefits due to semantic approaches over conventional ones. Because of its general foundational nature, semantic approaches are often hidden or at the core of the information solution. It is not always self-evident what the advantages of semantic approaches are because their results can be mimicked via conventional approaches (though at a higher cost with greater brittleness). Semantic concepts are not (generally) intuitive to content editors, information architects, project managers, or fellow developers or project vendors. It is imperative to engage in continuous training and knowledge transfer during a semantic deployment. Unlike just a few years back, we no longer see resistance to open-source solutions. In fact, for early semantic adopters, open source is a positive feature. However, open source in a complicated enterprise environment comes with challenges. Support is often weak and integrating the pieces becomes one of the project responsibilities and risks. Open APIs and Web service endpoints still can lead to integration challenges. Encoding mismatches or how error messages get generated or treated, as two examples, point to some of the challenges in creating an integrated enterprise environment from multiple open-source pieces.

Enterprise funding is still another concern. Enterprise IT budgets have come under pressure. The justification for many projects resides in being able to offset annual licensing and maintenance fees, which can impose delivery constraints based on renewal dates. Existing enterprise IT budgets have also been made more incremental, with milestone achievements often required for moving forward. These trends are putting a premium on agile development and the need for enterprise-scale deployment and testing tools. Repeatable build processes and scripts are an essential component now for complex stack deployments.

Many of the issues that emerge in enterprise deployments are ancillary to or independent of specific knowledge components. Logging, testing, security, access, service buses, and deployment builds are an umbrella over entire deployments. In these regards, too, we must adhere to enterprise build practices and standards. The

frequency of repeating builds and testing means we need to create scripts for these steps and improve deployment documentation and practices. In these regards, knowledge and semantic technologies are no different from other components in the broader, enterprise-wide stack.

Another reality of semantic technologies in the enterprise is that few champions and advocates exist within many organizations. We must find means to communicate to semantic newbies and to enlist the aid of champions in carrying the message forward within the organization. In multi-vendor deployments, we should seek single points of contact able to communicate with their colleagues. In turn, the consumers of knowledge applications—namely subject matter experts, employees, partners, and stakeholders—now become the active contributors to the graphs themselves, focusing on reconciling terminology and ensuring adequate entity and concept coverage. Graph-driven applications mean that those closest to the knowledge problems will also be those directly augmenting the graphs. These changes act to democratize the knowledge function and lower overall IT costs and risks.

A Web-Oriented Architecture

Web-oriented architecture, or *WOA*, is a subset of the service-oriented architectural (SOA) style, wherein we package discrete functions into modular and shareable elements ('services') that we make available in a distributed and loosely coupled manner. WOA uses the representational state transfer (REST) style, geared to the HTTP hypertext transfer model. Roy Fielding defined the REST architectural style in his 2000 doctoral thesis [8]. Fielding is also one of the principal authors of the Hypertext Transfer Protocol (HTTP) specification. We couch WOA guidelines within the framework of a generalized *architectural style,* and while not limited to the Web are a foundation for it.

Nick Gall, a Gartner analyst, was one of the first to coin the WOA moniker. Gall describes WOA as based on the architecture of the Web as a "globally linked, decentralized, and [with] uniform intermediary processing of application state via self-describing messages." REST provides principles for how resources are defined and used and addressed with simple interfaces without additional messaging layers such as SOAP or RPC. REST and WOA stand in contrast to earlier Web service styles known by the WS-* acronym (such as WSDL). WOA has proven highly scalable and robust for decentralized users since all messages and interactions are self-contained (convey 'state'). It is not surprising that the largest existing knowledge networks on the globe—such as Google, Wikipedia, Amazon, and Facebook—are Web based. These pioneers have demonstrated the wisdom of WOA for cost-effective scalability and universal access.

We recommend a WOA architecture for knowledge management and representation purposes. Like the Internet itself, WOA has the advantage of being scalable and distributed, all (mostly) based on open standards. RESTful application programming interfaces (APIs) extend interoperability to outside systems and provide flexibility for swapping in new features or functionality as new components or developments arise. Under this design, all components and engines ('services')

become in effect 'black boxes,' with information exchange via standard vocabularies and formats using APIs as the interface for interoperability.

Web Orientation and Standards

Two main reasons, plus a host of others, justify basing our KM architecture on the Web. The first main reason is a crowning achievement of the semantic Web, which is the simple use of uniform resource identifiers (URIs, now internationalized to IRIs) to identify data. Further, if the resource identifier can resolve to a representation of that data, it now becomes an integral part of the HTTP access protocol of the Web while providing a unique identifier for the data. The HTTP protocol is the second main reason, through which we gain access to a global, distributed network. These innovations provide the basis for distributed data at a global scale, all accessible via Web devices such as browsers and smartphones that are now a ubiquitous part of our daily lives. The combination of RDF with Web identifiers also means that we may expose any information from a given knowledge repository and make it available to others as linked data. This approach makes the Web a universal database.

We often think of HTTP as a communications protocol, but it is much more.[8] It represents the operating system of the Web as well as the embodiment of a design philosophy and architecture. Within its specification lies the secret of the Web's success. REST and WOA quite possibly require nothing more to understand than the HTTP specification. HTTP provides the distinctions of GET and POST and persistent IRIs and the need to maintain stateless sessions with an idempotent design. HTTP also provides for content and serialization negotiation, and error and status messages for HTTP requests. HTTP also includes language, character set, encoding, serialization, and mime type enforced by header information and conformance with content negotiation; common and consistent terminology to aid understanding of the universal interface; a resulting component and design philosophy that is inherently scalable and interoperable; and a seamless consistency between data and services. CRUD is readily applicable to HTTP.

Besides these reasons, WOA is consistent with the many open Web standards we use in KBpedia and our platform designs. See further Chap. 9.

A Modular Web Service Design

I have emphasized two themes throughout this chapter. The first theme is to scope and bound functionality related to design needs. The second theme is to integrate these functions within current content workflows. We express these themes using individual RESTful Web services in our design, as exposed and accessed through

[8] The current specification is RFC 2616 (June 1999), which defines HTTP/1.1; see http://tools.ietf.org/html/rfc2616. For those wanting an easier printed copy, a good source in PDF is http://www.faqs.org/ftp/rfc/rfc2616.pdf.

their *application programming interfaces (APIs*). We have already seen how the WOA approach enables us to use the HTTP protocol for accessing RESTful Web services. The specific scoping and design of the functional modules provide the complementary part of the overall design. Since the resulting APIs are independent of any particular operating environment, we can reduce implementation costs for multi-platform user agents and promote the development of multi-platform services.

We determine the modularity of the services through analysis of the work tasks (see Fig. 12.1). Where appropriate, we embed these modules into other current applications or workflows (Fig. 12.2). Enterprise considerations such as security, access control, or workflow management enter in at this point to help complete the roster of desired services. These definitions help provide the boundary responsibilities of each Web service and what types of API instructions they may need. Platform-wide requirements, such as access control, must inform some of these needs.

We tend to follow a few guidelines in designing our Web services. We emphasize (1) use of a canonical, internal data representation format; (2) unit testing for all services; (3) attentiveness to error numbering and conformity of error messages, some of which we discover during testing; (4) similar granularity and order for specifying parameters across the APIs; (5) provision of online demo examples; (6) standard import and export formats; and (7) dual access to the API via SPARQL and programmatically. We tend to use a 'triples' or N3 RDF format for our internal canonical representation, which has a standard specification. (We also allow multiple import or export formats beyond the internal canonical form.) The provision for dual access to the APIs gives us the standard query basis of SPARQL, plus faster programmatic calls when using internal network transfers.

The size of payloads in both query results and as result set objects can be a challenge for RESTful Web services. Long HTTP queries with many parameter requests and large result sets can be a problem to handle, especially in the security layer. In some cases, we may need to look at ways to minimize and package (consolidate) parameter options to make endpoint requests more efficient. Encoding mismatches are a further challenge. It is best, for example, to adhere to a standard UTF-8 encoding via all semantic component interfaces. Consistent encoding requires attention and coordination on both sides of the interface and in tool use, especially the use of spreadsheets or CSV files.

The more fundamental challenge, however, is one of mindset. Effective interfaces require effective communications of the participating vendors across the boundary. The terminology, concepts, logic, and open-world approach to knowledge management and semantic technologies are not easily communicated nor immediately understood by traditional vendors. We must continuously work on communications to overcome past practices and embrace the flexibilities provided by semantic technologies.

REST Web services [9] and linked data are naturally compatible approaches. Linked data is a set of best practices for publishing and deploying data on the Web using the RDF data model. The data objects are named using Web uniform resource identifiers (IRIs), emphasize data interconnections, and adhere to REST principles. We also see the ideas of RESTful Web services morph into ones with more limited and targeted functionality. These microservices have a broad swath of definitions.

Some of the narrower ones, including in their ideas of choreographing and aggregating multiple small services, bear a close resemblance to the particular flavor of Web services that we recommend.

An Interoperability Architecture

Figure 12.3 presents our generic architecture for this WOA design. The three tiers of the system are content acquisition, repository, and content consumption:

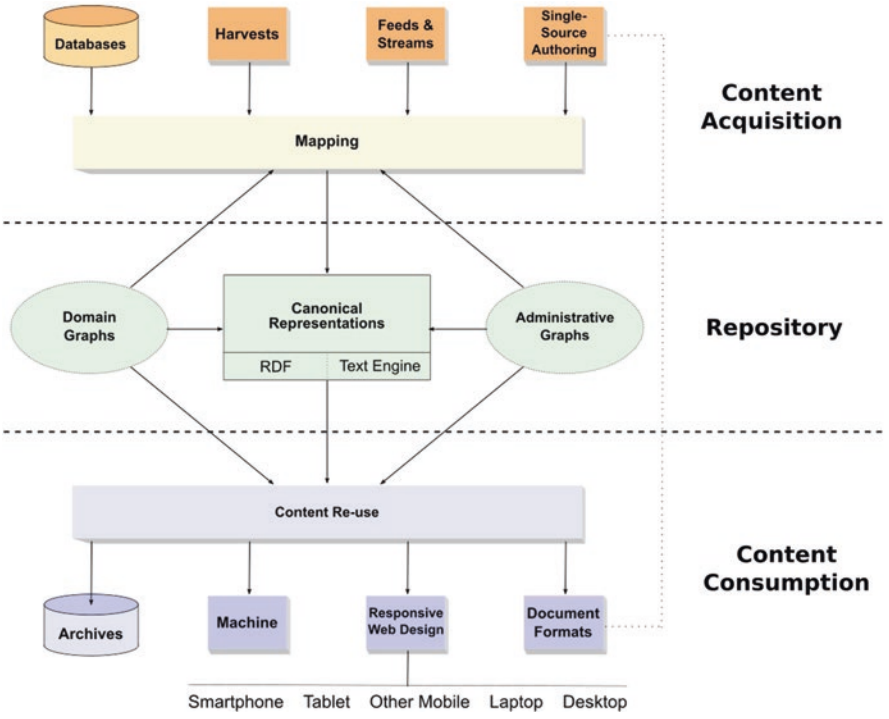


Fig. 12.3 An interoperability architecture

We have tended to abstract our WOA services into simple and compound ones (which are combinations of the simple). All Web services have uniform interfaces and conventions and share the error codes and standard functions of HTTP. We further extend the WOA definition and scope to include linked data, which is also RESTful. Thus, our WOA also sits atop an RDF (Resource Description Framework) database ('triple store') and full-text search engine.

The content acquisition tier is where all information comes into the system. For new sources, this involves mapping the concepts and other conformities to the existing knowledge graph. Already mapped sources and concepts require fewer integrity checks when we add instances or updates. Because we are using semantic technologies, we are agnostic as to the content source and can handle most any content. The content ingestion step is where we employ the limited number of canonical forms and use RDF as our data transfer model (see Chap. 9).

The repository tier is where the knowledge graph, knowledge base, triple store, OWL API, and full-text search engine reside. Most all knowledge management (KM) functions reside in this tier. All ontologies and their management reside at this tier. The full-text search engine and triple stores are mostly agnostic third-party systems. While some differences in open-source search engines and triple stores exist, we may plug most into the design. We have used Jena and Virtuoso as triple stores in the past, as well as the Lucene and Solr search engines. Many other options exist. Again, the main point here is to emphasize design principles over specific tools. Multiple tool options now exist for—and will continue to—the varied components of this architecture.

Many of the specialized work functions shown in the middle sections of Table 12.1 and Fig. 12.1 reside in the bottom (as shown in Fig. 12.3) content consumption tier. Within this tier, we may move some content to an archive data store, or we may transform subsets for machine learning purposes or to repurpose the existing content. Some of the transformations at this tier are merely transfer conventions with an external application. In addition to such tailored forms and their dedicated Web services, we also make available the general output in a variety of standard formats. Note that the content reuse and mapping layers, as well as the repository, use the internal canonical data representation.

References

1. T. Berners-Lee, O. Lassila, J. Hendler, The semantic web, in *Scientific American Magazine* (2001)
2. N.F. Noy, M. Sintek, S. Decker, M. Crubézy, R.W. Fergerson, M.A. Musen, Creating semantic web contents with Protege-2000. IEEE Intell. Syst. **16**, 60–71 (2001)
3. Y. Sure, M. Erdmann, J. Angele, S. Staab, R. Studer, D. Wenke, OntoEdit: Collaborative ontology development for the semantic web. The Semantic Web—ISWC **2002**, 221–235 (2002)
4. S. Bechhofer, I. Horrocks, C. Goble, R. Stevens, OilEd: a reasonable ontology editor for the semantic web, in *Working Notes of the 2001 International Description Logics Workshop (DL-2001)*. pp. 1–9
5. S. Bechhofer, R. Volz, P. Lord, *Cooking the Semantic Web with the OWL API* (Springer, International Semantic Web Conference, 2003), pp. 659–675
6. M. Horridge, S. Bechhofer, The OWL API: A Java API for OWL ontologies. Semantic Web **2**, 11–21 (2011)
7. M.K. Bergman, Large-scale RDF graph visualization tools, in *AI3:::Adaptive Information* (2008)
8. R.T. Fielding, *Architectural Styles and the Design of Network-Based Software Architectures*. Ph.D., University of California, Irvine (2000)
9. L. Richardson, S. Ruby, *RESTful Web Services* (O'Reilly Media, Inc., Sebastopol, CA, 2008)

Chapter 13
Building Out the System

So, you have looked at the evidence and the prospects, and are now ready to contemplate moving ahead seriously with a knowledge management installation. You have some hoped-for target goals in various kinds of analysis or data interoperability or knowledge-based artificial intelligence. Where does one begin? What is the plan? How can one proceed with initial implementation and testing to keep risks manageable and to demonstrate tangible benefits?

These are the topics of this chapter.[1] We begin by looking at what is involved in tailoring a new installation for specific domain purposes. We identify the checklist of items that you should consider for domain use. We discuss how to conduct an inventory of information assets that we might apply to the instance, and where external sources and information can contribute. We pay particular attention to how to construct a phased implementation plan based on our own experience with successful client projects and lessons learned.

We next discuss the critical work tasks of any new domain installation: the creation of the domain knowledge graph and its population with relevant instance data. We look at the state of the art in mapping methods and tools, and how we may apply those tools to these central tasks. We discuss methodologies and some of the publicly available databases—including those in KBpedia—that may be employed to help facilitate the new effort. We look at longer term extensions to the base installation that we may contemplate as the effort proceeds from a proof of concept to a full-blown knowledge management installation for the enterprise. These factors contribute to how we can make practical choices to proceed given limited time and budget.

[1] Some material in this chapter was drawn from the author's prior articles at the *AI3:::Adaptive Information* blog: "Open SEAS: A Framework to Transition to a Semantic Enterprise" (Mar 2010); "'Pay as You Benefit': A New Enterprise IT Strategy" (Jul 2010); "A Brief Survey of Ontology Development Methodologies" (Aug 2010); "A New Methodology for Building Lightweight, Domain Ontologies" (Sep 2010); "Research Shows Natural Fit between Wikipedia and Semantic Web" (Oct 2008); "Shaping Wikipedia into a Computable Knowledge Base" (Mar 2015); "Reciprocal Mapping of Knowledge Graphs" (Feb 2017).

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8_13

Besides the context of limited budgets, these efforts have high but uncertain expectations and a lack of trained creators and users of the system. We know that our efforts must meet the open-world nature of knowledge so that we can turn that fact to our advantage. It is just as defensible, and likely easier to implement and test, an incremental approach to our knowledge domain and data needs. With a predefined starting basis such as KBpedia, we can expand new portions with our domain scope and vocabulary in a piecemeal manner, tackling only the current new scope of the specific domain focus at hand, what we call 'pay as you benefit.'

Tailoring for Domain Uses

Our prior discussions of '*domain*' make clear that the size and scope of what it means are flexible, from the minute and focused on to the broad and general, for any branch of inquiry. Further, in a purposeful, incremental plan, the domain coverage should also grow and expand. That is one of the beauties of the open-world nature of knowledge.

Scoping the current domain of interest is thus a central task for any existing plan. Most of the implementation effort is to conceptualize (in a *knowledge graph*) the structure of the new domain and to populate it with instances (data). We also find that as our domain scope grows, so does the justification and need for more general knowledge management functions and applications. These general KM tasks, as well as increased maintenance and testing that should accompany any more widely used apps, should be added to the roster of task considerations as incremental plans move forward.

A Ten-Point Checklist for Domain Use

Each incremental expansion of the system, including the initial proof of concept, should consider, and incorporate as appropriate, specific points from a ten-point checklist:

1. Define potential scope, starting place; a starting place wants and needs champions; the scope is of much lesser importance.
2. Conduct inventory, interview stakeholders, and evaluate assets.
3. Develop a phased plan; budget, schedule, and staffing; ID analysis and testing; define platform and ontology scope and phasing.
4. Assemble assets (tools, data, structure, vocabularies).
5. Build and test domain ontology.
6. Build out platform.
7. Map and populate data.
8. Conduct and test target analysis.

9. Refine and use KM system.
10. Document and proselytize results.

You should consult this basic checklist for each increment of the plan. Some from this checklist may be active during any particular increment, and others not. However, these are the general task areas from which to construct the current increment of the plan. You may also need to formalize certain areas over time, such as documentation now exposing and describing workflows, or including deployment requirements.

An Inventory of Assets

Without exception, you must inventory your potential information assets for the installation. This inventory need not be exhaustive to start, just relevant and applicable to the particular domain space that is your starting or expansion scope. Think about what an information asset in this space is, and how one finds and uses such information. Since knowledge workers know their information assets, an essential and integral requirement is to interview users when assembling this inventory. It is with these same interviews that we identify and assemble the domain vocabulary. Discussions should also help identify early champions and possible project team members.

Recall that our system is capable of handling text and documents (*unstructured data*), markup documents and attribute-value pairs (*semi-structured data*), and *structured data* (database, spreadsheets, tables). Evaluate this possible content for consideration as part of the TBox knowledge graph or as new instance data (ABox). Different tests and checks apply to concepts and instances. Concept data comes from glossaries, tables of content, thesauri, sometimes bullet lists, or more formal schema, such as hierarchies in spreadsheets or relational schema or ontologies. Text definitions, or links to encyclopedias, or links to specific Web pages, may be desirable content to add to the characterizations. You may uncover instance data in infoboxes, spreadsheets, data tables, or text records with fixed fields. You should inspect the record form to identify the types to which the instances belong and their attributes. Favor complete structure, but gaps are OK given the open nature of knowledge. Text in the form of labels or *semset* entries that can accompany instance entries is desirable.

I do not advise beginning a KM project premised on paper conversion to digital. All first-iteration sources should be electronic, with the possible exception of subsumption hierarchies that you might obtain from paper listings or tables of content. If essential data only resides on paper, this kind of task should only be tackled in later increments after the basic system has justified itself. In the earliest phases of a project, avoid unusual formats or data that requires much wrangling or cleaning to stage for ingesting. Again, if essential, such sources can be tackled in later phases.

I do not advise beginning a KM project where security and access are a concern. I do recommend that proprietary and restricted access content be included in the initial inventory and interviewing steps to demonstrate the point that semantic KM systems do not depend on open data. Early designs can anticipate possible security expansions, even though you may defer specifics and implementation. Since each implementation increment of the plan involves a new boundary for the domain, it is also appropriate that an updated inventory be conducted for each new phase, perhaps putting on to the table sources that you chose to skip over in earlier phases.

These all constitute possible domain extensions. However, KBpedia and its 55,000 reference concepts and millions of organized instance data are also available for free. Many of these also have links to text entries on Wikipedia or Wikidata, supplying that valuable content form. You may already find much, if not perhaps nearly all, of a starting skeleton for a given domain in the KBpedia structure.

Phased Implementation Tasks and Plan

Too many KM and business intelligence (BI) projects in the past have failed. The relational schema and its closed logic and brittleness have been one contributor to this record. Another reason for failure has been too ambitious scope or expectations. By embracing open approaches to knowledge, we can also open up a development path that is phased and incremental. We can let the experience and results of prior phases justify new phases and expansions.

This philosophy fits well with a proof-of-concept approach, followed by staged and managed extensions. Repeating methods and continuing to refine tools as part of this phasing means we are climbing learning curves as more knowledge workers become exposed and facile in the use of the system. Expanding use and input helps provide continued knowledge and feedback into the plan and how we execute each incremental phase.

In a proof-of-concept phase, the least-effort path would be to leverage KBpedia or portions of it as is, make few changes to the knowledge graph, and populate and test local instance data. A next step may be to expand the knowledge graph with still more instances. As increments occur, consider more KM infrastructure for the system to accompany the expansion of domain scope.

Over time the plan should reflect its content and management pipeline. It is important to design the ability to swap in and out various options at multiple points from input to desired output. Then, because disparate sources and different formats must be accommodated, it is also important to use canonical syntaxes and standards for expressing the products and specifications at the various steps along that pipeline. The very notion of pipeline implies workflows, which are the actual drivers for how we design the pipeline. Evolve key workflow steps to include:

- Clean the input sources.
- Express the sources in a canonical form [1].

- Identify and extract concepts.
- Map the structure to KB concepts.
- Identify and extract entities.
- Identify and extract relations.
- Type the entities, concepts, and relations.
- Extract attributes and values for identified entities.
- Add new import and export formats according to the needs of data interoperability and use of third-party analyses, machine learners, and tools.
- Test these against the existing KB.
- Update reference structures, including placement of the new assertions, as appropriate.
- Characterize and log to files.
- Commit to the KB, perhaps through formalized deployment steps.
- Rinse and repeat.

Much information gets processed in these pipelines, and the underlying sources update frequently. Thus, the pipelines themselves should be designed for performance and based on solid code with appropriate workflow tagging and management.

Automation, within the demanding bounds of quality, is also an essential scalable condition. Functional programming languages align well with the data and schema in knowledge management functions. Ontologies, as structures, also fit well with functional languages. The ability to create domain-specific languages (DSLs) should continue to improve bringing the knowledge management function directly into the hands of its users, the knowledge workers. An essential design criterion is to have a methodology and workflow that explicitly accounts for interoperable and straightforward tools, following the scoping guidelines discussed in the previous chapter. You may need to include and justify specific tasking for any of these aspects in a given plan increment.

Over the timeframe of multiple increments for a phased project, consider clusters of work tasks to be drawn upon for next increments.

Domain Knowledge Graph

You may start with KBpedia, though eventually it is desirable to move toward a tailored domain knowledge graph. You may proceed to create the domain knowledge graph from prunings and additions to the base KBpedia structure, or from a more customized format such as the approach recommended in *Ontology Development 101* [2]. Some of your tasks in this area are to determine the domain and scope of the ontology; incorporate domain terminology; consider reusing existing ontologies; enumerate important terms in the ontology; define the types and the class hierarchy, especially into typologies; and define the attributes of the types. After providing a preferred label, I encourage you to seek relevant alternative labels (for building the semsets).

The build methodology should reuse 'standard' ontologies as much as possible, to help promote interoperability. The 20 or so core and extended ontologies mapped to KBpedia are one starting point. To this base, you should add other commonly used ontologies or those specific to your domain. You should make identification of these candidates an explicit part of the information inventory efforts. At a minimum, the ongoing working knowledge graph should conform to ontology building best practices (see Chap. 14) and complete enough such that it can be loaded and managed in an ontology editor or IDE. You can use this working structure with the OWL API for specialty tools and user maintenance functions.

Instance Data Population

Identifying, staging, transforming, incorporating, and vetting new instance data should be a continuous set of tasks for the installation. It is less risk to start with simple data formats populated with clean data. I suggest you cluster new, desired interfaces or translators with expansions into entirely new sources of instance data, such as from external sources or relational databases. Considerations like this can spread needed development and tests over a complete project. A proper inventory of information assets will include file types and possible conversion tools for those types that may exist in the marketplace, preferably as open source. For example, a single conversion to the system's canonical format for a tool such as Tika can open up a thousand new data formats to the system. As a general guideline, it is of much less effort and cost to investigate existing, available options, and then to adapt them to our data federation design, than it is to write converters from scratch.

For relational systems with large data stores, it may be justified to use third-party commercial tools for initial staging and conversion. We have had excellent experience with tools such as Safe Software's FME; many options exist for high-throughput situations or where updates are frequent.

Analysis and Content Processing

Each increment should target some form of analysis or content processing as its design objective. From the platform perspective, that means being able to select appropriate subsets from the knowledge base, process or transform them in some way, and then submit those result sets to an external tool to conduct the designated work. Per the design philosophy, transformations or submittals of result sets should occur via an adequately scoped Web service. You should identify each new tool required for a given design objective, with integration part of the new tasking. Internal communications should also conform to the canonical data form. Some tasks may also require injecting analyzed results back into raw Web pages for display or visualization. Other tasks may need to expand Web pages to enable control and setting of tool parameters. You can also convert or export the information in various forms for direct use or incorporation into third-party systems.

You may drive visualization systems and specialized widgets using the result sets obtained from such queries or analysis, in which case you should include such in the task list. Our methodology also provides for administrative ontologies whose purpose is to relate structural understandings of the underlying data and data types with suitable end use and visualization tools. You may therefore also need to consider tasks related to creating or modifying the administrative ontologies.

Use and Maintenance

The emerging knowledge system has practical uses including search, querying, filtering, discovery, information federation, data interoperability, analysis, and reasoning. During use, you may discover many enhancements and improvements. Examples include improved definitions of concepts; expansions of synonyms, aliases, and jargon (*semsets*) for concepts and instances; better, more intuitive preferred labels; better means to disambiguate between competing meanings; missing connections or excessive connections; and splitting or consolidating of the underlying structure. We want to see an evolution of tooling and incorporation into existing workflows such that we make these enhancements as encountered and without major work disruption. Today, practitioners most often do *not* pursue such maintenance enhancements because existing tools do not support such actions. Users and practitioners do not respond well to IDEs and tools geared to ontology engineering. A-start-small strategy, of course, lowers risk and is more affordable. However, for effectiveness, you must design an explicit strategy anticipating extension and expansion. Ontology growth thus occurs both from learning and discovery and from expanding the scope. Versioning, version control, and documentation (see below) therefore assume central importance as the system grows. Any of these items may form a nexus for work tasks in a given increment of the plan.

Testing and Mapping

As we generate new ontologies, we should test them for coherence against reasoning, inference, and other natural language processing tools. We also use gap testing to discover holes or missing links within the resulting ontology graph structure. Gap testing helps identify internal graph nodes needed to establish the integrity or connectivity of the concept graph. We may use coherence testing to find missing or incorrect axioms. Though used for different purposes, we may also use mapping and alignment tools to identify logical and other inconsistencies in definitions or labels within the graph structure. Mapping and alignment help establish the links that help promote ontology and information interoperability. We ask external knowledge bases to play crucial roles in testing and mapping. Depending on the phase, you may need to include such tasks for a given plan increment. Mapping is not always a part of a given increment. However, testing should be a part of all of your increments. Include unit tests for all new tools and converters or further target analyses.

Documentation

Ontologies give us a way to capture the structure and relationships of a domain—which is also always changing and growing. We can further use ontologies to document their development and versions. We need to apply better tools—such as vocabulary management and versioning—and better work processes to capture and record use of our ontologies. We can handle some of these aspects with utilities such as OWLdoc or wikis for standard knowledge capture and documentation. We have innovated many connectors to capture ontology knowledge bases on an ongoing basis. Still, these are rudimentary steps that we need to enforce with management commitment and oversight. Ongoing use and training demand that we adequately document the knowledge graphs, ontologies, tools, scripts, and instance record sources that support a given knowledge installation. Given the lack of tools or best practices in this area, you will need to commit to and monitor documentation.

Mapping Schema and Knowledge Bases

Two critical work areas in tailoring your implementation are in building out the schema (knowledge graph) and populating your installation with instance data. Various mapping methods and tools aid these two work areas. Given their importance, let's spend a bit of time discussing these work areas in more detail.

Mapping Methods and Tools

Mapping is the definition of a formal correspondence of objects in one knowledge source with objects in another knowledge source, with the latter most often being the reference knowledge graph. The correspondence takes the form of assigning a specific predicate linking an object in an external source to its subject in a reference source. Some mapping is straightforward; other mappings may be quite hard due to the vagaries of language and context. Mapping involves specific methods and algorithms to propose candidate matches, as well as tools or applications that embed these methods in user interfaces and workflows, often with the intent of supporting the broader mapping purpose. By making the reference knowledge graph the target, we only need to test the updated graph for coherency and consistency. The reference knowledge graph grows and changes shape and scope over time as new domain information is incorporated. Properly mapped external sources can become an integral part of the domain knowledge graph and participate in inferencing and other reasoning tasks.

Though some tout complete automation of mapping as desirable, there is no such thing, and even small assignment error rates can translate into noticeable errors in

the knowledge base.[2] For this reason, we support what we call a 'semi-automatic' approach to mapping. The method involves using multiple methods to score potential matches, perhaps differentially weighted, and ultimately reviewed and vetted by human editors before acceptance into the system. The individual review steps are what make the approach 'semi-automated,' though to make that process efficient it is also useful to automate away clear mismatches and other problems before the human review of candidates. By automating the process to reduce easily recognized noncandidates and score only candidates via the differing methods, we can reduce the number of uncertain candidates editors need to review. We can also apply this method for screening candidates for supervised machine learning. Efficiencies and learning curves should be fed back into the screening tools so that reviewers believe that their input is valued and gets reflected in constantly improving tools, two unarguable objectives when mounting a knowledge management initiative.

The mapping methods are varied and tend to reflect the same broad clusters of semantic heterogeneities as provided by Table 5.1. We may use various ways to classify these mapping types, but the central options tend to focus on areas in Table 5.1 such as schema, labels/lexical, labels/definitions/semantics, instances, relations, machine learning, or mediated, by using external KBs or thesauri or WordNet. The most straightforward approaches look only at labels and propose various kinds of string matches. Better ones look at attributes, external relations, subsumption hierarchies, and semantics of labels and concepts. Some of the tools provide multiple methods, and the user may combine or not multiples of them with user-assigned weights.

The heyday of tools developed in the areas of *ontology alignment*, or *mapping*, or *matching*, was in the 2000 to 2005 timeframe. Still, the sophistication and usability of these tools have continued to improve, even if the pace of new offerings has slowed. A major driver for these advances has been the annual OAEI (Ontology Alignment Evaluation Initiative) conference, which has provided a competitive contest and established evaluation test sets and criteria on a yearly basis since 2004. My recent survey specific to ontology mapping identified 30 different existing mapping tools, many embracing multiple methods, and most open source [3].

Building Out the Schema

If you ask most knowledgeable enterprise IT executives what they understand ontologies to mean and how to use them, you would likely hear that ontologies are expensive, complicated, and challenging to build. Reactions such as these (and not trying to set up strawmen) result from the relative lack of guidance on how one builds and

[2] As Chap. 14 explains, an F1 score of 95% is still based on an annotator agreement basis of perhaps 70–80%, which means an actual F1 score error rate of, say, 65%. With ten million assertions, this translates into as many as 3.5 million being in error. If actual F1 score is at 95%, that still means 500,000 errors.

maintains these beasties. The use of ontology design patterns is one helpful approach. Such patterns help indicate best design practice for particular use cases and relationship patterns. However, while such patterns should be part of a general methodology, they do not themselves constitute a methodology.

The focus here is on *domain ontologies*, which are descriptions of particular subject or domain areas. The last known census of ontologies in 2007 indicated that there were more than 10,000 then in existence, though today's count is likely in excess of 40,000.[3] Because of the scope and coverage of these general and domain representations, and the value of combining them for specific purposes, *ontology alignment* has been a topic of practical need and academic research. According to Corcho et al. [4] "a domain ontology can be extracted from special purpose encyclopedias, dictionaries, nomenclatures, taxonomies, handbooks, special scientific languages (say, chemical formulas), specialized KBs, and from experts." Another way of stating this is to say that a domain ontology—adequately constructed—should also be a faithful representation of the language and relationships for those who interact with that domain.

Overview of Approaches

There is a spectrum of approaches for how to conduct these mappings. At the simplest and least accurate end of the spectrum is string-matching methods, sometimes supplemented by regular expression processing and heuristic rules. An intermediate set of methods use concepts already defined in a knowledge base as a way to 'learn' representations of those concepts; while many techniques exist, two common ones are explicit semantic analysis and word embedding. Most of these intermediate methods require some form of supervised machine learning or other ML techniques. At the more state-of-the-art end of the spectrum are graph embeddings or deep learning, which also capture context and conceptual relationships as codified in the graph.

Aside from the string-match approaches, all of the intermediate and state-of-the-art methods use machine learning. Depending on the method, these machine learners require developing either training sets or corpora as a reference basis for tuning the learners. These references should be manually scoped, as in the case of training corpora for unsupervised learning, or manually scored into true and false positives and negatives (labeled results) for training sets for supervised learning. All of these techniques are useful, but you should, in any case, supplement them with logic tests and scripts to test coherence and consistency issues that may arise. You should test the coherency of the target knowledge graph after any new mappings.

Practitioners of ontology development have been documenting approaches since at least Jones et al. in 1998 [5]. That early study outlined common steps and noted

[3] A simple Web search of https://www.google.com/search?q=filetype:owl (OWL is the Web Ontology Language, one of the major ontology formats) shows nearly 39,000 results. Still, multiple ontology languages are available, such as RDF, RDFS, and others (though use of any of these languages does not necessarily imply that the artifact is a vocabulary or an ontology).

typical stages to produce first an informal description of the ontology and then its formal embodiment in an ontology language. The existence of these two descriptions is an important characteristic of many ontologies, with the informal description often carrying through to the formal description. Corcho et al. did the next major survey in 2003 [4]. This built on the earlier Jones survey and added more recent methods. The survey also characterized the methods by tools and tool readiness. More recently the work of Simperl and her colleagues has focused on empirical results of ontology costing and related topics. This series has been the richest source of methodology insight in recent years [6–8]. Though not a survey of methods, one of the more attainable descriptions of ontology building is Noy and McGuinness' well-known *Ontology Development 101* [2].

Another way to learn more about ontology construction is to inspect some existing ontologies. Though one may use a variety of specialty search engines and Google to find ontologies,[4] some current repositories also deserve inspection. Examples include the University of Manchester, VIVO, TONES, the Protégé ontology library, the Linked Open Vocabularies (LOV), the NanJing Vocabulary Repository, the Online Ontology Set Picker (OOSP), and the OBO (biomedical) Foundry. An older, but similar, repository is OntoSelect. Another way to learn about ontology construction is from a bottom-up perspective. In this regard, the Ontology Design Patterns (ODP) wiki is a source of building patterns and exemplary ontologies.[5] ODP is not likely the first place to turn to and does not give 'big picture' guidance, but it also should be a bookmarked reference once you begin real ontology development.

For the last 20 years, there have been many methods put forward for how to develop ontologies. Though new methodology developments have diminished somewhat in recent years, our reviews suggest that this is the current state of ontology development methodologies:

- Very few uniquely different methods exist, and those that do are relatively older in nature.
- The methods tend to cluster into either incremental, iterative ones or those more oriented to comprehensive approaches.
- There is a general logical sharing of steps across most methodologies from assessment to deployment and testing and refinement.
- Actual specifics and flowcharts are quite limited; except the UML-based systems, most appear not to meet enterprise standards.

[4] Specialty search engines for ontologies include Swoogle, FalconS, Watson, Sindice, and SWSE. In addition, one can use a general search engine such as Google with a search query such as <topic> owl:equivalentClass filetype:owl. Note that the filetype might also include RDF or a variant such as N3; we can substitute other language-specific constructs of interest for owl:equivalentClass.

[5] OntologyDesignPatterns.org (http://ontologydesignpatterns.org/wiki/Main_Page) is a semantic Web portal dedicated to ontology design patterns (ODPs). The portal was started under the NeOn project in 2009.

- Discussion of supporting toolsets is often lacking, and most of the examples, if even provided, are based solely on a single or governing tool. Tool integration and interoperability are almost nonexistent in narratives.
- Development methodologies are not as active an area of recent research.

While there is by no means unanimity in the community, we can see some consensus from these prior reviews [9]. We have taken these consensus items and added to them some points from our experience, resulting in these eight general guidelines for what you should consider in a domain ontology:

- Be lightweight and modular.
- Use reference structures.
- Reuse existing structure.
- Build incrementally.
- Use simple predicates.
- Test for logic and consistency.
- Map to external ontologies.
- Map reciprocally.

I expand further on these points in the next sections.

Some Design Guidelines

Effective ontology development is as much as anything a matter of mindset. This mindset is grounded in leveraging what already exists, 'paying as one benefits' (see below) through an incremental approach, and starting simple and adding complexity as we gain understanding and experience. Inherently this approach requires domain users to drive ongoing development with appropriate tools to support that emphasis. Ontologists and ontology engineering are important backstops, but not in the lead design or development roles. The net result of this mindset is to develop pragmatic ontologies that are understood—and used—by actual domain practitioners. Let's look more closely at the individual design guidelines just listed to see what goes into this mindset.

Be Lightweight and Modular

Begin with a *lightweight, domain ontology* [10], which is hierarchical or classificatory in nature. Ontologies built for the pragmatic purpose of interoperating different contexts and data should start lightweight with only a few predicates, such as `subClassOf`, `isAbout`, `narrowerThan`, or `broaderThan`. If done properly, these lighter weight ontologies with more limited objectives can be surprisingly robust in discovering connections and relationships. Moreover, they are a logical and doable intermediate step on the path to more demanding semantic analysis. Because we have this perspective, we also tend to rely heavily on the SKOS

vocabulary for many of our ontology constructs [11] and use *typologies* in our overall design.

Provide *balanced coverage* of the subject domain. The breadth and depth of the coverage in the ontology should be roughly equivalent across its scope. Build *modular* ontologies that split your domain and problem space into logical clusters. Try to *split domain concepts from instance records structurally*. Concepts represent the nodes within the structure of the ontology (also known as classes, types, or the *TBox*). Instances represent the data that populates that structure (also known as entities, individuals, or the *ABox*). Use *disjoint classes* to separate classes from one another where the logic makes sense, and let dissimilarities guide the bounding of types in the first place. An architecture of multiple ontologies often works together to isolate different work tasks to aid better ontology management. Also, try to use a core set of *primitives* to build up more complex parts. This approach is a kind of reuse within the same ontology, as opposed to reusing external ontologies and patterns. The corollary to this is that the same concepts should not be created independently multiple times in different places. Adhering to these practices is akin to object-oriented programming.

Try to think of your knowledge graph as also providing context, by explicitly considering what the best way is to describe what your content 'is about.' A good gauge for whether the context is adequate is whether one has sufficient concept definitions to disambiguate common concepts in the domain. As we add relationships and the complexities of the world get further captured, ontologies migrate from the lightweight to a more 'heavyweight' end of the spectrum.

Use Reference Structures

One benefit is that reference structures of any kind provide a focus, by definition, of common or canonical referents. This commonality leads to better defined, better understood, and more widely used referents. Common referents become a kind of common vocabulary for the space, upon which other vocabularies and datasets can depend. A common language, of sorts, can begin to emerge. Reference structures also provide a grounding, a spoke-and-hub design, that leads to an efficient basis for external vocabularies and datasets to refer to one another. Of course, any direct mapping can provide a means to relate this information, but such pairwise mappings are not scalable nor efficient. In a spoke-and-hub design, the number of mappings required goes down significantly with the number of datasets or items requiring mapping. The spoke-and-hub design,[6] for example, is at the heart of such disciplines

[6] The main advantage of a grounding reference is that it allows a spoke-and-hub design for data mapping, which is tremendously more efficient than pairwise mappings. In a spoke-and-hub design, where the reference ontology is the common node at the hub, only $n - 1$ routes are necessary to connect all sources, meaning that it scales linearly with the number of sources and attributes. Without a grounding reference, these same mapping capabilities would require routes in a pairwise (point-to-point) approach, which also scales poorly as a quadratic function. A system of ten datasets would require $n(n - 1)/2$ composite mappings in the reference grounding case, but

as master data management. Another benefit of common reference structures is that they provide a common target for the development of tools and best practices. These kinds of 'network effects' lead to still further tooling and practices.

Reuse Existing Structure

Reuse structure and vocabularies as much as possible. Fundamental to the whole concept of coherence is the fact that domain experts and practitioners have been looking at the questions of relationships, structure, language, and meaning for decades. Massive time and effort have already been expended to codify some of these understandings in various ways and at multiple levels of completeness and scope. A short list of these potential sources demonstrates the treasure trove of structure and vocabularies available to any enterprise for reuse: Web portals; databases; relational database schema; industry specifications and standards; spreadsheets; informal lists; legacy schema; metadata; taxonomies; controlled vocabularies; ontologies; master data (MDM) directories and catalogs; exchange formats, etc. Metadata and available structure may have value no matter where or how it exists, and a fundamental aspect of the build methodology is to bring such candidate structure into a standard tool environment for inspection and testing. It is wasteful to ignore prior investments that have been used to characterize or organize information assets.

We closely relate this guidance to our earlier advocacy that you should accompany each incremental phase of development with an update to the information inventory. The most productive methodologies for modern ontology building are those that reuse and reconcile prior investments in structural knowledge, not ignore them. These existing assets take the form of already proven external ontologies and internal and industry structures and vocabularies. Besides assembling and reviewing current sources, those selected for reuse must be migrated and converted to a proper ontological form (OWL in our case). Others have demonstrated some of these techniques for prior patterns and schema [10, 12]. In other instances, you may employ various converters or scripts to conduct the migration. Many tools and options exist at this stage, even though as a formal step practitioners often neglect this conversion.

Build Incrementally

Build ontologies *incrementally*. Much value can be realized by starting small, being simple, and emphasizing the pragmatic. It is OK to make those connections that are doable and defensible today while delaying until later the full scope of semantic complexities associated with complete data alignment. An open-world approach

45 in a pairwise approach. Of course, datasets themselves contain tens to thousands of attributes, compounding the map scaling problem further.

provides the logical basis for incremental growth and adoption of ontologies. You need to repeat the process of modifying a working ontology, testing it, maintaining it, and then revising and extending it over multiple increments. In this manner, the deployment proceeds and gets refined as learning occurs. Importantly, too, this approach also means that complexity, sophistication, and scope only grow consistent with demonstrable benefits. Thus, in the face of typical budget or deadline constraints, you may initially scope domains smaller or provide less coverage in depth or use a smaller set of predicates, all the while still achieving productive use of the ontology.

Use Simple Predicates

Define unambiguous *predicates* (also known as properties, relationships, attributes, edges, or slots), including a precise definition. Then, when relating two things to one another, use care in actually assigning these properties. Initially, assignments should start with a logical taxonomic or categorization structure and expand from there into more nuanced predicates. Though not involved in any reasoning, aggressively use *annotation properties* to promote the usefulness and human readability of the ontology, as well as to provide text support for the better characterization of entities and concepts.

Assign *domains* and *ranges* to your properties. Domains apply to the subject (the left-hand side of a triple), and ranges to the object (the right-hand side of the triple). You should not view domains and ranges as real constraints, but as axioms used by reasoners. In general, the domain for a property is the range for its inverse and the range for a property is the domain of its inverse. (You can envision this by understanding that domain applies to the subject, while range applies to the object. If you invert these roles, domain and range switch.) Use of domains and ranges will assist testing and help ensure the coherency of your ontology. Assign *property restrictions*, but do so sparingly and judiciously. Use of property restrictions will also support testing and provides possibly new features to machine learners.

Test for Logic and Consistency

We must always test our knowledge graphs for logic, consistency, completeness, and coherence. Test each increment; no official or public release should be made that does not pass all tests. As we learn, we should continue to add to the comprehensiveness of our tests. We test logic as we build with inference engines and reasoners. We look for completeness and consistency regarding standard ontology errors, such as what the tool OOPS! helps identify [13], and follow our best practices for completeness and the use of *semsets*.

The essence of *coherence* is that it is a state of logical, consistent connections, a logical framework for intelligently integrating diverse elements. So while context supplies a reference structure, coherence means that the structure makes sense. Is

the hip bone connected to the thigh bone, or is the skeleton askew? Coherence means that we draw the right connections (edges or predicates) between the right object nodes (or content) in the graph. Relating content coherently itself demands a coherent framework. At the upper reference layer, this begins with KBpedia, which begins as a coherent structure. If KBpedia continues as the basis for the modified domain ontology, and if incremental changes are tested for logic and consistency as they occur, then you should be able to continue to evolve the domain knowledge graph coherently. Absent starting reference structures, it is tough to create a cohesive starting knowledge graph, since any new assertion may not have been encountered in a related form before.

Map to External Ontologies

Mapping to external ontologies increases the likelihood of sharing and interoperability, but importantly from an ontology building perspective also helps to identify gaps or errors in the reference knowledge graph. Mapping helps expose the importance of 'punning,' since depending on use or context we may want to treat a given concept as either a class or an instance. Given our domain and our interoperability goals, we likely want to rely on a *set of core ontologies* for external reuse purposes. For interoperability purposes, we also want to write our ontologies in *machine-processable languages* such as OWL or RDF Schema.

Building Out the Instances (Knowledge Bases)

The conceptual and logical demands for adding instances are different in scope and kind than those for the conceptual knowledge graph. When adding instances, ensure the quality of the input data with reliable provenance; you may be required to justify your sources. An attributes ontology, embedded as one of the backbones in KBpedia, is a useful starting place to map data attributes and characteristics. We grow and mature the reference structure for this using similar considerations as to what we followed for the overall knowledge graph, including logic and consistency tests (though they will be of a different character, more akin to data validation). When adding instances, it is essential you relate all entities to a type and pay attention to other aspects of the instance's data record that may be useful to include as disambiguation cues.

In building out and then using instance data, we can see a cycle of ten or so broad guidelines. Note that I refer to the input instance source as a knowledge base, though, of course, any instance data repository may be a source. A relational data store, for example, would follow these guidelines, but also would need to go through

some form of relational to RDF converter. Other types of data stores may impose similar wrinkles.

Here are the ten guidelines for building out instances:

Update Changing Knowledge

We need to ensure that the input knowledge bases to the overall domain knowledge structure are current and accurate. Don't start with dated material! Depending on the nature of the KM system, there may be multiple input KBs involved, each demanding updates. Besides capturing the changes in the base information itself, many of the steps below may also be required to process this changing input knowledge correctly.

Process the Input KBs

Process the input KBs to be machine readable. We also desire processing to expose features for machine learners and to do other cleanup of the input sources, such as removal of administrative categories and articles, cleaning up category structures, consolidating similar or duplicative inputs into canonical forms, and the like. This step is highly contextual, and may require multiple steps or scripting.

Install, Run, and Update the System

The KBs themselves reside on their host databases or triple stores. Each of the processing steps may have functional code or scripts associated with it. All general management systems should be installed, kept current, and secured. The management of system infrastructure sometimes requires a staff of its own, let alone install, deploy, monitoring, and update systems. It is here that we may need to add specific source converters to the system.

Test and Vet Placements

New entities and types added to the knowledge base should be placed into the overall knowledge graph and tested for logical placement and connections. Though we should manually verify final placements, the sheer number of concepts in the system places a premium on semi-automatic tests and placements. Placement metrics are also valuable to help screen candidates. This task area requires similar tools and user interfaces, plus incorporation into existing workflows, as is required for concept placements into the governing knowledge graph.

Test and Vet Mappings

If we add new types or concepts to the governing knowledge graph, then these should be tested and mapped with appropriate mapping predicates to external or supporting KBs. Any new mappings to the base KB should be reinvestigated and confirmed.

Test and Vet Assertions

Testing does not end with placements and mappings. Attributes and values often characterize concepts; sometimes we may give them internal assignments as SuperTypes; and we must test all new assertions against what already exists in the KB. Though the tests may individually be straightforward, thousands may require testing, and cross-consistency is vital if one is adding large instance stores. Each of these assertions is subject to unit tests.

Ensure Completeness

Our standard practice calls to accompany each new concept in the KB with a definition, complete characterization, and connections, and synonyms or semsets to aid in natural language tasks. If updates are periodic or scheduled, as opposed to one-time batch incorporation, then we recommend writing scripts for the appropriate tests. Any activity that we can reasonably anticipate to occur three times or more deserves scripting attention.

Test and Vet Coherence

As we build and extend the broader structure, we apply system tests to ensure that the overall graph remains coherent. We address and correct outliers, orphans, and fragments when encountered. We do some of this testing via component typologies, and some we do using various network and graph analyses. You should flag possible problems and document or present them for manual inspection. Like other manual vetting requirements, confidence scoring and ranking of issues and candidates help speed up this screening process.

Generate Training Sets

A key objective of populating our knowledge system with instance data is to enable the rapid creation of positive and negative training sets for machine learning. We need to generate candidates; they should be scored and tested; and we need to vet their final acceptance. Once vetted, we may need to express the training sets in

different formats or structures (such as finite-state transducers, one of the techniques we often use) for them to perform well in actual analysis or use. Since machine learners may require many iterations to refine input parameters, your scripting attention is certainly required here.

Test and Vet Learners

We can then apply machine learners to the various features and training sets produced by the system. Each learning application involves the testing of one or more learners; the varying of input feature or training sets; and the testing of various processing thresholds and parameters (including possibly white and blacklists). This set of requirements is one of the most intensive on this listing, and requires you to document test results, alternatives tested, and other observations useful to a cost-effective application.

Rinse and Repeat

Each of these ten steps is not a static event. Instead, given the constant change inherent in knowledge sources, including the ongoing addition of new instances, we must repeat the entire workflow on a periodic basis. The inexorable pull is to automate more steps and generate more documentation to reduce the tension between updating effort and current accuracy. A lack of automation leads to outdated systems because of the effort and delays in updates. The imperative for automation, then, is a function of the change frequency in the input KBs or the use of learners.

Pay as You Benefit

As best as I can tell, Alon Halevy was the first to use the phrase 'pay as you go' in 2006 to describe the incremental aspect of the open-world approach applied to the semantic Web [14]. Others had earlier applied the 'pay as you go' phrase to data management and storage; it had also been used to describe phone calling plans. Unfortunately, the 'pay as you go' phrase has (and still is) largely confined to incremental, open-world approaches involving the semantic Web. Nonetheless, I like the phrase, and I think it evokes the right mindset. In fact, I think with linked data and many other aspects of the current semantic Web we see such approaches come to fruition. Inch by inch, brick by brick, we see useful data on the Web getting exposed and interlinked. 'Pay as you go' is incremental, and that is good.

Still, I think we can express this idea better. The idea of 'pay as you benefit' more directly ties the question of project funding and project staging to project benefits. It ties directly into the open nature of knowledge and dovetails nicely with the repeated recommendations to implement your knowledge management initiatives

incrementally. The idea of 'pay as you benefit' is purposeful, and may be planned and implemented on standard enterprise cost-benefit principles.[7] What the 'pay as you benefit' idea means is you can start small and be incomplete. You can target any domain or department or scope that is most useful and illustrative for your organization. You can deploy your first stand-ups as proofs of concept or sandboxes. Moreover, you can build on each prior step with each subsequent one. Of course, you must communicate with stakeholders to get this message out and to overcome the glazed eyes that might accompany the terminology of knowledge management and ontologies. 'Pay as you benefit' is a guiding pragmatic principle for how you can build out your domain knowledge management system. So, how does one move ahead with a 'pay as you benefit' strategy?

Placing the First Stake

The first step is always the hardest on a new journey. We can minimize risk by planning an incremental rollout and scoping and bounding our first step carefully, but it is still important the first step be successful to move the journey forward. I have discussed elsewhere the wisdom of designing the first step for success, and to limit unneeded or risky development. Leveraging existing KBpedia assets as supplemented by your domain instance data is one way to bound this risk.

The players in the first step of a KM initiative should be those with a need and who are supportive. It is perhaps essential that the initial team include champions, who are smart and willing to learn. We need to spread the seeds of knowledge management on fertile ground, which also has some visibility to other portions of the organization. We almost assuredly bake in failure when we attempt such initiatives too broadly or without local support. Because of the shortcomings of past 'solutions' such as BI or data warehousing, we also see a decline and a reluctance for IT to embrace new and transforming approaches. These considerations argue strongly for embedding first stakes in a KM project within a department or group directly involved in knowledge work or management. KM projects are almost always of some threat to IT departments as they presently understand their role. As a general rule, do not attempt to start KM projects there, and expect resistance and naysaying from some in IT.

Incremental Build-Outs Follow Benefits

We make much of 'incremental' or 'agile' deployments within enterprises, but the nature of the traditional data system (and its closed-world assumption) can act to undermine these laudable steps. The inherent nature of an open-world approach,

[7] Including, of course, explicit attempts to model intangible benefits realistically.

matched with methodologies and best practices, can work wonderfully with KM-related projects. We have seen how we can incrementally stage our phases, moving into more complicated and enterprise-visible areas over time.

Learn to Quantify and Document Benefits

The grounding of a KM system in the information that knowledge workers have, how they presently conduct their work, and what they need to improve it, provides the same bases for documenting benefits from a new initiative. You should document current practices to capture and model workflows, and you should record time and effort associated with ongoing work tasks. These are the required metrics to show whether KM initiatives are improving productivity or not and, if so, by how much. (Of course, you need to measure and document benefits as well.) These kinds of considerations should be central in the design of a KM initiative because, without you collecting and monitoring such data, it will be impossible to project the documented savings and improvements needed to justify ongoing commitments. James Hendler once stated that "a little semantics goes a long way."[8] That truth—and it is true—when combined with incremental deployment firmly tied to demonstrable results promises a different way to do business.

References

1. L. Galárraga, G. Heitz, K. Murphy, F.M. Suchanek, *Canonicalizing Open Knowledge Bases* (ACM Press, New York, NY, 2014), pp. 1679–1688
2. N.F. Noy, D.L. McGuinness, *Ontology Development 101: A Guide to Creating Your First Ontology* (Knowledge Systems Laboratory, Stanford University, 2001)
3. M.K. Bergman, 30 Active ontology alignment tools, in *AI3:::Adaptive Information.* http://www.mkbergman.com/?p
4. O. Corcho, M. Fernandez, A. Gomez-Perez, Methodologies, tools and languages for building ontologies: Where is the meeting point? Data Knowl. Eng. **46** (2003)
5. D.M. Jones, T.J.M. Bench-Caponand, P.R.S. Visser, Methodologies for ontology development, in *Proceedings of the IT and KNOWS Conference of the 15th FIP World Computer Congress* (1998)
6. E.P.B. Simperl, C. Tempich, Ontology engineering: a reality check, in *On the Move to Meaningful Internet Systems* (Springer, Berlin, Heidelberg, 2006), pp. 836–854
7. E. Simperl, C. Tempich, D. Vrandečić, A methodology for ontology learning, in *Frontiers in Artificial Intelligence and Applications 167 from the Proceedings of the 2008 Conference on Ontology Learning and Population: Bridging the Gap between Text and Knowledge* (2008), pp. 225–249
8. E.P.B. Simperl, C. Tempich, Y. Sure, ONTOCOM: a cost estimation model for ontology engineering, in *The Semantic Web—ISWC 2006*, ed. by I. Cruz, S. Decker, D. Allemang,

[8] James Hendler, "a little semantics goes a long way." See http://www.cs.rpi.edu/~hendler/LittleSemanticsWeb.html.

C. Preist, D. Schwabe, P. Mika, M. Uschold, L.M. Aroyo (Springer, Berlin, Heidelberg, 2006), pp. 625–639
9. E. Simperl, M. Mochol, T. Burger, Achieving maturity: The state of practice in ontology engineering in 2009. Int. J. Comput. Sci. Appl. **7**, 45–65 (2010)
10. F. Giunchiglia, M. Marchese, I. Zaihrayeu, Encoding classifications into lightweight ontologies, in *Proceedings of the 3rd European Semantic Web Conference* (ESWC, 2006)
11. *SKOS Simple Knowledge Organization System Reference: W3C Recommendation*, World Wide Web Consortium (2009)
12. M. van Assem, V. Malaisé, A. Miles, G. Schreiber, A method to convert Thesauri to SKOS, in *The Semantic Web: Research and Applications*, ed. by Y. Sure, J. Domingue (Springer, Berlin, Heidelberg, 2006), 95–109
13. M. Poveda Villalón, *Ontology Evaluation: A Pitfall-Based Approach to Ontology Diagnosis*. Ph.D., Universidad Politécnica de Madrid, ETSI_Informatica (2016)
14. A. Halevy, M. Franklin, D. Maier, Principles of Dataspace Systems (PODS), in *Proceedings of ACM Symposium on Principles of Database Systems* (2006), pp. 1–9

Chapter 14
Testing and Best Practices

Builds, in a knowledge environment, should be responsive to the nature of knowledge, open and always changing. This knowledge environment includes the knowledge graph, plus its knowledge bases, and its management and analysis tools. To maintain the integrity of this structure going forward, we must test the structure for consistency and coherence after every batch of updates or changes. This constant requirement demands that the entire structure be recompiled and tested quickly and frequently. Constant revision is the only correct mindset, subject to user input and scrutiny, for which we need tools and guidance to do so in an intelligent way. As we wrap up our discussion on building a KM system, we need to give equal weight to the practical activities that keep our knowledge structures relevant.

We will start the discussion by introducing two straightforward metrics, from which all of our statistical tests flow.[1] From these we derive many useful and common statistics that are good to know, and easy to calculate. Our approach leverages the knowledge aspects, including good populations of type instances, to continue to improve the quality of the domain representation. Enhanced domain representations improve the subsequent ability to test new candidate representations, all in a virtuous circle. To make these efforts practical, we need scripts for both building the structure and testing its integrity. We want the control of these skills to continue to migrate to knowledge workers. Knowledge is best captured by those discovering it. These guidances then lead us to the question of best practices, especially for the build steps covered in Chap. 13. As we wrap up this chapter, we also conclude our discussion of building the knowledge representation system. This chapter completes

[1] Some material in this chapter was drawn from the author's prior articles at the *AI3:::Adaptive Information* blog: "Listening to the Enterprise: Total Open Solutions, Part 1" (May 2010); "Using Wikis as Pre-Packaged Knowledge Bases" (Jul 2010); "A Reference Guide to Ontology Best Practices" (Sep 2010); "The Conditional Costs of Free" (Feb 2012); "Why Clojure?" (Dec 2014); "A Primer on Knowledge Statistics" (May 2015); "Literate Programming for an Open World" (Jun 2016); "Gold Standards in Enterprise Knowledge Projects" (Jul 2016).

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8_14

the stage of the why and wherefore of a KR system, enabling us in the next part to tackle the question of applications and potential practical uses.

A Primer on Knowledge Statistics

Semantics is a funny thing. All professionals come to know that communication with their peers and external audiences requires accuracy in how to express things. Even with such attentiveness, communications sometimes go awry. It turns out that background, perspective, and context can all act to switch circuits at the point of interaction. Despite, and probably because of, our predilection as a species to classify and describe things, all from different viewpoints, we can often communicate with terms and language that convey to others something different from what we intended. Alas! This reality is why, I suspect, we have embraced as a species things like dictionaries, thesauri, encyclopedias, specifications, standards, sacred tracts, and such, to help codify what our expressions mean in a given context. So, yes, while sometimes we are sloppy in language and elocution, many misunderstandings between parties are also a result of the difference in perspective.

When we process information to identify relations or extract entities, to type them or classify them, or to fill out their attributes, we need measures to gauge how well our algorithms and tests work, all attentive to providing adequate context and perspective. These very same measures can also tell us whether our attempts to improve them are working or not. We also use these measures, in turn, to establish effective 'gold standards' and create positive and negative training sets for machine learning. Still, despite their importance, it is not always easy to explain these measures. The truth is, sometimes we don't adequately understand these measures.

Two Essential Metrics, Four Possible Values

In our context, we can see a couple of differences from traditional scientific hypothesis testing.[2] The problems we are dealing with in information retrieval (IR), natural language understanding or processing (NLP), and machine learning (ML) are all statistical classification problems, specifically in binary classification.[3] For example, is a given text token an entity or not? What type among a discrete set is it? Does the token belong to a given classification or not? Binary classification makes it considerably easier to posit an alternative hypothesis and the shape of its distribu-

[2] The *Open Semantic Framework* wiki is a contributor to content in this chapter, particularly "NLP and Knowledge Statistics" (http://wiki.opensemanticframework.org/index.php/NLP_and_Knowledge_Statistics) and "Ontology Best Practices" (http://wiki.opensemanticframework.org/index.php/Ontology_Best_Practices).

[3] I refer here to statistical classification; clearly, language meanings are not binary but nuanced.

tion. What makes it binary is the decision as to whether a given result is correct or not. We now have a different set of distributions and tests from more common normal distributions. The most common scoring methods to gauge the 'accuracy' of natural language or supervised machine learning analysis involve statistical tests based on the ideas of two essential metrics: negatives or positives, and true or false. We can measure both of these metrics by scoring correct 'hits' for predictions compared to a 'gold standard' of known results. This gold standard provides a representative sample of what our actual population looks like, one we have characterized in advance. We can use this same gold standard repeatedly to gauge improvements in our test procedures. I talk more about gold standards at the conclusion of this section.

Statistical tests will always involve a trade-off between the level of false positives (in which a non-match is declared a match) and the level of false negatives (in which an actual match is not detected).[4] Let's see if we can simplify our recognition and understanding of these conditions:

1. *TN/True Negative:* case was negative and predicted negative.
2. *TP/True Positive:* case was positive and predicted positive.
3. *FN/False Negative:* case was positive but predicted negative.
4. *FP/False Positive:* case was negative but predicted positive.

Combining these thoughts leads to a much simpler matrix, sometimes called a confusion matrix, for laying out the true/false, positive/negative characterizations (Table 14.1):

Table 14.1 Two essential metrics, four possible values

	Test assertion	
Correctness	Positive	Negative
True	*TP* True positive	*TN* True negative
False	*FP* False positive	*FN* False negative

As we can see, 'positive' and 'negative' are simply the assertions (predictions) arising from our test algorithm of whether or not there is a match or a 'hit.' 'True' and 'false' merely indicate whether these assertions proved correct or not as determined by gold standards or training sets. A false positive is a false alarm, a 'crying wolf'; a false negative is a missed result. Thus, all true results are correct; all false results are incorrect. More formally, we can now define these four values as follows:

- *TP* = Test assertion is positive and correct; standard provides labels for instances of the same types as in the target domain; manually scored; test identifies the same entity as in the gold standard.

[4] See http://en.wikipedia.org/wiki/Type_I_and_type_II_errors.

- *FP* = Test assertion is positive but incorrect; manually scored for test runs based on the current configuration; test indicates as positive, but deemed not true; test identifies a different entity than what is in the gold standard (including no entity).
- *TN* = Test assertion is negative and correct; standard provides somewhat similar or ambiguous instances from disjoint types labeled as negative; manually scored; test identifies no entity, gold standard has no entity.
- *FN* = Test assertion is negative and incorrect; manually scored for test runs based on the current configuration; test indicates as negative, but deemed not true; test identifies no entity, but gold standard has one.

These measures are sufficient to calculate most of the relevant statistics for our knowledge management and representation purposes.

Conversely, we can relate these two metrics to the branch of statistics known as statistical hypothesis testing. This testing is likely the statistics that you were taught in school. In hypothesis testing, we begin with a hypothesis about what might be going on concerning a problem or an issue, but for which we do not know the cause or truth. After reviewing some observations, we formulate a hypothesis that some factor A is affecting or influencing factor B. We then formulate a mirror-image null hypothesis that specifies that factor A does *not* affect factor B; this is what we test using statistical hypothesis testing. The null hypothesis is what we assume the world in our problem context looks like, absent from our test. If the test of our formulated hypothesis does not affect that assumed distribution, then we reject our alternative (meaning our initial hypothesis fails, and we keep the null explanation).

We make assumptions from our sample about the distribution of the population, which enables us to choose a statistical model that captures the shape of assumed probable results for our measurement sample. These shapes or distributions may be normal (bell shaped or Gaussian), binomial, power law, or many others. These assumptions about populations and distribution shapes then tell us what kind of statistical test(s) to perform. (Misunderstanding the actual shape of the distribution of a population is one of the major sources of error in statistical analysis.) Different tests may also give us more or less statistical power to test the null hypothesis, which is that chance results will match the assumed distribution. Different tests may give us more than one test statistic to measure variance from the null hypothesis.

We then apply our test and measure and collect our sample from the population, with random or other statistical sampling necessary so as not to skew results, and compare the distribution of these results to our assumed model and test statistic(s). We reject the null hypothesis if we observe significant differences from the expected shape in our sample at a high level of confidence. If we reject the null hypothesis, but in fact it was correct, we call that a Type I error, or a false positive (FP), the same as FP in a binary classification. If we accept the null hypothesis, we reject the alternative hypothesis that some factor A is affecting or influencing factor B. However, if we accept a null hypothesis that is not correct, we term that a Type II error, or a false negative (FN), the same as FN in a binary classification. Statisticians often apply common rules for how differences and level of confidence may lead to

rejection of the null hypothesis, thereby leading us to accept the alternative hypothesis that factor A is affecting or influencing factor B.

The binary classification TP *v* FP *v* TN *v* FN approach is better than the statistical hypothesis approach because it explicitly recognizes either the sampling method or that our test may be in error. Further, the TP *v* FP *v* TN *v* FN approach is also easier to explain and understand.

Many Useful Statistics

Armed with these four characterizations—true positive, false positive, true negative, and false negative—we now can calculate nearly all essential statistical measures. Most of these measures also have exact analogs in standard statistics. The first metric captures the concept of *coverage*. In standard statistics, this measure is called sensitivity; in IR and NLP contexts it is called *recall*. It is the fraction of the documents that are relevant to the query that is successfully retrieved. Recall measures the 'hit' rate for identifying true positives out of all potential positives, and we also call it the true positive rate, or TPR:

$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN}$$

A high recall value, expressed as a fraction of 1.00 or a percentage, means the test has a high 'yield' for identifying positive results. We measure it as *true positives* divided by all potential positives in the corpus.

Precision is the complementary measure to recall, in that it is a measure of how efficient the system is to make correct identifications from the positive ones. Precision is the fraction of retrieved documents that are relevant to the query. We measure it as *true positives* divided by all measured positives (true and false):

$$Precision = \frac{Number\ of\ true\ positives}{Number\ of\ true\ positives + false\ positives}$$

High precision indicates a high percentage of true positives compared to all positive results. Precision is something, then, of a *quality* measure, which we express as a fraction of 1.00 or a percentage. It provides a positive predictive value, as defined as the proportion of the true positives against all the positive results (both true positives and false positives). So, we can see that recall gives us a measure as to the breadth of the hits captured, while precision is a statement of whether our hits are correct or not. Note also that false positives are a proper focus of attention in test development because they directly lower precision and the efficiency of the test.

One of the preferred overall measures of IR and NLP statistics is the F-score, which is the adjusted (beta) mean of precision and recall. It recognizes that precision and recall are complementary and linked. The general formula for positive real β is

$$F_\beta = \left(1+\beta^2\right) \times \frac{Precision \times recall}{\left(\beta^2 \times precision\right) + recall}$$

which we can express for TP, FN, and FP as

$$F_\beta = \frac{\left(1+\beta^2\right) \times true\ positive}{\left(1+\beta^2\right) \times true\ positive + \beta^2 \times false\ negative + false\ positive}$$

In many cases, the harmonic mean is used, which means a beta of 1, which is also called the F_1 statistic or the F_1 score:

$$F_1 = 2 \times \frac{Precision \times recall}{Number\ of\ true\ positives + false\ positives + false\ negatives}$$

However, F_1 displays a tension. Either precision or recall may be improved to achieve an improvement in F_1, but with divergent benefits or effects. What is more highly valued? Yield? Quality? These choices dictate what kinds of tests and areas of improvement need to receive focus. As a result, the weight of beta can be adjusted to favor either precision or recall. Two other commonly used F measures are the F_2 measure, which weights recall higher than precision, and the $F_{0.5}$ measure, which puts more emphasis on precision than recall.

Accuracy is another metric that can factor into our evaluations, though we use it less in the IR and NLP realm. Accuracy is a statistical measure of how well a binary classification test correctly identifies a condition. We calculate it as the sum of true positives and true negatives divided by the total population (TP + FP + TN + FN):

$$Accuracy = \frac{Number\ of\ true\ positives + number\ of\ true\ negatives}{Total\ population}$$

An accuracy of 100% means that the measured are the same as the given values.

All of the measures above only require the measurement of false and true, positive and negative, as do a variety of predictive values and likelihood ratios. We may

also calculate relevance, prevalence, and specificity, which use these same metrics in combination with the total population. By bringing in some other rather simple metrics, we can expand this statistical base to cover such measures as information entropy, statistical inference, pointwise mutual information, variation of information, uncertainty coefficients, information gain, AUCs, and ROCs. All of these still bridge from the basic four values that we need to measure of TP, FP, FN, and TN. We may accommodate these additional tests by keeping track of distributions, calculating confidence intervals, tracking joint or conditional distributions, or summing the area under the distribution curve, in addition to our standard four measures.

We can summarize across all of these basic statistical tests on a single chart, courtesy of a template on Wikipedia,[5] for which I have taken some minor liberties. I show this summary chart of IR and NLP statistical tests in Table 14.2.

Working Toward 'Gold Standards'

Academic researchers in natural language processing (NLP) and machine learning (ML) commonly compare the results of their studies to benchmark, reference standards. A *gold standard* is a reference, benchmark test set where we have already scored results, with a minimum (if not zero) amount of *false positives* or *false negatives*. We should also include *true negative* results in a proper gold standard approximate to the likely ratio expected in the overall population to improve overall accuracy [1]. Gold standards that contain false positives and false negatives, by definition, immediately introduce errors, as we noted for Type I and Type II errors above. A skewed baseline makes it difficult to test and refine existing IR and NLP algorithms.

[5] See http://en.wikipedia.org/wiki/Template:DiagnosticTesting_Diagram.

Table 14.2 Various statistical measures

		True condition			
	Total population	Condition positive	Condition negative	Prevalence = Σ Condition positive/ Σ Total population	Accuracy (ACC) = Σ True positive + Σ True negative/Σ Total population
Test condition	Predicted condition positive	True positive (TP) Power	False positive (FN) Type I error	Positive predictive value (PPV), Precision = Σ True positive/ Σ Predicted condition positive	False discovery rate (FDR) = Σ False positive/ Σ Predicted condition positive
	Predicted condition negative	False negative (FN) Type II error	True negative (TN)	False omission rate (FOR) = Σ False negative/ Σ Predicted condition negative	Negative predictive value (NPV) = Σ True negative/ Σ Predicted condition negative
		True positive rate (TPR), Recall, Sensitivity = Σ True positive/ Σ Condition positive	False positive rate (FPR), Fall-out = Σ False positive/ Σ Condition negative	Positive likelihood ratio (LR+) = TPR/FPR	Diagnostic odds ratio (DOR) = LR+/LR−
		False negative rate(FNR), Miss rate = Σ False negative/ Σ Condition positive	True negative rate (TNR), Specificity (SPC) = Σ True negative/ Σ Condition negative	Negative likelihood ratio (LR−) = FNR/TNR	F1 score = 2/ ((1/Recall) + 1/Precision))

Moreover, because gold standards also often inform training sets, errors there propagate into errors in machine learning. The requirement to compare research results to existing gold standards provides an empirical basis for how the new method compares to existing ones, and by how much. Precision, recall, and combined F_1 score are the most prominent among these statistical measures.

We often refer to specific standards, such as the NYT Annotated Corpus or the Penn Treebank [2], as gold standards because they have been in public use for some time, with many errors edited from the systems. Vetted standards such as these may have inter-annotator agreements in the range of 80–90%. More typical use cases in biomedical notes [3] and encyclopedic topics [4] tend to show inter-annotator agreements in the range of 75–80%. While a claimed accuracy of even, say, 95% sounds impressive, applied to a large knowledge graph such as KBpedia, with its 55,000 concepts, it translates into 2750 concept misassignments (actually, the problem is many orders of magnitude greater than that when we include all assertions). That sounds like a lot, and it is. Misassignments of some nature occur within any standard. When they occur, they are sometimes glaringly obvious, like being out of plumb. It is pretty easy to find most errors in most systems. Still, for the sake of argument, let's accept we have applied a method that has a claimed accuracy of 95%. Remember, this is a measure applied only to the gold standard. If we take the high end of the inter-annotator agreements for domain standards noted above, namely 80%, then we have this overall accuracy of the system:

$$0.80 \times 0.95 = 0.76$$

Whoa! Now, using this expanded perspective, for a candidate knowledge graph the size of KBpedia—that is, about 55,000 items—we could see as many as 13,200 concept misassignments (again, orders of magnitude greater for all assertions). Those numbers now sound huge, and they are. They are unacceptable.

A couple of crucial implications result from this analysis. First, we need to take a holistic view of the error sources across the analysis path, including and most especially the reference standards. (They are, more often than not, the weak link in the analysis path.) Second, we want to get the accuracy of reference standards as high as possible. Thus, we can see many areas by which gold standards may need attention:

1. They may contain false positives.
2. They may contain false negatives.
3. They have variable inter-annotator agreement.
4. They have variable mechanisms, most with none, for editing and updating the labels.
5. They may lack sufficient inclusion of true negatives.
6. They may derive from an out-of-context domain or circumstance.

You should be aware of these potential sources of error to improve test foundations.

An integral part of any knowledge representation or management effort must be to create gold standards for continuous quality improvements. The domain coverage

inevitably requires new entity or relation recognizers, or the mapping of new datasets. The nature of the content at hand may range from tweets to ads to Web pages or portions or academic papers, with specific tests and recognizers from copyrights to section headings informing new learners. Every engagement requires reference standards. One effort might favor instance records over concepts. Creating gold standards efficiently with a high degree of accuracy is a competitive differentiator.

We may use each type and its instances in KBpedia as a training set for learners. We can continue to improve the accuracy of instance assignments for each type by testing shared attributes or neighbors or type inheritance, plus errors fixed after inspection. One key to growing a consistent knowledge graph over time is to apply these virtuous improvements. Once you create a gold standard, you then run your current test regimes against it when you run your same tests against unknowns. Preferably, of course, the gold standard only includes true positives and true negatives (that is, the gold standard is the basis for judging 'correctness'; see confusion matrix above). If it does not, misassignments, when encountered, must be fixed, preferably as part of existing workflows (see Chap. 12).

More accurate standards and training sets lead to improved IR and ML algorithms, feeding the virtuous circle in knowledge-based artificial intelligence (KBAI) (see Fig. 4.2). Continuing to iterate better knowledge bases and validation datasets is a driving factor in improving both the yield and quality from the KBs. KBAI, then, is a practice based on a curated knowledge base eating its tail, working through cycles of consistency and logic testing to reduce misassignments, while continually seeking to expand structure and coverage. Adding and testing structure or mapping to new structures and datasets continually gets easier, and also produces a network effect. These efforts enable us to partition the knowledge structure efficiently for training specific recognizers, classifiers, and learners while also providing a logical reference structure for adding new data and structure.

We then use this basic structure—importantly supplemented by the domain concepts and entities relevant to the domain at hand—to create reference structures for training the target recognizers, classifiers, and learners. The process of testing and adding structure identifies previously hidden inconsistencies. As corrected, the overall accuracy of the knowledge structure to act in a reference mode increases. Through straightforward SPARQL queries, we can retrieve both positive and negative training sets for machine learning. Clean, vetted gold standards and training sets are thus a critical component to improve our knowledge bases going forward [5]. We need to give much attention to the practice of creating gold standards and training sets because, without it, we are shooting in the dark when we attempt to improve our learners or language analysis.

Builds and Testing

The implications of working with knowledge bases are clear. KBs are constantly in flux. Single-event, static processing is dated as soon as we run the procedures. The only way to manage and use KB information comes from a commitment to constant

processing and updates. Further, with each processing event, we learn more about the nature of the underlying information that causes the processing scripts and methods to need tweaking and refinement. Without us documenting what we have done with prior processing, it is impossible to know how to tweak next steps to avoid dead ends or mistakes of the past. KBAI processing cannot be cost effective and responsive without a memory. We find literate programming, discussed below, an enabler in this process.

Any knowledge management installation may involve multiple input sources, all moving at different speeds of change. We require numerous steps in processing and updating the input information, the 'systems,' if you will, to achieve our artificial intelligence and data interoperability purposes. The artifacts associated with these activities range from functional code and code scripts to parameter, configuration, and build files; to the documentation of those files and scripts; to the logic of the systems; to the process and steps followed to achieve desired results; and to the documentation of the tests and alternatives investigated at any stage in the process. The kicker is that you will need updates to all of these components. Without a systematic approach, you will not easily remember the script code of what you previously did, leading to costly rediscovery and rework.

Build Scripts

We seek simplicity in our code scripts through modularity and aggressive use of the OWL API. This API gives us the ability to manipulate the graph connections and structure, using direct triple assertions (often in N3 or Turtle). We seek modularity to segregate code for testing and debugging, and because of the use of simpler data input files, again based on triples. We tightly couple the scripting approach with the platform's Web service design, how we set parameters, and how we ingest or export datasets.

We initially require build scripts for installing the apps on the platform and installing other build scripts. Since we recommend open-source configurations for the platform, the multiplicity of tools included with the platform can impose installation challenges. Project build utilities such as Maven or Ant can be very helpful here. For a new build, you may need to create local directory structures, back up prior versions, install input knowledge structures, update metadata and any hard-wired script references (which, should, over time, evolve to more sophisticated control structures), log setups, and reboot. Your actual build process may take dozens or hundreds of runs as you test changes, errors get generated from various tests (see next section), and then you resolve them. Throughout the entire process of data ingest and error resolution, we strongly recommend you enforce UTF-8 encoding across all knowledge representations.

In our KBpedia experience, we employ a series of testing scripts during builds (next section) to debug the knowledge structure in its current implementation. We build the base knowledge graph (in the case of KBpedia, this is KKO) first. To that,

we add the various typologies used for classifying the structure instances. We perform separate build checks against the typologies, particularly the identification of orphan concepts (types) and fragments of types within the typologies. We make modifications to the basic input files that guide these scripts, often using a triples format. I overviewed the kinds of build steps requiring scripting in Chap. 13. These kinds of iterations are what account for the many runs necessary to produce an integral, connected knowledge structure. Though we could make these modifications via an ontology editor such as Protégé, which is still used during inspection to identify the needed changes, by working with the input file structure, we have a more streamlined basis for full, complete build routines. Modifying the input files keeps the build routines simpler and, therefore, repeatable.

We advise securing sufficient memory for the build process. In the case of KBpedia, we recommend a commodity server with a minimum of 8 GB of RAM. More is preferred, though your domain needs may raise or lower memory needs. Once build issues are worked out, and the graph appears complete and consistent, we advise adding further scripts to generate statistics, which we run at this time. You may want to add standard ontology metrics such as concept and assertion counts at this time. You may also invoke more detailed stats or fragments, including graph-wide statistics, at this point. Some of the statistics runs require a census of the knowledge structure involving multiple, repeated SPARQL queries, which can take quite a bit of time to run.

Only deploy the updated knowledge structure when the build scripts run to completion and all tests pass. You may find that small projects at the department level require little in the way of formal deployment; systems in enterprise-wide use may require staging through multiple servers with various approval steps before official deployment. If you have complicated deployments, perhaps involving multiple servers to host key platform components, you may need to give these aspects scripting attention. You may need to conform mature installations with broader enterprise workflow steps and procedures.

We have progressed KBpedia through this growth and renewal process uncounted times. Our automated build scripts mean we can regenerate KBpedia on a commodity machine from scratch in about 45 min. If we add all of the logic, consistency, and satisfiability checks, we can create a new build in about 2 h. One of our recent expansions to KBpedia involved reciprocal mapping (see Chap. 15) to Wikipedia, and added about 40% new nodes to KBpedia's then current structure. Remarkably, using the prior KBpedia as the starting structure, we were able to achieve this expansion with even better logical coherence of the graph in a few hundred hours of effort due to our build philosophy and scripts.

Testing Scripts

We invoke various test scripts as an integral part of the build procedure. We apply scripts against platform tools, for coherency and consistency checks, for reference standards used for placements or machine learning, or for general incremental

improvements to the KBpedia structure. Each build invokes some structure tests. Standard tests include (1) 'unsatisfied' classes, which lack characterizations sufficient to standards or degree of connectedness; (2) misassigned classes, where subsumption relationships are contradicted; (3) wrong relations in terms of degree or probability of relationship; (4) wrong SuperTypes; (5) missing, new, or misassigned attributes; (6) general ontology checks such as orphans, fragments, or splits; (7) various graph and connectedness measures; (8) OOPS! consistency and completeness checks; (9) Protégé and add-on checks; and (10) others of your choosing. We try to provide common details and organization to error messages and labels, a consistent approach we also try to extend to Web services and tools. Every error type should have an error code and adequate explanatory text.

Incremental builds (with version numbers, even if not released) are the secret to being able to maintain these knowledge structures. Accumulating too many changes between builds can lead to multiple error sources and greater difficulty to debug. Incremental builds surface errors quickly and fewer at a time. Still, we may require various build runs before we fix all errors. We only release builds that pass all tests, when we assign a new version number.

We can train using 'dirty' training bases that have embedded error no better than the quality of their inputs. If we want to train our knowledge applications with Dick and Jane reader inputs, too often in error to begin with, we will not get beyond the most basic of knowledge levels. We need clean reference standards. On average, we can create a new reference standard for a given new type in 20–40 labor hours. Specifics may vary, but we typically seek, at a minimum, about 500 true-positive instances per standard, with 20 or so true negatives. This criterion is a minimum for a reference standard. For machine learning purposes, more is better. We could conceivably lower the requirement for a reference standard below 500 true-positive instances as we see the underlying standards improve. We are not seeking definitive statistical test values but a framework for evaluating different parameters and methods. In most cases, we have seen our reference sets grow over time as new wrinkles and perspectives emerge that require testing.

In all cases, our most critical success factor is to engage users, the knowledge workers and managers, in manual review and scoring of the reference standards. We document and train this process so that we may repeat and refine it. User analysts understand and detect patterns that then inform improved methods. We believe that clean, vetted training sets and reference standards that move toward 'gold' ones are essential to any KM/KR project.

Literate Programming

The only sane way to tackle knowledge bases at these structural levels is to seek consistent design patterns that are easier to test, maintain, and update. Open-world systems must embrace repeatable and mostly automated workflow processes, plus a commitment to timely updates, to deal with the constant, underlying change in

knowledge. Code and scripts do not reside in isolation. We need to explain the operation of the code to others so they may fix bugs or maintain it. If the software is a processing system, we learn much from testing and refinement, which we should document for subsequent iterations. We must install and deploy our systems. We need to update libraries and languages frequently for security and performance reasons; we need to update executables and environments as well. When we update systems, we need to run tests for expected performance and accuracy. The severity of some updates may require revision to whole portions of the underlying systems. New employees need tech transfer and training, and managers need to know how to take responsibility for the systems. These are all needs that literate programming can help support.

Literate programming is a style of writing code and documentation first proposed by Donald Knuth [12]. In any aspect of a project that uses code or scripts—tests, configurations, installations, deployments, maintenance, or experiments—the developers or users write narratives and documentation to accompany it. The documentation should be robust by explaining what it is, the logic of it, and what it is doing and how to exercise it. This documentation far exceeds the best practices of inline code commenting. Literate programming narratives might provide background and thinking about what is being tested or tried, design objectives, workflow steps, recipes, listings of data or discussions of datasets, or whatever. The style and scope of documentation are similar to a scientist's or inventor's lab notebook. Indeed, the breed of emerging electronic notebooks, combined with REPL coding approaches, which allow the embedding of live code demos within notebooks, now enable interactive execution of functions and visualization and manipulation of results, including supporting macros. Thus, we can include working demos and code in-line with our narratives.

Notebook systems that support literate programming, such as Org-mode, can 'tangle' their documents to extract the code portions for compilation and execution. They can also 'weave' their documents to extract all of the documentation in the code now formatted for human readability, including using HTML. Some electronic systems can process multiple programming languages and translate functions. Some electronic systems have built-in spreadsheets and graphing libraries, and most open-source systems can be extended (though with varying degrees of difficulty and in different languages). Some of the systems interact with or publish Web pages.

Leading notebook software include the iPython Notebook (Jupyter), Org-mode, Wolfram Alpha, Zeppelin, Gorilla, and others. Literate programming requires a focused commitment. The objective of programmers should not be solely to write code but to write systems that can be used and reused to meet desired purposes at an acceptable cost. Documentation is integral to that objective. Experiments should be documented, codified, and improved. A lines-of-code (LOC) mentality is counterproductive to effective software for knowledge purposes. Literate programming is the most conducive workflow to achieve these ends, with notebooks as the medium for tracking and training work tasks.

One question is what language to use for the literate programming or scripts. Lisp (defined as a *lis*t *p*rocessing language) is one of the older computer languages

around, dating back to 1958, and has evolved to become a family of languages. 'Lisp' has many variants, with Common Lisp one of the most prevalent, and many dialects that have extended and evolved from it. Most recently our scripting choice has been Clojure, a modern language based on Lisp, but able to run in the Java virtual machine (JVM), which makes integrating with existing Java tools much easier. In the context of knowledge management and semantic uses, fully 60% of existing applications can now interoperate with Clojure apps, an instant boon for leveraging many open-source capabilities. Java gives us certain advantages, including platform independence and the leverage of debugging and profiling tools, among others. Clojure is a functional programming language, which means it has roots in the lambda calculus and functions are 'first-class citizens.' Functions can pass as arguments to other functions, and return as values, or assign as variables in data structures. These aspects make the language well suited to mathematical manipulations and the building up of more complicated functions from simpler ones. Because of Clojure's REPL (read-event-print-loop) abilities, we can interpret code immediately as we execute instructions at the time of input, leading to a very dynamic and responsive code-development and -testing environment, also well suited to literate programming.

Alternatives like Scheme, Erlang, Haskell, or Scala offer some of the same JVM benefits. Further, tooling for Clojure is still limited, and it requires Java to run and develop. Even with extensions and DSLs, learning Lisp's mindset may be awkward for some. The point here is not to point to a specific language alternative but to enumerate the kinds of evaluation criteria that may go into such a software decision. External factors, too, such as popularity and skill knowledge, certainly can and should enter into language decisions.

As a summary observation, a knowledge management project brings substantial *technical debt*, defined as the overhead and overlooked consequences of adopting a given technological solution, and then needing to develop, stage, manage, and use it. Technical debt is broader still for knowledge management projects because all aspects of the source knowledge are dynamic. Keeping current with changes is a positive thing, and no responsive KM solution would last long without it. Literate programming captures all of these dynamics.

Some Best Practices

We have discussed at length build components and practices over the past five or six chapters. While we have not been prescriptive, since techniques and tools are continually improving, we have tried to cover the major steps and background that go into building a knowledge representation and management platform. We have also discussed the approaches for building the knowledge structures and graphs upon which these systems run. As we wrap up these discussions, let me recount some of the best practices we apply in these steps. We have learned most of these best practices from client deployments in areas such as data treatment and dataset management, creating and using knowledge structures, and in testing, analysis, and documentation.

No bright line separates recommended steps and best practices, so we have touched upon many key arguments already in our presentation. Modularity in knowledge graphs, or consistent attention to UTF-8 encoding in data structures, or emphasis on 'semi-automatic' approaches, or use of literate programming and notebooks to record tests and procedures are just a few of the examples where lines blur between standard and best practices. The key point is that best practices are also an integral part of doing standard tasks right.

Data and Dataset Practices

We have emphasized the importance of using only one or a few internal canonical forms for representing our data, including the importance of testing and ensuring that we maintain UTF-8 encoding throughout. UTF-8 is important to maintain multi-language capabilities and the uniform treatment of different language character sets and accents. We have noted the use of basic triple assertions (often in N3 or Turtle) for use in our data transfer protocols. We have also noted the importance of using language tags for all of our labels as one means to promote multilingual use and internationalization (sometimes referred to as *i18n*). We want to add on to these points in this section by pointing out best practices in dataset packaging and the use of linked data.

Dataset Best Practices

Datasets are one of the fundamental dimensions for organizing content within our recommended design. Some consideration needs to go into how best to bound these structures. The first consideration relates to the domain, or the scope of the dataset: What is the applicable scope or business purpose of this information? It is best to think of this question from a perspective of access, which is, after all, the most pragmatic way to think of it. We also want to capture the source of the data, and whether it may vary by publisher or source location. For example, provenance or download location or format may be an important distinguishing factor in release or access and may have copyright or royalty implications. That leads to the need to record when we create the data, perhaps adding metadata for whether the data has periodic update or creation times. It may be helpful to distinguish between preliminary data and final data or to segregate data because of workflow or processing considerations. We also need to record all data by type; that is, does the data vary by class or kind? For example, we may find it desirable to keep records about schools separate from records about churches, though at a different level both may be considered buildings. We should be attentive to the data attributes for specific instances, and to use common vocabulary and schema for organizing those characteristics. We may also want to record the completeness of records in regard to attributes or descriptions, since we may want to prefer using better characterized data in parts of our analysis or may want to flag areas needing future attention. Any of these differences may warrant creating a separate dataset or adding new metadata. Ultimately, these

considerations for how to organize data structurally come down to possible differences in access rights, both at the record and attribute levels. Access differences may warrant altered dataset organization. No limits occur to the number of datasets that may be managed by a given KM instance.

Once you set such boundaries, then think about common attributes or metadata that should be applied. The KBART Recommended Practice is worth review since it suggests a file format and common sense set of metadata fields and formats for transmission of metadata from content providers useful to linked knowledge bases [6]. Still, further, datasets and their records (as all decision or information artifacts in an enterprise) go through natural work stages or progressions. Even the lowliest written document goes through the steps of being drafted, reviewed, characterized, approved, and then possibly revised. Whatever such workflow steps may be, including versioning, you may argue to assign some records to a different dataset. Lastly, whatever operational mode you devise, find naming conventions to reflect these variations in your dataset files. These considerations show that datasets are meaningful information artifacts in and of themselves.

Linked Data

Linked data is a set of best practices for publishing and deploying instance and class data using the RDF data model, naming the data objects using uniform resource identifiers (URIs or IRIs), and exposing the data for access via the HTTP protocol while emphasizing data interconnections, interrelationships, and context useful to both humans and machine agents. The challenge is not the mechanics of *linking data*, but the meaning and basis for connecting that data. Connections require logic and rationality sufficient to inform inference and rule-based engines reliably. It also needs to pass the sniff test as we 'follow our nose' by clicking the links exposed by the data.

Most linked data uses a woefully small vocabulary of data relationships, with even a smaller set used for setting linkages *across* existing linked datasets. Linked data techniques are a part of the foundation of overall best practices, but not the whole foundation. We do not, for example, have sufficient and authoritative linking predicates to deal with common 'sort of' conditions. Just as SKOS is a generalized vocabulary for modeling taxonomies and simple knowledge structures, we need a similar vocabulary for predicates that reflect real-world usage for linking data objects and datasets with one another.[6] KBpedia provides this. Practice to date suggests that uncurated, linked datasets in the wild are unlikely to be useful nor to be used in combination with other datasets. On the other hand, users desire and readily consume quality linked data. Where you want your KM installation to interact with outside parties, employing linked data is one way to help ensure interoperability.

[6] A vocabulary of linking predicates would capture the variety and degrees to which individuals, instances, classes, and concepts are similar or related to objects in other datasets. This purpose is different than, say, voiD (Vocabulary of Interlinked Datasets), which has as its purpose in providing descriptive metadata about the nature of particular datasets.

Knowledge Structures and Management Practices

A central role of ontologies is to describe a 'worldview,' and in specific organizations this means a shared understanding of the concepts, relations, and terminology to describe the participants' shared domain. In turn, these shared understandings establish the semantics for how to effect communication and understanding within the population of domain users. All of this means that finding ways to identify and agree upon shared vocabularies and understandings is central to the task of modeling (creating an ontology) for the domain, and it involves practices in collaboration, naming, and use of these knowledge structures. Sometimes this perception of shared views is too strictly interpreted as needing to have one and only one understanding of concepts and language. Far from it.

Organizational and Collaborative Best Practices

One of the strengths of ontologies and language modeling within them is that we can accommodate multiple terms for the same concept or slight differences in understandings about nearly similar concepts. It is perfectly OK to have differences in terminology and concept understandings so long as those differences are also captured and explicated within the ontology. Embedding collaboration as an implementation best practice is important. We should understand that prior investments in agreed-upon structures and vocabularies deserve respect and we should review them for incorporation. We should capture essential differences, not smudge or obscure them. We want to organize our work teams, and support processes for consensus making, including tool support, so that our teams identify and decide upon terminology, definitions, alternative labels (*semsets*), and relations between concepts. These processes need not be at the formal ontology level, but at the level of the concept graph that underlies the ontology.

Naming and Vocabulary Best Practices

We recommend in our standard build practice to define all concepts and terminology, use semsets to capture alternative ways to name things, and sometimes treat concepts as either classes or instances. While consensus building and collaboration methods are at the heart of effective ontology building, we should not impose language and concepts by fiat. Try to name all *concepts as single nouns*. Use *CamelCase notation* for these classes (that is, class names should start with a capital letter and not contain any spaces, such as MyNewConcept). Name all *properties as verb senses* (so that we may easier read triples), e.g., hasProperty. Try to use *mixedCase notation* for naming these predicates (that is, begin with lower case but still capitalize each word after and do not use spaces). Try to use common and *descriptive prefixes and suffixes* for related properties or classes (while they are just labels and

their names have no inherent semantic meaning, it is still a useful way for humans to cluster and understand your vocabularies). For example, properties about languages or tools might contain suffixes such as 'Language' or 'Tool' for all related properties. Provide *inverse properties* where it makes sense, and adjust the verb senses in the predicates to accommodate. For example, <Father> <hasChild> <Janie> would be expressed inversely as <Janie> <isChildOf> <Father>.

Give all concepts and properties a *definition*. We conduct the matching and alignment of things by concepts (not merely labels), which means that each concept must be defined.[7] Provide clear definitions (along with the coherency of its structure) to give your ontology its semantics. Remember not to confuse the label for a concept with its meaning. (This approach also aids multilinguality.) Provide a *preferred label* annotation property that is used for human-readable purposes and in user interfaces. KBpedia uses the property of `skos:prefLabel`. Include a class *sem-set*, robustly harvested and populated, for all concepts and ambiguous entities. Try to assign *logical and short names to namespaces* used for your vocabularies, such as `kbpedia:XXX` or `skos:XXX`, with a maximum of five letters preferred. Enable *multilingual capabilities* in all definitions and labels. This language requirement is a somewhat complicated best practice in its own right. For the time being, it means attending to the `xml:lang = "en"` (for English, in this case) property for all annotation properties.[8]

Best Ontology Practices

To my knowledge, the most empirical listing of ontology best practices comes from Simperl and Tempich [8]. In that 2006 paper they examined 34 ontology-building efforts and commented on cost, effectiveness, and methodology aspects. Various collective ontology efforts also provide listings of principles or best practices. The OBO (*The Open Biological and Biomedical Ontologies*) effort, for example, offers a useful, organized listing of criteria[9] for an exemplary ontology. Their recommendations include their own best practices and to formulate and use a unified methodology. Simperl and Tempich emphasize modularity in their findings, consistent with our standard recommendation. They also recommend metrics for ontology evaluation and tools to extract ontology components from existing data sources, also consistent with our recommended standards.

One best practice we recommend is to embrace a mindset that ontologies can, and should, start small, and may grow incrementally. Another best practice is to keep relationships (predicates) simple at first until you gain fluency. Use simple, well-defined, and documented attributes. Aggressively mine and reuse existing

[7] As another commentary on the importance of definitions, see http://ontologyblog.blogspot.com/2010/09/physician-decries-lack-of-definitions.html.

[8] The Protégé manual [7] is also a source of good tips, especially with regard to naming conventions and the use of the editor.

[9] See http://obofoundry.org/wiki/index.php/OBO_Foundry_Principles.

knowledge and structure. Knowledge graphs, like knowledge, must be a continuous, dynamic structure, designed for (comparatively easy) updates and automatic builds. Another best practice is to enter items once, and relate them only to direct parents, not more removed upper categories. The upper categories can be inferred, and single, proper placements lead to a cleaner graph structure that is easier to interpret. Be cognizant of the many internal platform needs in workflow management and user interfaces and widgets where administrative ontologies may also contribute. If this mindset is followed, your initial ontology development need not be comprehensive nor expensive. You may grow efforts as you realize benefits. You can adopt pragmatic approaches to testing and then building out a knowledge management system. Starting with a stable structure like KBpedia is likely your most efficient path.

Testing, Analysis, and Documentation Practices

The usefulness of a KM platform depends on its accuracy, consistency, and concurrency for the domain. We need to ensure that these factors remain true as we extend our use of the knowledge and grow the domain further. We need to identify, characterize, and vet concepts. We need to teach this process, and to master tools. We need to assign responsibilities and manage the practice. We find testing and documentation central to this process.

Testing Best Practices

You should embed testing for functionality and testing your knowledge structures for consistency and coherency in all phases and steps of your KM platform. Testing is best when it is incremental and best as part of any build process. You should assign domains and ranges to properties, and invoke reasoners and other tools during update efforts to find inconsistencies. You should test all new concepts and properties at the time of introduction, which you may batch so long as you can manage the increments. Test *external class* assignments because they work to 'explode the domain' [9] and surface other inconsistencies. Use already vetted knowledge bases as reference testbeds when testing the coherence of concepts in a new domain ontology; if the domain ontology describes concepts quite differently than standard practice, or if relationships between concepts are at variance, then you likely have coherency problems. As you work with the system, continue to evolve ontology specifications to include *necessary and sufficient conditions* for complete reasoner testing.

Analytical Best Practices

The two core opportunities of a KM system in data interoperability and knowledge-based artificial intelligence place a premium on analysis, principally in natural language understanding and machine learning. External search engines also fit into this

category. In all cases, these analytic tools or learners are third-party applications, with varying degrees of ease of use and documentation. Three areas of best practices apply to these external tools. First, it is important to discover, test, and select the tools. Second, employ documentation and support structures, such as input data files or run-control specifications, to help make analytic runs repeatable. Third, be cognizant of the technical debt that each adopted tool may bring.

Every new analytical task should begin with a survey of available tools. You should include standard search, plus a search of major code repositories such as GitHub, plus monitoring of technical publication sources (blogs, RSS feeds, arXiv.org, etc.), in your initial investigations. As you research the tools, you may need to migrate from simple spreadsheet listings to detailed characterization of the alternatives. You should download leading candidates, install them, and initially test. This research is a good place to use the notebook paradigm. The choice of tools is fundamental to a KM system built from multiple parts from multiple parties with multiple purposes. Performance and scalability may rapidly become concerns as a KM system grows within larger enterprises.

As tools progress from candidates to provisional, we need to integrate them into the existing platform. Though you likely used support for the internal canonical data forms as an initial screening criterion, you now need to stage and test data exchange for the tool. As in other areas, you should document steps thoroughly for broader training and adoption. In the evaluation process, every new analytic area, even more so than the specific tools involved, will also incur some degree of technical debt. Scully et al. provide one example for a machine learning installation of how to think about various categories of technical debt potentially arising from new tools [10]. Andrew Ng also provides a concise listing of practical machine learning tips [11].

In varied guises, other analytical tools impose similar or related overhead costs. Search, as we discussed in Chap. 12, also poses debt and changes to work procedures. We always have hard-to-quantify benefits and the costs at the more intangible ends of the spectrum when using tools. Our benefits might be qualitative; our costs may hide. If we are to include intangible benefits in the positive column, then we must also be expansive in how we think of the costs of adoption as well.

Documentation Best Practices

Let me emphasize strongly that we need to bake documentation into the cake. We need to document every step of our efforts, like is done for good science notebooks but leveraging today's modern electronic versions. We must adequately comment and annotate our ontologies. We should document the entire *ontology vocabulary* via a dedicated system that allows finding, selecting, and editing of ontology terms and their properties. We should document ontology maintenance and construction methodologies, including naming, selection, completeness, and other criteria. We find wiki documentation useful for training purposes, which is easily updated and maintained. Try to accommodate both standard wikitext and WYSIWYG editors; users have split preferences. Supporting the output of notebook files to wikis or Web pages is a best practice. Also, find large-scale graph and visualization tools so that

you can prepare and distribute navigable versions of your knowledge graphs. You may also find other diagrams and flowcharts, including UML diagrams, useful for documenting and training workflows or defining use cases for tools.

While it is not yet seamlessly achievable, try to move toward single-source publishing, where one can author once and then publish selected portions in a variety of formats (HTML, PDF, doc, csv). We want wiki-like environments where multiple authors may contribute, and we have easy collaboration and rollback of versions. Simple import and export versions, such as XHTML or XML, help facilitate this, though it is still difficult to theme or lay out content easily for multiple publication venues. We also want to adopt single-source publishing environments that enable us to characterize and label workflow steps as part of our natural interaction with the content. These systems should allow user-defined steps and labels and rules.

Best practices, like worldviews, depend on the circumstance and the players. We may need to modify broad guidelines that work in general for the specific. In all cases, KR and KM systems tailored to particular needs, and scoped to specific domains, will have their own set of capabilities and configurations. Today's requirements will evolve to different ones tomorrow. The work environment in which you need to embed these systems and their workflows will vary greatly. That is why it is practical to consider standard and best practices as guidelines, and not prescriptions.

References

1. G. Hripcsak, A.S. Rothschild, Agreement, the F-measure, and reliability in information retrieval. J. Am. Med. Inform. Assoc. **12**, 296–298 (2005)
2. E. Miltsakaki, R. Prasad, A.K. Joshi, B.L. Webber, *The Penn Discourse Treebank* (2004)
3. P.V. Ogren, G.K. Savova, C.G. Chute, Constructing evaluation Corpora for automated clinical named entity recognition, in *Medinfo 2007: Proceedings of the 12th World Congress on Health (Medical) Informatics; Building Sustainable Health Systems* (IOS Press, Amsterdam, 2007), pp. 2325
4. V. Stoyanov, C. Cardie, Topic identification for fine-grained opinion analysis, in *Proceedings of the 22nd International Conference on Computational Linguistics-Volume* (2008), pp. 817–824
5. K. Dellschaft, S. Staab, On how to perform a gold standard based evaluation of ontology learning, in *The Semantic Web-ISWC* (Springer, Berlin, Heidelberg, 2006), pp. 228–241
6. KBART Phase II Working Group, *KBART: Knowledge Bases and Related Tools Recommended Practice* (NISO, Baltimore, MD, 2014)
7. M. Horridge, S. Jupp, G. Moulton, A. Rector, R. Stevens, C. Wroe, *A Practical Guide to Building OWL Ontologies Using Protégé and CO-ODE Tools* (University of Manchester, Manchester, 2007)
8. E.P.B. Simperl, C. Tempich, Ontology engineering: a reality check, in *On the Move to Meaningful Internet Systems* (Springer, New York, 2006), pp. 836–854
9. F. Giasson, *Exploding the Domain* (Frederick Giasson, 2008)
10. D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J.-F. Crespo, D. Dennison, Hidden technical debt in machine learning systems, in *Advances in Neural Information Processing Systems* (2015), pp. 2503–2511
11. K. Jalan, *How to Improve Machine Learning Performance? Lessons from Andrew Ng*. https://www.kdnuggets.com/2017/12/improve-machine-learning-performance-lessons-andrew-ng.html
12. D. E. Knuth, Literate Programming. *The Computer Journal* **27**(2), 97–111 (1984)

Part V
Practical Potentials and Outcomes

Chapter 15
Potential Uses in Breadth

We begin this last *Part V* looking at *potential* applications. These knowledge management uses, made possible by following Peirce's guidelines, leverage KBpedia and domain extensions to it. I have assembled these examples to illustrate our intent in this *practionary* to attain what Peirce called "the third grade of clearness of apprehension":

> "It appears, then, that the rule for attaining the third grade of clearness of apprehension is as follows: Consider what effects, that might conceivably have practical bearings, we conceive the object of our conception to have. Then, our conception of these effects is the whole of our conception of the object." (1878, CP 5.402)

It is, of course, impossible to conceive of all practical effects from a thing. However, in this chapter, and the one that follows, I try to share what I see as some important practical effects of applying Peirce's guidelines to knowledge representation. To my knowledge, few have implemented the ideas listed in this or the next chapter. The practical effects of these ideas are strong potentials with reasonable prospects for being realized. These ideas, collectively, help us begin to apprehend this 'third grade' of clear understanding.

I have selected these case examples both to highlight the diversity of potential uses and to showcase those with the highest likelihood of impact. Because what manifests in the future often 'surprises,' I am likely overlooking some impactful and practical effects of what may unfold in the future. Nonetheless, this method of selection does conform to what Peirce called the pragmatic maxim as a way to sift through the myriad of possible explanations for things to focus on those with the most economy and likelihood of bearing fruit.

I introduce each case with some context and a problem statement, then an introduction of concepts and existing building blocks that pertain to it, and then possible generalizations and potential practical effects were the case implemented. I do not exhaust the potential high-impact applications in these chapters. Recall that we provided a long list of other possible uses of our approach in Table 4.1 in Chap. 4. Consult that list for a fuller picture of potential applications. You will see, for

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8_15

example, that the case studies in this concluding *Part V* do not include applications such as ontology-driven applications (ODapps), concept alignment, entity and concept extraction, and semantic search, to name just a few of the important missing ones.

In this first concluding chapter, I briefly present about a dozen possibilities in *breadth* that introduce a variety of practical KR uses. These possibilities are more of an overview than the in-depth cases in the following chapter, but in their totality provide a good sense of potentials. We split these dozen possibilities into near-term potentials, logic and representations, and other more speculative potentials. The order of presentation is from the near at hand to the speculative. In the next Chap. 16, we present three practical applications, also pretty near at hand, in a considerable amount of *depth*. The combination of these two chapters of breadth and depth, the dimensions of Peirce's definition of information (Chap. 2), broadly captures the sense of practical and potential uses of our approach. Chapter 17 concludes this last *Part V* of the book, recapping Peirce's guidelines for knowledge representation.

Near-Term Potentials

We have already discussed how ontologies may drive bespoke applications and Web services. We have seen the importance of organizing attributes and mapping to them for instance characterization and intensionality. Four further potentials are also near at hand in word sense disambiguation, relation extraction, reciprocal mapping, and extreme knowledge supervision. These potential applications all are examples of leveraging the rich structure of KBpedia and its extensions.

Word Sense Disambiguation

Word sense disambiguation is picking the correct meaning for a word where it has multiple meanings.[1] Vocabularies grow by either minting new words or giving new meanings, also called *senses*, to existing words. Multiple senses for common words is a historical linguistic result of the bifurcated chaining of new word senses for new uses based on adjacent metaphors [1]. This mode of how new word senses get coined conforms to the least 'cognitive cost' for generating, interpreting, and learning them [2]. Some of these senses, such as *game* for *hunted fowl* or *game* for an *amusing pastime*, may have diverged long ago with a broad span of meaning.

The traditional approach to word sense disambiguation (WSD) uses dictionaries to look up the various senses of a word. Lesk is a leading method, wherein we

[1] WSD is also closely related to named-entity recognition or named-entity disambiguation. The dictionary basis shifts from word senses to entity characterizations (attributes), but much else in approach is similar.

search the various word senses in a dictionary based on the neighboring text for the search term. The Lesk algorithm calculates the overlap of the sense definition of a word and the contextual definitions of the terms that surround its use, with variants allowing us to control the sliding window or other parameters [3]. The limitation of the Lesk approach is that it depends on the wording of the definitions. We may also base word embeddings on other factors, including structure and other features [4].

Unsupervised learning surfaces other rules and insights useful to WSD. Nearly a quarter-century ago Yarowsky showed a strong tendency for one word sense per discourse and collocation [5]. Choosing the most frequent sense for a multisense term is one of the best performing heuristics [6]. On a more abstract front, Sun has shown a framework that regularizes the structure of feature-rich corpora, which can derive training models that can converge rapidly and reduce generalization risk [7].

Methods for word sense disambiguation may also learn from large knowledge sources, with Banko and Brill providing one of the first [8]. One of their findings was that the larger the number of annotations for term entries, the better the resulting accuracy. More recently, Ponzetto and Navigli have demonstrated that knowledge harvested from Wikipedia can be efficiently used to improve the performance of a WSD system [9]. Adding Wikipedia links to baseline approaches can further enhance disambiguation performance [10]. Still, WSD for state-of-the-art systems has 2–5% error, not including inter-annotator differences. These performance figures are also for very limited domains with corpora and training sets known in advance. Word sense disambiguation applied to new domains needs to overcome what is known as the knowledge acquisition bottleneck, which is the cost of finding, structuring, or annotating knowledge for WSD and other natural language processing applications. Many difficulties occur in acquiring tagged senses for WSD.

The potential of KBpedia and how it is structured to improve the WSD picture are profound. First, we have an instance-rich knowledge structure. Not only does that structure bring direct benefits, but also the hundreds to thousands of instances per type also provide a rich content base for various word- and subgraph embedding models. Second, the KBpedia structure is coherent. Third, we base KBpedia on Peircean ideals of knowledge representation. Its features are mostly lexically based (relations, attributes, senses, and meanings), which means that abstraction layers through the use of neural nets have a higher prospect for being interpretable (and coherent). Fourth, because of the degree of semantic relatedness in the structure, chances are greater that neighbor-based methods to WSD will perform better than alternatives. Fifth, the KBpedia features, as *Appendix C* describes, are a richer base for structure regularization methods than what Sun has analyzed [7].

So, what we see with a KBpedia-based approach to WSD is one that combines all of the best methods in a single package. Its contextual understandings can extend to entity recognition and disambiguation, as well as concepts and relations. KBpedia's graph structure, with its emphasis on trichotomies and typologies, should also promise better performance because of its comparative simplicity and cleanliness. We have strong dictionary and synonym bases, combined with a coherent and robust graph structure with millions of instances with content, which is expandable for new domains, and testable with the potential for continuous improvement.

Relation Extraction

In the context of relation extraction, most define 'relation' as a form of connection between two objects. The objective of relation extraction, then, is to identify and extract this relation. In contrast, we have seen in the context of Peirce that a general relation may specialize into one of the three forms: attributes, external relations, or representations. Our Peircean approach also gives us better tools to identify and extract general relations, and then to organize and reason over them.

Relation extraction attempts to correctly identify and extract what is essentially an RDF triple of *subject-predicate-object*. Sometimes the subject placeholder is blank or unknown; sometimes the object placeholder is blank or unknown. (Theoretically, we could also treat the predicate slot as a blank.) Because of these structural aspects to a relation, the earliest forms of extraction put forward by Hearst in 1992 used many heuristics applied to lexico-syntactic patterns [11]. These techniques are surprisingly effective for many relation patterns; many systems still use them. The kernel method builds on this approach by looking for patterns within generalized tuples. Supervised approaches can also work quite well since we can pose the problem as one of binary classification. Relation extraction was also one of the first applications of the use of knowledge bases to inform labeled examples, what we now call distant supervision [12], which remains one of the better performing methods. More recently, joint inference on both entities and relations looks to improve extraction efficiency further [13].

Relation extraction has some unique uses within NLP methods. First, of course, it is the method for extracting relations (though, as mentioned, this has not yet been distinguished from attributes and representations). Second, we may find patterns to help narrow the identification of new concepts or entities by analogy to existing complete patterns. In the most effective sense, we should be able to narrow the applicable types for the new concepts or entities as well, but that is little applied. The potential exists to improve significantly our ability to identify previously unseen entities, not already in our dictionaries or gazetteers. Third, because of its patterned nature, we also value relation extraction as a technique used in data mining and question answering [14]. Last, the potentials for relation extraction are even more vast, which I get to in a moment.

The TextRunner and then KnowItAll and ReVerb efforts from the Etzioni lab at the University of Washington, and more recently the Nell project from Carnegie Mellon University, have been mining Web sources to discover relations and their associated entities. These efforts use open information extraction as a technique for knowledge base population. These approaches are useful, for example, to identify new entity members for specific types, sometimes called 'slot filling,' with millions of candidates identified. Another application is to disambiguate entities based on context. Besides these university efforts, commercial entities have been doing the same. Still, relation extraction is a comparatively inaccurate NLP task due to the variability in the triple structure in language and the immense number of potential entities.

KBpedia can improve all aspects of extracting, identifying, reconciling, and organizing the three aspects of relations, which also should lead to new capabilities in ontology learning and better capabilities in question answering and data mining. Inspections of the object slot may also aid in error detection of values and other possible misassignments. We can realize these potentials due to the better characterization and structure of KBpedia.

If we someday want to create ontologies from raw input text, the dream of ontology learning, we will require broad and accurate characterizations of relations to decompose the meaning of text structure. We have already mentioned the fine-grained structure of relations in KBpedia. The three segments of attributes, external relations, and representations, organized by types, provide better structure for evaluating relations. An initial task is to inspect and map relations from VerbNet and the open IE projects. The Nell project also provides domains and ranges, which should be helpful to relate types to specific predicates. These characterizations, in turn, would enable better mapping and inference of entity types to predicates and other patterns. The feedback from this process would undoubtedly surface improvements to KBpedia, which would feedback into better extractions anew. Computerized machine reading or natural language understanding will need these capabilities. The area of relation extraction should be a fruitful research focus for many years to come [15].

Reciprocal Mapping

The standard method of mapping is to relate new concepts and entities in an external knowledge 'source' (B) to the master or governing one in a 'target' resource (A). The use case typically uses the target resource as a reference for external sources, possibly for data federation or integration. The mapping statements take the form of A:B or B:A. However, the external source may also be a valuable contributor to new concepts or entities for the target resource. In this use case, our interest is adding more A' to A, rather than simply mapping statements. We call this use case 'reciprocal mapping,' a topic in Chap. 13. Reciprocal mapping is not warranted in all cases, and only best applies when we encounter a quite complete external source, as is the case of Wikipedia contributing to KBpedia.[2] It is also a particularly useful technique where one wants to augment an existing knowledge graph, perhaps in adding domain extensions to a starting basis in KBpedia.

First, let's assume that we have already mapped the matching concepts between B → A and B itself is a rich external source.[3] What we want to do is to use this linkage to propose a series of *new* subclasses that we could add to A (KBpedia in our

[2] Significant portions of this section are drawn from the KBpedia Web site for its reciprocal mapping use case; see http://kbpedia.org/use-cases/extending-kbpedia-with-kbpedia-categories/

[3] Any sufficiently complete or robust external ontology closely related to the current domain needs may fulfill this role.

example case) based on the subcategories that exist in B for each of these mappings. The challenge we face by proceeding in this way is that our procedure potentially creates tens of thousands of new candidates. Because the B category structure has an entirely different purpose than the KBpedia knowledge graph (A, in this case), and because B's creation rules are completely different from those of A (KBpedia), many candidates are inconsistent or incoherent to include. A cursory inspection shows that we should drop most of the candidate categories. It is not tenable to review hundreds of thousands of new candidates manually, as is the case when B is the size of Wikipedia; we need an automatic way to rank potential candidates.

Several factors differ for reverse (reciprocal) mapping from our standard B → A mapping case. First, we need to find missing clusters or new concepts or types in B that fit, but are missing, in A. Second, we need to ensure that the scope and boundaries of concepts or types in B are roughly equivalent to those in A. We may expend considerable effort to clean the source B type and category structure prior to the reciprocal mapping. Third, we also need to capture structural differences in the source knowledge graph (B). Possible category matches fall into three kinds: (1) *leaf* categories, which represent child extensions to existing KBpedia (A) terminal nodes; (2) *near-leaf* categories, which also are extensions to existing KBpedia terminal nodes, but which also are parents to additional child structure in the source; and (3) *core* categories, which tie into intermediate nodes in KBpedia that are not terminal nodes. By segregating these structural differences, we can train more precise placement learners.

We automate this process with an SVM classifier trained over graph-based embedding vectors generated using the DeepWalk method [16]. DeepWalk learns the subcategory patterns that exist in the B category structure in an unsupervised manner. The result is to create graph embedding vectors for each candidate node. Our initial B → A maps enable us to create training sets with thousands of preclassified subcategories quickly. We split 75% of the training set for training, and 25% for cross-validation. We also employ some hyperparameter optimization techniques to converge to the best learner configuration. Once we complete these three steps, we classify all of the proposed subcategories and create a list of potential subclass of candidates to add into KBpedia, which we then filter by relevance score and vet manually.

The reference 'gold' standards in the scored training sets (see Chap. 14) provide the basis for computing all of these statistics. We score the training sets as to whether a given mapping is true or false (correct or not). (False mappings should be purposefully introduced.) Then, when we parse the test candidates against the training set, we note whether the learner result is either positive or negative (indicated as correct or indicated as not correct). When we match the test to the training set, we thus get one of the four possible scores: true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). Those four simple scoring categories are sufficient to calculate any of the statistical measures, as we discussed in Chap. 14.

We capture the reciprocal mapping process using a repeatable pipeline with the reporting of these various statistical measures, enabling rapid refinements in parameters and methods to achieve the best performing model. Once appropriate candidate

categories are generated using this optimized model, we then manually inspect results and make final selections. We then run these selections against the logic and coherency tests for the now-modified graph and keep or modify or drop the final candidate mappings depending on how they meet the criteria. Our experience suggests that this semiautomated process may take as little as 5% of the time it would typically take to conduct this process by comparable manual means.

So, machine learning methods may reduce the effort required to add new concepts or structure by 95% or more. Machine learning techniques can filter potential candidates automatically to reduce greatly the time a human reviewer has to spend to make final decisions about additions to the knowledge graph. A reusable pipeline leads to fast methods for testing and optimizing parameters used in the machine learning methods. We can systematically tune and rapidly vet this pipeline.

Extreme Knowledge Supervision

Recall from Chap. 4 that *knowledge supervision* is the purposeful use and structuring of knowledge sources and graphs to provide features and training sets for *KBAI*. *Distant supervision* uses the same sources, though employed as is and not purposefully staged. In knowledge supervision, we design and prep the knowledge base so that its structure enables query selection of labeled positive (and, with repeatable techniques, negative) training sets for supervised machine learning. This pre-staging of the knowledge sources eliminates 80% of the effort or more required for most supervised learning tasks. We also showed a virtuous circle of interaction between properly designed knowledge bases and a knowledge graph such that we can add new assertions and facts to the knowledge base and improve its quality by a higher ratio of true positives (see Fig. 4.2).

When repeatedly and purposefully carried out through many cycles, we can call this *extreme* knowledge supervision. In the case of KBpedia, remember, we already have important structural splits between concepts, entities, events, attributes, external relations, and representations, all organized according to the triadic *universal categories* of Charles Peirce, and further subtyped by scores of modular *typologies*. Theoretically, we may use the intersection of any of these dimensions to create and train supervised learners. Further, because of this richness of structure, we also can develop better language parsers (see Chap. 16) and reasoners (see next) to apply to our tasks. Also, combinations of these features through inference over category structures are a patented way[4] that brings significant efficiencies [17]. Here is the breadth of tasks to which we may apply extreme knowledge supervision:

[4] See https://patents.google.com/patent/US8484245B2/en, "Large Scale Unsupervised Hierarchical Document Categorization Using Ontological Guidance," Viet Ha-Thuc and Jean-Michel Renders, 2013, US Patent No. 8484245B2.

- Entity identification (recognition) and extraction
- Attribute identification and extraction ('slot filling')
- Relation identification and extraction
- Event identification and extraction
- Entity classifiers
- Phrase (n-gram) identification
- Entity linkers
- Mappers
- Topic clusterers
- Topic classifiers
- Disambiguators
- Duplicate removal
- Semantic relatedness
- Inference and reasoning
- Subgraph extraction
- Ontology matchers
- Ontology mappers
- Sentiment analysis
- Question answering
- Recommendation systems
- Language translation
- Multi-language versions
- Artificial writing
- Ongoing knowledge base improvements and extensions

I have listed these areas in rough order from the simpler to the more complex analyses. Distant supervision efforts have concentrated on information extraction, the first items on the list. However, all are amenable to knowledge supervision with ML.

A vetted knowledge graph with millions of supporting instances also provides some graph-level benefits. The first area is in 'deep graphs' [18]. The basic idea behind 'deep graphs' is to segregate graph nodes and edges into *types*, which form supernodes and superedges, respectively. In our terminology, 'deep graph' node types are akin to types of similar *attributes,* and edge types are akin to types of *relations*. The 'deep graph' algorithm can partition these grouped types into lattices, which can be intersected (combinations of nodes and edges) into representing deeper graph structures embedded in the initial graph. We can use these deeper graph structures as new features for machine learning or other applications. A second area, important to data interoperability, is in 'symbol grounding' [19] (also see next chapter). The usefulness of symbol grounding resides in associating symbol tokens as understood by the computer with actual language meanings. Besides interoperability, such groundings are crucial to natural language understanding.

The idea of large knowledge bases providing enabling technology for knowledge sharing goes back at least 30 years [20]. We are still in the early phases of such iterative refinements of KBpedia. As this process continues, expect to see faster and

more accurate learners, the incorporation of still additional knowledge sources and datasets, and more sophisticated combinations of features and methods for extreme knowledge supervision. Song and Roth provide an excellent current survey with hundreds of references for how machine learning based on using world knowledge may create such potentials [21].

Logic and Representation

The previous section begins to scratch the surface for how KBpedia, as structured using the guidelines of Peirce, may improve many knowledge-based tasks, especially in the areas of natural language processing (NLP). I would now like to move beyond this traditional baseline and address more fundamental questions of logic, reasoning, and representation. These kinds of fundamental questions can take the use and contributions of knowledge-based systems to new levels. The four initial topics we cover in this section include automatic hypothesis generation, encapsulating KBpedia for deep learning, measuring classifier performance, and the thermodynamics of representation itself.

I do not touch on all of the logical potentials in this section. For example, the use of fuzzy logic, or intensional logic, or methods of inductive reasoning provides enormous potential. Areas in nonclassical logics such as three-valued logics [22] or triadic logic [23] also deserve attention given their relationship to Peirce's universal categories. These are worthy topics for future attention.

Automatic Hypothesis Generation

One of Peirce's signal contributions was to bring the importance of abductive reasoning to the fore in matters of epistemology. We discussed the now three classical logical methods of deduction, induction, and abduction in Chap. 8. *Deduction*, the most widely employed method in the semantic Web and knowledge graphs, evaluates correct placement by traceable logic chains, most of a hierarchical nature. *Induction*, little used but with great promise for knowledge graphs, can look to shared or common features to make probable assertions. *Abduction*, which Peirce brought to the fore, is the logic of new knowledge and scientific discovery. It is rarely used and not well understood, some due to Peirce's own changing views.

What Peirce early called abduction he later acknowledged was, in fact, induction. Peirce's confusion—and how he eventually worked out the issue—is instructive. What Peirce initially called abduction is what we now call inference to the best explanation (IBE). The basic idea is that given a particular outcome, what is the most likely path through the knowledge graph that leads to that outcome. It is a form of backward chaining, where all parts of the syllogism are known, and therefore it is a true inferential method. Still, many combinations are possible, and reasoning

backward across available choices can soon become computationally intractable. Since in abductive reasoning we are ultimately seeking the explanation to a question or phenomenon, this kind of IBE reasoning is quite valuable for knowledge graphs in general [24] and has applicability to instance characterizations in the ABox as well [25].

Still, this view of abductive reasoning is but a part of what Peirce intended in his mature formulation. Peirce was seeking no less than an understanding of how the scientific method (purposeful inquiry) worked and its logic, in a broad sense. His characterizations redound with expressions of 'surprising facts,' 'flashes,' 'guesses,' 'instinct,' and 'new knowledge.' Dewey, a fellow pragmatist, saw similar things, but particularly looked toward abductive reasoning also as a way to explicate learning [26]. Peirce well understood the combinatorial problem and sought to understand how we winnow through the myriad of options, recognizing the factors of economy, effort, likelihood of producing results, and all of those things we now understand as 'pragmatic.' Peirce understood that there was a transitional space between perception and hypothesis that held the key to this unique logic. Flach, throughout his many writings, has noted the importance of abductive and inductive logic to the development of scientific knowledge, and also usefully split Peirce's ideas of abduction into explanatory and confirmatory reasoning [27]. The nut to crack around abduction resides in explanatory reasoning. Flach has attempted to refine Peirce's conception of explanatory reasoning into a form amenable to logical analysis [28].

Prying open the heart of the logic of science is an exciting prospect. Kapitan made a powerful argument for why IBE was not the nub of abductive reasoning, and suggested that heuristic aids, while not inferences, could still be used to discover new knowledge, often based on analogy [29]. Kapitan also compiled eight reasons from Peirce for what we should seek in a candidate hypothesis to explain an observation or a surprising fact:

1. The cost (in time, money, and effort) of testing the hypothesis (1901, CP 6.533; 1901, CP 7.230)
2. The intrinsic value of the hypothesis regarding its 'naturalness' and 'likelihood' (1901, CP 7.223)
3. The fact that the hypothesis can be readily broken down into and elements and studies (1901, MS 692:33)
4. The simplicity of the hypothesis (*i.e.*, it is more readily apprehended, more facile, more natural or instinctive) (1902, MS L75:286; 1901, CP 6.532; 1908, CP 6.477)
5. The breadth of the hypothesis or the scope of its predictions (1902, MS L75:241, 457:37)
6. The ease with which we may falsify the hypothesis (1902, MS L75:285)
7. The testability of the hypothesis using severe tests based on 'incredible predictions'
8. The analogy of the hypothesis with familiar knowledge (1901, MS 873:16).

These guidelines feel incomplete. As part of his treatment of logic within the universal categories, Peirce held abductive reasoning as irreducible from the other two forms of logic, deductive (2 ns) and inductive (3 ns). We are still missing the essence of what makes abductive reasoning different. If we can truly get at the essence of the scientific method and purposeful inquiry, we will have unlocked a tremendously powerful door to new knowledge and discovery.

Kapitan held that missing piece was the creative, what it is that underlies knowledge [30]. He did not see this as an inferential step, but as one 'suggested' by the facts, by a general cognition. Kapitan likened the transition from the perception that leads to the idea as arising by analogy, from the unconscious. Selected quotes by Peirce support parts of this interpretation.

More recently, Tschaepe questioned some of this interpretation, choosing to focus more on 'guessing' [31]. Successful guessing is both piecemeal and done in an orderly fashion, guided by ethics and aesthetics, situated to logic as Peirce did. Tschaepe notes that a more metaphorical kind of logic is in play, and is indeed playful ('musement' in Peirce's term). Some scholars see it as likely based on the detection of patterns. Yes, the process is logical in a broad sense but is also a rapid surfacing and evaluation of candidate explanations arising from patterned similarities and metaphors. This critical stage between perception and hypothesis evaluation is a multifactorial, synthetic, broad contrast of iconic options rapidly screened for pragmatic likelihood. The methods of this critical phase in abduction appear more oriented to pattern matching than inference, which, in any case, appears weak. Once a potential hypothesis is chosen for some level of evaluation, it becomes indexical.

KBpedia, or its derivatives, has the raw grist to begin feeding tests of these broad factors. In the near to intermediate term, backward chaining and IBE look quite tractable within the KBpedia structure. Longer term, however, getting at the true 'guessing' game involved with abductive reasoning—unique and broad—is where, I think, some surprisingly useful payoffs may result where KBpedia may contribute.

Encapsulating KBpedia for Deep Learning

Geoffrey Hinton is a founder of deep learning. He and his team at the University of Toronto helped promote the idea of backpropagation as a way to send weights to adjust supervised labels to unsupervised layers in a neural network, with the increasingly propagated layers leading to the term of deep learning. Deep learning is exceptionally effective for image and pattern recognition tasks, less so for natural language. Unfortunately, the representations at all layers of deep learning are opaque, meaning we can glean no meaning from the information at a given layer. This 'black box' aspect is the weakness of deep learning. The concern, of course, with methods that lack explanation is that it is hard to know how to make further

improvements. Inexplicable methods always seem to top out at some limit of performance.

Hinton likely understands these limits better than anyone. Well before deep learning became such a buzz phrase, Hinton and his team in 2011 were experimenting with how to package features together to act as a unit during the deep learning process [32]. Hinton's group has been more focused on image representations than text. Still, this paper was the first to mention of defining these feature packages as 'capsules.' Hinton has continued to work on this 'capsule' concept and has come to understand that clean features about single entities are the best ones to include [33].

'Capsules' may offer a path for better aggregating natural language features into discernable packages. KBpedia's unique way of organizing and classifying related feature types based on the universal categories may also offer a better way to create meaningful 'capsules' for NLP. The 'capsule' approach, or other similar ways to package features into meaningful sets, may provide the missing technique for making deep learning more understandable in the context of natural language.

Measuring Classifier Performance

We presented statistical measures for binary classification and NLP tasks in Chap. 14. We touched upon but did not elaborate two additional measures of ROC and AUC. ROC, the receiver operating characteristic (also called the relative operating characteristic), is a curve that plots the true-positive rate versus the false-positive rate at various settings. AUC measures the area under this curve and reduces the standard error from the use of ROC alone. Researchers use these two measures to compare the performance of machine learning classifiers, though they are noisy methods with challenges in interpretation [34]. We need better performance measures.

In the 1930s the Italian statistician Bruno de Finetti wrote much on probability [35], and likely was instrumental in resurrecting interest in Bayesian conditional probabilities. de Finetti developed a method of plotting three variables against one another called the ternary plot. It has found wide use in genetics, for example, in plotting the frequency of diploid genes (AA—Aa—aa) against one another using the display within an equilateral triangle, which can, for example, capture the distribution of the Hardy-Weinberg frequency of a gene, a standard measure.

About 15 years ago, the Spanish statistician Valverde-Albacete and his team adopted the de Finetti ternary plot to provide a more accurate means to compare machine learning classifiers. The plot uses the three corresponding values of change in entropy, versus what they termed the variation of information, and the mutual information surfaced by the classifier [36]. The group calls this display the 'entropy triangle.' One can see a striking resemblance of these de Finetti entropy triangles to the semiotic triangles of Peirce (see Fig. 2.1). Further, the relation to Shannon entropy and the potential correspondences to object-representamen-interpretant at the apexes also draw attention. Though tentative, intuition about these correspon-

dences suggests two possible lines of inquiry. First, we may apply de Finetti ternary plots to a more quantitative treatment of the Peircean sign representation. Second, the existing entropy calculations and insights might have either a Peircean interpretation or an applicability to signs about Shannon information theoretics. For now, we should view these correspondences as wholly speculative, but thought provoking nonetheless. Whether these intuitions bear fruit, the apparent superiority of the entropy triangle as a measure of classifier performance remains.

Thermodynamics of Representation

The close relation of information to energy as discussed in Chap. 2—and the findings of Landauer showing the energetic and physical aspects of information—provides possible guidance for how we should think about and model knowledge representations going forward. Susanne Still has taken this viewpoint to heart, and routinely uses the thermodynamic and informational aspects of information engines in her work [37]. This area, too, applies to measuring classifier performance, as well as other relevant topics.

For example, Still has shown information engines to require predictive inference to function well, which requires memory and favors a minimum of redundant information. In nonequilibrium conditions (namely, life), the most favored information engine is that which is most efficient in predictive power for a given level of memory. Of course, no information engine may extract more work than is contained in its useful informational inputs, and the best engines use more available information and dissipate less. (Dissipation under nonequilibrium conditions is average work minus the change in nonequilibrium free energy.) Still has also related her work to learning theory [38], data representations [39], and information bottlenecks [40].

The idea of information bottlenecks to test for better data representations or better predictive inferences is but one method where we may exploit the convergence of information theory and knowledge. It is clear that we can apply these methods of entropy measurement to help screen data representations and models and even to test model parameters. These kinds of tests are hardly standard in ontology building and maintenance, though such efforts using the proxies of information engines provide a useful means for doing so.

We can apply these same perspectives and tests to evaluate the use of Peirce's universal categories as an organizational framework for knowledge graphs. We also should consider monitoring reference concepts by use to discover over-specified or redundant information in our systems. As was pioneered in biomedical research with 'knockout' mice, we can remove selected pieces or portions of our knowledge graphs to measure their after and before information theoretic contributions.

As we pull together more evidence for the linkage between information theory and various entropy and free energy measures, we will undoubtedly discover more insights regarding composition and construction of our knowledge systems to make them more efficient. The beautiful thing about information-theoretic metrics is that

we can negate empty arguments about philosophy or ideology and focus on what works with the most efficiency, a clear reflection of Peirce's admonitions for pragmatism. Routinely testing for information bottlenecks should also aid our ability to continue to refine better performing predictive inferences. Still states,

> "Predictive inference can be interpreted as a strategy for effective and efficient communication: past experiences are compressed into a representation that is maximally informative about future experiences. The information bottleneck (IB) framework can thus be applied, either in a direct way, or in its recursive form (RIB). Both methods find, asymptotically, the causal state partition, i.e., minimal sufficient statistics. RIB additionally recovers, asymptotically, the ε-machine, which is a maximally predictive and minimally complex deterministic HMM [hidden Markov model], believed to be the best predictive description of a stochastic process that can be extracted from the data alone." (p. 985)

It appears pretty evident that we should adhere more to energetic factors (dissipation, entropy) in evaluating alternatives. These methods may also help us better quantify the benefits of organizing our knowledge structures using Peirce's universal categories and typologies as compared to traditional dichotomous representations.

Potential Methods and Applications

New applications and uses for knowledge graphs remain untapped. We have listed some of these areas as potential applications for years, such as self-service business intelligence or semantic learning. We conclude this section and chapter by discussing the relation of Peirce's ideas and guidance to nature and questions of the natural world.

Self-Service Business Intelligence

I have been hearing about self-service business intelligence for more than two decades, yet it remains as elusive as ever. The definitions have changed over time and now include concepts like 'big data,' but the basic idea is to enable users, who lack IT or coding skills, to access enterprise data for their queries and reports [41]. The genesis of the idea arises from the promise of placing data analytics directly in the hands of the users who need it, matched with frustrations for how long specific requests to IT for queries or reports take to fulfill [42]. Part of the problem in achieving this ideal is that parties tackled early attempts at self-service BI as some new application, only 'dumbed down' with slick user interfaces (UIs) to overcome the lack of computing skills by its users. In retrospect, it is not hard to see how attempts to fulfill this need settled upon supplying still another application as a separate product. Enterprise-level applications were the rage over those same decades. Naturally, to address the need of business analysts, the trick was to modify the business

intelligence tools, such as they were, used by IT and then repackage them for easier use. The joke through at least the 1990s was that an 'executive information system' was the one with the big buttons with the big labels.

Those older visions fail for at least two reasons. The first reason is to consider business intelligence as some form of separate application. Early attempts at business intelligence or data warehousing failed and disappointed at high rates. We discussed at length in Chaps. 3 and 4 the challenges in data interoperability and impediments to information access and sharing. The general challenges of business intelligence and knowledge management remain unsolved. The second reason for failure is to consider the hurdle for nontechnical users as mainly one of user interfaces. Sure, UI considerations are important. However, the real hurdles are fitting with existing work tasks and flows. The users of business intelligence create that intelligence. These knowledge workers must be involved in feeding and adding to the enterprise knowledge stores, as well as tapping them. Knowledge workers should steward their knowledge assets. This imperative needs to put users in the knowledge recording role, as well as the knowledge using one. Knowledge is not an afterthought, but part and parcel of the daily activities seamlessly integrated into current work tasks and flows.

Though KBpedia and its structure are well suited to knowledge capture and use, the question of self-service goes beyond that. Self-service is not a matter of user interfaces and buttons, though at some point those items are worthy of attention, but a matter of mindset and making knowledge management integral to current work tasks. As we discussed in Chap. 12, this approach includes being attentive to workflows and piecing apart specific tasks such that they can integrate well with current daily activities (see further Chap. 16). As for knowledge creation, we must integrate new concepts and add and modify instances as we encounter them. These activities occur while researching online, writing or reading documents, or interacting with co-workers and colleagues.

We need to deploy our specific KM apps where we engage in these activities—be they browsers, word processors, spreadsheets, calendar systems, or chat. BI systems would benefit from a similar distribution with standard work tasks. We want to encourage continuous access and constant availability. In these senses, we solve the UI challenge more by embedding knowledge functionality in existing applications than by dashboards or big buttons. While we have not been prescriptive in this book, I do think that the guidelines we offer provide pragmatic approaches for how to adopt self-service ways in your organization.

Semantic Learning

Many aspects of what some anticipated as a semantic learning Web by 2020 have failed to materialize [43]. We have tried and used both latent and explicit ways to learn from text. We do not have multiple knowledge bases talking to each other or annotated or guided educational resources or commercial semantic browsers. We

lack the connectedness portions of the vision. We have achieved talking to personal devices and leveraging massive knowledge bases like Wikipedia, mostly through supervised means, but the learning and interoperability aspects still appear weak. The lack of connection or connected learning sources is not one of technology or standards but provenance and authoritativeness. We have learned in our two decades of using the Web that it is a medium as prone to spamming and misuse as it is for access and convenience. We have found that the latent methods, applied to either text or images, do not perform as well as supervised methods. Still, though, even with supervised learning, we do not see much active learning or connectivity (defined as two separately maintained sources interacting automatically with one another).

We will not see marked improvements in latent semantic indexing—and unsupervised methods in general—until we have better parts-of-text segmentation and classification. We need a true foundational set of semantic primitives. I believe Peirce offers such (see next chapter). We have not yet tested this premise. Further, with its graph structures and inherent connectedness, we also have some exciting graph learning methods that we can apply to KBpedia and its knowledge bases. The perhaps best known method for conducting unsupervised learning on a subgraph is the k-nearest neighbor method, with the latent Dirichlet allocation and conditional random field (CRF) methods growing in popularity. We also have emerging subgraph alternatives. With KBpedia's rich feature set, we have many additional options for discovering better performing semantic learning. Whether the approach is Peircean or not, we likely need to see a more grounded set of semantic primitives emerge before we see production-grade performance with latent indexing or vectors. Without these primitives, there remains too much of a 'black box' aspect for these methods, similar to what we see with the opaque explanations for deep learning.

We do, however, have adequate means for production-grade methods for meaningful semantic connections using supervised learning with human editorial vetting. We need to take care of what resources we select for our learning purposes. We need automated ways to screen through the myriad of candidates. Then, we need to review those manually that remain ambiguous after tests, feeding our final selections back into the system to improve the performance of the learner when next used.

KBpedia thus is a potential contributor to semantic learning in two ways. The first way is to move toward a more logical, defensible set of semantic primitives for characterizing and indexing text, perhaps including unsupervised methods. The second way is by mindset and example, where builds and testing are constant against an already coherent structure. A key insight is in how to construct and maintain our knowledge structures. Users of the open Web, as is, do not trust it as a coherent knowledge source. Still, we will use quality sources, determined by editorial oversight or supplied by trusted brands. We need to discriminate and then depend on vetted resources, like from industry standard groups or proven resources like the Wikimedia properties. A key lesson is that we cannot fully automate the entire process of discovery, harvesting, vetting, and connecting; humans must be in the loop, only accepting what meets editorial standards.

Nature as an Information Processor

It is clear that information is central to the idea of life (through DNA) and language and communications (through symbols). We also saw in the discussion in Chap. 2 that Landauer had shown the physical nature of information and from Jaynes onward that many had pointed to the energetic nature of information. These indicators suggest that nature acts as an information processor.

The least controversial interpretation of information processing in nature occurs through genetic and cultural information. This overlap has led Sweller and Sweller to posit five common principals of natural information processing systems, which I have taken the liberty to edit slightly [44] (Table 15.1):

Table 15.1 Natural information processing system principles

Principles	Cognitive case	Evolutionary case	Function
Store information	Long-term memory	Genome	Store information for indefinite periods
Borrow and reorganize	Transfer information to long-term memory	Transfer information to the genome	Permit the rapid building of an information store
Random genesis	Create novel ideas	Create novel genetic codes	Create novel information
Narrow limits of change	Working memory	Epigenetic system related to environmental information	Input environmental information to the store
Organize and link	Long-term working memory	Epigenetic system related to genetic information	Use information from the information store

Wiesner—after reviewing developments in dynamical systems theory, information theory, physics, and computation theory—goes much further [45]. She claims that formal language theory, such as the examples of transformations provided by Noam Chomsky [46], provides the key to understanding information processing in natural systems. Her synthesis leads to methods based on how quantum processes store and manipulate information, what Wiesner calls 'intrinsic quantum computation.'

In a broader sense, the mathematician Burgin and his co-authors over the years have been looking at commonalities and classification of various kinds of computational algorithms (for example, see [47]). Burgin claims that the basic structure of the world is triadic (physical, structural, mental), which corresponds to Plato's triad (material, ideas/forms, mental) or may be related to Peirce's semiotic sign triad of object, sign, and interpretant. This existential triad leads Burgin and Dodig-Crnkovic to propose the three following types of computations [47]:

1. Physical or embodied (object) computations
2. Abstract or structural (sign) computations
3. Mental or cognitive (interpretant) computations.

The authors note that the abstract or mental forms are themselves based on physical or embodied computations. In any case, the authors stress that we need a much better understanding of computation as an activity of information processing.

Quax, I believe, in his 2013 Ph.D. thesis [48] and associated papers, may have done just that. Returning to the roots of computation in Shannon information theory, as discussed in Chap. 2, Quax et al. notes that the topological analysis of network interactions, while often posited as an explanatory basis, has proven insufficient to identify which nodes "drive the state" of networks [49]. Their idea, which supplements the topological relationships, is grounded in Shannon entropy and mutual information. Information theory is often applied to statistical inference when an external observer describes the state of a system. As applied to dynamical systems, such as knowledge systems, each component of the system (*e.g.*, a chunk of information) is an observer that stores the information and records state.

Quax and his co-authors derive two dynamic measures from these aspects of Shannon information. First, the authors calculate the influence of this information as it moves further from the source node, incurring losses on the way. They call this the 'information dissipation length.' (They measure IDL to the 50% dissipation level since the decay rate is asymptotic with a long tail.) IDL is a measure of the size of the subsystem that is affected by a particular element. IDL is somewhat akin to 'influence' in traditional graph measures that lack dynamic considerations (that is, are only topological). Second, the authors also calculate how the usefulness of the information dissipates over time. IDT is a proxy for how long the network remembers the particular state of a node, another measure of its influence.

This combination of structural (topographic) and dynamic (IDL and IDT) may not be exactly the right mix, but it does show how basing the analysis on information theoretics offers up new ways for understanding the nature of graphs and their interactions over time. For example, one finding is that it is intermediate players, not the central hubs or most popular nodes, that may have the most influence on dynamical processes within complex networks [50]. We may apply IDL and IDT to any complex, dynamic network. Mutual information (I) is that which nodes share. Figure 15.1 is a two-node example.

'Deep information networks' use somewhat similar information-theoretic approaches to reduce the dimensionality of knowledge graphs [51], though with potentially better understandability of the intermediate layers than deep learning. As we apply such techniques to more systems, we should gain further insights to improve our predictive power, perhaps getting to such seemingly intractable questions such as emergence, state transitions, or self-organization.

We see the potential relatedness or interactions between Peirce's semiosis, universal categories, and information theory. If we find that Peirce's universal categories indeed capture some fundamental truths about nature, for which some combination of the categories and information theory provides insight, then we can begin to apply lessons from natural science to the questions of language, knowledge, and representation. Each subsequent insight will feedback upon those that came before to improve our ability to model and predict our natural world.

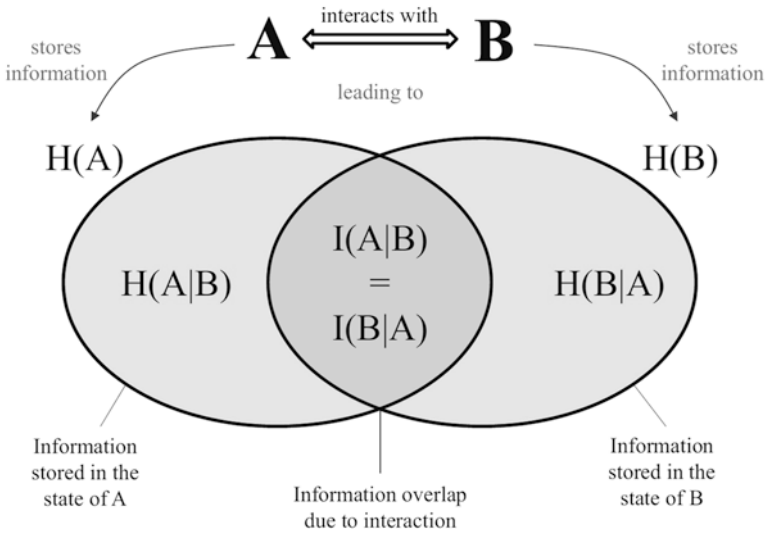


Fig. 15.1 Shannon and mutual information (reprinted by permission of the European Physical Journal—Special Topics) [49]

Gaia Hypothesis Test

The chemist James Lovelock first posed the Gaia hypothesis[5] in the 1970s, soon getting collaborative support from the microbiologist Lynn Margulis.[6] They hypothesized that life is an integral part of the Earth's development. Organisms have coevolved with changes in Earth geology and chemistry and climate; high oxygen levels, which are highly reductive, grew in the atmosphere due to the presence of life; life adapted to salinity changes due to salt runoff from terrestrial sources; and a complete weave of interacting forces and effects intertwined. The hypothesis has led some to consider the Earth a form of 'living thing.' Though derided when first postulated, advocates have refined the hypothesis to reflect emerging science better and scientists now largely embrace the idea of an evolving and interacting biosphere.

We also see another trend. The initial understanding of entropy as something that led to disorder caused thoughtful physicists, such as Erwin Schrödinger, discussed in Chap. 2, to posit explanations in the 1940s for how life did not violate the second law of thermodynamics. That subject, too, has evolved much, whereas now a significant portion of scientists see entropy as operating in either equilibrium or non-equilibrium circumstances. The Earth, with massive influxes of solar radiation and

[5] Gaia was the Greek goddess who personified Earth.

[6] I discuss Margulis in a different context in Chap. 3.

the evolution of life that has created its 'Gaia-like' effects, is the quintessential nonequilibrium case.

Under nonequilibrium conditions with massive external influxes of energy, the equilibration principle, what one might also think of as selective pressure, is to dissipate this free energy as rapidly as possible. That idea, in turn, promoted on both statistical mechanics and biological terms by some, is known as the maximum entropy production (MEP) principle [52]. The principle favors structures that utilize and then dissipate free energy fastest and most efficiently. Ludwig Boltzmann, the explicator of entropy and statistical mechanics, is now praised by some for quantifying what is *not* (that is, entropy), akin to the contribution of the Arabian mathematicians who invented the number zero [53].

Researchers have applied MEP to the Earth at planetary scale [54] and related it to more prosaic observations like water flows in soils [55]. Kleidon, in a comprehensive treatment of this topic with wonderful illustrations of various global fluxes, stated, "This seeming contradiction [of standard interpretations of entropy] is resolved by considering planet Earth as a coupled, hierarchical and evolving nonequilibrium thermodynamic system that has been substantially altered by the input of free energy generated by photosynthetic life" [56].

Herrmann-Pillath has woven these threads of the Gaia hypothesis, MEP, Charles Peirce's semiotics, and other factors into a complete speculation [57]. He includes the 'fourth law of thermodynamics' from Stuart Kauffman [58], another theorist on the origin of life, who posed the role of work and the "tendency for self-construction biospheres to construct their own workspace" (p. 244). This view bridges from Peirce's statements about semiosis and its applicability to crystals and bees. We call the application to living organism biosemiotics, and for inanimate or broader applications, such as what Herrmann-Pillath proposes, 'physiosemiosis.' This term arises from the proposition that "the biosphere is a system of generating, processing and storing information, thus directly treating information as a physical phenomenon," and follows the triadic semiotic model. A few researchers have speculated that Peirce's ideas of semiosis may even extend as far as the formation of matter after the Big Bang,[7] though it would be 15 years after Peirce's death before Hubble discovered the redshift. Still, Peirce intended his views on semiosis to infuse nature.

Peirce's advocacy that first, second, and third are the necessary and sufficient building blocks for all of reality may provide some missing insight into these basic questions of evolution and cosmology. His placement of randomness and chance into Firstness appears to conform to what we continue to learn about what is possible and where it arises. Peirce's prescience about signs, universal categories, and roles of chance and continuity quite possibly was truly cosmic. If indeed Peirce did grok the nature of nature at its most fundamental levels, then how we can apply his insights to our understanding of existence and reality is but at the beginning stages.

[7] See footnote 11 in Chap. 10.

References

1. Y. Xu, B.C. Malt, M. Srinivasan, Evolution of word meanings through metaphorical mapping: systematicity over the past millennium. Cogn. Psychol. **96**, 41–53 (2017)
2. C. Ramiro, B.C. Malt, M. Srinivasan, Y. Xu, Mental algorithms in the historical emergence of word meanings. Cogn. Sci., 986–991 (2017)
3. D. Oele, G. Van Noord, Distributional lesk: effective knowledge-based word sense disambiguation, in *IWCS* 2017*—12th International Conference on Computational Semantics—Short papers* (2017)
4. T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, *arXiv:1301.3781 [cs]* (2013)
5. D. Yarowsky, Unsupervised word sense disambiguation rivaling supervised methods. Association for Computational Linguistics, in *Proceedings of the 33rd annual meeting on Association for Computational Linguistics* (1995), pp. 189–196
6. P.S. Resnik, *Selection and Information: A Class-Based Approach to Lexical Relationships*. Ph.D., University of Pennsylvania (1993)
7. X. Sun, Structure regularization for structured prediction, in *Advances in Neural Information Processing Systems* (2014), pp. 2402–2410
8. M. Banko, E. Brill, Scaling to very large corpora for natural language disambiguation, association for computational linguistics, in *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics* (2001), pp. 26–33
9. S.P. Ponzetto, R. Navigli, Knowledge-rich word sense disambiguation rivaling supervised systems. Association for Computational Linguistics, in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics* (2010), pp. 1522–1531
10. E. Agirre, A. Barrena, A. Soroa, Studying the Wikipedia Hyperlink Graph for Relatedness and Disambiguation. *arXiv:1503.01655 [cs]* (2015)
11. M.A. Hearst, Automatic Acquisition of Hyponyms from Large Text Corpora, in *Proceedings of the 14th Conference on Computational Linguistics-Volume 2*. Association for Computational Linguistics (1992), pp. 539–545
12. M. Mintz, S. Bills, R. Snow, D. Jurafsky, Distant supervision for relation extraction without labeled data, in *Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP*, vol. 2–7 (Suntec, 2009), pp. 1003–1011
13. D.B. Nguyen, M. Theobald, G. Weikum, *J-REED: Joint Relation Extraction and Entity Disambiguation* (ACM Press, Singapore, 2017), pp. 2227–2230
14. N. Bach, S. Badaskar, A review of relation extraction, in *Literature Review for Language and Statistics II*, vol. 2. (2007), pp. 1–15
15. H. Paulheim, Knowledge graph refinement: A survey of approaches and evaluation methods. Semantic Web **8**, 489–508 (2017)
16. B. Perozzi, R. Al-Rfou, S. Skiena, DeepWalk: Online learning of social representations, in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (2014), pp. 701–710
17. V. Ha-Thuc, J.-M. Renders, *Large-Scale Hierarchical Text Classification Without Labelled Data* (ACM Press, New York, NY, 2011), pp. 685–696
18. D. Traxl, N. Boers, J. Kurths, Deep graphs—a general framework to represent and analyze heterogeneous complex systems across scales. Chaos **26**, 1–27 (2016)
19. B.G. Johnston, M. Williams, in *Conference on Artificial General Intelligence*. A formal framework for the symbol grounding problem (Atlantis Press, Paris, 2009)
20. R. Neches, R.E. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, W.R. Swartout, Enabling technology for knowledge sharing. AI Magazine **12**, 36 (1991)
21. Y. Song, D. Roth, Machine learning with world knowledge: the position and survey, *arXiv:1705.02908 [cs, stat]* (2017)
22. P. Cobreros, P. Égré, D. Ripley, R. van Rooij, Foreword: three-valued logics and their applications. J. Appl. Non-Classical Logics **24**, 1–11 (2014)

23. R. Lane, Triadic logic, in *Digital Encyclopedia of Charles S. Peirce* (2001)
24. C. Elsenbroich, O. Kutz, U. Sattler, A case for abductive reasoning over ontologies, in *CEUR* (2006), pp. 1–12
25. J. Du, K. Wang, Y.-D. Shen, A tractable approach to ABox abduction over description logic ontologies, in *AAAI* (2014), pp. 1034–1040
26. R.S. Prawat, Dewey, Peirce, and the learning paradox. Am. Educ. Res. J. **36**, 47–76 (1999)
27. P. Flach, A. Kakas, O. Ray, Abduction, induction, and the logic of scientific knowledge development, in *Workshop on Abduction and Induction in AI and Scientific Modelling.* (Citeseer, 2006), p. 21
28. P.A. Flach, in *Abduction and Induction*. On the logic of hypothesis generation (Springer, Dordrecht, 2000), pp. 89–106
29. T. Kapitan, Peirce and the autonomy of Abductive reasoning. Erkenntnis **37**, 1–26 (1992)
30. T. Kapitan, In what way is Abductive inference creative? Trans. Charles S. Peirce Soc. **26**, 499–512 (1990)
31. M. Tschaepe, Guessing and abduction. Trans. Charles S. Peirce Soc. **50**, 115–138 (2014)
32. G.E. Hinton, A. Krizhevsky, S.D. Wang, in *International Conference on Artificial Neural Networks*. Transforming auto-encoders (Springer, Berlin, Heidelberg, 2011), pp. 44–51
33. S. Sabour, N. Frosst, G.E. Hinton, Dynamic routing between capsules, in *Advances in Neural Information Processing Systems* (2017), pp. 3859–3869
34. F.J. Valverde-Albacete, C. Peláez-Moreno, 100% classification accuracy considered harmful: the normalized information transfer factor explains the accuracy paradox. PLoS One **9**, 1–10 (2014)
35. B. de Finetti, Foresight: its logical Laws, its subjective sources. Annales de l'Institut Henri Poincare **7**, 94–158 (1937)
36. F.J. Valverde-Albacete, C. Peláez-Moreno, Two information-theoretic tools to assess the performance of multi-class classifiers. Pattern Recogn. Lett. **31**, 1665–1671 (2010)
37. S. Still, Thermodynamic Cost and Benefit of Data Representations. *arXiv:1705.00612 [cond-mat]* (2017), pp. 1–8
38. S. Still, Information theoretic approach to interactive learning. EPL (Europhysics Letters) **85**, 1–6 (2009)
39. S. Still, D.A. Sivak, A.J. Bell, G.E. Crooks, Thermodynamics of prediction. Phys. Rev. Lett. **109**, 1–5 (2012)
40. S. Still, Information bottleneck approach to predictive inference. Entropy **16**, 968–989 (Feb. 2014)
41. TechTarget, in *SearchBusinessAnalytics*. What Is Self-Service Business Intelligence (BI)? http://searchbusinessanalytics.techtarget.com/definition/self-service-business-intelligence-BI
42. J. Stangarone, Self-Service Business Intelligence, *mrc's Cup of Joe Blog*, 101 (2015)
43. A. Stutt, E. Motta, Semantic learning webs. J. Interact. Media Educ. **2004**, 1–32 (2004)
44. J. Sweller, S. Sweller, Natural information processing systems. Evol Psychol **4**, 434–458 (2006). https://doi.org/10.1177/147470490600400130
45. K. Wiesner, Nature computes: information processing in quantum dynamical systems. Chaos **20**, 037114 (2010)
46. N. Chomsky, Three models for the description of language. IRE Trans. Inf. Theory **2**, 113–124 (1956)
47. M. Burgin, G. Dodig-Crnkovic, Typologies of Computation and Computational Models. *arXiv preprint arXiv:1312.2447* (2013), pp. 1–24
48. R. Quax, *Information Processing in Complex Networks*. Ph.D., University of Amsterdam (2013)
49. R. Quax, A. Apolloni, P.M.A. Sloot, Towards understanding the behavior of physical systems using information theory. Eur. Phys. J. Spec. Top. **222**, 1389–1401 (2013)
50. R. Quax, A. Apolloni, P.M.A. Sloot, The diminishing role of hubs in dynamical processes on complex networks. J. R. Soc. Interface **10**, 1–10 (2013)
51. G. Franzese, M. Visintin, Deep Information Networks. *arXiv:1803.02251 [cs]* (2018), pp. 1–10

52. A. Kleidon, R. Lorenz (eds.), *Non-Equilibrium Thermodynamics and the Production of Entropy: Life, Earth, and Beyond* (Springer, Berlin, New York, 2005)
53. R.E. Ulanowicz, S.J. Goerner, B. Lietaer, R. Gomez, Quantifying sustainability: resilience, efficiency and the return of information theory. Ecol. Complex. **6**, 27–36 (2009)
54. A. Kleidon, Non-equilibrium thermodynamics, maximum entropy production and Earth-system evolution. Philos. Trans. A Math. Phys. Eng. Sci **368**, 181–196 (2010)
55. E. Zehe, T. Blume, G. Bloschl, The principle of 'maximum energy dissipation': a novel thermodynamic perspective on rapid water flow in connected soil structures. Philos Trans R Soc Lond B Biol Sci **365**, 1377–1386 (2010)
56. A. Kleidon, Life, hierarchy, and the thermodynamic machinery of planet earth. Phys Life Rev **7**, 424–460 (2010)
57. C. Herrmann-Pillath, *Revisiting the Gaia Hypothesis: Maximum Entropy, Kauffman's 'Fourth Law'and Physiosemeiosis* (Frankfurt School of Finance & Management, Frankfurt am Main, 2011)
58. S.A. Kauffman, *Investigations* (Oxford University Press, New York, 2000)

Chapter 16
Potential Uses in Depth

Information, Peirce tells us, is like a spatial function covering the complete *area* of a topic. We have just covered a dozen topics in *breadth* regarding the potential of Peircean ideas to the subjects of knowledge and its representation. Now, let's turn our attention to the treatment of three additional topics, only now in *depth*. These three new topics are not as speculative as some from the prior Chap. 1 have chosen them based on highest impact, near-term potentials. The three areas are workflows and business process management (BPM), semantic parsing, and applications and interactions in robotics. The challenge in all three areas is to automate as much as possible while leaving to the knowledge worker what is uniquely human. For space reasons, I shorten the two bookends to allow a fuller treatment of the middle case.

Workflows and BPM

Business processes and their management are the most neglected areas of business. Business process management, or BPM, is about how businesses identify, select, implement, and refine their bases of production, and the actions to perform them. BPM embraces two kinds of knowledge: that needed to do a task, and the knowledge of what to do when a process goes off track. BPM is the *action* side of the business, analogous to the *things* of business, which is what KM manages.[1] Indeed, in this very manner of *verb-noun,* we see similar failings and lack of attention for BPM as we see for knowledge management.[2] It is telling that service-oriented,

[1] BPM may also refer to business process *modeling*. We retain the management sense here, noting that the modeling part only comes after thinking through the management portion.

[2] The exception to this observation is advanced manufacturing. Some of these businesses, inherently action oriented, have pioneered BPM's related cousin of manufacturing process management. However, it is an open question whether manufacturing businesses are better at KM as well.

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8_16

knowledge-based businesses still do not see that the fundamental basis of their products is knowledge. Attending to the production and consumption of knowledge warrants as much attention to efficiencies as do the actions or processes on the factory floor.

We first touched upon the subject of content workflows in Chap. 12. Here, I give better detail to these flows and argue that knowledge management itself deserves business process management. Ideas in Peirce and KBpedia enable us to better represent actions and events, critical aspects of workflow. We will see real differences from BPM once we learn how to incorporate it into our actual, daily workflows.

We may group BPM activities in many ways but, fundamentally, they represent how to satisfice multiple business objectives, not always in concert. The goals of profit and public service, for example, are not weighed equally by companies and nonprofit institutions. A business process initiative should consider a scope that at least includes:

1. A logical conceptual model of process and terminology for the business process agreed by the user community of workers and managers
2. An agreed design and implementation plan
3. Technology support to implement that system
4. How users and administrators interact with the system, including user interfaces and approval steps and actions
5. Agreed and documented process and governance.

Initiatives require both semantic technologies and management commitment.

Today, however, we are not even at the preliminaries. For the majority of companies, agreed workflow procedures for business processes or operational workflow management systems do not exist. Gaps arise because BPM deals with abstract processes and intimately involves people, work practices, and management. Workflows cut across organizational boundaries and thus need to be attentive to terminology and semantics. To raise the question of 'what is your workflow?' is to disrupt the workflow of your standard knowledge worker. Many BPM efforts are bass-ackwards. We do not need a separate application upon which to focus our 'workflow' attention. We need workflow considerations rooted in how we currently work. While it is OK to disrupt the knowledge worker for a short period to help understand their implicit workflows, it is not OK to put in place BPM systems that divert knowledge workers from their standard work.

Semantic technologies are essential to the task because shared communication is at the heart of workflow management. Semantic business process management has been a steady research interest over the past 15 years or so (see Hepp et al. for one of the seminal early papers [1]). One theme is the potential role of ontologies in BPM [2]. Through this work we learn the imperative of incorporating 'action' in our ontologies and the need to handle branching and merging in evolving workflows. Unfortunately, however, besides some notable research initiatives from Europe, we have not seen broad commercial success for semantic BPM.

Concepts and Definitions

We focus on digital *content* in the context of BPM for knowledge work, which refers to all documents, datasets, records within them, ontologies used by the system, and internal control vocabularies and structures. A *content life cycle* embraces all content stages and workflow steps within those stages, from inception to use. In its entirety, we capture a content life cycle by the complete *controlled vocabulary* and sequencing of all stages and steps. A *content stage* is a broader concept than workflow step and represents a state within the content life cycle ranging from experimental to working to archived. Figure 12.2 is one example of various content stages.

A *state* is an instantiation of a workflow step that represents a change in content or an evaluation of it (for example, a manager approving the release of content even without making any changes to the actual content). An *event* is an occurrence that causes a change in state. A *workflow* is a sequence of connected steps representing a business activity where each step follows without delay or gap and ends just before the next subsequent step may begin. A *workflow step* is a discrete step in a workflow that has an explicit label, where the general progression proceeds from creation through editing and review to approvals. Workflow step is a narrower concept than content stage within a content life cycle. We organize workflows around discrete and definable business objectives such as authoring, harvesting (ingest), archiving, and the like. Upon completion of various workflows, we may deem the content ready for different stages in a content life cycle. In an authoring workflow, for example, new content may proceed from creation to editing to completion of tagging and then review (with potential approval). Approval of authored content thus represents a change in the content stage from working to readiness for public release and use.

We can thus see the 'stages' of a content life cycle as a broader organizing framework than the individual 'steps' of the workflows we embed within these content stages. All workflows, though individual steps may differ, share the basic conceptual progression of drafting or creation proceeding through editing or modification and then to testing and review before acceptance for public use or readiness. Because of these conceptual similarities, we can develop and share controlled vocabularies to represent these progressions across workflows, preferably using consistent terminology we draw from actual work practice.

We need to intimately relate these stages and steps to governance and quality control. We introduce known checks, reviews, and sign-offs into the workflow to ensure that releases meet the organization's quality standards. 'Business processes' are the combination of an orderly progression in workflows with governance. By adopting such BPM practices, we help ensure the repeatability and generalizability of our organizational efforts.

Providers have developed *workflow engines* to keep these specifications persistent and to execute some of them, incorporating the various decision rules and triggers that enable the progression to proceed step by step through the workflows. The workflow engine is a software application that manages and executes modeled computer processes, and thus provides a coherent and standard way for specifying

various business workflows and then executing them depending on the governance and quality checks desired by the organization. We may base events or triggers for moving from state to state within the progression on user or manual review, timing or duration checks, applications of scripts or automated tests, and the like. *Workflow management* is the collection of processes and governance by which we oversee the entire content life cycle, including guidance for how steps progress across the cycle and how we conduct reviews and approvals.

We may represent these steps and activities with controlled vocabularies or graphic notations. *BPEL* (business process execution language) was the leading standard in the early days of service-oriented architectures. Today, *BPMN* (business process model and notation) is the leading approach whereby we use graphical editors to create visual workflows that generate XML instructions to the workflow engine. Optionally, we may represent the individual steps in a workflow process using a *work breakdown structure* (WBS), a management approach used for many decades.

The BPM Process

The business process management (BPM) cycle should begin with a *logical model*, starting, in the case of KM, from the viewpoint of content life cycle and work stages. We should involve creators and users of the content, especially including responsible managers, in agreeing to the major stages and terminology of this model, as well as transition and decision points for state changes. We need to inspect and define specific stages of this cycle; adding steps to the process requires consultation with the stakeholders. The products of this part of the cycle are the initial controlled vocabularies and representations of content stages, similar to what we show in Fig. 12.2. The result of this process creates the 'backbone' to the overall BPM effort.

We then need to express this logical model and its controlled vocabulary and terminology in an ontology to take advantage of semantic technologies. The stages and steps naturally lend themselves to class specifications. Review and approval levels become properties, all again governed by the agreed common vocabularies. At this juncture, we advise to discuss and decide upon required or optional annotation guidelines and metadata for events and state transitions. Noting the name of the approving employee and timestamps are a couple of standard fields. Rejection or re-review steps may warrant more elaborate notes. Some of these annotation standards may require input from legal counsel or be attentive to the regulatory requirements of the business.

It is important to balance current terminology in use with consistency across the full business process for:

- Content life cycle stages
- Workflow steps
- Events
- Actors (agents) and roles
- Transitions and alerts.

We must update the logical model and ontologies to reflect any changes to the conceptual model. The net result should be an updated workflow model in specification form that uses consistent terminology. This model will now become the baseline for workflows moving forward, which we may further refine or update or use as templates for other workflows as needed.

We configure the workflow engine in parallel to this effort. The major aim is to add the workflow logic and business rules, in addition to the install and configuration parameters. Many proprietary and open-source workflow engines exist for both BPEL and BPMN, though choosing the right one is a complicated task involving a panoply of anticipated functions and the types of integration desired with other tools (such as graphical changes to a BPMN specification flowing through to the engine). As noted earlier, we have seen semantic integrations in research projects, but we appear to lack turnkey off-the-shelf systems that incorporate semantics. Aside from the need to integrate a workflow engine, prior chapters provide the guidelines for one.

Optimal Approaches and Outcomes

One has to wonder about the relatively low uptake of BPM for knowledge management functions. I think we can point to two reasons for this lack. First, as we discussed for the lack of use of information and knowledge in Chap. 3, managers do not have a bred-in-the-bones belief in the importance of process and workflows in knowledge management. Somehow we know we are involved with important tasks of discovery and pursuit of knowledge, but we do not understand that these are purposeful and refinable activities. Second, I think the implementation of BPM has been bass-ackwards. Business process or workflows are not applications; they are an articulation of what we do. Recording or changing workflows must occur at the point of work, not in a separate app. A workflow system that gets used must be unobtrusive and linked to the content work at hand. This point-of-action imperative means we should split the BPM functions into atomic operations distributed across current applications, all governed by consensual workflows and terminology.

We need to embed state transitions and state designation changes into existing workflow screens. We need to look to our major content platforms (word processors, spreadsheets, content management systems, and the like) and find where we can embed simple workflow-related functionality. Some of this may be as simple as recording state transitions; others might be specific tabs or operations. Plug-ins are a proven model and can emit simple data structs recording their actions, invoked manually or automatically depending on a REST-ful Web service linked with the content ontology. From a UI perspective, this should be done consistently in context with the host tool for all workflows. Some of these activities, such as editing or managing ontologies (knowledge graphs), tagging content, mapping content, or further refining terminology and semantics, are new tasks and not merely state changes of current tasks. These functions become a bit more complicated Web services, as we discussed in Chap. 12.

Based on today's standards, it would be wise to link our ontology design to some form of meta-model that would enable us to talk directly with BPMN. This notation covers the range of known and anticipated BPM and workflow activities and states. Third-party tools allow users and analysts to inspect and modify workflows graphically and to emit their specifications in canonical forms.[3] A good design would let users and analysts examine and refine workflows directly.

What we are seeking is a framework and workflow that naturally allows us to present all existing and new content through a pipeline that extends from authoring and review to metadata assignments. Making final assignments for subject tags from the candidates and then ensuring that we correctly assign all other metadata may be either eased or impeded by the actual workflows and interfaces. The trick to such semi-automatic processes is to get these steps right. Analysts need manual overrides when suggested candidate tags are not right. Sometimes new terms and entries are found when reviewing the processed content; these need to be entered and then placed into the overall knowledge graph as discovered. The process of working through steps on the tag processing screens should be natural and logical. Some activities benefit from focused, bespoke functionality, rather than calling up a complicated or comprehensive app.

In business settings these steps need to be recorded, subject to reviews and approvals, and with auditing capabilities should anything go awry. Potential revision means there needs to be a workflow engine underneath the entire system, recording steps and approvals and enabling things to be picked up at any intermediate, suspended point. These support requirements tend to be unique to each enterprise. Thus, we favor an underlying workflow system that can be readily modified and tailored—perhaps through scripting or configuration interfaces. We also want version control systems for our knowledge graphs so that we may record, compare, and roll back changes as required.

Respect for workflows is also a first principle, expressed in two different ways. The first way is that we should not unduly disrupt existing workflows when introducing interoperability improvements. While workflows can—and should—be improved or streamlined over time, initial introduction and acceptance of new tools and practices must fit with existing ways of doing tasks to see adoption. Workers resist jarring changes to their existing work practices. The second way that workflows should be respected is the importance of being aware of, explicitly modeling, and then codifying how we do tasks. This focus becomes the 'language' of our work and helps define the tooling points or points of interaction as we merge activities from multiple disciplines in our domain. These workflow understandings also help us identify useful points for APIs in our overall interoperability architecture. These considerations provide the rationale for assigning metadata to characterize our information objects and structure, based on controlled vocabularies and relationships as established by domain and administrative ontologies.

Peirce's guidelines and KBpedia provide some unique strengths to a BPM initiative. Events, states, roles, and actions are well characterized and structured. We have repeatedly seen the semantic technology influence in KBpedia, an essential perspec-

[3] See, for example, the open-source Yaoqiang BPMN editor (http://bpmn.sourceforge.net/).

tive for capturing consensus and terminology related to business processes and workflows. We have put forward a 'pay-as-you-benefit' strategy for incremental testing and adoption of new scope and functionality, an approach that also fits well with implementing what may prove to be a broad, or business-wide, BPM implementation plan. Moreover, from an architectural standpoint, we have put forward a WOA design that supports atomic and distributed functionality interacting via Web services, the only approach that makes logical sense for a workflow management system.

An effective BPM system would bring tangible benefits in three ways. The first is that for us to gain efficiencies by climbing the learning curve, we must have a documented business process that we can repeat and refine. The second benefit is that we gain a basis for learning about learning. Today, knowledge gets produced. Still, we have little insight into how to do it nor how we can do it better. The third and longer term benefit is that with a better understanding of states, actions, and events, we provide a possible entry point into real-time knowledge supervision. Processes are a Thirdness, and mediation is a dynamic process of what exists and how chance and change may affect it.

Semantic Parsing

Parsing is the identification and segregation of string symbols into the constructs of a formal grammar.[4] A formal grammar is a set of rules for how to process these symbols, often including defined classes (lexemes) to which the processed symbols may be assigned, the aggregate of which is called the lexicon. The processing of text is like Pac-Man chewing through tokens in either left-right or right-left directions, top-down or bottom-up, by character, word, or phrase, deciding at each token how to transform it or terminate. The parser might be simple, perhaps relying only on heuristics or regular expressions and seeking only to define token boundaries. The parser might be quite sophisticated and based on machine learning of the optimal methods and parameters to parse domain content for specific domain purposes.

Different NLP methods may benefit from different parsers or grammars. Some output from the parse such as tables, trees, or vectors may be suitable for different purposes or content. The tree structure, for example, is a proven storage structure for parsed documents and Web pages. Some parser generators can also effectively operate in 'reverse,' in which case they may perform as compilers (for computer languages) or syntax or grammar checkers (for languages) or theorem provers (if logic based). Thus, much research is potentially transferrable among disciplines.[5]

Lexical analysis is often the first stage of parsing, wherein the system 'chunks' or tokenizes the string into lexical units. For natural language understanding, the lexical constructs are parts of speech, word senses, sentence structure (syntax), and

[4] The word *grammar* is derived from a Greek word meaning "writing," though at one time the knowledge of Latin grammar was viewed as endowing one with magical power, from which arose our word *glamor* [4].

[5] In many areas of computational linguistics, care should be taken when comparing findings from the contributing disciplines.

the like. These constructs intimately link to the formal grammar. Parsers need to sequence the string in specific ways and often rely on recursion to keep the algorithm simpler and better performing. The recursion method may thus impose other requirements on chunking order or storage or perhaps add pre- or post-processing steps. We may impose simplifications or feature reductions to keep the language analysis decidable or have it complete in acceptable time. Circumstances of use may also require us to attend to other tasks during the parse steps, including normalization, word forms, word segmentation, stemming, case adjustments, lemmatization, or sentence detection (end, beginning).

Formal grammars have different degrees of expressiveness. Once we process a natural language into a formal grammar, its reverse translation may not retain the same expressiveness, making the grammar 'lossy.' We may choose to accept some loss, since capturing the full expressiveness of natural language may require larger and more complicated formal grammars with weaker performance and greater storage [3]. Parsing and their accompanying grammars are the keys to natural language understanding. Peirce has much to offer in these areas, though toolmakers have yet to exploit parsers and grammars based on Peircean principles to any real degree.

A Taxonomy of Grammars

At the syntax level, we can classify grammars into phrase-structure (or constituency) ones and dependency ones.[6] The guiding idea behind constituency grammars is that groups of words may act as a single unit, such as a noun phrase (NP) or verb phrase (VP). Dependency parsing can express word dependencies (such as some semantic relationships) and is getting more attention because of its suitability to some forms of machine learning. Dependency parsing works well for natural languages that have free word orders (*e.g.*, Turkish, Czech). Dependency parsing can also be used to generate treebanks, which have become popular reference structures for use by tokenizers or text annotators. Example dependency grammars include word grammar, functional generative description, and link grammar.[7] However, the more common parsers use constituency grammars.

A formal grammar provides a set of transition rules for evaluating tokens and a lexicon of types that can build up, or generate, representative language structures. The tokens are either terminal or symbolic, with the terminal ones causing the pro-

[6] This split somewhat reflects a similar one for *discriminative* versus *generative* machine learning models. Discriminative models, also called conditional models, are a class of models used in machine learning for modeling the dependence of unobserved (target) variables y on observed variables x. Example discriminative models include support vector models (SVM), conditional random fields (CRF), neural networks (xNN), linear regression, maximum entropy Markov, and random forests. Generative models use algorithms to try to reconstruct how the original data was generated, often through probabilistic means. Example models include hidden Markov models (HMM), naive Bayes, generative adversarial networks (GANs), Gaussian mixture model, and other types of the mixture model.

[7] Nivre argues that a dependency grammar is not a grammar formalism, rather a specific way to describe the syntactic structure of a sentence [49].

cess to continue to the next token or to stop entirely. Formal grammars act like abstract machines (or automata). Automata theory, the basis of finite-state machines, is also closely related in that an automaton is a finite representation of a formal language that may be an infinite set. Grammar with a larger lexicon of types or more sophisticated steps encourages more straightforward representations and better generalizations, including recursion, to reduce evaluation times.

Categorial grammar, a constituency grammar derived from the simply typed lambda calculus, is based on types and is built according to the principle of compositionality, wherein we understand complex expressions from the meaning of their components and the rules (grammar) of their construction. This grammar is a phrase-structure grammar, better known as a context-free grammar, in which a terminating symbol never appears on the left-hand side of a transformation step. Context-free languages are the theoretical basis for the syntax of most programming languages.

Since formal grammars are a branch of formal language, we can draw upon a rich mathematics literature of theory, constructs, and algorithms. In the 1960s, theoretical research in computer science on regular expressions and finite automata led to the discovery that context-free grammars are equivalent to non-deterministic pushdown automata.[8] This discovery led to the interaction of formal grammars with compiler construction. It also led to the design of deterministic context-free grammars that could be parsed sequentially by a deterministic pushdown automaton, a requirement in early programming language designs due to computer memory constraints.

Noam Chomsky was the first to formalize the idea of the hierarchical constituency with a phrase-structure grammar in 1956, which he proceeded to expand upon and develop over the ensuing decades, called the Chomsky hierarchy,[9] which splits into four types. Deterministic context-free grammars (DCFGs), an intermediate grammar in the Chomsky hierarchy, are a proper subset of the context-free grammars, which can derive from deterministic pushdown automata. As benefits, we can parse DCFGs in linear time, and a parser generator can automatically generate them.

For natural languages, practitioners favor context-free grammars, another intermediate type in the Chomsky hierarchy. Here is a sampling of the methods or grammars that have emerged from context-free grammars (CFGs):

- Affix grammar
- Attribute grammar
- Categorial grammar
- CYK algorithm
- Earley algorithm
- Generalized context-free grammar
- Generalized phrase-structure grammar
- Head-driven phrase-structure grammar
- ID/PL grammar
- GLR parser
- Lambek calculus [4]

[8] See https://en.wikipedia.org/wiki/Deterministic_context-free_grammar

[9] Also known as the Chomsky–Schützenberger hierarchy.

- Lexical functional grammar
- LL parser
- Minimum recursion semantics
- Parsing expression grammar
- Pregroup grammar
- Phrase-structure grammar
- Stochastic context-free grammar.

Categorial grammars create fixed lexicons that assign a category (type) to each symbol and inference rules for what type of symbol follows, sufficient to specify a particular language grammar. The CYK algorithm is widely taught and implemented and is a good basis for understanding context-free grammar aspects [5]. Head-driven phrase-structure grammar (HPSG) marks entries with a hierarchy of types. As more rules get added to HPSG, the approach takes on the form of what researchers call a construction grammar. Minimal recursion semantics is a meta-level language for describing semantic structures in a typed formalism that the authors claim is an easy way to decompose, relate, and compare semantic structures [6]. In phrase-structure grammars feature sets are attribute-value pairs, where the value may be single, multiple, or complex, including lists, sets, or functions.[10] Another nice aspect of a feature structure is that we can represent them as a directed acyclic graph (DAG), with the nodes corresponding to the variable values and the paths to the variable names. Further, we can effectively transform every context-free grammar (CFG) into a weakly equivalent one without unreachable symbols (unprocessed tokens in the string).

Researchers strive to find a sufficiently expressive grammar, perhaps with some heuristics for rare edge cases, to capture and rewrite back natural language sufficient for effective communication. It is clear that some features of languages are not context free. It turns out, as Joshi showed for some leading-edge grammars, that we need to only capture partial aspects of context sensitivity to obtain sufficient expressivity, what he classed as mildly context-sensitive grammars [54]. Here are some prominent options:

- Combinatory categorial grammar
- Embedded pushdown automaton
- Head grammar
- Linear-indexed grammar
- Tree-adjoining grammar.

We may associate the elements of combinatory categorial grammar (CCG, which is grounded in combinatory logic), such as verbs or common nouns, with a syntactic 'category' that has a function with specified arguments and a type of result [7]. CCGs combine descriptive adequacy—that is, applicability to the constructions and interpretations of a wide range of diverse languages—with explanatory adequacy, in the sense of having the fewest expressions to obtain an adequate level of theoretical linguistic competence. This level of 'mildly context-sensitive grammars' is the current 'sweet spot' within the Chomsky hierarchy for trading off performance with expres-

[10] We talked of this simple data struct in Chap. 9.

siveness. The generalized linear context-free rewriting system has proven an enabler for formulating and testing new grammars at this leading edge of performance.

Not all formal grammars are generative, either. Constraint grammars are entirely rule based, often embracing hundreds of rules. Constraint-based grammars state the rules that are disallowed, with many acting as constraint analogs to standard generative models. Functional theories of grammar try to model the way language is used in communications under the assumption that formal relations between linguistic elements are functionally motivated.

Computational Semantics

So, we now have a broad view of the mechanics of formal grammars and parsing, but what about the meaning of language, its semantics? We can see the processing rules and approach; we still need to understand the semantics of the chunks involved and their contribution to a representation of meaning. Computational semantics is the study of how to automate the process of constructing and reasoning with meaning representations of natural language expressions [8]. Semantic parsing breaks natural language into logical forms [9]—that is, an unambiguous artificial language—with the logic intended to express the meaning of the language components.[11] Shallow semantic parsing uses discriminative models, like recurrent neural networks, to label the roles in a sentence.

Joachim Lambek was one of the pioneers of the mathematics of sentence structure and syntax and formulated many algebraic approaches of early computational linguistics using his Lambek calculus. He acknowledged that the idea behind this approach could be traced back to Charles Sanders Peirce's ideas about valency in chemistry [4]. Lambek grammars, built using the Lambek calculus, extend basic categorial grammars. The Lambek calculus helped stabilize approaches and notations and was a forerunner to Montague grammar.

The central idea of Richard Montague's first paper in 1970 was to frame linguistic semantics as a homomorphic mapping between two algebras, one syntactic and the other semantic. In a series of three papers in the early 1970s Montague[12] fleshed out a formal theory that represents the standard theory for computational semantics for most of the last of the twentieth century. We call this basis the Montague grammar (MG) [3]. Montague expressed the semantics of the source into a logical form based on a theory of the semantics. He provided a functional mapping between the syntax and the logical form that preserves the structure and equivalences. While the statement of this approach seems straightforward, maintaining the homomorphism (same shape) between the forms is the tricky part [2]. The intensional logic of Montague grammars is a typed lambda calculus [53].[13] Before Montague, linguists had no methods for

[11] For a sample detailed description see SLING, a frame-based semantic parser using a dependency grammar [50].

[12] Montague's contributions came to an untimely end when he was violently murdered at age 40.

[13] This makes these grammars well suited to functional languages like Lisp.

assigning a compositional semantics to natural language syntax due to the mismatch with first-order logic. Montague's type theory represents a solution to Gottlob Frege's desire to use function arguments as the basic 'glue' to combine meanings, a view unknown in linguistics at the beginning of the 1970s, yet now viewed as standard [10].

Montague grammars have been a stepping stone in many different directions. One direction is that MGs presume a tree structure, which favors FSTs, HMMs, and other finite-state methods of syntax analysis [3]. Another direction is to generalize into algebraic terms, making the system more functional with better information theoretics. One direction has been to combine different ideas of semantic primes as the starting lexicon.

Much of the work in semantic linguistics has focused on the commonalities between human languages. One way is to use semantic primes, a basic list of primitives under which to categorize terms. Anna Wierzbicka first posed these ideas in a seminal work in 1972 [11]. The universal dependencies provide shared starting points between scores of human languages. The 'universal semantic tagset' (a different concept) provides a (growing) set of cross-language primitives [12]. These initiatives show how data gathering and comparisons between human languages, made available through the Web, are remaking how computational semantics may move forward.

Three Possible Contributions Based on Peirce

Let us now weave Peirce and his potential contributions into the narrative. Consistent with the Peircean guidelines through this book, we should look to be:

- *Real*—Peirce advocated empirical truth for describing and organizing the things in the world. Definitions or arrangements based solely in the mind are psychological and not phenomenological. Hewing to a test of reality means what we retain should be true in relation to what we have already modeled, helping to ensure that our methods remain consistent and coherent.
- Organized according to the *universal categories*—Continuing to maintain reasoned splits into Firstness, Secondness, and Thirdness may offer some surprising keys and insights for our knowledge representations going forward.
- *Logical*—Since logic is at the heart of the Peircean view, logic fits well with the ideal of formal grammars.
- Consistent with the *logic of relations*—Peirce has already provided us with significant guidance in his identification of relations and his logical treatments of them, including algebraic notions to inform modeling.
- A good *entity-attribute distinction*—We have already pointed to the importance of separating out attributes (a Firstness) from entities (a Secondness).
- Capable of distinguishing *generals* from *particulars*—We want discrete class-level types (generals, a Thirdness) and item-level (particulars, a Secondness) ones.
- Attentive to the *sign representativeness* in Peircean semiosis—Peirce's ten sign classes (see Table 2.2), or even analysis of his later 28- and 66-sign classifications, are a rich target for applying mathematical or logical analysis for teasing out rules for analyzing problems.

- Reflective of the *probabilistic nature* of truth—We should favor learning models that support inductive reasoning and allow the use of probability distributions to characterize some nodes.
- *Contextual*—In that we capture both the intensionality and extensionality of our lexemes and choose word senses based on the overlap with accompanying text. The inclusion of inference and background world knowledge supports this aim [13].

I discuss below three different approaches by which we may embrace these Peircean guidelines in whole or part. I present the approaches in relative order of complexity of implementation, starting with the simplest. We begin with Peircean part-of-speech tagging, move to machine learning, and then conclude with a dedicated Peircean grammar. We can also combine these three approaches in various ways.

Peircean POS Tagging

The first and most direct incorporation of Peircean themes likely resides in relating these constructs to off-the-shelf part-of-speech taggers. One quick approach might be to map an existing schema to the KBpedia components, which we have already organized in a Peircean manner. Chen, the originator of the E-R modeling ideas, understood the entity-attribute split well. I have taken Chen's mapping of word senses [14] and related it to existing KBpedia components (Table 16.1):

Table 16.1 Simple POS mapping

Word sense	KBpedia component
Common noun	Concept/type/entity/event
Proper noun	Entity/event
Transitive verb	Relation
Intransitive verb	Attribute
Adjective	Attribute
Adverb	Attribute (property)

These senses map pretty well but lack consideration of subtypes within entities, external relations, or types. They also neglect many of the 'gluing' parts of speech such as determiners, conjunctions, or prepositions.[14] Some modifications to an E-R model approach might be undertaken to embrace the full structure of languages better as found in some reference tagsets, but that is a demanding, manual task. Ninio, in a recent review informed by Peirce, also put forward an approach to syntactically label parts of speech [15].

We need to go deeper into Peirce's ideas about signs, language, and grammar to understand how a Peircean approach to POS tagging might better proceed. Peirce had strong interests in word categories, more from a semantic than syntactical per-

[14] Tokenizers and POS taggers, plus any reference tagsets employed, should be attentive to syntax that is declinable (noun, pronoun, verb, adverb). The indeclinable terms (proposition, conjunction, interjection, particles, modals) are less of a problem since only single terms are required. Declensions of tense, case, plurality, or gender are very important topics in some languages, though I do not speak further of it here.

spective, with original ideas about common nouns, proper nouns, pronouns, verbs, and prepositions [16]. Peirce understood a sentence as a formal proposition split into two fundamental parts, the subject and the predicate (1902, CP 2.318).[15] In a formal proposition, the subject is definite. Subjects often begin as indefinite individuals (such as 'selectives,'[16] e.g., *some person*), proceed as better understood and characterized into a definite individual (a 'proper noun,' e.g., *Jimmy Johnson*), and then may be related to a type, a definite general (a 'common noun,' e.g., *football coach*) (1905, MS 280:41). However, common nouns are not universal,[17] with proper nouns providing the ultimate subjects [17]. Some predicates bring with them the need to also specify a direct object as a complement to the intended subject, as in 'Cain killed Abel' (1899, CP 2.230). Other sentence constructions may also require multiple subjects through the use of conjunctions ('Bob and Mary went shopping') or triadic relations ('Bob donated a scholarship to Mary').

It is these kinds of constructions that help instruct what Peirce meant by a predicate, or what he termed a *rheme*. A rheme is an 'unsaturated' term (1902, CP 2.317), meaning it stands as the function within a propositional phrase that lacks (is 'blank') a subject. Here is Peirce's definition for verb:

> A *verb*, being understood in a generalized sense, may be defined as something logically equivalent to a word or combination of words, either making a complete proposition, or having certain *blanks*, or quasiomissions, which being filled each with a proper name, will make the verb a complete proposition. (1896, NEM 4:278)

Peirce goes on to say that "The places at which lines of identity can be attached to the verb I call its *blank subjects*" (1898, NEM 4:338).

This idea of *blank subjects*, and the role of the index in relation to them, is one of Peirce's pivotal perspectives. As Nöth notes, "indexical signs had traditionally not been associated with the concept of representation, and indeed, the terminological tradition had been to subsume only iconic and symbolic signs under this term" [18]. Peirce helps show and generalize the range of relations between things, between subjects and predicates, which indicate ranges of determinacy or selectiveness. We see, for example, that we may characterize clauses, verb and noun phrases, prepositions, indicatives,[18] adverbs, and adjectives according to their indexicality and whether the subject is determinant.

---

[15] By formal proposition I mean a sentence in the indicative mood, "for a proposition is equivalent to a sentence in the indicative mood" (1903, CP 2.315), for which Peirce was mostly concerned. Contrast this to the other moods (1893, CP 2.291) or "quasi-propositions," see below.

[16] A "selective" (1903, CP 4.408) is an indeterminant individual such as indicated by selective pronouns (any, every, all, no, none, whatever, whoever, everybody, anybody, nobody) or particular selectives (some, something, somebody, a, a certain, some or other, one) (1903, CP 2.289).

[17] Peirce did not hold the *common noun* to be a universal part of speech (POS). He states, "I do not regard the common noun as an essentially necessary part of speech. Indeed, it is only fully developed as a separate part of speech in the Aryan languages and the Basque, –- possibly in some other out of the way tongues" (1904, CP 8.337).

[18] Such as *this*, *that*, *something*, *anything*.

Of course, the initial split of sentences into subject and predicate masks the fact their syntax may be somewhat complex. Peirce applies the same logic, though, to noun phrases and verb phrases as well as to other constructs he calls 'quasi-propositions.' This construct, which Peirce named a dicisign or dicent, is information bearing and adds further characterization to its subject. When decomposed, a dicisign acts like a value pair with its two signs, like a full proposition, providing a function sign (predicate) and a denotation sign (subject). The subject may itself be an index or indeterminate, one of the reasons why Peirce called them 'quasi-propositions.' Like Peirce's viewpoint of the *breadth* of information, the *dicent sinsign* points to more characteristics, or attributes, of the intended subject. Like the *depth* of information, the *dicent indexical legisign* points to subsumption ('___ *is a man*' implied by the *man* common noun) or external relations. Hilpinen provided some of the first detailed analyses of how Peirce viewed the proposition [19].

Table 16.2 combines these insights to characterize Peirce's ten signs linguistically. Note that the order in this table changed from Table 2.2 where the dominant ordering was qualisign-sinsign-legisign (consistent with Peirce's 1903 ordering in his *Syllabus* [EP 2:294-295) to rheme-dicisign-argument (consistent with Peirce's 1904 ordering in a letter to Lady Welby [CP 8.341]):

Table 16.2 Peirce's ten signs for KR relation to linguistics

Sign name (redundancies)	Comments	KBpedia	POS[a]
(Rhematic iconic) qualisign	Onomatopoeic [20], ideophones, uncountable (mass) nouns (?)	Abstractives	
(Rhematic) iconic sinsign	Indefinites, conjunctives, disjunctives	Attributes	N, VI, ADJ, ADV
Rhematic indexical sinsign	Direct experience, relations	External relations	DT, N,VT
(Rhematic) iconic legisign	Metaphors, puns, analogies, diagram-like, genres (?)	Associations	N, VI, ADJ, ADV
Rhematic indexical legisign	Demonstratives, pronouns, proper names	Particulars	N, VI
Rhematic symbol (legisign)	Common nouns	Types	N, AUX
Dicent (indexical) sinsign	'Quasi-propositions' for adjectives, adverbs, modifiers (in depth)	Attributes	NP, VP, ADJP, ADVP
Dicent indexical legisign	'Quasi-propositions' for information in breadth	External relations, subsumption, is-a	NP
Dicent symbol (legisign)	Proposition (full), sentence	OPEN	NP + VP
Argument (symbolic legisign)	Multiple sentences	Graph measures	---

[a]I view these assignments as provisional. There is not much in the literature (or Peirce directly) on these assignments. I anticipate further research to refine these assignments somewhat

The sense that emerges is that Peirce's strong links to information and representation mean there is much of value in Peirce's linguistic views to KR and knowledge management. While the information in this section could be used to set different bases for labeling syntactic categories, it may be better to logically continue the effort to develop a tokenizer more attuned to Peirce's unique views, as I discuss next.

Machine Learning Understanding Based on Peirce

The index acts as a reference to the object, ultimately representing an individual thing (including individual collections) (1907, EP 2:407). The interpretant must have some previous (or 'collateral') acquaintance with the object to identify that individual thing, what Peirce termed 'collateral observation.'[19] By this term, Peirce meant "previous acquaintance with what the Sign denotes" (1909, EP 2:494). Nothing of this observation is psychological since the interpretant contributes no part to the observation. The collateral observation plays a parallel role to context but is not the same. Collateral observation is central to disambiguation since presently observed characters may be compared with previous observations to separate out identities. Collateral observations extend beyond the boundaries of the sentence.

Context and most arguments also extend beyond the boundaries of the sentence.[20] If we are to hope for acceptable levels of *natural language understanding* through KR formalisms, we must also tackle these issues of collateral observation, context, and reasoned argument. To get at these aspects we perhaps want to combine some form of semantic parser, better attuned to our organization of KBpedia by the universal categories and types, and machine learning, possibly leveraging *knowledge supervision*, as we discussed in Chap. 4.

A semantic parser requires us to formulate a semantics, including language construction, and then to learn how to apply it to new content with acceptable computational times [9]. Sarbo and Farkas suggest a rather simple parsing method grounded in their interpretation of Peirce [20]. I agree with the authors that we want simple models because most formal models of natural language are too complex. I also like their approach using a pushdown automaton, which is more capable than a finite-state machine. However, I do not agree with their grammar basis or some of their construction rules. We have a different mindset and structure in KBpedia in the universal categories and typologies.

A couple of approaches look promising for next steps. One of the approaches, combinatory categorial grammar (CCG), we introduced above. The other approach, Lambek categorial grammar (LCG), is closely related. CCG is an efficiently parseable, yet linguistically expressive, grammar formalism. Because of its strong lexicon approach and suitability to types, CCG should be a good match with the

[19] Peirce also termed this *collateral experience*, *collateral information*, and *collateral acquaintance*.

[20] Of course, it is possible to write out full syllogisms in the confines of a single sentence, but most often arguments are made over multiple sentences.

typology design of KBpedia. Researchers have developed a probabilistic CCG from question-answer pairs using supervised learning from ontologies and knowledge bases [21]. As shown for transitive verbs, we can substitute meaning vectors as the learning basis [22]. Perhaps more promising is a tensor-based semantic framework that can be "seamlessly integrated" with CCG for a "practical, type-driven compositional semantics based on distributional representations" [23]. Edward Grefenstette's thesis provides excellent guidance on how to relate distributional representations of meaning to CCG [24].

Another useful aspect of KBpedia is the availability of a text corpus (largely from Wikipedia) for all of the reference concepts in the system. By leveraging this content, we can create distributional representations that enable us to overcome fixed-pair inputs (such as question-answer) used to train many of these CCGs. This additional distributional component improves generality beyond the fixed input vocabulary used in training, making it more suitable for open-vocabulary applications [25]. Lastly, CCGs warrant testing against KBpedia due to the availability of open-source implementation kits. OpenCCG is perhaps the best known, with helpful online tutorials.

Though invented before CCGs and overlooked for a period, LCGs provide a simpler and more transparent mapping between phrase-structure trees, dependency structures, and semantic terms [26]. CCGs, in practice, have tended to need a large number of non-categorial rules, making them harder to understand and less generalizable [26].

One trend we see in computational linguistics is to combine logical and statistical approaches to natural language.[21] Logical, or compositional, approaches relate syntactical phrases to the meanings of their parts and how they are combined. These are the traditional approaches of semantic parsers to map messages to logical forms, which lend themselves to dealing with inference, ambiguity, and vagueness. On the other hand, statistical approaches, including machine learning, focus more on the individual word and phrase meanings or broader notions of content (and context) beyond the sentence [27]. What is exciting about a combined approach is that we can look at the compositional and semantic aspects of our language, mapped into the categorial perspectives of Peirce's logic and semiosis, and then convert those formalisms to distributions over broad examples provided by KBpedia's knowledge.

Peirce Grammar

At a more speculative level, we can look to going to the heart of the matter: fully adopting Peirce's views on logic and relations regarding how we specify our grammars. Full adoption is not such a wild idea since there have been attempts and

[21] One genesis of this grand synthesis is a 2010 paper by Coecke et al., "Mathematical Foundations for a Compositional Distributional Model of Meaning" [51], first unveiled in 2008 [52].

probes around a 'Peircean grammar' for at least a couple of decades. The basic advantage of this approach is that we do not need to shoehorn Peircean ideas into existing approaches, but are free to set up a clean infrastructure from scratch.

Patrick Suppes was one of the first to question the traditional approach of translating sentences into the formal notation of predicate logic [28]. He observed that inference in predicate logic bore little resemblance to the informal reasoning in English. He began to explore what he called *extended relation algebras*, which were a model-theoretic semantics for English that used neither quantifiers nor variables, but only constants on operations on sets and relations.

Chris Brink, in his 1978 thesis, was explicit about the influence of Peirce [29]. He noted that Peirce's first 1870 paper on the "Logic of Relatives" [30] was instrumental in guiding his thinking. Peirce classified logical terms into three classes—absolute terms, (simple), relative terms, and conjugatives—which correspond roughly to monadic, dyadic, and triadic predicates [31]. Within a decade Brink and his students were referring to this approach as the 'Peirce algebras,' a term which has stuck. One of the basic operations was the 'Peirce product,' R:A, which is the set of all elements related by R to some element in A (a basic matrix algebra). Relatives, as dyadic relations, can be represented algebraically rather than by conventional model theoretics. We can perform arithmetic over the individual identities. Boolean modules formalize the calculation of sets interacting with relations via the Peirce product. This approach enables us to treat the system as a two-sorted algebra (Boolean algebra with multiplication via the Peirce product). A two-sorted algebra makes explicit the implicit relation of concepts and roles, a concern for early KR languages for AI.

Eventually, this algebra was shown able to express the semantics of reasoning over sets, including for subsumption relations [32]. Using an algebraic approach to reason over sets becomes simpler since equations are sufficient to capture first-order reasoning using the calculus of relations for unions, intersections, and complements. De Rijke provided proof for full Peirce algebras in his 1993 thesis and showed the link with dynamic modal logics [33]. This confluence of work naturally led to efforts to try to find a grammar to match this algebra.

Michael Böttner defined a 'Peirce grammar' in 2001 and applied it to natural language [34]. It is a context-free grammar. The basic idea of the grammar is a direct one using Peirce algebra. Rather than translating the semantics first to a set-theoretic metalanguage, the Peirce grammar uses only algebra with the equation sign '=' as the single predicate. A Peirce grammar allows computations on strings directly rather than to variables or a pre-computation of an intermediate representation. Böttner introduced tree structures to handle more efficiently the inherently 'flat' nature of a Peirce algebra representation (useful for programming languages; it may be less so for natural language sentences, which are shorter). A Peirce grammar can support references outside of the sentence, again as a function of storage design, attractive for context. Böttner presents a strong definition of a Peirce grammar for English (see his Table 16.2 in the reference). While the approach neglects

some nuances, the approach does appear to handle the ideas of context and language fragments (such as clauses) in a computationally efficient manner.

Hans Leiß later took up some of the weaknesses and provided some extensions to overcome them [35]. Leiß noted that prior Peirce grammars had only modeled extensional aspects of natural language. It was unclear how to handle intensional aspects (attributes) or verbs with propositional arguments. The 'trick' of coding linguistic strings directly means that the unit boundaries (sentence, paragraph, arbitrary window) should be finite; the limits have not been tested. Leiß raises questions about whether and how we should handle noun phrases.[22] On the other hand, the approach does not use variables. We can construct meanings from a few basic ones with equality and subsumption capturing the relations between sets and their relations. Leiß, as well, looks to the tree structure to provide a more tractable approach to the inherent 'flat' structure of a pure Peirce grammar. As Leiß concludes [35]:

> Peirce grammar differs from other grammatical theories in that meanings are first-order objects, abstract sets and relations, which can only be composed by algebraic operations. Extended Peirce grammar adds a further sort of meanings, finite trees of sets and relations, from which constituent meanings can be extracted. These 'second-class' values have no ontological motivation—they only serve as intermediate stages in the evaluation of sentences, allowing us to give the context of an expression an access to the meaning *constituents* of the expression. (p. 162)

From there, for more than 10 years,[23] the trail goes cold. Relational grammars, in general, have gone out of favor. It very well may be fundamental limits exist with relational grammars, or particularly Peirce grammars, that relegate them to a minor footnote in the history of computational linguistics. However, I suspect that Peirce grammars, as they may evolve, may yet prove a seminal player in that history.

Cognitive Robotics and Agents

Robotics is a potential testbed for Peircean ideas about representation and KBpedia. One reason, of course, is an economic demand for greater autonomy combined with cognition. Advances tend to appear where the most imperative resides. A Peircean approach will also aid robotics designers, and much in the ideas about representation will benefit robotics, particularly in the interface with cognitive systems. Further, while language is symbolic, cognition and understanding are more. For the idea of Thirdness to become a general, it must pass the threshold of the habitual as Peirce explains the matter. An emphatic 'Fire!' is five symbols combined into a string symbol on a page, but shouted in a dark theatre with an intonation that signals

[22] We are also missing a design or approach to compositionality.

[23] Leiß's publication is dated in 2009, but based on a conference paper presented in 2005.

real fear invokes immediate action. There is nothing to 'think through' before acting, though that starts immediately as well.

Cognitive robots embrace the ideas of learning and planning and interacting with a dynamic world. When combined with mobility and perception sensors, this leads to greater autonomy. When combined with speech recognition and natural language understanding (NLU), we can instruct the robot by voice commands or interact with it as a virtual agent for Q&A or knowledge assistance. Over time, researchers have tested and designed various cognitive architectures for integrating cognitive and robotic functions, with a preference for a modular design with generic interfaces [36].

Time coordination for how long it takes modules to process their tasks requires trade-offs in expressiveness; simpler and more abstract representations appear best. We also see that open-source, modular robot languages and operating systems emerge (such as the Robot Operating System, ROS), the Robobrain knowledge engine initiative and even relatively affordable autonomous robot platforms (such as iCub or ROBOTIS OP2). Cognitive robotics promises to improve our baseline understanding of knowledge representation. Perception of and interaction with the external world are integral to sign communications. If we are ever to approach anything like true natural language understanding, then we need to incorporate all of the universal categories in our reality. Autonomous, cognitive robots are the anvil upon which these understandings may get hammered out.

Lights, Camera, Action!

Peirce's semiosis is not consistent with traditional computational views of cognition, which are a variation of input-output models. How the symbol gets interpreted is neglected by the traditional view [37]. Peirce's semiosis more closely represents the theory of embodied cognitive science, which differs from the tradition in pursuing three goals. These goals are to elevate the importance of the body as an explanation for various cognitions, to understand the body as a contributor to cognition, and to broaden our view of how agents use the environment to affect cognition (mood lighting, staging, arranging, and positioning) [38].

Similar to our example of shouting 'Fire!' awareness of the environment is an essential factor in cognition. The homunculus argument of the little man interpreting things in our heads must be grounded in some external reality to keep that interpretation from cascading into an infinite regress. This 'interpretation solely in the mind' is the fallacy in Descartes' worldview, as well as in the traditional view of cognition. Successful cognition also requires learning, and epigenetic robots depend on perceptions or kinesthetics (learning from physical actions) for the new information from which to learn. The forming of symbols from that learning is one way to achieve a certain Thirdness. Further, as long as we look at knowledge as only symbolic, we will miss the importance of Firstness and Secondness. Everything has its place.

Peirce indeed acknowledged the unconscious reaction and the role of instinct in signs and how we interpret them. Peirce believed in an unconscious aspect of the mind (1903, EP 2.188; 1903, CP 5.108; 1882, CP 7.64; *c.f.*, 1902, CP 7.363-367). Formal reasoning, the sphere of theory and analytics and logic, is part of the conscious mind, but the realm of action and pragmatic knowledge is a function of the unconscious.

> Reasoning, properly speaking, cannot be unconsciously performed. A mental operation may be precisely like reasoning in every other respect except that it is performed unconsciously. But that one circumstance will deprive it of the title of reasoning. For reasoning is deliberate, voluntary, critical, controlled, all of which it can only be if it is done consciously. An unconscious act is involuntary: an involuntary act is not subject to control; an uncontrollable act is not deliberate nor subject to criticism in the sense of approval or blame. A performance which cannot be called good or bad differs most essentially from reasoning (1903, CP 2.282).

Peirce held belief, which we saw in Chap. 2, as an important aspect of knowledge that occurs mostly in the unconscious (1905, EP 2:336; 1905, CP 5.417). Habitual stuff and reactive actions are part of common sense and not (generally) part of consciousness. However, the informal 'reasonings' in the unconscious are often more reliable than conscious reasoning and logical inference:

> Association is the only force which exists within the intellect, and whatever power of controlling the thoughts there may be can be exercised only by utilizing these forces; indeed, the power, and even the wish, to control ourselves can come about only by the action of the same principles. Still, the force of association in its native strength and wildness is seen best in persons whose understandings are so little developed that they can hardly be said to reason at all. Believing one thing puts it into their heads to believe in another thing; but they know not how they come by their beliefs, and can exercise no control over the inferential process. These unconscious and uncontrolled reasonings hardly merit that name; although they are very often truer than if they were regulated by an imperfect logic, showing in this the usual superiority of instinct over reason, and of practice over theory. They take place like other mental suggestions according to the two principles of similarity and connection in experience (1886, CP 7.453).

I think two implications arise from Peirce's observations. First, we begin to unveil a bit of the role of knowledge bases as 'belief' bases insofar as they make direct assertions. We can know that balls are round and squares have four equilateral sides and can act on these assertions without further analysis. Second, 'instinct,' as ephemeral as it is to define, plays an essential role in guiding actions. Still, what does Peirce mean by 'instinct,' one of his most common descriptors?

Peirce first sees two sources of instinct, one innate or biological, the result of evolution, what he calls *inherited*, and the other one of infant training and inculcation, what he calls *traditional*. Peirce notes that instincts may change when circumstances change, but that is rare since circumstances hone our instincts over generations of trial and error. Peirce splits his views of logic into a *logica docens*, the logic of theory and study, and a *logical utens* [39], the internal logic of practice as influenced by instinct. The first logic is that of the scientist, and the second that of the practical actor.

> If an action, although complicated, has very often to be performed, and is almost always performed in nearly the same way, it frequently happens that we have an instinct for performing it. The action of walking is an example; the action of throwing a stone is another. Now instinct is remarkable for its great accuracy, as well as for its adaptedness to its purpose; and it would usually be unwise in the extreme to attempt to perform such an act under the guidance of theory; for theories have to be studied very long and very deeply before they can be entirely freed from error; and even then the application of them is laborious and slow (n.d., NEM 4:187)

Peirce did not view instinct as inferior to formal reasoning while noting that "action in general is largely a matter of instinct" (1905, CP 5.499). He saw that "we all have a natural instinct for right reasoning" (1902, CP 2.3)

> If so, in what respect do you hold reasoning to be superior to instinct? Birds and bees decide rightly hundreds of times for every time that they err. That would suffice to explain their imperfect self-consciousness; for if error be not pressed upon the attention of a being, there remains little to mark the distinction between the outer and the inner worlds (1902, CP 2.176).

> Of excessively simple reasonings a great deal is done which is unexceptionable. But leaving them out of account, the amount of logical reasoning that men perform is small, much smaller than is commonly supposed. It is really instinct that procures the bulk of our knowledge; and those excessively simple reasonings which conform to the requirements of logic are, as a matter of fact, mostly performed instinctively or irreflectively (1902, CP 2.181)

Peirce saw that instinct and abduction are linked[24] (1903, CP 5.171). While Peirce held that pragmatism is a conscious discipline (and thus in the realm of *logica docens*), there may be instinctual aspects of how we conduct it. We often instinctively screen through the multiplicity of abductive options to select those for more expensive inductive attention. We need to evaluate options and potential outcomes based on practical measures and instincts. Our conscious reasonings need to incorporate an inspection and role for instinct.

Practical implications for robotics arise from this discussion. We can envision subsystems that deal with direct knowledge—based on the ABox and perhaps direct typing—tied more closely to the perceptual sensors and kinesiology of the agent. These subsystems may be linked more closely to instinctual actions such as resource acquisition, risk avoidance, and protection. Perhaps this perspective offers insight into how to monitor the external environment while all inputs are within expected parameters versus outliers or disruptions that may need to trigger more analytic modules. It would also seem that instinctual subsystems may be more attractive areas for rule-based approaches or unsupervised machine learning. The underlying idea of instinct is what is the best way to act under given contexts and events.

[24] Note that Peirce specifically excludes consideration of instinct in the scientific method and its quest for truth, since all assumptions should be open to question. Pragmatism, however, adds action and instinct to the equation.

Cognitive versus reaction subsystems also require different time demands and impose different time delays. Robot information architectures need to define modules and optimize effectiveness-performance trade-offs. The integration of NLU requires complements to perceptions and actions, a current area of active interest [40]. The conventional perspective is to ask how we can better import existing knowledge representation and systems into cognitive robots. However, perhaps we need to pose that question the other way around.

> Wisdom lies in nicely discriminating the occasions for reasoning and the occasions for going by instinct. Some of my most valued friends have been almost incapable of reasoning; and yet they have been men of singularly sound judgment, penetrating and sagacious (1903, CP 7.606).

Grounding Robots in Reality

One way to avoid the homunculus' infinitely regressing explanations is to 'ground' our symbols into some base truth, called the 'symbol grounding problem.' Grounded symbols no longer have free variables and become the 'indecomposable' primitives of the representation. The implication is that higher order concepts are derivations of lower level concepts until the concepts can no longer be divisible. Ultimately, for cognitive robots, the processing of natural language, be it from commands or interacting with humans, must be part of this grounding. It may take the form of a semantic model underlying both language and robot commands that is also related to robotic perception and actuation; see combinatory categorial grammars [40] as mentioned in the prior use case. Cangelosi sees the symbol grounding problem in similar terms, where the questions of perception and action and how they are represented mentally are a core issue in cognitive robotics [41]. Deb Roy sees the representation more broadly, embracing the idea of symbols as included in semiosis based on Peircean concepts [42]. He also wants to specifically relate "sensing and motor action to words and speech acts."

While essential considerations, some argue that the question of grounding goes well beyond mere representation [43]. The nature of the question is evolving to one of *meaning*[25] and how that relates to *grounding*. That is because for a cognitive robot to formulate new knowledge and symbols, there must be some guiding principles or functions that go beyond a representative grounding. Some mechanism must exist for emergence or how incorporation of new information may lead to new actionable knowledge. Roy, again, tries to get at this question by posing a split of 'meaning' into *referential*, *functional*, and *connotative* forms. He prefers a simple base language, modeled on that used by young children [44]. Stanton believes

[25] Note that *meaning* has many connotations including existential, linguistic, philosophical, psychological, semiotic, of life, and others. Our use embraces all of these senses.

that we should try to mimic the evolution of the brain and include intrinsic 'value systems' in autonomous robots to process novel experiences [45]. Stanton nibbles around the edges for how Peircean concepts may contribute to this task. We have also tended to overlook the adaptability and emergent properties of symbol systems for robotic intelligence [46]. Ricardo Gudwin, steeped in Peircean viewpoints, takes the question to a different level when he notes that we do not have a symbol grounding problem, yet one of grounding to the icon [47]. Icons are a representation of Firstness and perhaps better intimately tied to the inputs of sensors in robots.

No matter how framed, KBpedia provides three solid contributions to the grounding problem. First, of course, it provides a complete and coherent view of representation, knowledge, actions, events, and relations. Second, KBpedia as a reference knowledge graph grounds the system ultimately in Firstness (monads), Secondness (particulars), and Thirdness (generals) no matter where we start the inspection. Third, we construct KBpedia with multiple knowledge bases that can provide the reference base for both analytic and instinctual purposes and tests. The structural recursion and richness of KBpedia structure appear an excellent fit for cognitive robotic architecture and purposes.

Robot as Pragmatist

As crucial as symbol grounding is, I think we still may be missing the pivotal importance of robotics to knowledge-based artificial intelligence. Up to this point, we have framed the challenge as one of getting AI advances—including ideas of representation and meaning—*into* robots. What of the other way around?

What this short survey has shown us is that robots may bring their own contributions to these questions. We have seen how important it is to integrate the dimensions of perception and action into a cognitive processing robot. We recognize both analytic (cognitive, thinking) tasks and activities more dominated by instinct and kinesthetic action. Cognitive robots demand that we deal with the challenges of integration, coordination, and choreography. I think it is fair to observe that doing KBAI in a purely symbolic, unembodied state is likely to provide an incomplete testbed for knowledge, cognition, and learning. Human intelligence evolved in a mobile, interactive environment. We may need to embed artificial intelligence in dynamic, physical contexts to approximate similar capabilities.

Parisi et al. argue how multimodal representations can improve the robustness of recognizing actions, action-driven perception, sensory-driven motor behavior, and human-robot interaction [48]. Robot vision systems are providing a different perspective on how we need to represent best things like shapes, vectors, and objects. Brain studies show that 'what' and 'where' have separate recognition areas, lending credence to the need for modularity.

Cognitive robots promise to help us improve our knowledge representations and AI efforts. Cognitive robots may be the drivers for better capabilities in planning, coordinated action-cognition responses, human-robot interactions, and learning to perform an ever-growing list of functions in real-time settings. Solving the competing demands for cognitive robots can only make us more pragmatic.

References

1. M. Hepp, F. Leymann, J. Domingue, A. Wahler, D. Fensel, in *e-Business Engineering*. Semantic business process management: A vision towards using semantic web services for business process management (IEEE, Beijing, China, 2005), pp. 535–540
2. F. Smith, M. Proietti, Ontology-based representation and reasoning on process models: A logic programming approach, *arXiv:1410.1776 [cs]* (2014)
3. A. Kornai, *Mathematical Linguistics* (Springer, London, 2008)
4. J. Lambek, *From Word to Sentence: A Computational Algebraic Approach to Grammar*, Polimetrica sas (2008)
5. M. Lange, H. Leiß, To CNF or not to CNF? An efficient yet presentable version of the CYK algorithm. Informatica Didactica **8**, 1–21 (2009)
6. A. Copestake, D. Flickinger, C. Pollard, I.A. Sag, Minimal recursion semantics: An introduction. Res. Lang. Comput. **3**, 281–332 (2005)
7. M. Steedman, *A Very Short Introduction to CCG*. Unpublished draft note (1996), p. 8
8. J. Bos, A survey of computational semantics: Representation, inference and knowledge in wide-coverage text understanding. Lang. Linguist. Compass **5**, 336–366 (2011)
9. P. Liang, Learning executable semantic parsers for natural language understanding, *arXiv:1603.06677 [cs]* (2016)
10. B.H. Partee, Montague grammar, ed. by N.J. Smelser and P.B. Bates, in *International Encyclopedia of the Social and Behavioral Sciences* (Pergamon/Elsevier Science, Oxford, 2001). p. 7 pp.
11. A. Wierzbicka, *Semantics: Primes and Universals* (Oxford University Press, Oxford, 1996)
12. L. Abzianidze, J. Bos, Towards Universal Semantic Tagging, *arXiv:1709.10381 [cs]* (2017)
13. S. Hassan, *Measuring Semantic Relatedness Using Salient Encyclopedic Concepts*. Ph.D., University of North Texas (2011)
14. P.P.-S. Chen, English, Chinese and ER diagrams. Data Knowl. Eng. **23**, 5–16 (1997)
15. A. Ninio, Learning a generative syntax from transparent syntactic atoms in the linguistic input. J. Child Lang. **41**, 1249–1275 (2014)
16. W. Nöth, Charles Sanders Peirce, pathfinder in linguistics, in *Digital Encyclopedia of Charles S. Peirce* (2000)
17. A.-V. Pietarinen, Peirce's pragmatic theory of proper names. Trans. Charles S. Peirce Soc. **46**, 341 (2010)
18. W. Nöth, Representation and reference according to Peirce. Int. J. Signs Semiotic Syst. **1**, 28–39 (2011)
19. R. Hilpinen, On CS Peirce's theory of the proposition: Peirce as a precursor of game-theoretical semantics. Monist **65**, 182–188 (1982)
20. J. Sarbo, J. Farkas, A Peircean ontology of language, in *International Conference on Conceptual Structures* (Springer, Berlin, Heidelberg, 2001), pp. 1–14
21. T. Kwiatkowski, E. Choi, Y. Artzi, L. Zettlemoyer, Scaling semantic parsers with on-the-fly ontology matching, in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing* (Association for Computational Linguistics, Seattle, WA, 2013), pp. 1545–1556

22. S. Clark, Type-driven syntax and semantics for composing meaning vectors, in *Quantum Physics and Linguistics: A Compositional, Diagrammatic Discourse* (2013), pp. 359–377
23. J. Maillard, S. Clark, E. Grefenstette, *A Type-Driven Tensor-Based Semantics for CCG* (Association for Computational Linguistics, 2014), pp. 46–54
24. E. Grefenstette, *Category-Theoretic Quantitative Compositional Distributional Models of Natural Language Semantics*. Ph.D., Balliol College, University of Oxford (2013)
25. M. Gardner, J. Krishnamurthy, Open-vocabulary semantic parsing with both distributional statistics and formal knowledge, in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*. (Association for the Advancement of Artificial Intelligence, 2017), pp. 3195–3201
26. T.A.D. Fowler, *Lambek Categorial Grammars for Practical Parsing*. Ph.D., University of Toronto (2016)
27. P. Liang, C. Potts, Bringing machine learning and compositional semantics together. Ann. Rev. Linguist. **1**, 355–376 (2015)
28. P. Suppes, Direct inference in English. Teach. Philos. **4**, 405–418 (1981)
29. C.H. Brink, *The Algebra of Relations*. Ph.D., University of Cambridge (1978)
30. C.S. Peirce, Description of a notation for the logic of relatives, resulting from an amplification of the conceptions of Boole's calculus of logic. Mem. Am. Acad. Arts Sci. **9**, 317–378 (1870)
31. C. Brink, The algebra of relatives. Notre Dame J. Form. Logic **XX**, 900–908 (1979)
32. C. Brink, R.A. Schmidt, Subsumption computed algebraically. Comput. Math. Appl. **23**, 329–342 (1992)
33. M. de Rijke, *Extending Modal Logic*. Ph.D., Universiteit van Amsterdam, Institute for Logic, Language and Computation (1993)
34. M. Böttner, Peirce grammar. Grammars **4**, 1–19 (2001)
35. H. Leiß, The proper treatment of coordination in Peirce grammar, in *Proceedings of FG-MoL 2005*. (Edinburgh, Scotland, 2009), pp. 149–166
36. M. Scheutz, J. Harris, P. Schermerhorn, Systematic integration of cognitive and robotic architectures. Adv. Cogn. Syst. **2**, 277–296 (2013)
37. P. Steiner, CS Peirce and Artificial Intelligence: Historical heritage and (new) theoretical stakes, in *Philosophy and Theory of Artificial Intelligence* (Springer, New York, 2013), pp. 265–276
38. L. Shapiro, The embodied cognition research programme. Philos. Compass **2**, 338–346 (2007)
39. P. Chiasson, Logica Utens, in *Digital Encyclopedia of Charles S. Peirce* (2001)
40. C. Matuszek, E. Herbst, L. Zettlemoyer, D. Fox, Learning to parse natural language commands to a robot control system, in *Experimental robotics* (Springer, Heidelberg, 2013), pp. 403–415
41. A. Cangelosi, Solutions and open challenges for the symbol grounding problem. Int. J. Signs Semiotic Syst. **1**, 49–54 (2011)
42. D. Roy, Semiotic schemas: A framework for grounding language in action and perception. Artif. Intell. **167**, 170–205 (2005)
43. M.-A. Williams, Representation = grounded information, in *Trends in Artificial Intelligence*, ed. by T.-B. Ho, Z.-H. Zhou (Springer Berlin Heidelberg, Berlin, Heidelberg, 2008), pp. 473–484
44. D. Roy, A mechanistic model of three facets of meaning, in *Symbols and Embodiment: Debates on Meaning and Cognition*, ed. by M.D. Vega, A.M. Glenberg, A.C. Graesser (Oxford University Press, Oxford, 2008), pp. 195–222
45. C.J. Stanton, The value of meaning for autonomous robots, in *Proceedings of the Tenth International Conference on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems*, ed. by B. Johansson, E. Sahin, C. Balkenius (Lund University, Lund, 2010), pp. 129–136
46. T. Taniguchi, T. Nagai, T. Nakamura, N. Iwahashi, T. Ogata, H. Asoh, Symbol emergence in robotics: A survey. Adv. Robot. **30**, 706–728 (2016)
47. R. Gudwin, The icon grounding problem. Int. J. Signs Semiotic Syst. **1**, 73–74 (2011)
48. G.I. Parisi, J. Tani, C. Weber, S. Wermter, Emergence of multimodal action representations from neural network self-organization. Cogn. Syst. Res. **43**, 208–221 (2017)
49. J. Nivre, *Dependency Grammar and Dependency Parsing* (Växjö University, Växjö, 2005)

50. M. Ringgaard, R. Gupta, F.C.N. Pereira, Sling: A framework for frame semantic parsing. *arXiv:1710.07032 [cs]* (2017)
51. B. Coecke, M. Sadrzadeh, S. Clark, Mathematical foundations for a compositional distributional model of meaning. Linguist. Anal. **36**, 345–384 (2010)
52. S.C.B. Coecke, M. Sadrzadeh, A compositional distributional model of meaning, in *Proceedings of the Second Quantum Interaction Symposium (QI-2008)* (Oxford University Press, Oxford, 2008), pp. 133–140
53. J.R. Hobbs, S.J. Rosenschein, Making computational sense of Montague's intensional logic. Artif. Intell. **9**, 287–306 (1977)
54. A. K. Joshi, Context-sensitive grammars, in *Oxford International Encyclopedia of Linguistics, 2nd Edition* (K. Vijay-Shanker and D. Weir, eds., Oxford University Press, Oxford, UK, 2003), pp. 1–4

Chapter 17
Conclusion

Peirce posited a "third grade of clearness of apprehension" to better understand a topic at hand, what he claimed as the ultimate expression of his *pragmatic maxim*. One of the favorite quotes I have used in this book is Peirce's first formulation of this maxim:

> "Consider what effects, which might conceivably have practical bearings, we conceive the object of our conceptions to have. Then, our conception of these effects is the whole of our conception of the object." (1874, CP 5.402, EP 1:132, W 3:266)

Peirce came to believe that this initial formulation did not capture his exact intent. Here is how Kelly Parker summarized it [1]:

> "In the proposal for Memoir 32, Peirce expressed his discomfort with this formulation of the pragmatic maxim. He wrote that the paper 'was imperfect in tacitly leaving it to appear that the maxim of pragmatism led to the last stage of clearness' (NEM 4:30). Indeed, the phrasing of the maxim is potentially misleading. One might read this statement as providing guidelines for an alternative means of *defining* concepts. If we think of standard dictionaries as giving the 'second-grade' linguistic definitions of concepts, we might take the pragmatic maxim as a guide to producing a super-dictionary of 'third-grade' definitions. Such a book (a '**practionary**'?) might endeavor to list all the practical effects a thing could have in experience, and thus furnish the reader with a better conception of the object." (p. 182) (bold added)

Throughout this book, I have attempted to adhere strictly to this form, the first such attempt to apply Peirce to the interpretation of a single concept, which, in our case, is *knowledge representation*. This book is the first attempt to produce a *practionary*.

As I stated in the beginning, knowledge representation is a field of artificial intelligence dedicated to representing information about the world in a form that a computer system can utilize to solve complex tasks. We have explored this topic from background to practice and then on to implications. In *Part I*, we set the stage for the context of the concept by discussing the nature of information, knowledge, and representation, as well as the challenges and opportunities facing KR. In *Part II* we provided a speculative grammar for KR, including the structural role of the universal categories of Firstness, Secondness, and Thirdness, and the terminology, languages,

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8_17

logic, and models for knowledge representation. That foundation lets us discuss existing frameworks and KR constructs in typologies and knowledge graphs in *Part III*. In our next part, we used these components to build KR and knowledge management systems, including what to construct and what to test and best practices. With a working system in hand, we were then able in *Part V* to discuss 15 possible application areas of a Peircean approach to KR, covering informative examples in both breadth and depth. These 15 cases were in addition to the three main application thrusts for this book in knowledge management, data interoperability, and knowledge-based artificial intelligence. As we wrap up our survey of Peirce and KR, we conclude by teasing out some cross-cutting threads and implications from our journey.

I will let you judge whether this practionary achieved its objective of attaining a "third grade of clearness of apprehension" covering "all of the conceivable practical effects" of a Peircean interpretation of knowledge representation. For me, the author struggling to understand a lone genius working in isolation more than a century ago, I have found Peirce's guidance invaluable. Now, as we wrap up our discussion, I would like to stand back from this framework of a practionary and offer some thoughts as to where this journey has led us.

The Sign and Information Theoretics

Peirce's understanding of semiosis and signs connects intimately with his views and understanding of logic. Both, I have argued, are themselves prescinded from Peirce's universal categories of Firstness, Secondness, and Thirdness. Indeed, I have argued that the universal categories provide the overarching framework for how we need to organize and categorize our world. The reality of the universal categories is that fundamental.

In Peirce's descriptions of *prescission*, which we introduced in Chap. 7:

> "Now, the categories cannot be dissociated in imagination from each other, nor from other ideas. The category of first can be prescinded from second and third, and second can be prescinded from third. But second cannot be prescinded from first, nor third from second. The categories may, I believe, be prescinded from any other one conception, but they cannot be prescinded from some one and indeed many elements. You cannot suppose a first unless that first be something definite and more or less definitely supposed. Finally, though it is easy to distinguish the three categories from one another, it is extremely difficult accurately and sharply to distinguish each from other conceptions so as to hold it in its purity and yet in its full meaning" (1880, CP 1.353).

By this understanding, we prescind Firstness and Secondness from Thirdness, which Peirce reaffirms many places in other ways, often using the term 'degenerate.' Thirdness, too, as we have seen, is where *meaning* resides and is also perhaps best characterized as 'continuity,' the force which Peirce calls *synechism*.[1]

[1] This topic is more fully discussed in Appendix A.

We certainly need to place knowledge representation in Thirdness. Knowledge representation is ultimately symbolic because we need to relate it to a computer. KR also is tied intimately to meaning, since we have come to understand that knowledge is information we believe and upon which we may act.

At the same time, we have ample evidence for information, the basis for knowledge, being energetic and physical. Shannon and following researchers have provided quantitative ways and relationships for understanding the nature of information, messages, transmission losses, and entropy. We have seen how structure is also intimately related to this theory, providing the substrate by which free energy gets dissipated in a high-energy input nonequilibrium system, characteristic of life here on Earth. These ideas of structure and canonical forms further help us to think about the architectural designs of our information systems.

Peirce's ideas about information being a limit function comprised of the full breadth and depth of what we can know about any given thing, approximate, for the totality of things, what may be the limit of information in the absolute. It affirms that much over which we may reason is best expressed as statistics or probabilities. The absolute limit of information, though unknowable, should perhaps be estimable on information theoretic bases.

As we first diagrammed in Fig. 2.1, I think a deep relationship exists between Shannon's information theoretics and Peircean semiosis. We have the building blocks to tie together absolute information, messages and losses, recipient response, and meaning and entropy. My intuition, still to be tested, is that the absolute limit of what we have come to understand as information is energetic and physical. Nadin, also from a Peircean but different perspective, sees a similar complementarity between information processes and semiotic processes [2].

In inspecting these relationships, we have seen the advantages of the simple over the complex in our structures, and how recursion and automata make simpler patterns act like engines. The combination of logic (broadly defined to include abduction as Peirce did) and mathematics and entropy, informed by the guidelines of the universal categories and semiosis, should prove a fruitful playground for musing about knowledge representation and our tools to work with it.

Peirce: The Philosopher of KR

I discuss Peirce the person and some of his unrelated aspects of philosophy in *Appendix A*. But, as our constant companion through this book, it is now apparent that Peirce is something like a patron saint of knowledge representation. There has not been a single topic within KR for which Peirce does not offer trenchant insights. This illumination is not limited to the direct items of information, knowledge, and representation. Most importantly, Peirce's insights relate to how we think about and conduct knowledge representation, and how we choose practically among alternatives moving forward.

It is not surprising that most perhaps best know Peirce as the founder of pragmatism, despite the depth and breadth of his contributions in other areas. The logical endpoints of his inquiries most often lead to the practical aspects of how to act. Paradoxically, most still treat Peirce as a subject of theoretical discussion and rarely put his guidance into practice. In computer science, for example, no working Peircean semiotic systems exist to my knowledge, and the field has effectively ignored abductive reasoning. The lack of applying Peirce's ideas of pragmatism to real problems feels disappointing. We are overlooking manifest opportunities. It is time to square this circle.

Knowledge and Peirce

Peirce wrote a series of papers arguing against a Cartesian view of the world, a view which places truth solely in the mind and refuses to accept the primacy of external reality.[2] The world is not exclusively one of deductive logic. Objective truth can be approximated by the scientific method. This approximation of truth can always be the subject of inquiry based on different perspectives, or new facts or insights. Beliefs imposed from without or driven by social pressures alone are dead ends to knowledge and understanding. What is real is mostly external to us that we collectively adjudicate through reason and consensus. How we think about, organize, and define our problem spaces is central to that process. In the words of Qiwei Chen [3]:

> "Peirce teaches us that the human capacity for knowledge is both unlimited and limited. It is limited in the sense that perfect knowledge cannot be fulfilled in any one individual person and any one particular moment, but as the history of science has shown, every presumed limit has been proved to break down and to be overcome by the progress of knowledge from generation to generation. If it is considered as a process realized in all human beings both past and future, human knowledge is constantly increasing and 'may increase beyond any assignable point,' that is, there is no absolute limit that might restrict it. Indeed, 'an absolute termination of all increase of knowledge is absolutely incognizable, and therefore does not exist' (CP 5.330)" (p. 47).

Peirce insists that probabilities and chance amidst continuity also direct us to use inductive and abductive logic to anticipate the future. Peirce provides clear guidance on what is information, with meaning defined ultimately upon what we believe and act. Information is a product function of what is intensional that characterizes something with what are extensional connections to external things. Through habit or repeated observations, we may come to believe this information sufficient to act, at which point we are responding to knowledge. This knowledge is not immutable, though it does require a 'surprising fact' or loss of belief to stimulate new inquiry. Abductive reasoning and then the choice of working hypotheses to test follow.

---

[2] Peirce wrote three articles for the *Journal of Speculative Philosophy* in 1868-9: "Questions Concerning Certain Faculties Claimed for Man" (CP 5.213-63; EP 2:11-27); "Some Consequences of Four Incapacities" (CP 5.264-317; EP 1:28-55); and "Grounds of Validity of the Laws of Logic: Further Consequences of Four Incapacities" (CP 5.318-357; EP 1:56-82).

Generating new ideas and testing the truth of them is a logical process that we can formalize. Critical to this process is the proper bounding, definition, and vocabulary upon which to conduct the inquiries. As Peirce argued, we need to express the potentials central to the inquiries for a given topic through a suitable speculative grammar. The guiding lens for how we do this thinking comes from the purpose or nature of the inquiries at hand. In the case of machine learning applied to knowledge bases, this lens, I have argued, should be grounded in Peirce's categories of Firstness, Secondness, and Thirdness, all geared to feature generation upon which machine learners may operate. The structure of the system should also be oriented to enable (relatively quick and cheap) creation of positive and negative training sets upon which to train the learners. In the end, the nature of how to structure and define knowledge bases depends upon the uses we intend them to fulfill [4].

We also see, however, that knowledge representation, while symbolic, is not limited to the realm of the symbolic. Some of our knowledge is unconscious or instinctual and may be triggered by the dyadic kinesthetic or by the sudden alarm or alert. The nature of the stimulus (or predicate) giving rise to these signals helps direct what kind of signal and action-response might get triggered. Looking to embed our efforts to understand human language and communication in robotic testbeds should help continue to guide our understanding of these factors.

These strands of argument point to Peircean insights about the nature of knowledge. Peirce's contributions extend to the representational as well. The general ideas of signs and sign-making are the first level of contribution. We also gain much from Peirce's concepts of denotations and indexicality. The rationale for splitting our predicates into the broad groupings of attributes, external relations, and representations is a significant advance over conventional upper ontologies. The fact we have a working knowledge artifact, KBpedia, available for free to use in semantic technology and knowledge representation instantiations is a crucial basis for testing and extending Peirce's ideas about knowledge further.

Enticing connections occur between Peirce's ideas and very topical fields in computer science beyond machine learning, natural language understanding, and robotics. Two of these are possible bridges between description logics and category theory [5, 6] and the emerging field of homotopy type theory. We also have the links to the many promising approaches to computational linguistics as discussed in Chap. 16.

Time to Move from Theory to Practice

The semantic Web needs to play a central role in data integration and interoperability. Fortunately, as we have seen in other areas, semantic technologies lend themselves to generic functional software that can be designed for reuse in most any knowledge domain, chiefly by changing the data and ontologies guiding them. This design means that we can build reference libraries of groundings, mappings, and transformations over time and reuse them across enterprises and projects. Functional

programming languages align well with the data and schema in knowledge management functions and ontologies and DSLs, domain-specific languages. These prospects parallel the emergence of knowledge-based AI (KBAI), which marries electronic Web knowledge bases with improvements in machine learning algorithms.

We have ample evidence of the possible areas for which Peirce's ideas may offer unique and valuable insights to all areas of semantic technologies, knowledge representation, and information science. It is time, after a hundred years and many books and learned papers, for how we learn from and use Peirce to move from the theoretical to the practical.

I hope that this practionary and KBpedia stimulate more practical use and testing of Peirce's insights. Whether KBpedia, an outgrowth of it, or something entirely different, seeing a reference standard emerge for interoperating across multiple datasets and communities would be a potent seed to nucleate still further insights and understanding. We have not yet seen the catalyst that will trigger the cascade of emergent properties one would see from the network effect.

I think one of the reasons we have seen theory prevail over practice with Peirce is the fear of failing, the intimidation of trying to encapsulate a working system that captures the breadth and depth of C.S. Peirce's genius. However, Peirce himself had a pretty sanguine view of his limitations, as he stated in 1906 in "Pragmatism in Retrospect: A Last Formulation" [7]:

> "I here owe my patient reader a confession. It is that when I said that those signs that have a logical interpretant are either general or closely connected with generals, this was not a scientific result, but only a strong impression due to a life-long study of the nature of signs. My excuse for not answering the question scientifically is that I am, as far as I know, a pioneer, or rather a backwoodsman, in the work of clearing and opening up what I call semiotic, that is, the doctrine of the essential nature and fundamental varieties of possible semiosis; and I find the field too vast, the labor too great, for a first-comer. I am, accordingly, obliged to confine myself to the most important questions. The questions of the same particular type as the one I answer on the basis of an impression, which are of about the same importance, exceed four hundred in number; and they are all delicate and difficult, each requiring much search and much caution. At the same time, they are very far from being among the most important of the questions of semiotic. Even if my answer is not exactly correct, it can lead to no great misconception as to the nature of the logical interpretant. There is my apology, such as it may be deemed" (CP 5.488).

Besides espousing 'fallibility,' Peirce took fallibility to heart. We have surely made many mistakes in our efforts to apply Peirce's guidance to a working knowledge representation system in KBpedia. I have perhaps misunderstood what Peirce had to say in multiple areas. Perhaps, some areas where we have accurately followed Peirce, his guidance may be wrong. We provide facilities on the KBpedia Web site to communicate those mistakes to us and to participate in KBpedia's ongoing improvement. Charles Sanders Peirce, the philosopher of knowledge representation, would undoubtedly prefer to see us struggle, fail, and improve upon his insights in making our knowledge representations practical, than not trying at all.

Reasons to Question Premises

One often finds at the end of a journey that what one thought they would discover or experience on the journey did not prove out. We learn things while on the journey that may cause us to change our initial premises. We encounter new things and take forks in the road. These shifting directions are the idea of fallibility in action, and it is also useful to look at why we got some of our premises wrong and what we have learned.

I remain convinced that enormous opportunities exist for applying Peircean semiotics to knowledge representation. I started with that premise, and end with that premise. With half of the modern US economy based on information, with a rapidly growing percentage globally doing the same, figuring out how to turn that information into knowledge and then to leverage that for economic benefit would be a Rosetta Stone. I also began with the premise that the failures to adopt working knowledge management systems were a combination of technology and culture. That premise, too, remains unchanged, but I also do not have a better idea as to which of culture or technology is more operative. What is clear is that a change in perspective is required to unleash new growth, one which demands energy and management attention.

AI Is a Field of KR

I have found Peirce's idea of prescission powerful and subtle. It is powerful because it is an entirely logical, nonpsychological way to decide a subsumption relationship.[3] Prescission, or its verbs *prescind* or *prescinded from*, is the process of comparing two items and seeing if either may exist independent of the other. If so, we say that the independent one is prescinded from the dependent one; it is one way to determine a subsumption relationship. The idea of prescission is subtle because, personally, I find getting the direction of the predicate correct is sometimes difficult, and some cases require much thought to discern. In Peirce's terms, 'prescission' is not yet so general for me as being habitual.

When I began this book, I blithely assumed that knowledge representation was a subfield of artificial intelligence. Every taxonomy that I have seen about AI subfields and that included consideration of knowledge representation shows KR as a subsidiary field. I frankly had never questioned the relationship.

However, when considered, mainly using prescission, it becomes clear that KR can exist without artificial intelligence, but AI requires knowledge representation. We can only pursue artificial intelligence via symbolic means, and KR is the transla-

[3] Or a sibling relationship where prescission works in both directions, as for *red* and *blue*, or *squares* and *triangles*.

tion of information into a symbolic form to instruct a computer. Even if the computer learns on its own, we represent that information in symbolic KR form. This changed premise for the role of KR now enables us to think, perhaps, in broader terms, such as including the ideas of instinct and kinesthetics in the concept. This kind of reconsideration alters the speculative grammar we have for both KR and AI, helpful as we move the fields forward.

So, rather than the definition at the beginning of this book as repeated a few pages prior, we should now state that *knowledge representation* is dedicated to symbolizing information about the world in a form that a computer system can utilize to solve complex tasks and useful to subfields such as artificial intelligence.

Hurdles to Be Overcome

Unfortunately, many of the lessons learned deal with the impediments to effective knowledge use and management. Most of the critical obstacles to overcome are not technological, rather social or attitudinal. We need to break away from dichotomous or Cartesian thinking. We need to inculcate a better appreciation for information and knowledge as assets, including the value of purposeful discovery and management. We need to understand the nature of signs and representation and commit to the use of semantic technologies to bridge differences and capture meaning. These are skills that can be learned. However, without the commitment of top-level managers, incentives and processes will not be put in place to encourage their adoption.

Besides these failures of attitude and management, the manner in which we promulgate knowledge management in the organization fails for a further two reasons. One failure is to view knowledge management as its own 'application,' somehow separate and independent of standard work tasks. As we have argued, we need to include distributed, specific functions within current applications, coordinated as services to some form of governing workflow engine and ontologies. At the same time, this realization also opens up opportunities across the board in business process improvements. Knowledge management is itself a leading candidate for these improvements.

The second further failure is in not driving the KM function directly to the knowledge workers and users. Knowledge nurturing, discovery, definition, and use should be directly in the hands of those we pay for those responsibilities. KM, let alone the questions of KR, should not be the responsibility of IT (and RTFM while you are at it). Information technology has rightful responsibility for the security, operations, and maintenance of the information infrastructure, and should hold sway on those aspects for KM as well. Hegemony should stop there.

I noted before the advances shown in manufacturing in many of these areas. We are also now witnessing how product and distribution fulfillment centers are starting to see the fruits of automation and robotics. The next frontier is in the white-collar, knowledge-oriented portions of the economy. Here is where the next innovation

wave is due. Peircean approaches to knowledge representation combined with semantic technologies are the bright path to follow moving forward.

Of Crystals and Robots

As first noted in Chap. 11, Peirce famously claimed that thought does not necessarily occur in the brain, and that we may find thought in the work of crystals and bees, inanimate matter, and insects (1906, CP 4.551). We have also talked about its applicability to robots and AI. The most important lesson to emerge from our investigations might well be that some fundamental truths underlie the universal categories. During the second great wave of artificial intelligence in 1988 Daniel Dennett wrote that [8]:

> "AI is, in large measure, philosophy. It is often directly concerned with instantly recognizable questions: What is mind? What is meaning? What is reasoning and rationality? What are the necessary conditions for the recognition of objects in perception? How are decisions made and justified?" (p. 283)

Peirce, I believe, gives us guidance on all of these questions. Still, as a voice of theory, not yet validated by practice, Peirce may point the way, yet leaves many questions tantalizingly open.

Peirce understood graph structures. His language formulations and understanding of relations are at the forefront of much current computational linguistic research. His conception of mind embraced the external world if not was dominated by it. His interest in moving algebra to geometric forms and then topology fits well with the probability landscapes that now inform much thinking in machine learning and statistical mechanics. His writing about logic machines and electrical computation indicates that he was anticipating much that has come to pass.[4] His attempts to construct more elaborate and structured sign systems foreshadowed many aspects of ontologies and knowledge graphs. We can construct every idea that Peirce advocated from realities in the external world agreed to by the community. He was clear about the fundamental concepts of reality, existence, actuality, being, truth, chance, and continuity.

The neuroscientist Eugene Izhikevich in a recent debate with Roger Penrose said [9]: "We are at the stage of understanding consciousness as we were for information before Shannon. We lack a theory and definition for it that is agreed as likely correct." That is a fair assessment. Hopefully, we have taken some tiny steps on the path to that theory.

We want a theory grounded in reality, including quantum reality. We want a theory that embraces Shannon's information theory, yet one that extends its embrace to include meaning. We want a theory of signification and representation that can model energy fluxes that extend from inanimate matter to human symbol systems.

[4] See discussion on this topic in Appendix A.

We want a theory that captures the logic and message content of human language, one that can effectively communicate a symbolic representation to computers. We want a theory with a set of primitives that give us these capabilities while being small and straightforward. It will take many minds and much tinkering to complete the journey on this path that Charles Sanders Peirce has blazed for us.

References

1. K.A. Parker, *The Continuity of Peirce's Thought* (Vanderbilt University Press, Nashville, 1998)
2. M. Nadin, Information and semiotic processes the semiotics of computation. Cybern. Hum. Knowing **18**, 153–175 (2011)
3. Q. Chen, Some aspects of Peirce's theory of knowledge, in *Living Doubt (*G. Debrock and M. Hulswit, eds., Springer Netherlands, Dordrecht, 1994), pp. 43–53
4. M.K. Bergman, A speculative grammar for knowledge bases, in *AI3:::Adaptive Information* (2016)
5. E. Patterson, Knowledge representation in bicategories of relations, in *arXiv*, vol. 1706 (2017)
6. F. Zalamea, *Synthetic Philosophy of Contemporary Mathematics* (Urbanomic Sequence Press, New York, NY, 2012)
7. C.S. Peirce, *Philosophical Writings of Peirce: Selected Writings* (Justus Buchler, ed., Routledge and Kegan Paul Ltd., reissued by Dover Publications, New York NY, 1940)
8. D.C. Dennett, When philosophers encounter artificial intelligence. Daedalus **117**, 283–295 (1988)
9. Qualcomm Corporation, *AI Debate: How Far Can the AI Revolution Go?* (2018)

Appendix A: Perspectives on Peirce

Charles Sanders Peirce (1839–1914), pronounced 'purse,' was an American logician, scientist, mathematician, and philosopher of the first rank. His profound insights and writings spanned a half-century, and cover topics ranging from the nature of knowledge and epistemology to metaphysics and cosmology. Well known in the Americas and Europe early in his career, his foibles, and challenges to social orthodoxies, led to a precipitous decline in his fortunes, such that he died nearly penniless and unable to publish. Still, Peirce had a deep influence on many leading thinkers of his time, and as transcription and publishing of his voluminous writings move to completion, an influence that will continue for generations.

My first attraction to Peirce began with my professional interests in the semantic Web.[1] My earliest exposure to the semantic Web kept drawing my attention to questions of symbolic knowledge representation (KR). Like the genetic language of DNA in biology, my thought has been that there must be better (more 'truthful') ways of representing knowledge and information in digital form. My sense is that syntax or specific language is not the key, but that the basic building blocks of grammar and primitives hold that key. We further need a set of primitives well suited to natural language understanding, since humanity embodies so much of its cultural information in text. Structured data, such as from databases, is not an appropriate starting point; we critically need means to represent natural language. In Peirce, I have found the guide for those interests.

I have maintained throughout this book that Peirce is the greatest thinker ever in the realm of knowledge representation. Yet, KR, as a term of art, was not a phrase used in Peirce's time. Granted, Peirce wrote much on relations and representation (via his semiotic theory of signs) and provided many insights into the nature of

[1] Some material in this appendix was drawn from the author's prior articles at the *AI3:::Adaptive Information* blog: "The Importance of Being Peirce" (Sep 2016); "Being Informed by Peirce" (Feb 2017); "How I Study C.S. Peirce" (Aug 2017); "Why I Study C.S. Peirce" (Aug 2017); "A Foundational Mindset: Firstness, Secondness, Thirdness" (Mar 2016); "How I Interpret C.S. Peirce" (Sep 2017).

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8

information and knowledge, but he never used the specific phrase of 'knowledge representation.' He never attempted to categorize knowledge such as what we have undertaken with the KBpedia Knowledge Ontology (KKO), though he did make multiple attempts to classify the 'sciences' (fields of study in today's parlance). While Peirce had more than a glimmer of an idea that reasoning machines might someday be a reality, there was no need within his time to attempt to provide the specific representational framework for doing so.

Because of his influence—and his nearly constant presence throughout this book—I wanted to share what I have learned about Peirce the person, the polymath, the philosopher, and as a polestar guiding new directions in KR. I hope to convey a bit of the perspective about why you, too, should study Peirce, and help add to the interpretation of his fecund mind. I conclude this appendix with suggested resources you may find helpful to study this most remarkable human thinker.

Peirce, the Person

Charles S. Peirce was born into privilege in 1839 and was brought up among the intellectual elite in Cambridge, Massachusetts. His father, Benjamin Peirce, was a professor at Harvard and one of the prominent mathematicians of the 1800s. Charles received a first-rate education, including much personal tutoring by his father, and was given preference and sinecures at a young age, mainly through his father's connections.

Trained as a chemist at Harvard's Lawrence Scientific School where he graduated *summa cum laude* in 1863, Peirce was able to secure a deferment with his father's assistance from serving in the Civil War. Peirce was a working scientist for most of his employed career at the Coast and Geodetic Survey, then perhaps the premier US Government research facility, on gravitational differences around the globe, based on meticulous measurements using pendulums, often of Peirce's innovative designs. His early writings in the mid-1860s in areas of logic and metaphysics received wide acclaim. He was frustrated in securing a teaching position at Harvard,[2] but eventually became a lecturer at Johns Hopkins University, which was innovating in American graduate education, from 1879 to 1884, when he was summarily dismissed under unclear clouds of scandal. He subsequently had sporadic engagements in various entrepreneurial activities and wrote and translated articles for hire, but never had a permanent position again. His last decades were spent writing at his Milford, Pennsylvania home, Arisbe, which itself was in various stages of construction and disrepair based on vacillating, but declining, financial fortunes. By his death in 1914, he and his second wife, Juliette, were essentially penniless, having been sustained in part due to loans and charity from friends and family, orchestrated by his brother, James, himself a Harvard mathematician, and his lifelong friend, William James. Peirce had no children.

[2] In fact, due to enmity at Harvard, Peirce was barred from lecturing on campus for 30 years, only relaxed when Peirce was in his 60s.

In a stellar biography, Brent often refers to Peirce as a dandy in his earlier years [1]. Playing on the pronunciation of his name, two of Peirce's favorite self-descriptions were that he had 'Peirce-istence' and 'Peirce-everence.' He was certainly an iconoclast, and also flaunted society's conventions, living with Juliette before marriage and after being abandoned by his first wife, Zina. Peirce was a prodigious writer and very hard worker over 50 years, but was cavalier, if not unethical, in the abuse of his positions and public funds. He was reportedly a user of morphine and cocaine, ostensibly for neuralgia, but a factor that may have contributed to his sometimes perplexing inconstancies. Peirce often pursued his intellectual interests at the expense of his paid responsibilities. He created powerful enemies that ultimately kept him from securing a professorship at a leading university, which he and his family believed his birthright. He made poor decisions concerning money and finances, often disastrous ones, and died virtually penniless, with no fame and little notoriety. Still, Peirce befriended and influenced many of the leading thinkers of his time, including William James, Josiah Royce, John Dewey, and Oliver Wendell Holmes.

After Peirce's death, Harvard was scandalous in how it (mis-) handled his donated papers and restricted access for many years to his unpublished writings,[3] a continuation of the vendetta brought by Charles W. Eliot, the long-standing Harvard president. His supposed supporter and family friend, Simon Newcomb, routinely undercut Peirce. Thankfully, within two decades of his death, anthologies were published, and his reputation and stature began to grow. The understanding of his insights and accomplishments continues to grow as researchers study and release his voluminous unpublished writings. Peirce's reputation now is the highest it has ever been in the 100 years since his death, growing, and surely greatly exceeds whatever fame he saw during life.

Peirce was often the first to acknowledge how he changed his views, with one set of quotes from early 1908 showing how his thinking about the nature of signs had changed over the prior 2 or 3 years [3]. That example is but a small snapshot of the changes Peirce made to his sign theories over time, or of his acknowledgments that his views on one matter or another had changed.

Of course, it is not surprising that an active writing career, often encompassing many drafts, conducted over a half of a century, would see changes and evolution in thinking.[4] Most Peircean scholars acknowledge changes in Peirce's views over time, particularly from his early writings in the 1860s to those after the turn of the century and up until his death in 1914. Where Peirce did undergo major changes or refinements in understanding, Peirce himself was often the first to explain those changes. Many scholars have looked to specific papers or events to understand this evolution in thinking. Max Fisch divided Peirce's philosophy development into three periods: (1) the Cambridge period (1851–1870); (2) the cosmopolitan period (1870–1887);

[3] See further Nathan Houser, "The Fortunes and Misfortunes of the Peirce Papers" [2].

[4] Peirce's lifetime writings have been estimated at 100,000 pages, and Case has estimated that as many as three-quarters of his writings still wait transcription [4]. I doubt this estimate, but in any case discovery of new entire manuscripts is unlikely, since untranscribed pages seem to constitute mostly drafts of prior manuscripts.

and (3) the Arisbe period (1887–1914) [5]. Murphey split Peirce's development into four phases: (1) the Kantian phase (1857–1866); (2) three syllogistic figures (1867–1870); (3) the logic of relations (1879–1884); and (4) quantification and set theory (1884–1914) [6]. Brent has a different split more akin to Peirce's external and economic fortunes [1]. Parker tends to split his analysis of Peirce into early and mature phases [7]. It is a common theme of major scholars of Peirce to note these various changes and evolutions. Some of this analysis asserts breakpoints and real transitions in Peirce's thinking. Others tend to see a more gradual evolution or maturation of thinking. Some of the arguments bolster whatever particular thesis the author is putting forward. Such is the nature of scholarship.

For me, I take a pragmatic view of these changes. First, some of Peirce's earliest writings, particularly his 1867 "On a New List of Categories' [8], but also mid-career ones, are amazingly insightful and thought provoking. Tremendous value resides in these earlier writings, often infused with genius. Peirce, after all, was in the prime of his powers. Sure, I can see where some points have evolved, or prior assertions or terminology have changed, but Peirce is also good at flagging those areas he sees as having been important and earlier in error. I, therefore, tend to rely most on his later writings, when a hard life lived, maturity and experience added wisdom and perspective to his thoughts. I tend to see his later changes more as nuanced or mature, rather than radical breaks with prior writings. I see tremendous continuity and consistency of worldview in Peirce over time.[5]

Peirce considered himself foremost as a scientist, who probed and questioned premises with logic and purpose. Peirce's critical attention and refinement of the scientific method place him in the top tier of philosophers of science. Peirce believed that all questions lend themselves to scrutiny and logical analysis. Among the myriad of possibilities available to us for inquiry as scientists, Peirce's methods help point to those options most likely to yield fruit within limited time and resources, the essence of his philosophy of pragmatism. The universal categories provide us with a constant and consistent framework for representing, analyzing, and organizing knowledge.

Though many intellectual giants of history were recognized as such in their own times—Newton, Einstein, Darwin, and Aristotle come to mind—all of us like the story of the genius unjustly ignored in his lifetime. In science, famous examples include Copernicus, Galileo, Wegener, and Mendel. Charles Sanders Peirce belongs in this pantheon as well, a possible outcome I think he realized himself. The failure of his grant application to the Carnegie Institution in 1903 to synthesize his life's work, supported no less by Andrew Carnegie and Teddy Roosevelt, was Peirce's last attempt at broad-scale recognition. Ill, in poverty, and shunned by the establishment of his time, Peirce worked feverishly in his last years to get down on paper as much

[5] I make this assertion despite major shifts in some of Peirce's positions. For instance, Max Fisch in his 1967 paper, "Peirce's Progress from Nominalism Toward Realism," in *The Monist* (vol. 51, pp. 159–178) charts Peirce's evolution from some nominalist positions in his early writings to a full-blown "three-category realist" (Fisch's phrase) by the turn of the century. Still, this evolution—and others—only augments Peirce's lifelong theses regarding sign-making, logic, and universal categories.

as he could, pretty much laboring alone and in obscurity. We are still plumbing these handwritten papers, gaining new insights and perspectives of what we think we know about Peirce's philosophy and perspectives.

Philosophers, logicians, scholars, and laypersons study Peirce as a passion, many for a living. Though Peirce was neglected by many during the heyday of analytical philosophy throughout the twentieth century, that is rapidly changing. Walker Percy and Umberto Eco were two noted writers who have studied Peirce closely and written on him. The reason for Peirce's ascendancy, I think, is precisely due to the Internet, with then ties to knowledge representation and artificial intelligence. Peircean views are directly relevant to those topics. His writings in logic, semiosis (signs), pragmatics, existential graphs, classification, and how to classify are among the most direct of this relevancy.

However, relevant does not mean agreed upon, and researchers understand Peirce through their own lenses, as the idea of Peirce's Thirdness affirms. Given Peirce's own constant questioning and revision of his theories, plus the fragmented nature of the written record he left behind, I think it is fair to assert that we will never come to understand Peirce's 'truth' fully. Peirce was a man of complexity, unlikely to be fully plumbed. On the other hand, I also think we are only still beginning to understand how Peirce's insights can inform our understanding of the world.

Peirce, the Philosopher

Peirce did not view himself as a philosopher but as a scientist and logician. These distinctions are mere shadings in Peirce's philosophy, one that places high stock on truth, logic, representation, and the scientific method. Much of Peirce's philosophy figures prominently in the main body of this book, specifically in the role of semiosis and sign-making (Chap. 2); his universal categories, "truth" and fallibility, and categorization (Chap. 6); logic of relations and logic types (Chaps. 7 and 8); the role of natural classes (Chaps. 5 and 10); and pragmatism (Chap. 14). Here, however, I want to highlight the more cross-cutting aspects of Peirce's philosophy, not so directly related to KR, but also essential to understand his worldview.

Peirce's Architectonic

Peirce's *architectonic*, a word applied to the worldview for certain influential philosophers such as Kant or Aristotle, is built around the structure of all human knowledge. The pivotal elements of Peirce's architectonic are his universal categories, as manifested in logic, and evaluated through the pragmatic maxim. Peirce organized his classifications of science into disciplines using this system, in which he also embedded such topics as ethics, esthetics (his spelling), philosophy, and metaphysics, in addition to the classical sciences and humanities. Peirce evolved his

classification of the sciences considerably over time. Beverly Kent conducted a thorough analysis in 1987, much based on unpublished manuscripts at the time, that documents at least 20 different classifications over the period of 1866 to 1903 (the last, final one called the 'perennial'), with minor ones in between [9]. The three main branches of Peirce's perennial classification are mathematics, cenoscopy (philosophy), and idioscopy (the special sciences of traditional science and the humanities). Peirce believed that we should place philosophy within this systematic account of knowledge as science, and adopted the idea of the architectonic from the philosopher he idealized the most, Immanuel Kant. Peirce increasingly relied on this structural sense and the irreducible universal categories in most all of his later thinking.

Logic, as defined by Peirce, is only another name for semiotic (1897, CP 2.227). The clear thread through Peirce's writings is the respect and attention he gives to the primacy of logic, but also the role of community in deciding belief and terminology. Though, as a normative science (along with ethics and esthetics), logic is not the center root of his categorization of science, Peirce still bases all of his major arguments and insights on logic. Those insights include ones about the role and principles of logic itself.

> I do not, for my part, regard the usages of language as forming a satisfactory basis for logical doctrine. Logic, for me, is the study of the essential conditions to which signs must conform in order to function as such. How the constitution of the human mind may compel men to think is not the question; and the appeal to language appears to me to be made (and logicians generally do make it; in particular their doctrine of the copula appears to rest solely upon this), it would seem they ought to survey human languages generally and not confine themselves to the small and extremely peculiar group of Aryan speech (1904, NEM 4:243).

Via the classification of the sciences, Peirce attempts to organize and relate all aspects of knowledge and inquiry, and via the logic of semiosis Peirce provides a way to think about and represent that knowledge. Peirce subsumes these considerations under the irreducible foundation of the universal categories, though Peirce placed the study of these categories within phenomenology, another branch of philosophy. (Thus, phenomenology, normative science, and metaphysics provide the three branches of cenoscopy, or philosophy.) Peirce is also clear about these same groundings for his pragmatism. In a 1902 letter to William James, Peirce stated:

> [M]y three categories, … in their psychological aspect, appear as Feeling, Reaction, Thought. I have advanced my understanding of these categories much since Cambridge days; and can now put them in a much clearer light and more convincingly. The true nature of pragmatism cannot be understood without them (1902, CP 8.256).

Though we can see the universal categories subsuming logic, semiosis, and pragmatism, we can also see a tight nexus between all of the concepts. For example, Ika, in an overlooked doctoral thesis, provides lengthy analysis that places Peirce's universal categories at the foundation of his pragmatism [10]:

> … it can be said that Peirce's general philosophical project was most fundamentally concerned with some kind of *methodological* quest; a quest that seeks to establish the most fundamental categories that are both logically and metaphysically presupposed in any inquiry. The categories are logical presuppositions in the sense that they are principles or norms to be necessarily followed in the process of inquiry. They are also metaphysical presuppositions in the sense that Peirce rightly regarded them as reflections or representa-

> tions of reality. Peirce's unique brand of pragmatism, with its blend of logical rigour, practical orientation and realist metaphysical foundations was the end result of his methodological quest (p. 23).

which also ties into the idea and importance of logic:

> According to his classification of the sciences, metaphysics depends on logic for its fundamental principles, and logic depends on metaphysics for the data on which to operate. Although this relation of inter-dependence between metaphysics and logic is useful for determining certain aspects of his overall philosophical position, it is too rigid to account for another sense in which logic is dependent on metaphysics for Peirce, namely, that the whole end or intention of logic is contained within metaphysics [10] (p. 139).

but also recognizes that the categories subsume semiosis, providing the more general tenets:

> While Peirce appears to be preoccupied with his theory of signs, and sees sign and sign-action in every phenomenon, he did not seek to reduce reality to a semiotic system, where the real would be construed as only that which is sign-like. For Peirce, such a reductionist view of reality would result either in a dismissal of metaphysics or require that metaphysics be reducible to logic … both these views are inconsistent with Peirce's overall philosophical position, which recognises the distinction between the logical and the real as important [10] (p. 152).

We stride into a world with an uncertain future. We need to act and make decisions in the face of that uncertainty. We evaluate that world by the three logical methods of deduction, induction, and abduction. Peirce's architectonic provides a nexus of logic, signs, and universal categories to give us the tools we need to move forward, what Peirce calls pragmatism:

> Pragmatism … had been designed and constructed … architectonically. Just as a civil engineer, before erecting a bridge, a ship, or a house, will think of the different properties of all materials, and will use no iron, stone, or cement, that has not been subjected to tests; and will put them together in ways minutely considered, so, in constructing the doctrine of pragmatism the properties of all indecomposable concepts were examined and the ways in which they could be compounded. Then the purpose of the proposed doctrine having been analyzed, it was constructed out of the appropriate concepts so as to fulfill that purpose. In this way, the truth of it was proved (1905, CP 5.5).

We can thus understand Peirce's *architectonic* as the building blocks that go into constructing our structure of knowledge. How we go about thinking about these building blocks and then applying them to a given problem at hand, such as capturing a domain or inquiring where we have doubt, is what I refer to as a *mindset*. The universal categories are foundational to either of these two meanings.

Chance, Existents, and Continuity: Real

The three universal categories, as noted, are appropriately studied under the phenomenology section of the cenoscopic (philosophic) branch of the sciences. Though we earlier, in *Table 6.2*, listed many examples of Firstness, Secondness, and Thirdness, let's single out some phenomenological aspects of these categories that Peirce

emphasized in his writings. These three aspects are absolute chance for Firstness; actual, existing individuals for Secondness; and continuity for Thirdness. In some ways, these concepts are firsts among equals given their prominence in Peirce's thinking. If a grounding exists for the three universal categories, these may be it.

Chance

The fount of Peirce's universal category of Firstness is absolute chance. Peirce brings two remarkable insights about chance in his writings. The first insight, now somewhat prosaic but new for its time, was the importance of probability to many problems. The results, for many problems, are not absolute, but probable across a distribution of possible outcomes. It is essential to sample randomly, or by chance, to test these probabilities. Peirce was an early explicator about random sampling and statistics. Indeterminant problems are common, and an understanding of chance and probabilities is the only tractable way to assess them.

The second remarkable insight is more fundamental, and perhaps even more critical. It is what Peirce called *tychism*. Peirce was an early supporter of Darwin's theory of evolution and understood the role of variation. Peirce was first exposed to the ideas of evolution at least since the time of the Metaphysical Club, under the strong influence of friend and fellow club member, Chauncey Wright. Peirce's probability studies also enabled him to see that our world was one of 'surprising facts.' A completely random world would signal no variety. Absolute chance must, therefore, be leading to variants that cause us to inspect and understand emerging properties. Chance is itself offering up variants, some which have the character of persistence because of their stronger probability to be reinforced. These forces of chance give our world the variety and diversity it possesses. Laws and habits lead to regularities that both tend to perpetuate themselves as generalities, but also flash surprising variation that causes us to take stock and categorize and generalize anew.

In Peirce's cosmogony, these primitives of chance (Firstness), law (Secondness), and habit (Thirdness) can explain everything from the emergence of time and space to the emergence of matter, life, and then cognition. Though it is true that Thirdness (continuity) is the more synthesizing concept, the role of chance alone to drive this entire reality suggests its essential character [11]. Tychism is thus a philosophical doctrine that absolute chance is real and operative in the world, and it is the source of irregularity and variety and the underlying force of evolution.

Chance alone could be the variant that led to the minute differences arising during the Big Bang, which is posited to have led to matter and its structure. Chance is what enabled self-perpetuating life to emerge from inanimate matter. Chance is how forms of life could symbolically capture these variations via cognition and language. While all of this may now seem inevitable—though unexpected in how manifested—Peirce would maintain that they are events arising from chance. Perhaps most events have a cause, but the fundamental ones result from chance. 'Surprising facts,' a favorite phrase of Peirce, mean the world is unpredictable and ultimately probabilistic. The limits of Cartesian logic, the 0's and 1's, are likely never achievable. Reality is shaded and nuanced.

When Peirce began putting forth these ideas, specifically in his *Popular Science* series in 1878 in "On the Nature" [12], these were radical ideas. At the time of these publications, science was still decades away from quantum mechanics and the Heisenberg uncertainty principle. Moreover, even though Einstein (in) famously said that "God doesn't play dice with the world,"[6] Einstein himself, and his unsettling of Newtonian physics, was still three decades away. These examples are but a few of where Peirce had insight and prescience well in advance of later supporting science.

The reason for such insights, Peirce would say, and I would agree, is not that he was somehow miraculously able to see the future. But, through the rigorous application of logic, Peirce was able to see the requisite primitives of existence. As he wrote,

> The endless variety in the world has not been created by law. It is not of the nature of uniformity to originate variation, nor of law to beget circumstance. When we gaze upon the multifariousness of nature we are looking straight into the face of a living spontaneity. A day's ramble in the country ought to bring that home to us (1887, CP 6.553).

Peirce posited five reasons to believe in the reality (objective existence) of absolute chance: [6] (1) mechanical forces cannot explain growth and complexity in nature; (2) the sheer variety of nature; (3) uniformity develops from some state of determinacy; (4) no empirical evidence supports determinism; and (5) we can draw verifiable consequences from the hypothesis of chance (from 1892, CP 6.58-62).

Existents

Existents are what is actual, what exists, and consists of events and entities. Everything that exists is an individual and has an identity. Existents reside entirely in Secondness. "… existence (not reality) and individuality are essentially the same thing" (1901, CP 3.613). Existents have the nature of 'haecceity,' the idea of 'thisness' from the Latin, that gives them their particular uniqueness and identity.

Existents are thus an instantiation, something actual with identity, in comparison to the possibilities or qualities of Firstness, and in contrast to the generalities or continuities of Thirdness.[7] Existents embody qualities as found in Firstness, and may be generalized or related to continuous collections as found in Thirdness. Existents have some limits that bound their thisness, or haecceity, in either space (entities) or time (events). They exist whether we think them to do so or not. In Peirce's semiosis, actual existents are sinsigns.[8] We may indicate

[6] See https://en.wikiquote.org/wiki/Albert_Einstein

[7] Peirce sometimes also refers to relations between two existent objects as also being *existent*.

[8] Jon Alan Schmidt takes exception to this wording, noting that it entails that everything exists is a sign, which perhaps Peirce never stated in exactly this way. The quote we do have from Peirce is "that all this universe is perfused with signs, if it is not composed exclusively of signs" (1906, EP 2:394).

existents (via an index) as an object. Existents are real, but reality is not limited to them. Secondness is the most straightforward of the universal categories.

Continuity

Synechism, which Peirce equated to continuity,[9] is the notion that space, time, and law are continuous and form an essential Thirdness of reality in contrast to existing things and possibilities. Peirce notes that continuity is one of "the most difficult, the most important, the most worth study of all philosophical ideas" (1893, MS 717; NEM 4:310). I tend to agree.

> Now if we are to accept the common sense idea of continuity (after correcting its vagueness and fixing it to mean something) we must either say that a continuous line contains no points or we must say that the principle of excluded middle does not hold of these points. The principle of excluded middle only applies to an individual (for it is not true that 'Any man is wise' nor that 'Any man is not wise'). But places, being mere possibles without actual existence, are not individuals. Hence a point or indivisible place really does not exist unless there actually be something there to mark it, which, if there is, interrupts the continuity… On the whole, therefore, I think we must say that continuity is the relation of the parts of an unbroken space or time… The precise definition is still in doubt; but Kant's definition, that a continuum is that of which every part has itself parts of the same kind, seems to be correct. This must not be confounded (as Kant himself confounded it) with infinite divisibility, but implies that a line, for example, contains no points until the continuity is broken by marking the points. In accordance with this it seems necessary to say that a continuum, where it is continuous and unbroken, contains no definite parts; that its parts are created in the act of defining them and the precise definition of them breaks the continuity… Breaking grains of sand more and more will only make the sand more broken. It will not weld the grains into unbroken continuity (1902, CP 6.168).

Peirce clearly excludes individuals from continuity; indeed, they are disruptions to it. The principle of excluded middle also does not apply, since we are also dealing with generalities. He illustrates these ideas in multiple passages with the idea of points on a continuous line, such as this next example:

> A true continuum is something whose possibilities of determination no multitude of individuals can exhaust. Thus, no collection of points placed upon a truly continuous line can fill the line so as to leave no room for others, although that collection had a point for every value towards which numbers, endlessly continued into the decimal places, could approximate; nor if it contained a point for every possible permutation of all such values. It would be in the general spirit of synechism to hold that time ought to be supposed truly continuous in that sense (1902, CP 6.170).

[9] Peirce states, "I have proposed to make *synechism* mean the tendency to regard everything as continuous" (1893, CP 7.565). He goes on to say, "I carry the doctrine so far as to maintain that continuity governs the whole domain of experience in every element of it. Accordingly, every proposition, except so far as it relates to an unattainable limit of experience (which I call the Absolute,) is to be taken with an indefinite qualification; for a proposition which has no relation whatever to experience is devoid of all meaning" (CP 7.566).

We cannot distinguish things without making the line discontinuous. If something is inexplicable, it cannot be continuous:

> … synechism amounts to the principle that inexplicabilities are not to be considered as possible explanations; that whatever is supposed to be ultimate is supposed to be inexplicable; that continuity is the absence of ultimate parts in that which is divisible; and that the form under which alone anything can be understood is the form of generality, which is the same thing as continuity (1902, CP 6.173).

We now begin to see the intimate connection between continuity and generality. "True generality is, in fact, nothing but a rudimentary form of true continuity. Continuity is nothing but perfect generality of a law of relationship" (1902, CP 6.172). We can also relate continuity to the concepts of regularity:

> That continuity is only a variation of regularity, or, if we please so to regard it, that regularity is only a special case of continuity, will appear below, when we come to analyze the conception of continuity. It is already quite plain that any continuum we can think of is perfectly regular in its way as far as its continuity extends. No doubt, a line may be say an arc of a circle up to a certain point and beyond that point it may be straight. Then it is in one sense continuous and without a break, while in another sense, it does not all follow one law. But in so far as it is continuous, it everywhere follows a law; that is, the same thing is true of every portion of it; while in the sense in which it is irregular its continuity is broken. In short, the idea of continuity is the idea of a homogeneity, or sameness, which is a regularity. On the other hand, just as a continuous line is one which affords room for any multitude of points, no matter how great, so all regularity affords scope for any multitude of variant particulars; so that the idea [of] continuity is an extension of the idea of regularity. Regularity implies generality; and generality is an intellectual relation essentially the same as significance, as is shown by the contention of the nominalists that all generals are names. Even if generals have a being independent of actual thought, their being consists in their being possible objects of thought whereby particulars can be thought. Now that which brings another thing before the mind is a representation; so that generality and regularity are essentially the same as significance. Thus, continuity, regularity, and significance are essentially the same idea with merely subsidiary differences. That this element is found in experience is shown by the fact that all experience involves time. Now the flow of time is conceived as continuous. No matter whether this continuity is a datum of sense, or a quasi-hypothesis imported by the mind into experience, or even an illusion; in any case it remains a direct experience. For experience is not what analysis discovers but the raw material upon which analysis works. This element then is an element of direct experience (1908, CP 7.535).

At one point, Peirce claims that "continuity represents Thirdness almost to perfection" (CP 1.337), and Haack notes that abductive reasoning is the preferred logic for positing continuities [13]. Peirce relates his concept of time to continuity (CP 6.132), and claims that his ideas about fallibility are grounded in it:

> The principle of continuity is the idea of fallibilism objectified. For fallibilism is the doctrine that our knowledge is never absolute but always swims, as it were, in a continuum of uncertainty and of indeterminacy. Now the doctrine of continuity is that all things so swim in continua (1897, CP 1.171).

Peirce notes that classifying and typing things are also grounded in continuity:

> … it will be found everywhere that the idea of continuity is a powerful aid to the formation of true and fruitful conceptions. By means of it, the greatest differences are broken down and resolved into differences of degree, and the incessant application of it is of the greatest value in broadening our conceptions (1878, CP 2.645).

We thus see that Peirce's conception of continuity is a metaphysical theory as well as a methodological principle. Peirce and others have noted that the presence of continuity is not a construct of the human mind, but is part of reality [14].

What Is Real

Peirce grew over his working life to believe that all of these universal categories were real, and not merely figments of the human mind. "If I *truly know* anything, that which I know must be *real*" (EP 2:181). Fisch dated this transition to about 1897 [15] when Peirce accepted the reality of the category of Firstness, *i.e.*, of possibility, in addition to his then acceptance of the reality of the categories of Thirdness and Secondness, becoming what Fisch called a 'three-category realist.'[10] In Peirce's words:

> 'Truth is the conformity of a representation to its object,' says Kant. One might make this statement more explicit; but for our present purpose it may pass. It is nearly correct, so far as it is intelligible. Only, what is that 'object' which serves to define truth? Why it is the *reality*: it is of such a nature as to be independent of representations of it, so that, taking any individual sign or any individual collection of signs (such, for example, as all the ideas that ever enter into a given man's head) there is some character which that thing possesses, whether that sign or any of the signs of that collection represents the thing as possessing that character or not. Very good: now only tell me what it means to say that an object possesses a character, and I shall be satisfied. But even now, in advance of our study of definition, [we can] sufficiently see that we can only reach a conception of the less known through the more known, and that consequently the only meaning which we can attach to the phrase that a thing 'has a character' is that something is *true* of it. So there we are, after threading the passages of this labyrinth, already thrown out at that very conception of truth at which we entered it. Indeed, when one comes to consider it, how futile it was to imagine that we were to clear up the idea of *truth* by the more occult idea of *reality*! (1902, CP 1.578)

Reality, for Peirce, is that which has character independent of what we might think about it in our minds. It rejects the Cartesian mind-body duality. The measures of a character of a thing arise from its disruptions in continuity. These disruptions arise from evolving design and absolute chance.

Leaning into Pragmatism

In a probabilistic world, which it is, we see lines of evidence everywhere for inferring various aspects of the world, now and into the future. The truth is, as Peirce often makes clear, only the here and now are knowable; what might come next (into the future) is a probability. The stronger, or more definitive, means of inference, deduction, and induction can never apply to the future. I am not sure Peirce understood that his formulation of abductive reasoning was the needed pathway here, but it is also true that abductive reasoning is the only path to new knowledge or novelty. We must make practical choices in our limited time. From Ika's dissertation: [10].

[10] h/t to Jon Alan Schmidt; also see EP 2:186-195.

> The point is that pragmatism as a logical maxim is set to serve the assertion that there are real things; for without that assertion, pragmatism would be a meaningless enterprise, no matter how hard we think of it as only a logical principle. In his classification of the sciences, Peirce describes logic as the science of the category of Secondness, and metaphysics as the science of Thirdness. His whole point is that the sciences are just as closely related to one another as are the three categories. That is, according to his theory of categories, Secondness is meaningful because of the Thirdness it involves. Similarly, pragmatism as a logical maxim would simply remain meaningless if it did not involve metaphysics [10] (p. 149).

The future is not given. The future may be changed via action, or via chance. Some future conditions are more favorable to me as an entity in the present than other future conditions. The choice of next actions among many possible next actions should be guided, in Peirce's view, by pragmatic considerations for three reasons. One, not all alternatives may be tested simultaneously. Two, some alternatives are more likely instrumental than others. Three, any alternative has its own unique set of actions and steps, what might be called costs. Peirce developed the pragmatic maxim to provide guidance for what we should attend to next, and how.

Peirce, the Polymath

C.S. Peirce was a man of many capabilities and many accomplishments. His contributions spanned all three of the sciences of discovery (as he named them)—mathematics, cenoscopy, and idioscopy—previously discussed. Peirce's advances in mathematics, logic, physical sciences, and the scientific method are legion. He was the first to develop a theory of signs (semiosis), is the acknowledged 'father' of American pragmatism, developed diagrammatic ways to represent logic via existential graphs, and explicated a new kind of inference, abductive reasoning. He made contributions to linguistics, the categorization of the sciences, geodesy, and topology. His precise work on physical measures with pendulums and in chemistry led him to make advances in probability, statistics, and instrument errors. He was a realist and understood the limits to truth. His advances appear grounded in a relentless questioning of premises and a rigorous application of logic to the most basic questions. These quests led him to a fundamental cosmogony built around the irreducible and universal categories of Firstness, Secondness, and Thirdness.

One of the best general introductions to Peirce's lifelong accomplishments is provided in the hard-to-find "Introductory Note" by Max Fisch to Chap. 2 of Thomas Sebeok's 1981 book, *Play of Musement* [16]. For those keenly interested in Peirce's life and accomplishments, this obscure paper is worth tracking down. One thing we do know is that Peirce was a classifier throughout his life. His classifications range among the foundations of cosmology to phenomena, relations, natural classes, sciences and knowledge, signs and knowledge representation, logic, and mathematics. In keeping with that spirit, I, too, will classify Peirce's accomplishments according to the sciences of discovery.

Mathematics

Peirce's father, Benjamin, was a noted mathematician, and Charles grew up being tutored and challenged in math, including mathematical games. Peirce made substantial contributions to the field of mathematics in many areas throughout his working career, though he did admit to backing away from rigorous mathematical problems late in his life.

Peirce's deepest and broadest contributions were in mathematical logic, where he pioneered many new areas [17]. Putnam provides a good overview of Peirce's many intellectual contributions to logic [18]. We have already noted his explication of the third mode of logical reasoning, abductive logic, and his invention of the notation and rigor of existential graphs. The term 'first-order logic' is due to Peirce. He was a keen student and critic of the leading logicians of his era, De Morgan, Boole, Schröder, and Venn. Peirce considered his work on the logic of numbers and the analysis of the infinite as being one of his major mathematical contributions [1]. Peirce also referenced a three-valued logic, many methods for which he had already developed [19]. We can also point to a second major area of contributions in probability theory, then known as the Doctrine of Chances (1878, CP 2.645-66). He explicated important ideas in randomness and sampling and analytic methods. We have already seen how important probability and continuity were to Peirce's metaphysics.

In his earlier years, Peirce developed a calculus founded on the actualness of infinity and infinitesimals. He suggested a cardinal arithmetic for infinite numbers, years before any work by Cantor (who completed his dissertation in 1867) and without access to Bolzano's 1851 work. In 1880–1881, he showed how to do Boolean algebra via a repeated and sufficient single binary operation.

In 1881 he set out the axiomatization of natural number arithmetic, a few years before Dedekind and Peano. In the same paper, Peirce gave, years before Dedekind, the first purely cardinal definition of a finite set. In that same year of 1881, Peirce provided the first successful axiom system for the natural numbers. Soon after, he distinguished between first-order and second-order quantification. In the same paper, he set out what can be read as the first (primitive) axiomatic set theory, anticipating Zermelo by about two decades. He also made contributions in the areas of finite differences and linear associative algebra [20].

Peirce was intrigued with the ideas of geometric or notational expressions of algebra, often regarding notions of continuity. He was an explicator of some of the earliest foundations of topology, and his invention of existential graphs is a direct expression of this interest. Peirce is the inventor of the quincuncial projection of the sphere. He also claimed a proof that only four colors are needed to color a spheroidal map (the so-called four-color problem).

Peirce was also the first to apply statistical methods to comparative biography [1], and he also applied his mathematical approaches to what is today known as political economy and econometrics. In 1880 Peirce was elected as a member of the London Mathematical Society.

Cenoscopy

A prior section and many references throughout this book deal with Peirce's contributions to the science of cenoscopy, or philosophy, which are legion, including his co-founding of the Metaphysical Club in Cambridge in 1872. I only want to add one further point here, and it relates to the idea of the 'highest good,' or *summum bonum*. Peirce sees striving for the *summum bonum* as moving toward the perfection of Thirdness of continuity or generality by the process of evolution:

> … the pragmaticist does not make the *summum bonum* to more consist in action, but makes it to consist in that process of evolution whereby the existent comes more and more to embody those generals which were just now said to be *destined*, which is what we strive to express in calling them *reasonable*. In its higher stages, evolution takes place more and more largely through self-control, and this gives the pragmaticist a sort of justification for making the rational purport to be general (1905, CP 5.433).

In his sciences of discovery, Peirce places esthetics and ethics with logic as the three normative sciences, with their dependence on one another:

> My own view in 1877 was crude. Even when I gave my Cambridge lectures [1898] I had not really got to the bottom of it or seen the unity of the whole thing. It was not until after that that I obtained the proof that logic must be founded on ethics, of which it is a higher development. Even then, I was for some time so stupid as not to see that ethics rests in the same manner on a foundation of esthetics… (1910, CP 8.235).

Regarding which, Peirce elaborates further on the definition of esthetics and its relations to the *summum bonum*:

> Esthetics is the science of ideals, or of that which is objectively admirable without any ulterior reason. I am not well acquainted with this science; but it ought to repose on phenomenology. Ethics, or the science of right and wrong, must appeal to Esthetics for aid in determining the *summum bonum*. It is the theory of self-controlled, or deliberate, conduct. Logic is the theory of self-controlled, or deliberate, thought; and as such, must appeal to ethics for its principles. It also depends upon phenomenology and upon mathematics (1900, CP 1.191).

Peirce alludes to what is goodness, the esthetics to which ethics impels action as the governing principle of logic, in many areas throughout his writings [21]. His ideals of looking to the community to help guide inquiry and adjudicate truth are also grounded in his practical ethics. We can see an esthetic core to the ethics that govern Peirce's overall philosophy.

Idioscopy

In the area of the special sciences, that is, the standard sciences plus the humanities (nature and mind) that Peirce termed the *idioscope*, Peirce's contributions occur in three different areas. The first area, and most prolific, was Peirce's contributions as a scientist. The second area was as an inventor. The last area of contribution comes from Peirce's special skills as a person, a humanist.

Scientist

For most of his employed life, apart from his teaching at Johns Hopkins and the piecework that constituted much of his later employment, Peirce was a practicing physical scientist. He made notable contributions in geodesy, astronomy, metrology, and chemistry. As a practicing scientist, Peirce gained much appreciation for the difficulty and lack of precision and repeatability in measurements. He understood the importance of accurate tools and measurement standards for capturing small differences. These first-hand experiences contributed greatly to his views on probabilities and the role and significance of the scientific method.

Geodesy was a primary responsibility for Peirce during his more than three-decades-long employment at the US Coast and Geodetic Survey, as introduced previously. He proposed using the wavelength of sodium light as a means to measure the length of pendulums more accurately, anticipating the metric standard by many decades. These studies also helped improve our understanding and calculation of the exact shape of the earth [17].

Peirce made many contributions to astronomy, including computations of theoretical astronomy, stellar observations, and theories of error. He was among the first to propose (correctly) that the Milky Way forms a disc, and did pioneering work on the magnitude of stars in the Milky Way [1]. The only full-length book authored solely by Peirce during his lifetime was an 181-page monograph in 1878, *Photometric Researches*, on the applications of spectrographic methods to astronomy [22].

In many areas, various researchers have noted Peirce's foresight in his scientific endeavors. For example, Ilya Prigogine claimed that Peirce's "Design and Chance" article, written in 1884, with its view of time and the second law of thermodynamics, anticipated the 'new physics' of the twentieth century. We note other areas for which Peirce foresaw or alluded to future science or discoveries throughout this book. Peirce was elected as a Fellow of the American Academy of Arts and Sciences in 1867, and a member of the National Academy of Sciences in 1877.

Inventor

Besides inventions such as map projections, semiotics, and pendulum design mentioned elsewhere, Peirce was also a prolific developer of notations and classical inventions. In notations, his existential graphs certainly stand out. However, he was also an inventor of the Peirce arrow symbol for the logical 'neither nor,' also called the Quine dagger (NOR), and its NAND complement, the Sheffer stroke. Peirce also created a unique method of iconic handwriting, which he dubbed 'Art Chirography.'

In 1892, Peirce developed an electrolytic bleaching process for wood pulp. A few years later, he also invented an acetylene lamp generator, also later tied into a

hydroelectric project, that was competing with Edison's electric light. At this same time, after his dismissal from Hopkins, he also conducted stress engineering analysis for what would eventually become the George Washington Bridge in New York City [17].

His strengths in logic and his inventive mind also foreshadowed the modern computer era. Some claim that he invented the electronic switching-circuit computer [17]. In 1886, he saw that Boolean calculations could be carried out via electrical switches, anticipating Claude Shannon by more than 50 years. He also wrote on Charles Babbage and posited the use of electricity and logic gates for reasoning machines.[11]

Humanist, as Person

Along with his student Joseph Jastrow at Hopkins, Peirce was one of the first experimental psychologists in the United States, pioneering experimental studies in 'subliminal' perception. He had definite views on the concept of higher academic education as a pursuit of collective research, an approach that he embodied in the *Studies in Logic* in 1893, a collection of essays by Peirce and his students. He wrote over 300 book reviews for the *Nation* magazine, and wrote a textbook in elementary mathematics, unpublished in his lifetime, that Carolyn Eisele painstakingly recreated in the *New Elements of Mathematics* series [19, 20, 23, 24]. For many years, Peirce documented individual studies of great men (1900, CP 7.256). He proposed the *Logotheca* as an updated replacement for *Roget's Thesaurus*.

Peircean ideas have been influential in linguistics, specifically in the fields of cognitive linguistics, diachronic linguistics, linguistic semantics and pragmatics, and text linguistics, most driven by his semiotic insights [25]. Peircean ideas have also informed computational approaches to linguistics [26] and language parsing [27] (see also Chap. 16). He produced an important work on pronunciation of Shakespearean English [17]. Peirce was also an avid book collector and adviser to the New York Public Library for the purchase of scientific books [1]. As will be noted in the next section, Peirce was an accomplished lexicographer, specializing in definitions of technical topics. He also was a translator, sometimes for hire, in Greek, Latin, French, and German.

According to Brent [1], Peirce was a practiced actor, belonging to many amateur acting groups, and his wife, Juliette, was reportedly an actress of some ability. He was lauded at times as a storyteller, orator and debater, teacher, and lecturer, though

[11] References to Charles Babbage may be found at CP 2.56 and CP 4.611. For electrical logical machines, see Charles S. Peirce, 1993, "Letter, Peirce to A. Marquand" dated 30 December 1886, in Kloesel, C. et al., eds., Writings of Charles S. Peirce: A Chronological Edition: Volume 5: 1884–1886. Indiana University Press: 421–422, with an image of the letter page with the circuits on p. 423.

other occasional reports characterize certain of his lectures as rambling, unintelligible, or dislocated. Peirce even knew card tricks and practiced occasional magic tricks. He was very much interested in mazes and games and published a series in *The Monist* on "Amazing Mazes" later in his life. As a hobby and because of family illnesses, Peirce was also well versed in the history and theory of medicine.

An Obsession with Terminology

Though Peirce frequently railed against nominalism, arguing instead for a realistic view of the world, he was also very attuned to names, labels, and definitions. He sought the 'correct' way to label his constructs. As one instance, at various times, Peirce called abductive reasoning *hypothesis*, *abduction*, *presumption*, and *retroduction.* He also called the methodeutic *speculative rhetoric*, *general rhetoric*, *formal rhetoric*, and *objective logic*. Such changing names were not uncommon with Peirce.

In his lifetime, Peirce both enjoyed and made money as a lexicographer defining terms. He personally wrote 6000 entries for the 12-volume *Century Dictionary* [28], and oversaw a total of 16,000 entries where he had primary responsibility in such areas as logic, mathematics, mechanics, measurement, philosophy, astronomy, and universities [17]. Peirce held that the understanding of a language symbol is a process of shared consensus among its community of users; he loathed to use common terms for his constructs. Indeed, when one of his terms, pragmatism, was adopted by William James who gave it a different spin and interpretation, Peirce disavowed his earlier term and replaced it with the term *pragmaticism*.

> So then, the writer [Peirce], finds his bantling 'pragmatism' so promoted, feels that it is the time to kiss his child good-bye and relinquish it to its higher destiny; while to serve the precise purpose of expressing the original definition, he begs to announce the birth of the word 'pragmaticism', which is ugly enough to be safe from kidnappers (pp 165–166) [29].

Peirce should have realized that understandability holds sway over individualized perspective. He was silly to argue with James about the term pragmatism, as James was doing so much to promote awareness of Peirce's ideas.

Table A.1 Examples of obscure Peirce terminology

Agapism	Cenoscopy	Interpretant	Phaneroscopy	Semeiotic
Anancasticism	Cyclosy	Legisign	Pragmastic Definition	Sinsign
Apeiry	Dicent	Medisense	Pragmaticism	Speculative rhetoric
Antethics	Entelechy	Methodeutic	Precission	Stecheotic
Architectonic	Fallibilism	Objective Idealism	Qualisign	Synechism
Axiagastics	Hylozoism	Percipuum	Representamen	Transuasion
Ceno-pythagorean	Hypostatic Abstraction	Periphraxy	Retroduction	Tychasticism
Chorisy	Idioscopy	Phaneron	Rheme	Tychism

This penchant for 'ugly' terms was not uncommon for Peirce. As examples, Table A.1 presents some terminologies from Peirce's writings. Changing and 'ugly' terminology is but the first of the difficulties in reading and understanding Peirce. His evolution as a thinker, plus the interpretations of those who study Peirce, also complicates matters. A real point about interpretation, I think, is to try to get past his sometimes off-putting terminology. Mostly what is hard to understand are terms you may be encountering for the first time.

I can appreciate Peirce's preference for precision in how he described things. I can also understand scholars sometimes concentrating more on literalness than meaning. But the use of obfuscatory terms or concentrating on labels over the conceptual is a mistake. When looking for a precise expression of new ideas I try to harken to key Peircean terms and concepts, but I sometimes find that alternative descriptions within Peirce's writings better communicate to modern sensibilities. Concepts attempt to embody ideas, and while it is useful to express those concepts with clear, precise and correct terminology it is the idea that is real, not the label. In Peirce's worldview, the label is only an index. I concur. In the semantic Web, we sometimes refer to this as 'things, not strings.'

Peirce, the Polestar

That we live in an age of information and new technologies and new developments is a truth evident to all. These developments lead to a constant barrage of new information, often leading to new or revised assertions ('facts'). What we believe and how we interpret that information are what we call knowledge. New facts connect to or change our understanding of old 'facts'; those connections, too, are a source of new knowledge. Our (1) powers of observation and learning and discovery; (2) interactions and consensus-building with communities; and (3) the methods of scientific inquiry, all cause us to test, refine, and sometimes revise or discard what we thought were prior truths. Knowledge is thus dynamic, continually growing, and subject to revision.

What I call a Peircean *mindset* can help inform answers to new problems, problems that Peirce did not directly address himself. Indeed, the problems that set this context are machine learning and natural language understanding, all driven by computers and electronic data unimagined in Peirce's day. Because my views come from my own context, something that Peirce held as an essence of Thirdness, I cannot say that I *base* my views on Peirce's views. Who knows if he would endorse my views more than a century after his death? However, my take on these matters is the result of much reading, thought, repeat reading, and study of Peirce's writings. So while I cannot say I *base* my views on Peirce, I can certainly say that he *guides* me.

Peirce's universal categories of Firstness, Secondness, and Thirdness provide the mindset for how to think about and organize knowledge. The tasks of defining and organizing knowledge demand that we bring meaning, context, and perspective to the task. I believe Peirce's universal categories and what they imply offer the next adaptive climb upward for knowledge representation. The overarching framework of Peirce's philosophy—his architectonic—is grounded in these categories. As a scientist and logician, Peirce applied this mindset in pragmatic and testable ways. These methods, indeed the scientific method itself, further guide how and where to apply this mindset in ways that are economical and promise the most knowledge among all of the possible paths of inquiry.

Peirce's fierce realism, his belief in reality beyond our minds, and his insistence that this reality is subject to inquiry and the fixation of belief leading ever closer to truth are distinctly different from the mind-body duality put forward by Descartes. Richard Bernstein in a recent book [30] called this viewpoint a sea change:

> Pragmatism begins with a radical critique of Cartesianism. In one fell swoop, Peirce seeks to demolish the inter-related motifs that constitute Cartesianism [mind-body duality; primacy of personal experience; doubt as a starting condition; there are incontrovertible truths to be discovered] … We can view the development of pragmatism from Peirce until its recent resurgence as developing and refining this fundamental change of philosophical orientation—this sea change. A unifying theme in all the classical pragmatists as well as their successors is the development of a philosophical orientation that replaced Cartesianism (in all its varieties) (pp 18–19).

Our real world is always changing, continually unfolding. Our real world is viewed by all of us differently, based on background, predilection, perspective, and context. What we think we know about the world today is subject to inquiry and new insights. New factors are arising to shift what we think we know about ourselves and our place in the world.

How I interpret Peirce, and why he has become a polestar in my thinking about knowledge representation, embraces three perspectives. First, given the breadth of Peirce's insights, I try to read as much by him and about his writings by others as I can. This exposure helps set a fertile milieu, useful to interpretation and critical judgment. Second, despite my awe of Peirce's genius, I do not treat his writings as gospel. Were he alive today, I do not doubt that the massive increase in knowledge and information since his day would cause him to alter his viewpoints—perhaps substantially so in some areas. Third, no similar reason compels us to shy away from questioning any of Peirce's assertions. Nonetheless, given Peirce's immense powers of logic, one better be well prepared with evidence and sound reasoning before undertaking such a challenge.

Resources About Peirce

Slowly at first, and then growing after the publication of the *Collected Papers*,[12] a legion of researchers and academics have labored to preserve, understand, and explicate Peirce's insights. Virtually every author and name mentioned in this book have played such a role, with hundreds more, some more active than those cited, contributing their part. I share in this section some of my preferences and personal selections for useful resources about Peirce.

My first recommendation to begin learning about Peirce is to start with Wikipedia. Its English entry on Charles Sanders Peirce is quite good and rather complete. An entire category is dedicated to Peirce on Wikipedia, with some 40 articles listed. I think the articles on semiosis, abductive reasoning, and pragmaticism are some of the better ones. Unfortunately, the article on Peirce's universal categories is pretty weak. To compensate, however, the Wikipedia Peirce bibliography is an excellent reference source.

Peirce is hardly easy to read, and most of what others write about him is also pretty dense. Though those seasoned in Peirce studies might find it covering standard ground, the 2013 Cornelis de Waal guide to Peirce [17] is an accessible introduction to Peirce and his contributions. I no longer consult it for facts or details, but as an intro it is helpful and a relatively quick read. If this piques your interest, then it is probably worth your time to start exploring Peirce in more depth. I also like de Waal's labeling of the 'doctrine of the categories.'

After introductions, it is best to study Peirce in his own words. The earliest known compilation of Peirce's writings was by Cohen in 1923 [31], nearly a decade after Peirce's death, and is both a good intro and starting compilation. An even better starting compilation is that of Buchler [32]. However, I did not start with either of these nor with de Waal, because my initial research discovered that searchable PDF versions of the first 'complete' compilations of Peirce's writings could be obtained for free online [33]. The *Collected Papers* are available online in a version easier to read than the PDF versions, and which you can search.[13] The problem with these CP sources, however, is that the editorial order of CP is not chronological, gaps exist because of the sources initially chosen, and the formatting and editorial decisions are not equal to later standards. The online version is better for learning and reading purposes, but the lack of editorial oversight hurts CP irrespective of format. (A prior CD library is also no longer available [28].)

[12] The jumbled nature of the original *Collected Papers* means they should be used with caution, since they have no chronology. Many contemporary Peirce scholars now tend to date by year the passages they quote in order to overcome this problem.

[13] You can use Google to search within the textlog.de site, even though it is in German and does not have its own search function, by using a query similar to the following: https://www.google.com/search?hl=en&as_q=peirce+abduction&as_sitesearch=www.textlog.de. Note: Include "peirce" in the request, because there are other philosopher papers on the textlog.de site. Also note that this approach is tailored for English, with the example querying for "abduction"; replace your own search query in the query string.

Of course, many editors have compiled Peirce's writings. In mathematics, you likely want to focus on the fantastic four-volume series from Eisele [34], which can often be found for free online. As a nonmathematician, I found Volume 4 the most useful. For my interests in logic and knowledge representation, I have found *Vol 1* of *The Essential Peirce* [35] the best single compilation of relevant writings. In fact, you can reassemble the entire contents of EP (as it is abbreviated) from free, online PDFs, and I have, but that also means you lose the fantastic Nathan Houser introduction and the excellent packaging and portability of an actual paperback book. Many other compilations are also available (see the various bibliographic sources).

I almost uniformly find the introductions by the editors of these compilations provide useful insights about Peirce. The introductions often weave in relevant personal details to help evaluate Peirce as a person. The editors bring a perspective and context to Peirce's accomplishments since they offer an external vantage. Under the category of editorial compilations, I especially like Nathan Houser's introduction to EP. However, from different perspectives, the intros by both Brent and Murphey (see below) helped bring him alive to me.

After this kind of a dive into Peirce's writings, again usefully supported by the editor's intros, I find I want a big picture of Peirce, which covers his motivations, circumstances, discoveries, and maturation. I suspect these are the hardest of the books about Peirce to write. It requires a breadth of familiarity and a deep understanding of (at least what the author thinks are) Peirce's intentions. There also are variants of this approach, focusing on specific slices (such as religion [36]) or particular concepts or academic perspectives.

The online Arisbe, the Peirce Gateway,[14] lists some 210 books published on Peirce and related topics since 1995 or so, with 114 published since 2006 alone. The site further lists 357 doctoral dissertations about Peirce, most in the last few decades. Note that many of these sources are not in English since Peirce is studied worldwide, with a strong following in Latin America, especially Brazil and Colombia. The Arisbe site is helpful in that most entries include at least a paragraph of description, and often with links to more extended online excerpts. Arisbe is a good resource should specific topics pique your interest while studying Peirce.

Among the comprehensive studies covering the entirety of Peirce's life work, I will mention two. The first is the book from Kelly Parker in 1998 [7] that focuses on Peirce's emphasis on continuity (synechism). Parker writes well, is lucid, and has an excellent notes section. The second compilation, and one of my favorite Peirce reads, is the earlier 1993 book by Murphey [6] on the development of Peirce's philosophy. Some other scholars, notably Hillary Putnam, have suggested that Murphey's interpretations are often controversial. Murphey did, indeed, change some of his opinions of Peirce, especially about continuity, in the second edition. However, I find Murphey's analysis of the phases of Peirce's developments to conform to my sense. The latter section of his book is excellent. I find it strange that many other general recommendations for Peirce readings tend to overlook this

[14] http://www.iupui.edu/~arisbe/

book. Perhaps a bit of this neglect came from Putnam's early comments, but Murphey is one of the resources I most often consult.

When first learning about Peirce, it is striking how dominant semiosis and his theory of signs (and logic) pervade many of the resources. These are critical Peircean topics, but I find that it took me a while to probe beyond these topics into others I find even more fascinating. I have focused on Peirce's universal categories of Firstness, Secondness, and Thirdness. I have also been studying abductive reasoning, language grammars, link between logic and mathematics, and how Peirce's views dovetail into current topics in topology and category theory. About these last topics, I recommend Fernando Zalamea [37]. Zalamea's scholarship is quite advanced, and I do not recommend as a starting point, but after some exposure to Peirce I like the synthetic view that Zalamea brings to the table. His scholarship shows that Peirce continues to produce major insights for modern logic and mathematics.

Biographies are another useful source. Louis Menand won a Pulitzer Prize for his recounting of the birth of pragmatism in the United States [38]. He told the story through the lens of the major participants in the Metaphysical Club, really more of an informal grouping of intellectuals. William James, Chauncey Wright, and Oliver Wendall Holmes figured prominently in that group, but none perhaps more so than Peirce. (Peirce and James were lifelong friends, but Peirce tremendously respected Wright for his insight and intellect, and they were very close friends; Wright, unfortunately, died young at 45.) What is great about this book is that the author frames the movement to pragmatism through the prism of slavery and abolition, the Civil War, and rapid intellectual and technological change. This perspective makes for an excellent read because it does such a marvelous job of placing Peirce into the context of his times, as well as providing equivalently fascinating looks at his very accomplished colleagues. However, this is not the single book to read if you want to probe deeply into Peirce's theories and worldview.

My favorite biography of Peirce, whose publication is a pretty astonishing story in its own right, is Brent's life biography of C.S. Peirce [1]. Brent first began his biography of Peirce to answer the question of who invented the US philosophy of pragmatism, triggered by clues in a biography of Peirce's friend, William James. He completed his dissertation in 1960 and intended to publish it, but ran into permission difficulties from Harvard, which was still acting poorly about Peirce's archival papers. Brent had to drop the project and moved on to other things. Then, in 1988, Thomas Sebeok, himself a then emerging Peirce scholar, encountered a description of the dissertation in a footnote in another book. He was able to get the dissertation through interlibrary loan and finally read it in 1990. He was astonished by what he learned and the quality of the work and set out to find Brent, whom he eventually tracked down in Washington, DC. Through Sebeok's catalyst, a publisher was found; Brent agreed to update his 30-year-old dissertation, itself an effort of considerable labor, and the work was finally published in 1993. Brent provides an unvarnished and, I believe, fair look at Peirce, the person, and shows insight into his accomplishments and unique ways of thinking about the world. Brent tackled head

on all of Peirce's foibles and weaknesses as well. The resulting biography is a masterpiece, what Sebeok termed a 'tragicomic thriller.' Brent himself came to believe "in philosophy [Peirce] was one of the most original thinkers and system builders of any time, and certainly the greatest philosopher the United States has ever seen." Brent came to feel 'deep affection' for his subject, despite those foibles and weaknesses. However, some Peirce scholars, such as Gary Richmond, think the biography unfair, with too much prominence given to Peirce's critics.

The Brent biography is an incredibly intelligent treatment of an incredibly intelligent man. As might be expected from a work that began as a dissertation, it is thorough and well referenced. As might not be expected from a dissertation, it is well written. Brent uses Peirce's own 'architectonic,' a term new to me then but studied by me now, a term drawn from Aristotle but modified by Kant and then Peirce, as a way of framing his treatment. Brent is also attuned to shifts in Peirce's thinking over time, a great boon to better understand the development of his theories. Since I believe others will study Peirce for centuries, along with other great thinkers of humankind, Brent's biography will be a must-include companion to Peirce's writings. As I note in the close to this chapter, Brent and Sebeok are but two of the hundreds of individuals who have made it their life's work and passion to understand Peirce better, to convey what he was trying to tell us, and to bring awareness of him to broader audiences. Also, a fictionalized biography of Peirce's mysterious second wife, Juliette, has some voyeuristic interest but is an unsuitable source for any reliable information about either Charles or Juliette [39].

The bulk of commentary, of course, about Peirce may be found in the academic literature. I often find when studying Peirce that a new topic (or one that finally gets my attention) will arise about which I want to learn more. As with all such topics, I first consult Wikipedia for a starting article, if one exists, to get a bit of background and then some key links and useful search terminology. However, my real focus in such investigations centers on Google Scholar. Google Scholar contains nearly 40,000 articles about or discussing Peirce, with the bulk, perhaps 70%, in English.[15] When searching Scholar, I always use "peirce" as one of my keywords and keep that search term in quotes (without the quotes, Scholar will also give you results from "pierce" since it seems to assume "peirce" is a misspelling). For papers of keen importance, I will also click the link 'Cited by xx' link on Scholar and do a secondary inspection of those to find other interesting papers that have cited the one of interest. I have assembled a complete electronic Peirce library of hundreds of documents over time in this manner.

A society is devoted to Peirce. One may find many Web sites such as Arisbe at the University of Indiana, online forums including for biosemiotics, annual conferences, and many individuals with their Web sites and writings who analyze and

[15] Here is an example query: https://scholar.google.com/scholar?q="peirce"+abduction. Substitute your own topic keywords for "abduction" in the example query string.

pronounce strong views as to what Peirce meant and how we should interpret him. I have often mentioned the influence of John Sowa in first getting me interested in Peirce, so his site (with query specific to Peirce) is a good one to include on your list. Sowa tends to focus on existential graphs, knowledge representation, logic, and natural language understanding. You may also find a good source for Web writers on Peirce on the Arisbe site; check out the blogroll on the left column. Of course, I, too, write not infrequently about Peirce. You may obtain my Peirce articles under my blog's Peirce category. I hope the dozen or so others who often write on Peirce forgive me for not directly mentioning them. Thank you, and I hope we see more.

For broad electronic resources on Peirce, probably the best is Arisbe, noted already.[16] Two high-quality, online philosophy sites, the Stanford Encyclopedia of Philosophy and the Internet Encyclopedia of Philosophy, are often useful introductory resources when beginning to learn about a new topic. Authoritative scholars write many of the Peirce articles. A site not updated since the early 2000s, but which has some unique and high-quality articles by outside experts, is the Digital Encyclopedia of Peirce. A useful site to see some different uses of specific Peircean terms is the Commens Web site. The Charles S. Peirce Project was established in 1976 to continue the mission of making Peirce's writings available, started by the *Collected Papers* (CP) project [33] going back to the 1930s. The Project continues to produce a multivolume chronological and critical edition of Peirce's writings. Romanini's Minute Semeiotic Web site is a fun way to explore what Peirce might have intended with his (incomplete) 66-sign schema.

Since first established by Joe Ransdall in 1993, a dedicated discussion list, Peirce-L, with often lively discussion, has nearly daily activity. That link will allow you to search archives going back to 2011, and to subscribe to the list. A similar mail list exists for a group in biosemiosis, another field that Peirce played no small role in helping to gestate. A useful piece of information if you study Peirce further, given that so much of his writing appeared long ago or has been transcribed or compiled by editors, is how to decipher the citation schemes. Good sources on Peirce citation standards are the Wikipedia CSP abbreviations and Robin catalog for citing papers and manuscripts. For the truly dedicated, you can help crowd-translate Peirce's unpublished manuscripts via the SPIN project co-directed by the Peirce scholar, Jeffrey Brian Downard.

[16] See http://www.iupui.edu/~arisbe/faqs/whyarisb.HTM for the history of the term Arisbe as used by Peirce for his Pennsylvania home.

References

1. J. Brent, *Charles Sanders Peirce: A Life* (Indiana University Press, Bloomington, 1998)
2. N. Houser, The fortunes and misfortunes of the Peirce papers, in *Signs of Humanity*, ed. by M. Balat, J. Deledalle-Rhodes (Mouton de Gruyter, Berlin, 1992)
3. C.S. Peirce, Peirce edition project. The ten main trichotomies of signs, in *The Essential Peirce – Volume 2: Selected Philosophical Writings (1893–1913)* (Indiana University Press, Bloomington)
4. S. Case, The man with a kink in his brain, in *National Review (online)* (2014), vol. 21
5. M.H. Fisch, Peirce's Arisbe: the greek influence in his later philosophy, in *Peirce, Semiotic, and Pragmatism: Essays*, ed. by M. H. Fisch, K. L. Ketner, C. J. Kloesel (Indiana University Press, Bloomington, 1986), p. 227
6. M.G. Murphey, *The Development of Peirce's Philosophy* (Hackett Publishing Company, Inc., Indianapolis, 1993)
7. K.A. Parker, *The Continuity of Peirce's Thought* (Vanderbilt University Press, Nashville, 1998)
8. C.S. Peirce, On a new list of categories, in *Proceedings of the American Academy of Arts and Sciences* (1867)
9. B.E. Kent, *Charles S. Peirce: Logic and the Classification of the Sciences* (McGill-Queen's University Press, Kingston, 1987)
10. S. Ika, *A Critical Examination of the Philosophy of Charles S. Peirce: A Defence of the Claim that his Pragmatism is Founded on his Theory of Categories*. Ph.D., University of Notre Dame Australia (2002)
11. P. Rose, Another guess at the riddle: more ado about nothing, in *Analecta Hermeneutica* (2013)
12. C.S. Peirce, The order of nature. Pop. Sci. Monthly **13**, 203–217 (1878)
13. S. Haack, Not cynicism, but synechism: lessons from classical pragmatism. Trans. Charles S. Peirce Soc. **41**, 239–253 (2005)
14. J.L. Esposito, Synechism: the keystone of Peirce's metaphysics, in *Digital Encyclopedia of Charles S. Peirce* (2001)
15. M.H. Fisch, Peirce's Progress from Nominalism Toward Realism. Monist **51**, 159–178 (1967)
16. M.H. Fisch, Introductory note, in *The Play of Musement* (Indiana University Press, Bloomington, IN, 1981), pp. 17–21
17. C. De Waal, *Peirce: A Guide for the Perplexed* (A&C Black, 2013)
18. H. Putnam, Peirce the Logician. Hist. Math. **9**, 290–301 (1982)
19. C.S. Peirce, *The New Elements of Mathematics – Volume 3* (Mouton, The Hague, 1976)
20. C.S. Peirce, *The New Elements of Mathematics – Volume 4* (Mouton, The Hague, 1976)
21. J. Feibleman, A Systematic Presentation of Peirce's Ethics. Ethics **53**, 98–109 (1943)
22. M. Bergman, C. S. Peirce: a short biographical sketch, in *Commens: Digital Companion to C.S. Peirce* (undated)
23. C.S. Peirce, *The New Elements of Mathematics – Volume 1* (Mouton, The Hague, 1976)
24. C.S. Peirce, *The New Elements of Mathematics – Volume 2* (Mouton, The Hague, 1976)
25. W. Nöth, Charles sanders Peirce, pathfinder in linguistics, in *Digital Encyclopedia of Charles S. Peirce* (2000)
26. G. Richmond, Outline of trikonic diagrammatic trichotomic, in *13th International Conference on Conceptual Structures, ICCS 2005*, ed. by F. Dau, M. L. Mugnier, G. Tumme (Springer-Verlag GmbH, Kassel, 2005), pp. 453–466
27. J.F. Sowa, Toward the expressive power of natural language, in *Principles of Semantic Networks* (Elsevier, Oxford, 1991), pp. 157–189
28. C.S. Peirce, *The Writings of Charles S. Peirce—A Chronological Edition*. Electronic Edition, InteLex
29. C.S. Peirce, in *The Monist.* What Pragmatism Is (1905), pp. 161–181
30. R.J. Bernstein, *The Pragmatic Turn*, Polity (2010)

31. C.S. Peirce, K.L. Ketner, *Chance, Love, and Logic: Philosophical Essays* (Lincoln, University of Nebraska Press, 1998)
32. C.S. Peirce, *Philosophical Writings of Peirce: Selected Writings* (Routledge and Kegan Paul Ltd., reissued by Dover Publications, New York, NY, 1940)
33. C.S. Peirce, *The Collected Papers of Charles Sanders Peirce*, vol 8 (Harvard University Press, Cambridge, MA, 1931)
34. C.S. Peirce, *The New Elements of Mathematics by Charles S. Peirce* (The Mouton Publishers, The Hague, 1976)
35. C.S. Peirce, *The Essential Peirce: Selected Philosophical Writings, Vol 1 (1867–1893)* (Indiana University Press, Bloomington, 1992)
36. M. Raposa, *Peirce's Philosophy of Religion* (Indiana University Press, Bloomington, 1993)
37. F. Zalamea, *Peirce's Logic of Continuity: A Conceptual and Mathematical Approach* (Docent Press, Boston, 2012)
38. L. Menand, *The Metaphysical Club: A Story of Ideas in America* (New York, NY, Farrar, Straus and Giroux, 2001)
39. M. Samuels, *The Queen of Cups* (Unlimited Publishing LLC, Bloomington, 2006)

Appendix B: The KBpedia Resource

KBpedia is a computable knowledge structure resulting from the combined mapping of six, large-scale, public knowledge bases—Wikipedia, Wikidata, OpenCyc, GeoNames, DBpedia, and UMBEL.[1] The KBpedia structure separately captures entities, attributes, relations, and concepts. KBpedia classes these into a natural and rich diversity of types, with their meaning and relationships, logically and coherently organized into about 80 typologies.

KBpedia is the first full-blown ontology based on Charles Sanders Peirce's universal categories and logic of relations. The KBpedia knowledge structure is written in the OWL semantic language; all underlying structures are represented in either OWL or RDF. KBpedia follows best practices, many of which were pioneered by KBpedia's editors, governing knowledge, and concept representation and annotations. All languages and knowledge representations are written in W3C-compliant standards.

The focal objective of KBpedia is to exploit large, public knowledge bases to support artificial intelligence using both supervised and unsupervised machine learning methods. KBpedia is explicitly designed to expose rich and meaningful feature sets to support the broadest range of machine learning methods. KBpedia is also specifically structured to enable useful splits across a myriad of dimensions from entities to relations to types that can all be selected to create positive and negative training sets, across multiple perspectives. The disjointedness of the SuperTypes that organize the 55,000 entity types in KBpedia provides a powerful selection and testing mechanism.[2] The coherency of KBpedia provides a basis for logic tests to further improve accuracy, including the creation of local gold standards, at an acceptable cost.

[1] Material in this appendix is drawn with permission from the KBpedia Web site at http://kbpedia.org

[2] This number is as of version 1.60 of KBpedia, based on the completion of this book in the first half of 2018. The current publicly released version of KBpedia likely has a different number of reference concepts. Back-of-the-envelope estimates suggest that KBpedia should eventually grow to be on the order of 80,000 reference types.

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8

KBpedia is a continually evolving reference structure for knowledge representation and management. KBpedia is staged to provide working levels of interoperability for the linked data ecosystem. Artificial intelligence (AI) and machine learning are revolutionizing knowledge systems. The most important factor in knowledge-based AI's renaissance has been the availability of massive digital datasets for the training of machine learners. Wikipedia and data from search engines are central to recent breakthroughs. Wikipedia is at the heart of Siri, Cortana, the former Freebase, DBpedia, Google's Knowledge Graph, and IBM's Watson, to name just a prominent few AI question answering or virtual assistant systems. Natural language understanding is showing impressive gains across a range of applications. To date, all of these examples have been the result of bespoke efforts leveraging Wikipedia, in whole or part. The tens of millions of instances captured by Wikidata add an entire ABox component to the knowledge structure. It is costly for standard enterprises to leverage these knowledge resources on their own.

Today's practices for leveraging these resources pose significant up-front and testing effort. Much latent knowledge remains unexpressed and not readily available to learners; it must be exposed, cleaned, and vetted. We need to spend further up front effort on selecting the features (variables) used and then to label the positive and negative training sets accurately. Without 'gold standards'—at still more cost—it is difficult to tune and refine the learners. The cost to develop tailored extractors, taggers, categorizers, and natural language processors is too high. KBpedia is meant to systematize a starter reference structure that new users may tailor to local domains at lower costs. Users may then apply integration and interoperability to structured, semi-structured, and unstructured data, that is, everything from text to databases. KBpedia proves that existing knowledge bases can be staged to automate much of the tedium and reduce the costs now required to set up and train machine learners for knowledge purposes. Besides labeled training sets for supervised machine learning, KBpedia, with its rich feature sets across all aspects of the knowledge structure, is also an excellent basis for selecting training corpora for unsupervised learning. It is often advisable to include some initial unsupervised learning in a more general supervised learning context.

Components

KBpedia is organized into a knowledge graph, KKO, the KBpedia Knowledge Ontology, with an upper structure based on Peircean logic. KKO sets the umbrella structure for how KBpedia's six constituent knowledge bases are related to the system. One of the three major branches of KKO, the Generals, represents the types in the system, with about 85% of the KBpedia's reference concepts residing there. These RCs are themselves entity types—that is, 47,000 natural classes of similar entities such as 'astronauts' or 'breakfast cereals'—which we organize into about

30 'core' typologies (among the 80 or so) that are mostly disjoint (nonoverlapping) with one another. The typologies provide a flexible means for slicing and dicing the knowledge structure; the entity types provide the tie-in points to KBpedia's millions of individual instances. The separate *Glossary* defines many of the terms used by KBpedia; Chap. 8 provides a more detailed discussion of KBpedia's vocabulary.

The KBpedia Knowledge Ontology (KKO)

We inform the upper structure that is the KBpedia Knowledge Ontology (KKO) using the triadic logic and universal categories of Charles Sanders Peirce. This trichotomy, also the basis for his views on semiosis (or the nature of signs), was in Peirce's view the most primitive or reduced manner by which to understand and categorize things, concepts, and ideas. We devote Chap. 6 to the universal categories and touch upon them and semiosis throughout the main book. We express KBpedia's knowledge base grammar in the semantic Web language of OWL 2. Thus, we may apply most W3C standards to the KBpedia structure. The resulting, combined structure, as shown in Fig. B.1, brings consistency across all source knowledge bases.

This diagram, drawn from KBpedia's online demo, shows the main topic areas, or typologies, that tie into KKO, which we list in their entirety in the *Structure* section below. The structure of KKO makes it a computable knowledge graph that supports inference, reasoning, aggregations, restrictions, intersections, and other logical operations. KKO's logic basis provides a powerful way to represent individual things, classes of things, and how those things may combine or emerge as new knowledge [1].

The KBpedia Knowledge Bases

Six, large-scale public knowledge bases are central to the KBpedia knowledge structure. These six sources are as follows:

- Wikipedia—five million articles that capture the fundamental concepts and entities of basic human knowledge, often including structured data and with many linkages
- Wikidata—structured data records for about 40 million individual entities
- OpenCyc—an extract of Cyc that represents the common sense and vetted relationships among KBpedia's base 55,000 concepts
- DBpedia—a machine-readable version of parts of Wikipedia in RDF
- GeoNames—a geographical database of some ten million places linked to about 800 distinct feature classes
- UMBEL—the initial organizational structure for the knowledge graph.

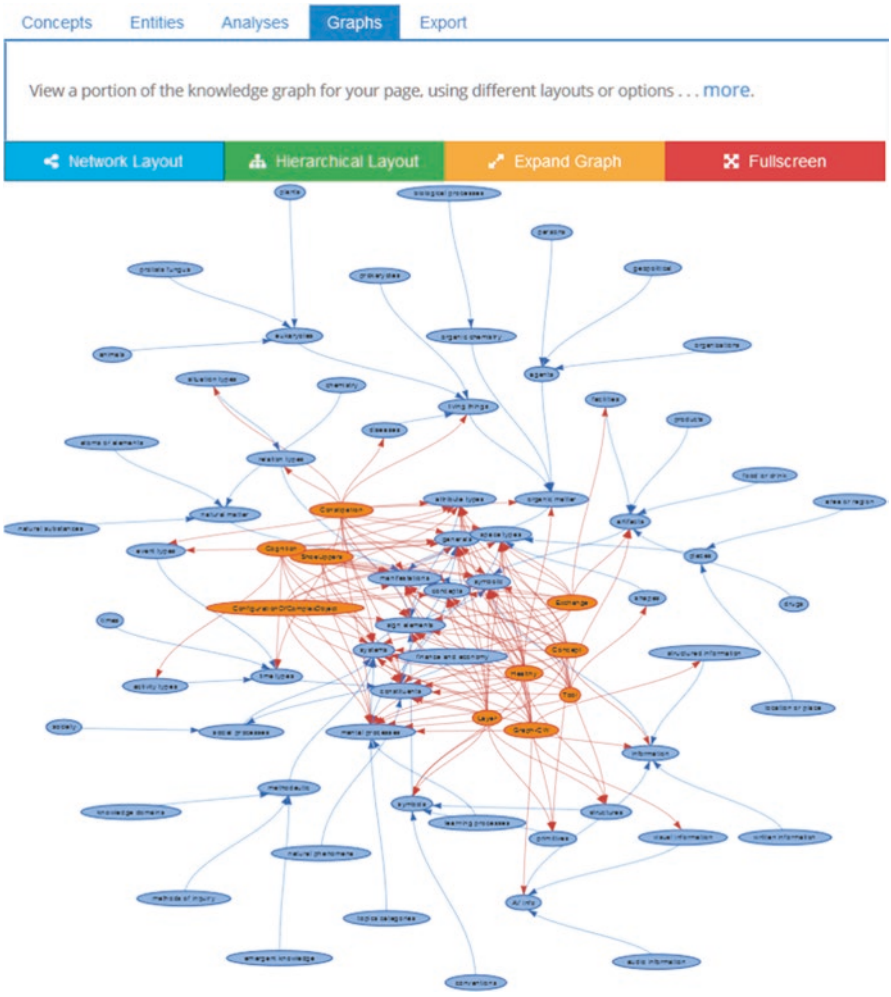


Fig. B.1 KBpedia (KKO) Network Graph

We have mapped and re-expressed each of these sources into the single, coherent knowledge system of KBpedia. We split the resulting KBpedia knowledge graph along the lines of concepts and topics, entities, events, attributes, annotations, and relations and their associated natural classifications or types. This resulting combination gives KBpedia a rich set of structural components.

Wikipedia and Wikidata are the two most important KB sources, Wikipedia for concepts, Wikidata for instances and properties, and both for multilingual capabilities. Certain aspects of Wikipedia have proven their usefulness for general knowledge acquisition, for example, using article (concept or entity) content to inform topical tagging using explicit semantic analysis (ESA) [2], automatic topic identification [3], information extraction [4], or a myriad of others. A weakness of Wikipedia has been its category structure, which was not part of the original design but added

in 2004. Various reviewers have likened Wikipedia more to a thesaurus than a classification scheme, others that it is different from classical knowledge organization systems in that it has no specified root or hierarchy. This situation improved a wee bit from 2006 to 2010, when editors organized the main Wikipedia topics according to top-level and main topics.[3] Still, typical commentaries point to the fact that Wikipedia's category structure is "noisy, ill-formed, and difficult to make sense of" [4]. Its crowdsourced nature has led to various direct and indirect cycles in portions of the category structure. All of these problems lead to the inability to do traditional reasoning or inference over the Wikipedia category graph [5]. We have done much to clean the Wikipedia categories and remap them and their instances to KBpedia, which have now made the structure computable.

The choice of Wikipedia's founders to make its full content available electronically for free and without restriction was a masterstroke, and now carries over to Wikidata, a sister project to Wikipedia under the Wikimedia banner. I only hope we can honor this philosophy. Wikidata takes as its starting point the structured data about entities evident in Wikipedia infoboxes. Rather than extracting and cleaning that entity information as DBpedia does, the roles of Wikidata are as a stand-alone reference base and as a multilingual source for all entities feeding the Wikimedia network, including Wikipedia. As of June 2018, Wikidata contained about 50 million data items in its system. The Wikidata approach leads to more uniformity and consistency and provides a central Wikimedia access point for structured data [6]. However, somewhat akin to Wikipedia, Wikidata also has struggled to find an appropriate typology (or ontology) for its millions of entities.[4] Again, KBpedia provides one such structure.

Besides these main six KBs, KBpedia has extended mappings to a further 20 other vocabularies, including schema.org, Dublin Core, the Bibliographic Ontology (BIBO), and others. KBpedia also supports exports in various formats and finite-state transducers or specialty lists, used as inputs to third-party analysis, management, and visualization tools. We also transform external and domain data into KBpedia's internal canonical forms for interacting with the overall structure.

The KBpedia Typologies

A *typology* is a grouping of similar types, sharing some essential characters. Each type is a parent to a particular group of instances, which also share essential traits or attributes. Chapter 10 is devoted to a discussion of KBpedia's typologies and the advantages of their modular, expandable design. *Table 10.2* provides a listing of the current typologies; *Table 10.3* describes those that are core.

[3] For Wikipedia's main topics, see http://en.wikipedia.org/wiki/Category:Main_topic_classificationsReference; for Wikipedia's top-level categories, see http://en.wikipedia.org/wiki/Category:Fundamental_categories

[4] As of May 2018, Wikidata contained more than 54 million entities. However, there has yet to emerge an overarching typology or ontology for these entities, with the typing system that does exist growing from the bottom up. For some background, see https://www.wikidata.org/wiki/Wikidata:Requests_for_comment/Migrating_away_from_GND_main_type

Structure

This section goes into a bit more detail on the structure of the KBpedia knowledge graph, KKO. At each level in the KBpedia Knowledge Ontology, we strive to organize each entry according to Peirce's universal categories of Firstness (1ns), Secondness (2ns), and Thirdness (3ns). Most of the reference concepts (RCs) in KKO are organized under the Generals (3ns) branch, though that is not evident from inspection of the upper nodes alone. All of KKO's SuperTypes (typologies) reside there.

Most of the 30 or so core typologies in KBpedia do not overlap with one another, what is known as disjoint. Disjointness enables us to perform powerful reasoning and subset selection (filtering) on the KKO graph. Upper typologies are useful to organize core entities, plus they provide homes for shared concepts. Living Things, for example, can capture concepts shared by all plants and animals, by all life, which then enables better segregation of those life forms. We apply such natural segregations across the KKO structure. Here is the upper structure of KKO with its 171 concepts (Table B.1):

Table B.1 The KKO upper structure organized by the universal categories

Level 1	Level 2	Level 3	Level 4	Level 5	Level 6	Level 7
Monads [1ns]						
	FirstMonads [1ns]					
		Suchness [1ns]				
			Accidental [1ns]			
			Inherent [2ns]			
			Relational [3ns]			
		Thisness [2ns]				
			Chance [1ns]			
			Being [2ns]			
			Form [3ns]			
		Pluralness [3ns]				
			Absolute [1ns]			
				Inclusive [1ns]		
				Exclusive [2ns]		
				Difference [3ns]		
			SimpleRelative [2ns]			
			Conjugative [3ns]			
	DyadicMonads [2ns]					
		Attributives [1ns]				
			Oneness [1ns]			
				Identity [1ns]		
				Real [2ns]		
					Matter [1ns]	
					SubstantialForm [2ns]	
					AccidentalForm [3ns]	

(continued)

Table B.1 (continued)

Level 1	Level 2	Level 3	Level 4	Level 5	Level 6	Level 7
				Fictional [3ns]		
			Otherness [2ns]			
			Inherence [3ns]			
				Quality [1ns]		
				Negation [2ns]		
				Intrinsic [3ns]		
		Relatives [2ns]				
			Concurrents [1ns]			
			Opponents [2ns]			
			Conjunctives [3ns]			
				Quantity [1ns]		
					Values [1ns]	
						Numbers [1ns]
						Multitudes [2ns]
						Magnitudes [3ns]
					Discrete [2ns]	
					Complex [3ns]	
				Subsumption [2ns]		
				Connective [3ns]		
					Unary [1ns]	
					Binary [2ns]	
					Conditional [3ns]	
		Indicatives [3ns]				
			Iconic [1ns]			
			Indexical [2ns]			
			Associative [3ns]			
				Denotative [1ns]		
				Similarity [2ns]		
				Contiguity [3ns]		
	TriadicMonads [3ns]					
		Representation [1ns]				
			Icon [1ns]			
			Index [2ns]			
			Symbol [3ns]			
		Mediation [2ns]				
		Mentation [3ns]				
Particulars [2ns]						
	MonadicDyads [1ns]					
		MonoidalDyad [1ns]				
		EssentialDyad [2ns]				
		InherentialDyad [3ns]				
	Events [2ns]					
		Spontaneous [1ns]				

(continued)

Table B.1 (continued)

Level 1	Level 2	Level 3	Level 4	Level 5	Level 6	Level 7
		Action [2ns]				
			Exertion [1ns]			
			Perception [2ns]			
			Thought [3ns]			
		Continuous [3ns]				
			TriadicAction [1ns]			
			Activities [2ns]			
			Processes [3ns]			
	Entities [3ns]					
		SingleEntities [1ns]				
			Phenomenal [1ns]			
			States [2ns]			
				Situations		
			Continuants [3ns]			
				Space		
					Points [1ns]	
					Areas [2ns]	
						2D dimensions
					SpaceRegions [3ns]	
						3D dimensions
				Time		
					Instants [1ns]	
					Intervals [2ns]	
					Eternals [3ns]	
		PartOfEntities [2ns]				
			Members [1ns]			
			Parts [2ns]			
			FunctionalComponents [3ns]			
		ComplexEntities [3ns]				
			CollectiveStuff [1ns]			
			MixedStuff [2ns]			
			CompoundEntities [3ns]			
Generals [3ns] (== SuperTypes)						
	Constituents [1ns]					
		NaturalPhenomena [1ns]				
		SpaceTypes [2ns]				
			Shapes [1ns]			
			Places [2ns]			
				LocationPlace		
				AreaRegion		
			Forms [3ns]			
		TimeTypes [3ns]				
			Times [1ns]			

(continued)

Table B.1 (continued)

Level 1	Level 2	Level 3	Level 4	Level 5	Level 6	Level 7
			EventTypes [2ns]			
			ActivityTypes [3ns]			
	Predications [2ns]					
		AttributeTypes [1ns]				
			IntrinsicAttributes [1ns]			
			AdjunctualAttributes [2ns]			
			ContextualAttributes [3ns]			
		RelationTypes [2ns]				
			DirectRelations [1ns]			
			CopulativeRelations [2ns]			
				ActionTypes		
			MediativeRelations [3ns]			
				SituationTypes		
		RepresentationTypes [3ns]				
			Denotatives [1ns]			
			Indexes [2ns]			
			Associatives [3ns]			
	Manifestations [3ns]					
		NaturalMatter [1ns]				
			AtomsElements [1ns]			
			NaturalSubstances [2ns]			
			Chemistry [3ns]			
		OrganicMatter [2ns]				
			OrganicChemistry [1ns]			
				BiologicalProcesses		
			LivingThings [2ns]			
				Prokaryotes [1ns]		
				Eukaryotes [2ns]		
					ProtistsFungus [1ns]	
					Plants [2ns]	
					Animals [3ns]	
				Diseases [3ns]		
			Agents [3ns]			
				Persons [1ns]		
				Organizations [2ns]		
				Geopolitical [3ns]		
		Symbolic [3ns]				
			Information [1ns]			
				AVInfo [1ns]		
					VisualInfo	
					AudioInfo	
				WrittenInfo [2ns]		
				StructuredInfo [3ns]		

(continued)

Table B.1 (continued)

Level 1	Level 2	Level 3	Level 4	Level 5	Level 6	Level 7
			Artifacts [2ns]			
				FoodDrink		
				Drugs		
				Products		
				Facilities		
			Systems [3ns]			
				ConceptualSystems [1ns]		
					Concepts [1ns]	
					TopicsCategories [2ns]	
					LearningProcesses [3ns]	
				SocialSystems [2ns]		
					FinanceEconomy	
					Society	
				Methodeutic [3ns]		
					InquiryMethods [1ns]	
					KnowledgeDomains [2ns]	
					EmergentKnowledge [3ns]	

Note that Table 10.2 in the main book provides an expansion on the typologies found under the Generals branch in the table above.

Capabilities and Uses

Online demos, various search and discovery facilities, use cases, and documentation on the KBpedia Web site provide further details about KBpedia. The primary purpose of KBpedia is to serve as a starting template for creating local domain knowledge graphs. However, as is, KBpedia also has the following capabilities:

- A consistent, coherent combination of six (6) large and leading public knowledge bases into the computable KBpedia knowledge structure
- Mappings to a further 20 'extended' knowledge bases
- A structured organization of the contributing knowledge sources that enables separate treatment of concepts, entities, events, attributes, and relations and their associated types
- Powerful and flexible manipulation and filtering capabilities
- Robust and configurable search and retrieval functions
- Pre-built taggers, classifiers, and mappers
- Ingest and export of multiple data formats
- All functions available via microservice-like APIs and Web services
- Use of open and accepted languages and standards

- A modular and expandable architecture
- A completely Web-based system, which we may deploy locally or in the cloud
- Integration and incorporation of all data assets—unstructured text, semi-structured and markup data, and structured datasets and databases
- A reference structure for interrelating and integrating your domain content
- Inherent multi-linguality, supported by the 200+ languages of the source knowledge bases
- Precise semantic representation for all items, enabling better selections and matches
- The ability to make selections via inference and other logical operations
- The potential to recognize and train up to 47,000 fine-grained entity types
- A knowledge graph suitable for network analytics such as influence, centrality, shortest paths, assortative mixing, and betweenness
- Faster, cheaper creation of positive and negative machine learning training sets
- Faster, cheaper configuration and testing of machine learners.

You may download the open-source KBpedia and other supporting materials and documentation from the project's GitHub site.

References

1. M.K. Bergman, Cognonto is on the hunt for big AI Game, in *AI3:::Adaptive Information* (2016)
2. E. Gabrilovich, S. Markovitch, Computing semantic relatedness using wikipedia-based explicit semantic analysis, in *Proceedings of the 20th International Joint Conference on Artificial Intelligence* (*IJCAI*) (Hyderabad, India, 2007)
3. M. Hassan, *Automatic Document Topic Identification Using Hierarchical Ontology Extracted from Human Background Knowledge* (2013)
4. F. Wu, D.S. Weld, Automatically refining the wikipedia infobox ontology, in *Proceedings of the 17th International Conference on World Wide Web* (ACM, 2008), pp. 635–644
5. A. Kittur, E.H. Chi, B. Suh, What's in wikipedia? Mapping topics and conflict using socially annotated category structure, in *Proceedings of the 27th Annual CHI Conference on Human Factors in Computing Systems (CHI'2009)* (New York, USA, 2009), pp. 1509–1512
6. H. Paulheim, C. Bizer, Type inference on noisy RDF data, in *International Semantic Web Conference* (Springer, Heidelberg, 2013), pp. 510–525
7. D. Vrandečić, M. Krötzsch, Wikidata: A Free Collaborative Knowledgebase. Commun. ACM **57**, 78–85 (2014)

Appendix C: KBpedia Feature Possibilities

The two most labor-intensive steps in machine learning for natural language are (1) feature engineering and (2) labeling of training sets. Supervised machine learning uses an input-output pair, mapping an input, which is a feature, to an output, which is the label. The machine learning consists of inferring ('learning') a function that maps between these inputs and outputs with adequate predictive power. We can apply this learned function to previously unseen inputs to predict the output label. The technique is particularly suited to problems of regression or of classification. Yet, despite the integral role of *features* in the machine learning process, we often overlook their importance compared to labels and algorithms.

Before we can understand how best to leverage features in our *knowledge-based artificial intelligence* (KBAI) efforts, we need first to define and name the feature space. Separately, we also need to study what exists on how to select, construct, extract, or engineer these features. Armed with this background, we can now assemble an inventory of what features might contribute to natural language or knowledge base learning.

We have followed these steps to produce a listing of possible KBpedia features.[1] We have organized that inventory a bit to point out the structural and conceptual relationships among these features, which enables us to provide a lightweight taxonomy for the space. Since others have not named or exposed many of these features before, we conclude this appendix with some discussion about what next-generation learners may gain by working against this structure. Of course, since much of this thinking is incipient, forks and dead ends may unfold, but there also will likely be unforeseen expansions and opportunities as well. A systematic view of machine learning and its knowledge and human language features—coupled with large-scale knowledge bases such as Wikipedia and Wikidata—can lead to faster and cheaper learners across a comprehensive range of NLP tasks.

[1] Features apply to any form of machine learning, including for things like image, speech, and pattern recognition. However, this chapter is limited to the context of natural language, unstructured data, and knowledge bases.

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8

What Is a Feature?

A "feature is an individual measurable property of a phenomenon being observed" [1]. It is an input to a machine learner, an explanatory variable, sometimes in the form of a function. Some equate features with attributes, but this is not strictly accurate, since a feature may be a combination of other features, or a statistical calculation, or an abstraction of other inputs (some would say it could be about anything!). In any case, we must express a feature as a numeric value (including Boolean as 0 and 1) upon which the machine learner can calculate its predictions. Machine learner predictions of the output can only be based on these numeric features, though they can be subject to rules and weights depending on the type of learner.

Pedro Domingos emphasizes the importance of features and the fact they may be extracted or constructed from other inputs [2]:

> At the end of the day, some machine learning projects succeed and some fail. What makes the difference? Easily the most important factor is the features used… Often, the raw data is not in a form that is amenable to learning, but you can construct features from it that are. This is typically where most of the effort in a machine learning project goes. It is often also one of the most interesting parts, where intuition, creativity and 'black art' are as important as the technical stuff.

Many experienced ML researchers make a similar reference to the art or black art of features. In broad strokes in the context of natural language, a feature may be a surface form, like terms or syntax or structure (such as hierarchy or connections); derived (such as statistical, frequency, weighted, or based on the ML model used); semantic (in terms of meanings or relations); or latent, either as something hidden or abstracted from feature layers below it. Unsupervised learning or deep learning features arise from the latent form.

For a given NLP problem domain, features can number into the millions or more. Concept classification, for example, could use features corresponding to all of the unique words or phrases in that domain. Relations between concepts could also be as numerous. We calculate some form of vector relationship over, say, all of the terms in the space so that we may assign a numerical value to 'high-dimensional' features.[2] Because learners may learn about multiple feature types, the potential combinations for the ML learner can be astronomical. This combinatorial problem has been known for decades and has been termed the curse of dimensionality for more than 50 years [3].

Of course, just because a feature exists says nothing about whether it is useful for ML predictions. Features may thus be one of the four kinds: (1) strongly relevant, (2) weakly relevant, (3) irrelevant, or (4) redundant [10]. We should favor strongly relevant features; we may sometimes combine weakly relevant to improve the overall relevancy. We should remove all irrelevant or redundant features from consideration. Often, the fewer the features, the better, so long as the features used are strongly relevant and orthogonal (that is, they capture different aspects of the prediction space) to one another.

[2] For example, in the term or phrase space, the vectors might be constructed from counts, frequencies, cosine relationships between representative documents, distance functions between terms, etc.

A (Partial) Inventory of Natural Language and KB Features

To make this discussion tangible, we have assembled a taxonomy of feature types in the context of natural language and knowledge bases. I drew this inventory from the limited literature on feature engineering and selection in the context of KBAI from the perspectives of ML learning in general [4–6], ML learning ontologies [7, 8], and knowledge bases [9–13]. My listing is only partial, but does provide an inventory of more than 200 feature types applicable to natural language.

I have organized this inventory into eight main areas in Table C.1, shown in *italics*, which tend to cluster into these four groupings:

- ***Surface features**—These are features that one can see within the source documents and knowledge bases. They include**lexical**items for the terms and phrases in the domain corpus and knowledge base;**syntactical**items that show the word order or syntax of the domain;**structural**items that either split the documents and corpus into parts or reflect connections and organizations of the items, such as hierarchies and graphs; or**natural language**items that reflect how we express the content in the surface forms of various human languages.*
- ***Derived features**—These are surface features that we transform or derive in some manner, such as the direct**statistical**items or the**model-based**ones reflecting the characteristics of the machine learners used.*
- ***Semantic features**—These are summarized in the**semantics**area, and reflect what the various items mean or how they are conceptually related to one another.*
- ***Latent features**—These features are not observable from the source content. Instead, these are statistically derived abstractions of the features above that are one- to N-levels removed from the initial source features. These**latent**items may either be individual features or entire layers of abstraction removed from the surface layer. These features result from applying unsupervised or deep learning machine learners.*

We may nucleate features and training sets based on the syntax, morphology, semantics (meaning of the data), or relationships (connections) of the source data in the knowledge base. Continuous testing and application of machine learning to the system itself create virtuous feedback where the accuracy of the overall system is constantly and incrementally improved.

The compiled taxonomy listing of features in Table C.1 exceeds any prior listings. In fact, most of the feature types we show have yet to participate in NLP machine learning tasks. We organize our taxonomy according to the same eight main areas, shown under the shaded entries, noted above:

Fully 50% of the features listed in the inventory in Table C.1 above arise from unique KB aspects, especially in the areas of ***semantics*** and ***structural***, including graph relationships. Many, if not most, of these new feature possibilities may prove redundant or only somewhat relevant or perhaps not at all. Not all features may ever prove useful. Some, such as *Case*, may be effectively employed for named entity or specialty extractions, applicable to copyrights or unique IDs or data types, but may prove of little use in other areas.

Still, because many of these KB features cover orthogonal aspects of the source knowledge bases, the likelihood of finding new, strongly relevant features is high. Further, except for the ***latent*** and ***model-based*** areas, each of these feature types may be used singly or in combination to create coherent slices for both positive and negative training sets, helping to reduce the effort for labor-intensive labeling as well. By extension, we can use these capabilities to more effectively bootstrap the creation of gold standards, useful when we are testing parameters.

The *Statistical* and *Meta-features* sections of Table C.1 are first derivatives of the base structure. The few listed here are examples of how we may include such measures in the feature pool, and they all are common ones. The point is that we may use derivatives and embeddings from other features in the table as legitimate features in their own right.

Table C.1 A (partial) taxonomy of machine learning features

Lexical				
	Corpus			
	Phrases			
		Averages		
		Counts		
		N-grams		
		Weights		
	Words			
		Averages		
		Counts		
		Cutoffs (top *N*)		
		Dictionaries		
		Named entities		
		Stemming		
		Stoplists		
		Terms		
		Weights		
Syntactical				
	Anaphora			
	Cases			
	Complements (argument)			
	Co-references			
	Decorations			
	Dependency grammar			
		Head (linguistic)		

(continued)

Table C.1 (continued)

	Distances			
	Gender			
	Moods			
	Paragraphs			
	Parts of speech (POS)			
	Patterns			
	Plurality			
	Phrases			
	Sentences			
	Tenses			
	Word order			
Statistical				
	Articles			
		Vectors		
	Information-theoretic			
		Entropy		
		Mutual information		
	Meta-features			
		Correlations		
		Eigenvalues		
		Kurtosis		
		Sample measures		
			Accuracy	
			F-1	
				Precision
				Relevance
		Skewness		
		Vectors		
		Weights		
	Phrases			
		Document frequencies		
		Frequencies (corpus)		
		Ranks		
		Vectors		
	Words			
		Document frequencies		
		Frequencies (corpus)		
		Ranks		
		String similarity		
		Vectors		
			Cosine measures	
			Feature vectors	

(continued)

Table C.1 (continued)

Structural				
	Documents			
		Node types		
			Depth	
			Leaf	
	Document parts			
		Abstract		
		Authors		
		Body		
		Captions		
		Dates		
		Headers		
		Images		
		Infoboxes		
		Links		
		Lists		
		Metadata		
		Templates		
		Title		
		Topics		
	Captions			
	Disambiguation pages			
	Discussion pages			
		Authors		
		Body		
		Dates		
		Links		
		Topics		
	Formats			
	Graphs (and ontologies)			
		Acyclic		
		Concepts		
			Centrality	
			Relatedness	
		Directed		
		Metrics (counts, averages, min/max)		
			Attributes	
			Axioms	
			Children	
			Classes	
			Depth	
			Individuals	
			Parents	
		Subgraphs		

(continued)

Table C.1 (continued)

	Headers			
		Content		
		Section hierarchy		
	Infoboxes			
		Attributes		
		Missing attributes		
		Missing values		
		Templates		
		Values		
	Language versions			
		Definitions		
		Entities		
		Labels		
		Links		
		Synsets		
	Links			
		Category		
		Incoming		
		Linked data		
		Outgoing		
		See also		
	Lists			
		Ordered		
		Unordered		
	Media			
		Audio		
		Images		
		Video		
	Metadata			
		Authorship		
		Dates		
		Descriptions		
		Formats		
		Provenance		
	Pagination			
	Patterns			
		Dependency patterns		
		Surface patterns		
			Regular expressions	
	Revisions			
		Authorship		
		Dates		
		Structure		

(continued)

Table C.1 (continued)

			Document parts	
				Captions
				Headers
				Infoboxes
				Links
				Lists
				Metadata
				Templates
				Titles
		Versions		
	Source forms			
		Advertisements		
		Blog posts		
		Documents		
			Research articles	
			Technical documents	
		E-mails		
		Microblogs (tweets)		
		News		
		Technical		
		Web pages		
	Templates			
	Titles			
	Trees			
		Breadth measures		
		Counts		
		Depth measures		
	Web pages			
		Advertisements		
		Body		
		Footer		
		Header		
		Images		
		Lists		
		Menus		
		Metadata		
		Tables		
Semantics	(most also subject to *syntactical* and *statistical* features above)			
	Annotations			
		Alternative labels		
		Notes		
		Preferred labels		
	Associations			
		Association rules		

(continued)

Table C.1 (continued)

		Co-occurrences		
		See also		
	Attribute types			
		Attributes		
			Cardinality	
			Descriptive	
			Qualifiers	
			Quantifiers	
				Many
		Values		
			Data types	
			Many	
	Categories			
		Eponymous pages		
	Concepts			
		Definitions		
		Grouped concepts (topics)		
		Hypernyms		
			Hypernym-based feature vectors	
		Hyponyms		
		Meanings		
		Synsets		
			Acronyms	
			Epithets	
			Jargon	
			Misspellings	
			Nicknames	
			Pseudonyms	
			Redirects	
			Synonyms	
	Entity types			
		Entities		
		Events		
		Locations		
	General semantic feature vectors			
	Relation types			
		Binary		
		Identity		
		Logical conjunctions		
			Conjunctive	
			Disjunctive	
		Mereology (part of)		
		Relations		
			Domain	
			Range	

(continued)

Table C.1 (continued)

		Similarity		
	Roles			
	Voice			
		Active/passive		
		Gender		
		Mood		
		Sentiment		
		Style		
		Viewpoint (worldview)		
Natural languages				
	Morphology			
	Nouns			
	Syntax			
	Verbs			
	Word order			
Latent				
	Autoencoders			
		Many; dependent on method		
	Features			
		Many; dependent on method		
	Hidden			
		Many; dependent on method		
	Kernels			
		Many; dependent on method		
Model based				
	Decision tree			
		Tree measures		
	Dimensionality			
	Feature characteristics			
		Data types		
		Max		
		Mean		
		Min		
		Number		
		Outliers		
		Standard deviation		
	Functions			
		Factor graphs		
		Functors		
		Mappings		
	Landmarking			
		Learner accuracy		
	Method measures			
		Error rates		

Though the literature most often points to classification as the primary use of knowledge bases as background knowledge supporting machine learners, in fact, many natural language processing (NLP) tasks may leverage KBs. Here is but a brief listing of application areas for KBAI:

See also Table 4.1. Undoubtedly other applications will emerge as this more systematic KBAI approach to machine learning evolves over the coming years.

Feature Engineering for Practical Limits

This richness of feature types leads to the combinatorial problem of too many features. Feature engineering is the way both to help find the features of strongest relevance and to reduce the feature space dimensionality to speed the ML learning times. Initial feature engineering tasks should be to transform input data, regularize them if need be, and create numeric vectors for new ones. These are preparation tasks to convert the source or target data to forms amenable to machine learning. This staging now enables us to discover the most relevant ('strong') features for the given ML method under investigation.

In a KB context, specific learning tasks as outlined in Table C.2 are often highly patterned. The most effective features for training, say, an entity recognizer, will only involve a limited number of strongly relevant feature types. Moreover, the relevant feature types applicable to a given entity type should mostly apply to other

Table C.2 NLP applications for machine learners using KBs

• Entity recognizers	• Ontology development
• Relation extractors	• Entity dictionaries
• Classifiers	• Data conversion and mapping
• Q & A systems	• Master data management
• Knowledge base mappings	• Specialty extractors

entity types, even though the specific weights and individual features (attributes and other type relations) will differ. This patterned aspect means that once we train a given ML learner for a given entity type, its relevant feature types should be approximately applicable to other related entity types. We can reduce the lengthy process of initial feature selection as training proceeds for similar types. It appears that we may discover combinations of feature types, specific ML learners, and methods to create training sets and gold standards for entire classes of learning tasks.

Probably the most time-consuming and demanding aspect of these patterned approaches resides in *feature selection* and *feature extraction*. *Feature selection* is the process of finding a subset of the available feature types that provide the highest predictive value while not overfitting.[3] Researchers typically split feature selection into three main approaches [14, 15, 16]:

[3] Overfitting is where a statistical model, such as a machine learner, describes random error or noise instead of the underlying relationship. It is particularly a problem in high-dimensional spaces, a common outcome of employing too many features.

- Filter—Select the N most promising features based on a ranking from some form of proxy measure, like *mutual information* or the *Pearson correlation coefficient*, which provides a measure of the information gain from using a given feature type.
- Wrapper—Test feature subsets through a greedy search heuristic that either starts with an empty set and adds features (forward selection) keeping the "strongest" ones or starts with a full set and gradually removes the "weakest" ones (backward selection); the wrapper approach may be computationally expensive.
- Embedded—Include feature selection as a part of model construction.

For high-dimensional features, such as terms and term vectors, we may apply stoplists or cutoffs (only considering the top *N* most frequent terms, for example) to reduce dimensionality. Part of the 'art' portion resides in knowing which feature candidates may warrant formal selection or not; this learning can be codified and reused for similar applications. One may also apply some unsupervised learning tests at this point to discover additional 'strong' features.

Feature extraction transforms the data in the high-dimensional space to a space of fewer dimensions. Functions create new features in the form of ***latent*** variables, which are not directly observable. Also, because these are statistically derived values, many input features are reduced to the synthetic measure, which naturally causes a reduction in dimensionality. Advantages of a reduction in dimensionality include:

1. Often a better feature set (resulting in better predictions) [17]
2. Faster computation and smaller storage
3. Reduction in collinearity due to a reduction in weakly interacting inputs
4. Easier graphing and visualization.

On the other hand, the latent features are abstractions, and so not easily understood as the literal. Deep learning generates multiple layers of these latent features as the system learns.

Of course, we may also combine the predictions from multiple ML methods, which then also raises the questions of ensemble scoring. We may also self-learn (that is, meta-learn) more systematic approaches to ML such that the overall learning process can proceed in a more automated way.

Considerations for a Feature Science

In supervised learning, it is clear that more time and attention have been given to the labeling of the data, what the desired output of the model should be. Much less time and attention have been devoted to features, the input side of the equation. The purposeful use of knowledge bases and structuring them is one way we can make progress. Still, progress also requires some answers to some fundamental questions. A scientific approach to the feature space would likely need to consider, among other objectives:

- Full understanding of surface, derived, and latent features
- Relating various use cases and problems to specific machine learners and classes of learners
- Relating specific machine learners to the usefulness of particular features (see also *hyperparameter optimization* and *model selection*)
- Improved methods for feature engineering and construction
- Improved methods for feature selection
- A better understanding of how to select supervised and unsupervised ML.

Some tools and utilities would also help to promote this progress. Some of these capabilities include:

- Feature inventories—how to create and document taxonomies of feature types
- Feature generation—methods for the codification of leading recipes
- Feature transformations—the same for transformations, up to and including vector creation.

Role of a Platform

The object of these efforts is to systematize how knowledge bases, combined with machine learners, can speed the deployment and lower the cost of creating bespoke artificial intelligence applications of natural language for specific domains. KBAI places primary importance on *features*. An abundance of opportunity exists in this area, and an abundance of work required, but little systematization.

The good news is we can build platforms that manage and grow the knowledge bases and knowledge graphs supporting machine learning, as we discussed in *Parts III* and *IV*. We can apply machine learners in a pipeline manner to these KBs, including orchestrating the data flows in generating and testing features, running and testing learners, creating positive and negative training sets, and establishing gold standards. The heart of the platform must be an appropriately structured knowledge base organized according to a coherent knowledge graph; this is the primary purpose of KBpedia.

Still, in the real world, engagements always demand unique scope and unique use cases. We should engineer our platforms to enable ready access, extensions, configurations, and learners. It is vital to structure our source knowledge bases such that slices and modules can be specified, and all surface attributes may be selected and queried. Mapping to the external schema is also essential. Background knowledge from a coherent knowledge base is the most efficient way to fuel this.

References

1. C. Bishop, *Pattern Recognition and Machine Learning* (Springer, Berlin, 2006)
2. P. Domingos, A Few Useful Things to Know About Machine Learning. Commun. ACM **55**, 78–87 (2012)
3. R.E. Bellman, *Dynamic Programming* (Princeton University Press, Rand Corporation, 1957)
4. I. Guyon, A. Elisseeff, An introduction to feature extraction, in *Feature Extraction* (2006), pp. 1–25
5. D. Haussler, *Convolution Kernels on Discrete Structures* (Technical report, Department of Computer Science, University of California at Santa Cruz, Irvine, CA, 1999)
6. M. Reif, F. Shafait, M. Goldstein, T. Breuel, A. Dengel, Automatic Classifier Selection for Non-Experts. Pattern. Anal. Applic. **17**, 83–96 (2014)
7. J. Tang, S. Alelyani, H. Liu, Feature selection for classification: a review, in *Data Classification: Algorithms and Applications* (2014), p. 37
8. M. Hilario, P. Nguyen, H. Do, A. Woznica, A. Kalousis, Ontology-based meta-mining of knowledge discovery workflows, in *Meta-Learning in Computational Intelligence* (Springer Berlin, Heidelberg, 2011), pp. 273–31
9. P. Panov, L. Soldatova, S. Džeroski, Ontology of Core Data Mining Entities. Data Min. Knowl. Disc. **28**, 1222–1265 (2014)
10. I. Anastacio, B. Martins, P. Calado, Supervised learning for linking named entities to knowledge base entries, in *Proceedings of the Text Analysis Conference (TAC2011)* (2011)
11. W. Cheng, G. Kasneci, T. Graepel, D. Stern, R. Herbrich, Automated feature generation from structured knowledge, in *Proceedings of the 20th ACM International Conference on Information and Knowledge Management* (ACM, 2011), pp. 1395–1404
12. L. Huang, D. Milne, E. Frank, I.H. Witten, Learning a Concept-Based Document Similarity Measure. J. Assoc. Inf. Sci. Technol. **63**, 1593–1608 (2012)
13. O. Medelyan, D. Milne, C. Legg, I.H. Witten, Mining Meaning from Wikipedia. Int. J. Hum. Comput. Stud. **67**, 716–754 (2009)
14. H. Shen, M. Chen, R. Bunescu, R. Mihalcea, Wikipedia Taxonomic Relation Extraction using Wikipedia Distant Supervision. Ann. Arbor. **1001**, 48109 (2014)
15. G.H. John, R. Kohavi, K. Pfleger, Irrelevant features and the subset selection problem, in *Machine Learning: Proceedings of the Eleventh International Conference* (1994), pp. 121–129
16. Z. Žabokrtský, *Feature Engineering in Machine Learning* (2015)
17. G.E. Hinton, Deep Belief Networks. *Scholarpedia***4**, 5947 (2009)

Glossary

ABox An ABox (for assertions, the basis for A in ABox) is an 'assertion component'; that is, a fact associated with a terminological vocabulary within a knowledge base. ABox are *TBox*-compliant statements about instances belonging to the concept of an ontology. Instances and instance records reside within the Abox.

Abductive reasoning Abduction (or abductive reasoning) is a mode of symbolic inference that involves the screening and selection from a domain D of the possible explanation paths to an outcome O possibly involving any element E of D, with the selection of candidate paths for inductive testing based on plausibility, economy, and potential impact; abduction does not produce probable results, only winnowed candidates.

Access control Access control is the protection of resources against unauthorized access, a process by which use of resources is regulated according to a security policy and is permitted by only authorized system entities according to that policy; see further RFC 2828.

Actions Actions are reactions to perceptions or stimuli, are energetic, or are thought, as understood to be broadly construed; actions reside in *Secondness*.

Activities Activities are sustained actions over durations of time; activities may be organized into natural classes.

Accuracy It is a statistical measure of how well a binary classification test correctly identifies or excludes a condition. It is calculated as the sum of *true positives* and *true negatives* divided by the total population.

Adaptive ontology An adaptive ontology is a conventional knowledge representational ontology that has added to it a number of specific best practices, including modeling the *ABox* and *TBox* constructs separately; information that relates specific types to different and appropriate display templates or visualization components; use of *preferred labels* for user interfaces, as well as alternative labels and hidden labels; defined concepts; and a design that adheres to the *open-world assumption*

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8

Administrative ontology Administrative ontologies govern internal application use and user interface interactions.

Annotation Annotations are indexes and the *metadata* of the *KB*; these cannot be inferenced over and do not participate in *reasoning* or *coherency* testing. But, they can be searched, and language *features* can be processed in other ways. Annotations may be grouped for convenience, but may not be typed.

API An application programming interface (API) defines how communication may take place between applications. Implementing APIs that are independent of a particular operating environment (as are the W3C DOM Level 2 specifications) may reduce implementation costs for multi-platform user agents and promote the development of multi-platform assistive technologies.

Architectonic Architectonic is a governing philosophy expressed iconically as a design, schema, or structure.

Architecture Architecture, as limited to the use herein, is the structure of a knowledge base (or bases) written in a knowledge representation formalism, and embedded in a general knowledge management platform. The architecture combines knowledge artifacts with software program(s) or computing system(s), the relationships among them, and the conditions on their use.

Artificial intelligence AI is the use of computers to do or assist complex human tasks or *reasoning*. There are many broad subfields from pattern recognition to robotics and complex planning and optimizations.

Aspects Aspects are aggregations of an *entity type* that are grouped according to *features* or views different from the type itself. As examples, the type of 'music composer' may have an aspect of being from the nineteenth century, or 'authors' may have the aspects of being Russian or writing novellas. The organization of aspects closely parallels that for *SuperTypes*.

Assertion Assertion is a *statement*, wherein a fact or logical expression with consequences is made.

Attributes *Attributes* are the *intensional* characteristics of an *object*, *event*, *entity*, *type* (when viewed as an *instance*), or *concept*. The relationship is between the individual particular and its attributes and characteristics, in the form of A:A. Attributes may be intrinsic characteristics or essences of single particulars, adjunctual or accidental happenings to the particular, or contextual in time or space or situations. Collectively known as depth, comprehension, significance, meaning, or connotation.

Attribute type Attribute types are an aggregation (or class) of multiple *attributes* that have similar characteristics among themselves (for example, colors or ranks or metrics). As with other types, attribute types do not have attributes.

Axiom An axiom is a premise or starting point of *reasoning*. In an *ontology*, each statement (*assertion*) is an axiom.

Base concept Base concepts are all of the *classes* in the overall KBpedia, comprised of the KBpedia Knowledge Ontology and all of its official *typologies*.

Belief Belief is a state of evidence sufficient to enable action.

Binding Binding is the creation of a simple reference to something that is larger and more complicated and used frequently. The simple reference can be used instead of having to repeat the larger thing.

Blank node Also called a bnode, a blank node in *RDF* is a *resource* for which a *URI* or literal is not given. A blank node indicates the existence of a thing, implied by the structure of the *knowledge graph*, but which was never explicitly identified by giving it a URI. Blank nodes have no meaning outside of their current graph and therefore cannot be mapped to other resources or graphs.

Cardinality Cardinality is where the number of members in a class or type is set or limited, such as `hasBiologicalParent` being set to two.

Class Class is a collection of one or more *instances* or *classes* that share the same potential *attributes* or *relations*; *concepts* and *entity types* are both classes.

Closed-world assumption CWA is the premise that what is not currently known to be true is false. CWA is the most common logic applied to relational database systems, useful for transaction-type systems. In knowledge management, CWA is used in at least two situations: (1) when the *knowledge base* is known to be complete, and (2) when the knowledge base is known to be incomplete, but a 'best' definite answer must be derived from incomplete information. See contrast to the *open-world assumption*.

Coherence Coherence is the state of being coherent for logic systems, where the knowledge base (domain) is consistent and has a high degree of conjunction for nondeductive assertions.

Collection See *class*.

Complete Complete is an evaluative criterion for logic systems where all *statements* which are true in all models are provable.

Concept See *class*.

Cyc Cyc is a common-sense *knowledge base* that has been under development for over 20 years backed by 1000 person-years of effort. The smaller OpenCyc version is available in OWL as open source; a ResearchCyc version of the entire system is available to researchers. The Cyc platform contains its own logic language, CycL, and has many built-in functions in areas such as *natural language processing*, search, inferencing, and the like. *UMBEL* is based on a subset of Cyc.

Cycle A cycle is where, in a graph, a path from a given *node* may reach itself (such as $A \rightarrow B \rightarrow C \rightarrow A$). In a subsumption hierarchy, this is an error.

Data integration Data integration is the bringing together of data from heterogeneous and often physically distributed data sources into a single, coherent view.

Dataset Dataset is a combination of one or more *records*, transmitted as a single unit (though it may be split into parts due to size).

Data types Data types are predefined ways that attribute values may be expressed, including various literals and strings (by language), *URIs*, Booleans, numbers, date-times, etc.

DBpedia A project that extracts structured content from *Wikipedia*, and then makes that data available as *linked data*. There are millions of entities characterized by

DBpedia in this way. As such, DBpedia is one of the largest—and most central—hubs for *linked data* on the Web.

Deductive reasoning Deductive reasoning extends from premises known to be true and clear to infer new facts.

Description logics Description logics and their semantics traditionally split *concepts* and their *relationship*s from the different treatment of *instances* and their *attributes* and roles expressed as *fact assertions*. The concept split is known as the *TBox* and represents the schema or *taxonomy* of the *domain* at hand. The TBox is the structural and *intensional* component of conceptual relationships. The second split of instances is known as the *ABox* and describes the attributes of instances (and individuals), the roles between instances, and other assertions about instances regarding their class membership with the TBox concepts.

Disjoint Disjoint is where membership in one *class* specifically excludes membership in another; this is a useful *property* in that it allows large, well-designed *knowledge bases* to be 'sliced and diced' for more effective processing or analysis.

Distant supervision A method to use *knowledge bases* to label *entities* automatically in text through *machine learning*, which is then used to extract features and train a machine learning classifier. The knowledge bases provide coherent positive training examples and avoid the high cost and effort of manual labeling.

Documents Documents (or articles or *records*) may be in the form of articles or data records. Whatever the form, extractions are needed to convert source information into the *triples* useful to the *knowledge base*.

Domain Domain is the bounded scope of real-world considerations that may contribute to the *knowledge representation* or information queries at hand. Scoping the domain is one of the first activities undertaken in a new KR project.

Domain (property) Domain, as applied to a property, is a statement that declares the classes or types from which the subject of the assertion must be drawn.

Domain ontology Domain (or content) *ontologies* embody more of the traditional ontology functions such as information interoperability, inferencing, reasoning, and conceptual and knowledge capture of the applicable domain.

Edges Edges, in a graphical representation of a *knowledge graph*, are the connections or the *relations* between *subjects* and *objects*; edges are the linking *properties* in a 'triple' *statement*.

Entailment Entailment is a consequence arising from a statement deemed to be true based on some underlying logic. The logical consequence is said to be *necessary* and *formal*; necessary, because of the rules of the logic (the conclusion is the consequent of the premises); and formal because the logical form of the statements and arguments hold true without regard to the specific *instance* or content.

Entity Entities are the basic, real things in our domain of interest; they are nameable things or ideas that have an identity, are defined in some manner, can be referenced, and may be related to *types*; entities are most often the bulk of an overall *knowledge base*. An entity is an *individual object* or *instance* of a *class*, a *Secondness*; when affixed with a proper name or label it is also known as a *named entity* (thus, named entities are a subset of all entities). Entities are

described and characterized by *attributes*. Entities are connected or related to one another through *external relations* and are referred to, signified, or indexed by *representations*. An entity may be independent or separate or can be part of something else. Entities cannot be *topics* or *types*.

Entity recognition The use of *natural language processing* to identify specific *entities* in text. Often used in conjunction with named entities, where it is abbreviated NER.

Entity type *Entity types* are the aggregations or collections or *classes* of similar entities, which also share some essence; entity types have the *attributes* (but not the same values) of instances of the type.

Essence The *attribute* or set of attributes that make an entity what it fundamentally is; it is a unique or distinguishing attribute that helps define a *type*.

Event Events are nameable sequences of time, are described in some manner, can be referenced, and may be related to other time sequences or types. Events are like entities, except that they have a discrete time beginning and end. Events are a Secondness and may be typed.

Extensional The extension of a *class*, concept, idea, or sign consists of the things to which it applies, in contrast with its *intension*. For example, the extension of the word 'dog' is the set of all (past, present, and future) dogs in the world. The extension is most akin to the *attributes* or characteristics of the *instances* in a set defining its class membership.

External linkages External linkages (or *mappings*) are any of the relational properties that may be used to map external datasets and schema to KBpedia. In its base form, which can be expanded, KBpedia has mappings to more than 20 external sources.

External relations External relations are assertions (relationships) between an *object*, *event*, *entity*, *type*, or *concept* and another particular or general. An external relationship has the form of *A:B*. External relations may be simple ones of a direct relationship between two different instances; may be copulative by combining objects or asserting membership, quantity, action, or circumstance; or mediative to provide meaning, context, relevance, generalizations, or other explanations of the subject with respect to the external world. External relations are extensional.

Fact A basic *statement* or *assertion* within a *knowledge graph* or *knowledge base*.

Fallibility Fallibility is the doctrine that truth is a limit function, unknowable in the absolute, and provides the logical basis for questioning premises.

False negative An error where a test result indicates that a condition failed, while it actually was successful. That is, the test result indicates a negative when the correct result should have been positive. Also known as a false-negative error or *Type II error* in statistics. It is abbreviated FN.

False positive An error where a test result indicates that a condition was met or achieved, while it actually should have failed. That is, the test result indicates a positive when the correct result should have been negative. Also known as a false-positive error or *Type I error* in statistics. It is abbreviated FP.

Feature A feature is a measurable property of the system being analyzed, equivalent to what is known as an explanatory variable in statistics.

Feature engineering Feature engineering is a process of creating, generating, and selecting the *features* to be used in *machine learning*, based on an understanding of the underlying data and choosing features based on their likely impact on learning results and effectiveness.

Firstness Firstness is possibility, the essences of what may be, the unexpected chance occurrence.

Folksonomy A folksonomy is a user-generated set of open-ended labels called *tags* organized in some manner and used to categorize and retrieve Web content such as Web pages, photographs, and Web links.

Function Function is any algebraic or logical expression allowable by the semantics and primitives used in the KR language where an input is related to an output.

Generals Generals are the mediating, continuous, vague, and indeterminate aggregations of instances into concepts, *classes*, *types*, collections, or sets. Generals are in *Thirdness*. Generals may often be considered real, and their understanding and identification may be shared through *representations*.

GeoNames GeoNames integrates geographical data such as names of places in various languages, elevation, population, and others from various sources.

Gold standard A gold standard is a reference, benchmark test set where the results are already scored and known, with a minimum (if not zero) amount of *false positives* or *false negatives*; good gold standards also include *true-negative* results.

Identifier Identifier is a reference pointer, a sign pointing to an *object*, but not the *object* itself. Identifiers should not be confused with the naming or defining label for the *object*; in practice, it is often a unique string assigned to the object. In *RDF* and KBpedia this identifier is a *URI*

Individual Individual in RDF and OWL (indeed, commonly in description logics) is synonymous with an *instance* or *entity*; we try not to use this term because of general terminological confusion; see *instance*.

Inductive reasoning A method of reasoning where lines of possible evidence are weighed to determine probable outcomes.

Inference Inference is the act or process of deriving logical conclusions from premises known or assumed to be true. The logic within and between *statements* in an *ontology* is the basis for inferring new conclusions from it, using software applications known as inference engines or *reasoners*.

Instance Instances are individual *entities* or *events*, the ground-level components of a *knowledge base*. Instances may include concrete *objects* such as people, animals, tables, automobiles, molecules, and planets, as well as abstract instances such as numbers and words; instances are in Secondness. An instance is also known as an *individual*, with *member* and *entity* also used somewhat interchangeably.

Instance record An *instance* with one or more *attributes* also provided.

Intensional The intension of a class is what is intended as a definition of what characteristics or properties its members should have. Intension is most akin

to the *attributes* or characteristics of the *instances* in a set defining its class membership. It is therefore like the key-attribute pair aspects of an instance (or *ABox*) in an *ontology*.

Inverse Inverse is when a *property*, say, `hasParent`, can be defined as the inverse property of `hasChild`.

Key-value pair Also known as a *name-value pair* or *attribute-value pair*, a key-value pair is a fundamental, open-ended data representation. All or part of the data model may be expressed as a collection of tuples `<attribute name, value>` where each element is a key-value pair. The key is the defined *attribute*, and the value may be a reference to another object or a literal string or value. In *RDF triple* terms, the subject is implied in a key-value pair by nature of the *instance record* at hand.

Kind Used synonymously herein with *class*.

Knowledge base A knowledge base (abbreviated KB or kb) is a special kind of database for knowledge management. As used in KBpedia, a KB includes *instances* and *classes* related to each other via triple statements.

Knowledge-based artificial intelligence Knowledge-based artificial intelligence, or KBAI, is the use of large statistical or *knowledge bases* to inform *feature* selection for machine-based learning algorithms used in AI.

Knowledge graph See *ontology*.

Knowledge management Knowledge management, or KM, is the practice of creating, sharing, finding, annotating, connecting, and extending information and knowledge for a given *domain*.

Knowledge representation A field of *artificial intelligence* dedicated to representing information about the world in a form that a computer system can utilize to solve complex tasks.

Knowledge supervision A method of *machine learning* to use *knowledge bases* in a purposeful way to create features, and negative and positive *training sets* in order to train the classifiers or extractors. *Distant supervision* also uses knowledge bases, but not in such a purposeful, directed manner across multiple machine learning problems.

Leaf nodes Leaf nodes are terminal nodes in a tree structure, often representing instances (but not always so).

Linkage A specification that relates an *object* or *attribute* name to its full *URI* (as required in the *RDF* language).

Linked data Linked data is a set of best practices for publishing and deploying *instance* and *class* data using the *RDF* data model, and uses *uniform resource identifiers* (URIs) to name the data objects. The approach exposes the data for access via the HTTP protocol while emphasizing data interconnections, interrelationships, and context useful to both humans and machine agents.

Lists Lists are unordered *members* or *instances*, with or without gaps or duplicates, useful for bulk assignment purposes. Lists generally occur through a direct relation assignment (*e.g.*, `rdf:Bag`). See *Sequences*.

Machine learning The construction of algorithms that can learn from and make predictions on data by building a model from example inputs. A wide variety

of techniques and algorithms ranging from *supervised* to *unsupervised* may be employed.

Mapping A considered correlation of *objects* in two different sources to one another, with the relation between the objects defined via a specific *property*. Linkage is a subset of possible mappings.

Member Used synonymously herein with *instance*.

Metadata Metadata are *annotations* and provide information about one or more aspects of the content at hand such as means of creation, purpose, when created or modified, author or provenance, where located, topic or subject matter, standards used, or other descriptive characteristics. In contrast to an *attribute*, which is an individual characteristic intrinsic to an *instance*, metadata is a description about that data.

Metamodeling Metamodeling is the analysis, construction, and development of the frames, rules, constraints, models, and theories applicable and useful for modeling a predefined class of problems.

Microdata Microdata is a proposed specification used to nest semantics within existing content on web pages. Microdata is an attempt to provide a simpler way of annotating HTML elements with machine-readable tags than the similar approaches of using *RDFa* or *microformats*.

Microformats A microformat (sometimes abbreviated μF or uF) is a piece of markup that allows expression of semantics in an HTML (or XHTML) web page. Programs can extract meaning from a web page that is marked up with one or more microformats.

Natural language processing NLP is the process of a computer extracting meaningful information from natural language input and/or producing natural language output. NLP is one method for assigning structured data characterizations to text content. NLP applications include automatic summarization, coreference resolution, machine translation, named entity recognition, question answering, relationship extraction, topic segmentation and recognition, word segmentation, and word sense disambiguation.

Named entity See *entity*.

Named entity recognition See *entity recognition*; also called NER.

Negation Negation is a unary operation that produces a value of *true* when its operand is false and a value of *false* when its operand is true.

Nodes Nodes, in a graphical representation of a *knowledge graph*, are the *subjects* and *objects* in a '*triple*' *statement*; they are connected to one another via *relations* (or *edges* in a graphical representation).

OBIE Information extraction (IE) is the task of automatically extracting structured information from unstructured and/or semi-structured machine-readable documents. Ontology-based information extraction (OBIE) is the use of an ontology to inform a 'tagger' or information extraction program when doing natural language processing. Input ontologies thus become the basis for generating metadata tags when tagging text or documents.

Object An object is an *entity*, *event*, *class*, *concept*, or *property* that can be referred to via an *identifier* of some sort; in KBpedia, every object has a URI *identifier*.

Ontology An ontology is a data model that represents a set of *concepts* or *instances* within a *domain* and the *relationships* between those concepts. Loosely defined, ontologies on the Web can have a broad range of formalism, or expressiveness or *reasoning* power.

Ontology-driven application Ontology-driven applications (or ODapps) are modular, generic software applications designed to operate in accordance with the specifications contained in one or more *ontologies*. The relationships and structure of the information driving these applications are based on the standard functions and roles of ontologies (namely as *domain ontologies*), as supplemented by UI and instruction sets and validations and rules.

Open Semantic Framework The open semantic framework, or OSF, is a combination of a layered architecture and an open-source, modular software stack. The stack combines many leading third-party software packages with open-source *semantic technology* developments from structured dynamics.

Open-world assumption OWA is a formal logic assumption that the truth value of a *statement* is independent of whether or not it is known by any single observer or agent to be true. OWA limits the kinds of *inference* and deductions to those that are known to be true. OWA is useful when we represent knowledge within a system as we discover it, and where we cannot guarantee that we have discovered or will discover complete information, typical of knowledge. See contrast to the *closed-world assumption*.

OWL The Web Ontology Language (OWL) is designed for defining and instantiating formal Web *ontologies*. An OWL ontology may include descriptions of *classes*, along with their related *properties* and *instances*. There are also a variety of OWL dialects.

Particulars Particulars are all *entities* and *events*; they are in *Secondness*.

Pragmatic maxim Pragmatic maxim is the understanding of a topic or an object by an apprehension of all of the practical consequences potentially arising from it.

Pragmatism Pragmatism, what Peirce came to term pragmaticism because of what he felt was a misappropriation of his idea, is the consideration and weighing of available alternatives or explanations in order to pick the most likely ones with a return.

Precision The fraction of retrieved documents that are relevant to the query. It is measured as *true positives* divided by all measured positives (true and false). High precision indicates a high percentage of true positives in relation to all positive results.

Predicate See *Property*.

Preferred label Preferred label (or *prefLabels* or *title*) is the readable string (name) for each *object* in KBpedia. The labels are provided as a readable convenience; the actual definition of the object comes from the totality of its description, prefLabel, altLabels, and connections (placement) within the *knowledge graph*. Labels of all kinds are *representations* and reside in *Thirdness*.

Prescission Prescission, or its verbs *prescind* or *prescinded from*, is the process of comparing two items and seeing if either may exist independent of the other.

If so, we say that the independent one is prescinded from the dependent one; it is one way to determine a subsumption relationship.

Property Property is an official term in *RDF* and *OWL* that includes what we term *attributes*, *external relations*, and *representations*; *w*e try to use the term sparingly, generally when only referencing those items in relation to RDF or OWL.

Punning In computer science, punning refers to a programming technique that subverts or circumvents the type system of a programming language, by allowing a value of a certain type to be manipulated as a value of a different type. When used for *ontologies*, it means to treat a thing as both a *class* and an *instance*, with the use depending on context.

Query Query is a request for information from an agent using a suitable knowledge representation.

Range (property) Range (property) declares the *classes* or data types from which the *object* data or *types* of an *assertion* must be drawn.

RDF Resource Description Framework (RDF) is a data model with a syntax that allows *statements* about *resources* in the form of *subject-predicate-object* expressions, called *triples* in RDF terminology. The subject denotes the resource, and the predicate denotes traits or aspects of the resource and expresses a relationship between the subject and the object.

RDFa RDFa uses attributes from meta and link elements and generalizes them so that they are usable on all elements allowing annotation markup with semantics. RDFa 1.1 is a W3C Recommendation that removes prior dependence on the XML namespace and expands HTML5 and SVG support, among other changes.

RDF Schema RDFS or RDF Schema is an extensible *knowledge representation* language, providing basic elements for the description of *ontologies*, otherwise called RDF vocabularies, intended to structure RDF resources.

Reasoner A semantic reasoner, reasoning engine, rules engine, or simply a reasoner, is a piece of software able to infer logical consequences from a set of asserted facts or *axioms*. The notion of a semantic reasoner generalizes that of an inference engine, by providing a richer set of mechanisms.

Reasoning Reasoning is one of many logical tests using *inference* rules as commonly specified by means of an ontology language, and often a description language. Many reasoners use first-order predicate logic to perform reasoning; inference commonly proceeds by forward chaining or backward chaining.

Recall The fraction of the documents that are relevant to the query that is successfully retrieved. It is measured as *true positives* divided by all potential positives that could be returned from the corpus. High recall indicates a high yield in obtaining relevant results.

Record As used herein, a shorthand reference to an *instance record*.

Reference concept Reference concepts (or *RefConcepts* or *RCs*), the *base concepts* in KBpedia, are any of the noun *objects* within KBpedia and abbreviated as RC. An RC may be either an *entity*, *entity type*, *attribute*, *attribute type*, *relation*, *relation type*, *topic*, or abstract *concept*. RCs are a distinct subset of the more broadly understood 'concept' such as used in the *SKOS RDFS* controlled vocabulary or formal concept analysis or the very general or abstract concepts common to some ontologies. The KBpedia *knowledge graph* is a coherently

organized structure of the nearly 60,000 RCs in KBpedia. All RCs are OWL *classes*.

Referent The *object* referred to by an *identifier*.

Reflexivity Reflexivity is when every element of X is related to itself, every *class* is its own subclass, such as every person is a person.

Reinforcement learning Reinforcement learning (RL) is a method of *machine learning* wherein actions are evaluated, most often as a Markov decision process, in accordance with stated performance objectives via a reward function to help converge to those desired goals.

Relation Relations are the way we describe connections between two or more things; *attributes*, *external relations*, and *representations* are all *instances* of the relations *class*.

Relation type An aggregation (or *class*) of multiple *relations* that have similar characteristics among themselves. As with other *types*, shared characteristics are subsumed over some *essence*(s) that give the *type* its unique character.

Representations Representations are signs, and the means by which we point to, draw or direct attention to, or designate, denote, or describe a particular *object*, *entity*, *event*, *type*, or *general*. A representational relationship has the form of *re:A*. Representations can be designative of the subject, that is, be icons or symbols (including labels, definitions, and descriptions); indexes that more or less help situate or provide a traceable reference to the subject; or associations, resemblances, and likelihoods in relation to the subject, more often of an indeterminate character.

Root Root is the name for the top-level node in a *taxonomy*, *knowledge graph*, *ontology*, or *typology*.

RSS RSS (an acronym for Really Simple Syndication) is a family of web feed formats used to publish frequently updated digital content, such as blogs, news feeds, or podcasts.

Satisfies Satisfies means that all *statements* have an interpretation that can be shown to be true.

schema.org schema.org is an initiative launched by major search engines to create and support a common set of schema for structured data markup on web pages. schema.org provided a starter set of schema and extension mechanisms for adding to them. schema.org supports markup in *microdata*, *microformat*, and RDFa formats.

Secondness Secondness is one of the three universal categories and refers to all actual *instances*, specifically including all individual *events* and *entities*.

Semantic enterprise An organization that uses *semantic technologies* and the languages and standards of the *semantic Web*, including *RDF*, *RDFS*, *OWL*, *SPARQL*, and others to integrate existing information assets, using the best practices of *linked data* and the *open-world assumption*, and targeting knowledge management applications.

Semantic technology Semantic technologies combine software and semantic specifications to encode meanings separate from data and content files and separate from application code. This approach enables machines as well as people to understand, share, and reason with data and specifications separately. Semantic

technologies provide an abstraction layer above existing IT technologies that enables bridging and interconnection of data, content, and processes.

Semantic Web The Semantic Web is a collaborative movement led by the World Wide Web Consortium (W3C) that promotes common formats for data on the World Wide Web. By encouraging the inclusion of semantic content in web pages, the Semantic Web aims at converting the current web of unstructured documents into a 'web of data.' It builds on the W3C's Resource Description Framework (*RDF*).

Semset Semsets (or *synsets* or *alternative labels* or *altLabels*) are collections of alternate labels and terms to describe a *concept* or *entity* or *event*. These semset alternatives include true synonyms, but may also be more expansive and include abbreviations, acronyms, aliases, argot, buzzwords, cognomens, derogatives, diminutives, epithets, hypocorisms, idioms, jargon, lingo, metonyms, misspellings, nicknames, nonstandard terms (see Twitter), pejoratives, pen names, pseudonyms, redirects, slang, sobriquets, or stage names, in short, any term or phrase that can be a reference to a given *instance* or *class*.

Sequences Sequences are ordered *members* or *instances*, with or without gaps or duplicates, useful for bulk assignment purposes. Sequences generally occur through a direct relation assignment (*e.g.*, `rdf:Seq`). See *Lists*.

SKOS SKOS or Simple Knowledge Organization System is a family of formal languages designed for representation of thesauri, classification schemes, *taxonomies*, subject-heading systems, or any other type of structured controlled vocabulary; it is built upon *RDF* and *RDFS*.

Sound An evaluative criterion in logic systems where all provable statements are true in all models.

SPARQL SPARQL (pronounced 'sparkle') is an *RDF* query language; its name is a recursive acronym that stands for SPARQL Protocol and RDF Query Language.

Statement A statement is the standard and most basic expression in *RDF* and *OWL*. A statement is comprised of a '*triple*' (*subject-property-object/value*).

Subject A subject is either a *concept*, *entity*, *event*, or *property* (also collectively known as a '*resource*' in *RDF*), and is the item currently under consideration or focus; it is equivalent to the linguistic *subject*.

Subject extraction Subject extraction is an automatic process for retrieving and selecting subject names from existing *knowledge bases* or datasets. Extraction methods involve parsing and tokenization, and then generally the application of one or more information extraction techniques or algorithms.

SuperType SuperTypes (also Super Types) are a collection of (mostly) similar r*eference concepts*. Most of the SuperType classes have been designed to be (mostly) disjoint from the other SuperType classes. SuperTypes are synonymous with the *typologies* used in KBpedia. SuperTypes (typologies) provide a higher level of clustering and organization of *base concepts* for use in user interfaces and for reasoning purposes.

Supervised learning A *machine learning* task of inferring a function from labeled training data, which optimally consists of positive and negative *training*

sets. The supervised learning algorithm analyzes the training data and produces an inferred function to determine the *class* labels for unseen instances correctly.

Symmetric Symmetric is when A relates to B exactly if it relates B with A.

Synechism Synechism is a philosophical doctrine that space, time, and law are continuous and form an essential *Thirdness* of reality in contrast to existing things and possibilities.

Tag A tag is a keyword or term associated with or assigned to a piece of information (*e.g.*, a picture, article, or video clip), thus describing the item and enabling keyword-based classification of information. Tags are usually chosen informally by either the creator or consumer of the item.

TBox A TBox (for terminological knowledge, the basis for T in TBox) is a 'terminological component,' that is, a conceptualization associated with a set of *facts*. TBox *statements* describe a conceptualization, a set of *concepts* and *properties* for these concepts. The TBox is sufficient to describe an *ontology*. Best practice often suggests keeping a split between instance records, the *ABox*, and the TBox schema.

Taxonomy In the context of knowledge systems, taxonomy is the hierarchical classification of *entities* of interest of an enterprise, organization, or administration, used to classify documents, digital assets, and other information. Taxonomies can cover virtually any type of physical or conceptual entities (products, processes, knowledge fields, human groups, etc.) at any level of granularity.

Thirdness Thirdness is one of the three universal categories and refers to all *classes*, *types*, or *generals*. The category conveys habitual law, continuity, or mediation.

Topic The topic (or theme) is the part of the proposition that is being talked about (predicated). In *topic maps*, the topic may represent any *concept*, from people, countries, and organizations to software modules, individual files, and events. Topics are in *Thirdness*.

Topic Map Topic maps are an ISO standard for the representation and interchange of knowledge. A topic map represents information using *topics*, associations (similar to a predicate relationship), and occurrences (which represent relationships between topics and information resources relevant to them), quite similar in concept to the *RDF triple*.

Training set A set of data used to discover potentially predictive relationships. In *supervised learning*, a positive training set provides data that meet the training objectives; a negative training set fails to meet the objectives.

Transitivity Transitivity is when item A is related to item B, and item B is related to item C, and then A is also related to A; this is the critical *property* for establishing inheritance chains.

Triple A basic *statement* in the *RDF* language, which is comprised of a *subject-property-object* construct, with the subject and property (and object optionally) referenced by *URIs*.

True negative A correct result, but one which fails (is negative) to meet the test objective. It is abbreviated TN.

True positive A correct result, and one which succeeds (is positive) to meet the test objective. It is abbreviated TP.

Tychism A philosophical doctrine that absolute chance is real and operative in the world; it is the source of irregularity and variety and the underlying force of evolution.

Type Types are the hierarchical classification of natural kinds of *instances* as determined by shared *attributes* (though not necessarily the same values for those attributes) and some common *essence*, which is the defining determinant of the type. All types may have hierarchy. Types are in Thirdness.

Typology Typologies are a natural organization of natural *classes* or *types*, with the most general types at the top of the hierarchy, and the more specific at the bottom. All types contained in a typology are children (subclasses) of the *root* type that is the basis for the character of the typology. Typologies provide a modular basis for expanding or collapsing the coverage of similar *instances* within a *knowledge base*. Typologies are central architectural components of KBpedia.

UMBEL UMBEL, short for Upper Mapping and Binding Exchange Layer, is an *upper ontology* of about 35,000 reference concepts, designed to provide common mapping points for relating different ontologies or schema to one another, and a vocabulary for aiding that ontology mapping, including expressions of likelihood relationships distinct from exact identity or equivalence. This vocabulary is also designed for interoperable *domain ontologies*.

Unsupervised learning A form of *machine learning*, this approach attempts to find meaningful, hidden patterns in unlabeled data.

Upper ontology An upper ontology (also known as a top-level ontology or foundation ontology) is an *ontology* that describes very general concepts that are the same across all knowledge domains. An important function of an upper ontology is to support very broad semantic interoperability between a large number of ontologies that are accessible ranking 'under' this upper ontology.

URI A uniform resource identifier is a Web-accessible address (string) for a specific piece of data; it is a more generalized form of a URL, which points to a page or resource location.

Value Value is a string, literal, or data value that pairs a numerical quantity, or quality or utility of a *subject* in relation to the meaning of its associated *attribute*, separate from the subject (but in association with it); these are known as key-value pairs; a value has no meaning or context absent its paired *attribute*.

Vocabulary A vocabulary, in the sense of knowledge systems or *ontologies*, is a *controlled vocabulary*. Vocabularies provide a way to organize knowledge for subsequent retrieval. They are used in formal declarations, subject indexing schemes, subject headings, thesauri, *taxonomies*, and other forms of knowledge organization systems.

Web-oriented architecture Web-oriented architecture, WOA, is a subset of the service-oriented architectural (SOA) style, wherein discrete functions are packaged into modular and shareable elements ('services') that are made available in a distributed and loosely coupled manner. WOA uses the representational state transfer (REST) style, geared to the HTTP model.

Wikidata This is a crowdsourced, open *knowledge base* of (currently) about 55 million structured *entity records*. Each record consists of *attributes* and values with robust cross-links to multiple languages. Wikidata is a key entities source.

Wikipedia Wikipedia is a crowdsourced, free-access, and free-content *knowledge base* of human knowledge. It has about five million articles in its English version. Across all Wikipedias, there are about 42 million articles in 288 different language versions.

WordNet WordNet is a lexical database for English that groups words into sets of synonyms called synsets; provides short, general definitions; and records the various semantic relations between these synonym sets. WordNet provides a combination of dictionary and thesaurus to support text analysis and artificial intelligence applications.

YAGO 'Yet another great ontology' is a WordNet structure placed on top of Wikipedia.

Index

M. K. Bergman, *A Knowledge Representation Practionary*,
https://doi.org/10.1007/978-3-319-98092-8

S