Semantic Web for the Working Ontologist

Modeling in RDF, RDFS and OWL

Dean Allemang

James Hendler

ELSEVIER

AMSTERDAM • BOSTON • HEIDELBERG • LONDON
NEW YORK • OXFORD • PARIS • SAN DIEGO
SAN FRANCISCO • SINGAPORE • SYDNEY • TOKYO
Morgan Kaufmann Publishers is an imprint of Elsevier

MORGAN KAUFMANN PUBL

Publisher: Denise E. M. Penrose
Publishing Services Manager: George Morrison
Project Manager: Marilyn E. Rash
Assistant Editors: Mary E. James
Copyeditor: Debbie Prato
Proofreader: Rachel Rossi
Indexer: Ted Laux
Cover Design: Eric DeCicco
Cover Image: Getty Images
Typesetting/Illustration Formatting: SPi
Interior Printer: Sheridan Books
Cover Printer: Phoenix Color Corp.

Morgan Kaufmann Publishers is an imprint of Elsevier.
30 Corporate Drive, Suite 400, Burlington, MA 01803

This book is printed on acid-free paper.

Library of Congress Cataloging-in-Publication Data

Allemang, Dean
Semantic web for the working ontologist modeling in RDF, RDFS and OWL / Dean Allemang, James A. Hendler.
p. cm.
Includes bibliographical references and index.
ISBN-13: 978-0-12-373556-0 (alk. paper)
1. Web site development. 2. Metadata. 3. Semantic Web. I. Hendler, James. II. Title.
TK5105.888.H465 2008
025.04—dc22
2007051586

For information on all Morgan Kaufmann publications, visit our Web site at *www.mkp.com* or *www.books.elsevier.com*.

ISBN-13: 978-0-12-373556-0
Printed in the United States
09 10 11 12 5 4 3 2

For our students

Preface

In 2003, when the World Wide Web Consortium was working toward the ratification of the Recommendations for the Semantic Web languages RDF, RDFS, and OWL, we realized that there was a need for an industrial-level introductory course in these technologies. The standards were technically sound, but, as is typically the case with standards documents, they were written with technical completeness in mind rather than education. We realized that for this technology to take off, people other than mathematicians and logicians would have to learn the basics of semantic modeling.

Toward that end, we started a collaboration to create a series of trainings aimed not at university students or technologists but at Web developers who were practitioners in some other field. In short, we needed to get the Semantic Web out of the hands of the logicians and Web technologists, whose job had been to build a consistent and robust infrastructure, and into the hands of the practitioners who were to build the Semantic Web. The Web didn't grow to the size it is today through the efforts of only HTML designers, nor would the Semantic Web grow as a result of only logicians' efforts.

After a year or so of offering training to a variety of audiences, we delivered a training course at the National Agriculture Library of the U.S. Department of Agriculture. Present for this training were a wide variety of practitioners in many fields, including health care, finance, engineering, national intelligence, and enterprise architecture. The unique synergy of these varied practitioners resulted in a dynamic four days of investigation into the power and subtlety of semantic modeling. Although the practitioners in the room were innovative and intelligent, we found that even for these early adopters, some of the new ways of thinking required for modeling in a World Wide Web context were too subtle to master after just a one-week course. One participant had registered for the course multiple times, insisting that something else "clicked" each time she went through the exercises.

This is when we realized that although the course was doing a good job of disseminating the information and skills for the Semantic Web, another, more archival resource was needed. We had to create something that students could work with on their own and could consult when they had questions. This was the point at which the idea of a book on modeling in the Semantic Web was conceived. We realized that the readership needed to include a wide variety of people from a number of fields, not just programmers or Web application developers but all the people from different fields who were struggling to understand how to use the new Web languages.

It was tempting at first to design this book to be the definitive statement on the Semantic Web vision, or "everything you ever wanted to know about OWL,"

including comparisons to program modeling languages such as UML, knowledge modeling languages, theories of inferencing and logic, details of the Web infrastructure (URIs and URLs), and the exact current status of all the developing standards (including SPARQL, GRDDL, RDFa, and the new OWL 1.1 effort). We realized, however, that not only would such a book be a superhuman undertaking, but it would also fail to serve our primary purpose of putting the tools of the Semantic Web into the hands of a generation of intelligent practitioners who could build real applications. For this reason, we concentrated on a particular essential skill for constructing the Semantic Web: building useful and reusable models in the World Wide Web setting.

Many of these patterns entail several variants, each embodying a different philosophy or approach to modeling. For advanced cases such as these, we realized that we couldn't hope to provide a single, definitive answer to how these things should be modeled. So instead, our goal is to educate domain practitioners so that they can read and understand design patterns of this sort and have the intellectual tools to make considered decisions about which ones to use and how to adapt them. We wanted to focus on those trying to use RDF, RDFS, and OWL to accomplish specific tasks and model their own data and domains, rather than write a generic book on ontology development. Thus, we have focused on the "working ontologist" who was trying to create a domain model on the Semantic Web.

The design patterns we use in this book tend to be much simpler. Often a pattern consists of only a single statement but one that is especially helpful when used in a particular context. The value of the pattern isn't so much in the complexity of its realization but in the awareness of the sort of situation in which it can be used.

This "make it useful" philosophy also motivated the choice of the examples we use to illustrate these patterns in this book. There are a number of competing criteria for good example domains in a book of this sort. The examples must be understandable to a wide variety of audiences, fairly compelling, yet complex enough to reflect real modeling situations. The actual examples we have encountered in our customer modeling situations satisfy the last condition but either are too specialized—for example, modeling complex molecular biological data; or, in some cases, they are too business-sensitive—for example, modeling particular investment policies—to publish for a general audience.

We also had to struggle with a tension between the coherence of the examples. We had to decide between using the same example throughout the book versus having stylistic variation and different examples, both so the prose didn't get too heavy with one topic, but also so the book didn't become one about how to model—for example, the life and works of William Shakespeare for the Semantic Web.

We addressed these competing constraints by introducing a fairly small number of example domains: William Shakespeare is used to illustrate some of the most basic capabilities of the Semantic Web. The tabular information about products and the manufacturing locations was inspired by the sample data provided

with a popular database management package. Other examples come from domains we've worked with in the past or where there had been particular interest among our students. We hope the examples based on the roles of people in a workplace will be familiar to just about anyone who has worked in an office with more than one person, and that they highlight the capabilities of Semantic Web modeling when it comes to the different ways entities can be related to one another.

Some of the more involved examples are based on actual modeling challenges from fairly involved customer applications. For example, the ice cream example in Chapter 7 is based, believe it or not, on a workflow analysis example from a NASA application. The questionnaire is based on a number of customer examples for controlled data gathering, including sensitive intelligence gathering for a military application. In these cases, the domain has been changed to make the examples more entertaining and accessible to a general audience.

We have included a number of extended examples of Semantic Web modeling "in the wild," where we have found publicly available and accessible modeling projects for which there is no need to sanitize the models. These examples can include any number of anomalies or idiosyncrasies, which would be confusing as an introduction to modeling but as illustrations give a better picture about how these systems are being used on the World Wide Web. In accordance with the tenet that this book does not include everything we know about the Semantic Web, these examples are limited to the modeling issues that arise around the problem of distributing structured knowledge over the Web. Thus, the treatment focuses on how information is modeled for reuse and robustness in a distributed environment.

By combining these different example sources, we hope we have struck a happy balance among all the competing constraints and managed to include a fairly entertaining but comprehensive set of examples that can guide the reader through the various capabilities of the Semantic Web modeling languages.

This book provides many technical terms that we introduce in a somewhat informal way. Although there have been many volumes written that debate the formal meaning of words like *inference, representation,* and even *meaning,* we have chosen to stick to a relatively informal and operational use of the terms. We feel this is more appropriate to the needs of the ontology designer or application developer for whom this book was written. We apologize to those philosophers and formalists who may be offended by our casual use of such important concepts.

We often find that when people hear we are writing a new Semantic Web modeling book, their first question is, "Will it have examples?" For this book, the answer is an emphatic "Yes!" Even with a wide variety of examples, however, it is easy to keep thinking "inside the box" and to focus too heavily on the details of the examples themselves. We hope you will use the examples

as they were intended: for illustration and education. But you should also consider how the examples could be changed, adapted, or retargeted to model something in your personal domain. In the Semantic Web, Anyone can say Anything about Any topic. Explore the freedom.

Second Printing: Since the first printing there have been advances in several of the technologies we discuss such as SPARQL, OWL 2.0, and SKOS that go beyond the state of affairs at the time of first printing. We have created a web site that covers developing technology standards and changing thinking about best practices for the Semantic Web. You can find it at http://www.workingontologist.org/.

ACKNOWLEDGMENTS

Of course, no book gets written without a lot of input and influence from others. We would like to thank a number of professional colleagues, including Bijan Parsia and Jennifer Golbeck, and the students of the University of Maryland MINDSWAP project, who discussed many of the ideas in this book with us. We thank Irene Polikoff, Ralph Hodgson, and Robert Coyne from TopQuadrant Inc., who were supportive of this writing effort, and our many colleagues in the Semantic Web community, including Tim Berners-Lee, whose vision motivated both of us, and Ora Lassila, Bernardo Cuenca-Grau, Xavier Lopez, Zhe Wu and Guus Schreiber, who gave us feedback on what became the choice of features for RDF-PLUS. We are also grateful to the many colleagues who've helped us as we've learned and taught about Semantic Web technologies.

We would also especially like to thank the reviewers who helped us improve the material in the book: John Bresnick, Ted Slater, and Susie Stephens all gave us many helpful comments on the material, and Mike Uschold of Boeing made a heroic effort in reviewing every chapter, sometimes more than once, and worked hard to help us make this book the best it could be. We didn't take all of his suggestions, but those we did have greatly improved the quality of the material, and we thank him profusely for his time and efforts.

We also want to thank Denise Penrose, who talked us into publishing with Elsevier and whose personal oversight helped make sure the book actually got done on time. We also thank Mary James, Diane Cerra, and Marilyn Rash, who helped in the book's editing and production. We couldn't have done it without the help of all these people.

We also thank you, our readers. We've enjoyed writing this book, and we hope you'll find it not only very readable but also very useful in your World Wide Web endeavors. We wish you all the best of luck.

About the Authors

Dean Allemang is the chief scientist at TopQuadrant, Inc.—the first company in the United States devoted to consulting, training, and products for the Semantic Web. He codeveloped (with Professor Hendler) TopQuadrant's successful Semantic Web training series, which he has been delivering on a regular basis since 2003.

He was the recipient of a National Science Foundation Graduate Fellowship and the President's 300th Commencement Award at the Ohio State University. Allemang has studied and worked extensively throughout Europe as a Marshall Scholar at Trinity College, Cambridge, from 1982 through 1984 and was the winner of the Swiss Technology Prize twice (1992 and 1996).

In 2004, he participated in an international review board for Digital Enterprise Research Institute—the world's largest Semantic Web research institute. He currently serves on the editorial board of the *Journal of Web Semantics* and has been the Industrial Applications chair of the International Semantic Web conference since 2003.

Jim Hendler is the Tetherless World Senior Constellation Chair at Rensselaer Polytechnic Institute where he has appointments in the Departments of Computer Science and the Cognitive Science. He also serves as the associate director of the Web Science Research Initiative headquartered at the Massachusetts Institute of Technology. Dr. Hendler has authored approximately 200 technical papers in the areas of artificial intelligence, Semantic Web, agent-based computing, and high-performance processing.

One of the inventors of the Semantic Web, he was the recipient of a 1995 Fulbright Foundation Fellowship, is a former member of the U.S. Air Force Science Advisory Board, and is a Fellow of the American Association for Artificial Intelligence and the British Computer Society. Dr. Hendler is also the former chief scientist at the Information Systems Office of the U.S. Defense Advanced Research Projects Agency (DARPA), was awarded a U.S. Air Force Exceptional Civilian Service Medal in 2002, and is a member of the World Wide Web Consortium's Semantic Web Coordination Group. He is the Editor-in-Chief of *IEEE Intelligent Systems* and is the first computer scientist to serve on the Board of Reviewing Editors for *Science*.

CHAPTER 1

What Is the Semantic Web?

This book is about something we call the Semantic Web. From the name, you can probably guess that it is related somehow to the famous World Wide Web (WWW) and that it has something to do with semantics. Semantics, in turn, has to do with understanding the nature of meaning, but even the word *semantics* has a number of meanings. In what sense are we using the word *semantics*? And how can it be applied to the Web?

This book is also about a working ontologist. That is, the aim of this book is not to motivate or pitch the Semantic Web but to provide the tools necessary for working with it. Or, perhaps more accurately, the World Wide Web Consortium (W3C) has provided these tools in the forms of standard Semantic Web languages, complete with abstract syntax, model-based semantics, reference implementations, test cases, and so forth. But these are like a craftsman's tools: In the hands of a novice, they can produce clumsy, ugly, barely functional output, but in the hands of a skilled craftsman, they can produce works of utility, beauty, and durability. It is our aim in this book to describe the craft of building Semantic Web systems. We go beyond coverage of the fundamental tools to show how they can be used together to create semantic models, sometimes called *ontologies,* that are understandable, useful, durable, and perhaps even beautiful.

WHAT IS A WEB?

The idea of a web of information was once a technical idea accessible only to highly trained, elite information professionals: IT administrators, librarians, information architects, and the like. Since the widespread adoption of the WWW, it is now common to expect just about anyone to be familiar with the idea of a web of information that is shared around the world. Contributions to this web come from every source, and every topic you can think of is covered.

Essential to the notion of the Web is the idea of an open community: Anyone can contribute their ideas to the whole, for anyone to see. It is this openness that has resulted in the astonishing comprehensiveness of topics covered by

the Web. An information "web" is an organic entity that grows from the interests and energy of the community that supports it. As such, it is a hodgepodge of different analyses, presentations, and summaries of any topic that suits the fancy of anyone with the energy to publish a webpage. Even as a hodgepodge, the Web is pretty useful. Anyone with the patience and savvy to dig through it can find support for just about any inquiry that interests them. But the Web often feels like it is "a mile wide but an inch deep." How can we build a more integrated, consistent, deep Web experience?

SMART WEB, DUMB WEB

Suppose you consult a Webpage, looking for a major national park, and you find a list of hotels that have branches in the vicinity of the park. In that list you see that Mongotel, one of the well-known hotel chains, has a branch there. Since you have a Mongotel rewards card, you decide to book your room there. So you click on the Mongotel website and search for the hotel's location. To your surprise, you can't find a Mongotel branch at the national park. What is going on here? "That's so dumb," you tell your browsing friends. "If they list Mongotel on the national park website, shouldn't they list the national park on Mongotel's website?"

Suppose you are planning to attend a conference in a far-off city. The conference website lists the venue where the sessions will take place. You go to the website of your preferred hotel chain and find a few hotels in the same vicinity. "Which hotel in my chain is nearest to the conference?" you wonder. "And just how far off is it?" There is no shortage of websites that can compute these distances once you give them the addresses of the venue and your own hotel. So you spend some time copying and pasting the addresses from one page to the next and noting the distances. You think to yourself, "Why should I be the one to copy this information from one page to another? Why do I have to be the one to copy and paste all this information into a single map?

Suppose you are investigating our solar system, and you find a comprehensive website about objects in the solar system: Stars (well, there's just one of those), planets, moons, asteroids, and comets are all described there. Each object has its own webpage, with photos and essential information (mass, albedo, distance from the sun, shape, size, what object it revolves around, period of rotation, period of revolution, etc.). At the head of the page is the object category: planet, moon, asteroid, comet. Another page includes interesting lists of objects: the moons of Jupiter, the named objects in the asteroid belt, the planets that revolve around the sun. This last page has the nine familiar planets, each linked to its own data page.

One day, you read in the newspaper that the International Astronomical Union (IAU) has decided that Pluto, which up until 2006 was considered a planet, should be considered a member of a new category called a "dwarf

planet"! You rush to the Pluto page, and see that indeed, the update has been made: Pluto is listed as a dwarf planet! But when you go back to the "Solar Planets" page, you still see nine planets listed under the heading "Planet." Pluto is still there! "That's dumb." Then you say to yourself, "Why didn't they update the webpages consistently?"

What do these examples have in common? Each of them has an apparent representation of data, whose presentation to the end user (the person operating the Web browser) seems "dumb." What do we mean by "dumb"? In this case, "dumb" means inconsistent, out of synch, and disconnected. What would it take to make the Web experience seem smarter? Do we need smarter applications or a smarter Web infrastructure?

Smart Web Applications

The Web is full of intelligent applications, with new innovations coming every day. Ideas that once seemed futuristic are now commonplace; search engines make matches that seem deep and intuitive; commerce sites make smart recommendations personalized in uncanny ways to your own purchasing patterns; mapping sites include detailed information about world geography, and they can plan routes and measure distances. The sky is the limit for the technologies a website can draw on. Every information technology under the sun can be used in a website, and many of them are. New sites with new capabilities come on the scene on a regular basis.

But what is the role of the Web infrastructure in making these applications "smart"? It is tempting to make the infrastructure of the Web smart enough to encompass all of these technologies and more. The smarter the infrastructure, the smarter the Web's performance, right? But it isn't practical, or even possible, for the Web infrastructure to provide specific support for all, or even any, of the technologies that we might want to use on the Web. Smart behavior in the Web comes from smart applications on the Web, not from the infrastructure.

So what role does the infrastructure play in making the Web smart? Is there a role at all? We have smart applications on the Web, so why are we even talking about enhancing the Web infrastructure to make a smarter Web if the smarts aren't in the infrastructure?

The reason we are improving the Web infrastructure is to allow smart applications to perform to their potential. Even the most insightful and intelligent application is only as smart as the data that is available to it. Inconsistent or contradictory input will still result in confusing, disconnected, "dumb" results, even from very smart applications. The challenge for the design of the Semantic Web is not to make a web infrastructure that is as smart as possible; it is to make an infrastructure that is most appropriate to the job of integrating information on the Web.

The Semantic Web doesn't make data smart because smart data isn't what the Semantic Web needs. The Semantic Web just needs to get the right data

to the right place so the smart applications can do their work. So the question to ask is not "How can we make the Web infrastructure smarter?" but "What can the Web infrastructure provide to improve the consistency and availability of Web data?"

A Connected Web Is a Smarter Web

Even in the face of intelligent applications, disconnected data result in dumb behavior. But the Web data don't have to be smart; that's the job of the applications. So what can we realistically and productively expect from the data in our Web applications? In a nutshell, we want data that don't surprise us with inconsistencies that make us want to say, "This doesn't make sense!" We don't need a smart Web infrastructure, but we need a Web infrastructure that lets us connect data to smart Web applications so that the whole Web experience is enhanced. The Web *seems* smarter because smart applications can get the data they need.

In the example of the hotels in the national park, we'd like there to be coordination between the two webpages so that an update to the location of hotels would be reflected in the list of hotels at any particular location. We'd like the two sources to stay synchronized, then we won't be surprised at confusing and inconsistent conclusions drawn from information taken from different pages of the same site.

In the mapping example, we'd like the data from the conference website and the data from the hotels website to be automatically understandable to the mapping website. It shouldn't take interpretation by a human user to move information from one site to the other. The mapping website already has the smarts it needs to find shortest routes (taking into account details like toll roads and one-way streets) and to estimate the time required to make the trip, but it can only do that if it knows the correct starting and end points.

We'd like the astronomy website to update consistently. If we state that Pluto is no longer a planet, the list of planets should reflect that fact as well. This is the sort of behavior that gives a reader confidence that what they are reading reflects the state of knowledge reported in the website, regardless of how they read it.

None of these things is beyond the reach of current information technology. In fact, it is not uncommon for programmers and system architects, when they first learn of the Semantic Web, to exclaim proudly, "I implemented something very like that for a project I did a few years back. We used. . . ." Then they go on to explain how they used some conventional, established technology such as relational databases, XML stores, or object stores to make their data more connected and consistent. But what is it that these developers are building?

What is it about managing data this way that made it worth their while to create a whole subsystem on top of their base technology to deal with it? And where are these projects two or more years later? When those same developers

are asked whether they would rather have built a flexible, distributed, connected data model support system themselves or to have used a standard one that someone else optimized and supported, they unanimously chose the latter. Infrastructure is something that one would rather buy than build.

SEMANTIC DATA

In the Mongotel example, there is a list of hotels at the national park and another list of locations for hotels. The fact that these lists are intended to represent the presence of a hotel at a certain location is not explicit anywhere; this makes it difficult to maintain consistency between the two representations. In the example of the conference venue, the address appears only as text typeset on a page so that human beings can interpret it as an address. There is no explicit representation of the notion of an address or the parts that make up an address. In the case of the astronomy webpage, there is no explicit representation of the status of an object as a planet. In all of these cases, the data describe the presentation of information rather than describe the entities in the world.

Could it be some other way? Can an application organize its data so that they provide an integrated description of objects in the world and their relationships rather than their presentation? The answer is "yes," and indeed it is common good practice in website design to work this way. There are a number of well-known approaches.

One common way to make Web applications more integrated is to back them up with a relational database and generate the webpages from queries run against that database. Updates to the site are made by updating the contents of the database. All webpages that require information about a particular data record will change when that record changes, without any further action required by the Web maintainer. The database holds information about the entities themselves, while the relationship between one page and another (presentation) is encoded in the different queries.

Consider the case of the national parks and hotel. If these pages were backed by the same database, the national park page could be built on the query "Find all hotels with location = national park," and the hotel page could be built on the query "Find all hotels from chain = Mongotel." If Mongotel has a location at the national park, it will appear on both pages; otherwise, it won't appear at all. Both pages will be consistent. The difficulty in the example given is that it is organizationally very unlikely that there could be a single database driving both of these pages, since one of them is published and maintained by the National Park Service and the other is managed by the Mongotel chain.

The astronomy case is very similar to the hotel case, in that the same information (about the classification of various astronomical bodies) is accessed from two different places, ensuring consistency of information even in the face of

diverse presentation. It differs in that it is more likely that an astronomy club or university department might maintain a database with all the currently known information about the solar system.

In these cases, the Web applications can behave more robustly by adding an organizing query into the Web application to mediate between a single view of the data and the presentation. The data aren't any less dumb than before, but at least what's there is centralized, and the application or the webpages can be made to organize the data in a way that is more consistent for the user to view. It is the webpage or application that behaves smarter, not the data. While this approach is useful for supporting data consistency, it doesn't help much with the conference mapping example.

Another approach to making Web applications a bit smarter is to write program code in a general-purpose language (e.g., C, Perl, Java, Lisp, Python, or XSLT) that keeps data from different places up to date. In the hotel example, such a program would update the National Park webpage whenever a change is made to a corresponding hotel page. A similar solution would allow the planet example to be more consistent. Code for this purpose is often organized in a relational database application in the form of *stored procedures*; in XML applications, it can be affected using a transformational language like XSLT.

These solutions are more cumbersome to implement, since they require special-purpose code to be written for each linkage of data, but they have the advantage over a centralized database that they do not require all the publishers of the data to agree on and share a single data source. Furthermore, such approaches could provide a solution to the conference mapping problem by transforming data from one source to another. Just as in the query/presentation solution, this solution does not make the data any smarter; it just puts an informed infrastructure around the data, whose job it is to keep the various data sources consistent.

The common trend in these solutions is to move away from having the presentation of the data (for human eyes) be the primary representation of the data; that is, they move from having a website be a collection of pages to having a website be a collection of data, from which the webpage presentations are generated. The application focuses not on the presentation but on the subjects of the presentation. It is in this sense that these applications are semantic applications; they explicitly represent the relationships that underlie the application and generate presentations as needed.

A Distributed Web of Data

The Semantic Web takes this idea one step further, applying it to the Web as a whole. The current Web infrastructure supports a distributed network of webpages that can refer to one another with global links called Uniform Resource Locators (URLs). As we have seen, sophisticated websites replace this structure locally with a database or XML backend that ensures consistency within that page.

The main idea of the Semantic Web is to support a distributed Web at the level of the data rather than at the level of the presentation. Instead of having one webpage point to another, one data item can point to another, using global references called Uniform Resource Identifiers (URIs). The Web infrastructure provides a data model whereby information about a single entity can be distributed over the Web. This distribution allows the Mongotel example and the conference hotel example to work like the astronomy example, even though the information is distributed over websites controlled by more than one organization. The single, coherent data model for the application is not held inside one application but rather is part of the Web infrastructure. When Mongotel publishes information about its hotels and their locations, it doesn't just publish a human-readable presentation of this information but instead a distributable, machine-readable description of the data. The data model that the Semantic Web infrastructure uses to represent this distributed web of data is called the Resource Description Framework (RDF) and is the topic of Chapter 3.

This single, distributed model of information is the contribution that the Semantic Web infrastructure brings to a smarter web. Just as is the case with data-backed Web applications, the Semantic Web infrastructure allows the data to drive the presentation so that various webpages (presentations) can provide views into a consistent body of information. In this way, the Semantic Web helps data not be so dumb.

Features of a Semantic Web

The World Wide Web was the result of a radical new way of thinking about sharing information. These ideas seem familiar now, as the Web itself has become pervasive. But this radical new way of thinking has even more profound ramifications when it is applied to a web of data like the Semantic Web. These ramifications have driven many of the design decisions for the Semantic Web Standards and have a strong influence on the craft of producing quality Semantic Web applications.

Give Me a Voice ...

On the World Wide Web, publication is by and large in the hands of the content producer. People can build their own webpage and say whatever they want on it. A wide range of opinions on any topic can be found; it is up to the reader to come to a conclusion about what to believe. The Web is the ultimate example of the warning *caveat emptor* ("Let the buyer beware"). This feature of the Web is so instrumental in its character that we give it a name: the *AAA Slogan*: "Anyone can say Anything about Any topic."

In a web of documents, the AAA slogan means that anyone can write a page saying whatever they please, and publish it to the Web infrastructure. In the case of the Semantic Web, it means that our data infrastructure has to allow

any individual to express a piece of data about some entity in a way that can be combined with information from other sources. This requirement sets some of the foundation for the design of RDF.

It also means that information is not managed as usual for a large, corporate data center, where one database administrator rules with an iron hand over any addition or modification to the database. A distributed web of data, in contrast, is an organic system, with contributions coming from all sources. It was this freedom of expression on the document Web that allowed it to take off as a bottom-up, grassroots phenomenon.

... *So I May Speak!*

In the early days of the document Web, it was common for skeptics, hearing for the first time about the possibilities of a worldwide distributed web full of hyperlinked pages on every topic, to ask, "But who is going to create all that content? Someone has to write those webpages!"

To the surprise of those skeptics, and even of many proponents of the Web, the answer to this question was that *everyone* would provide the content. Once the Web infrastructure was in place (so that Anyone could say Anything about Any topic), people came out of the woodwork to do just that. Soon every topic under the sun had a webpage, either official or unofficial. It turns out that a lot of people had something to say, and they were willing to put some work into saying it.

The document Web grew because of a virtuous cycle that is called the *network effect*. In a network of contributors like the Web, the infrastructure made it *possible* for anyone to publish, but what made it *desirable* for them to do so? At one point in the Web, when Web browsers were a novelty, there was not much incentive to put a page on this new thing called "the Web"; after all, who was going to read it? Why do I want to communicate to them? Just as it isn't very useful to be the first kid on the block to have a fax machine (whom do you exchange faxes with?), it wasn't very interesting to be the first kid with a Web server.

But because a few people did have Web servers, and a few more got Web browsers, it became more attractive to have both webpages and Web browsers. Content providers found a larger audience for their work; content consumers found more content to browse. As this trend continued, it became more and more attractive, and more people joined in, on both sides. This is the basis of the network effect: The more people who are playing now, the more attractive it is for new people to start playing.

A good deal of the information that will populate the Semantic Web is already available on the Web, typically in the form of tables, spreadsheets, or databases. Who will do the work of converting this data to RDF for distributed access? In the earliest days of the Semantic Web, there was little incentive to do so. As more and more data is available in RDF form, it becomes more useful to write applications that utilize this distributed data. The Semantic Web has been designed to benefit from the same network effect that drove the document Web.

What about the Round-Worlders?

The network effect has already proven to be an effective and empowering way to muster the effort needed to create a massive information network like the World Wide Web; in fact, it is the only method that has actually succeeded in creating such a structure. The AAA slogan enables the network effect that made the rapid growth of the Web possible. But what are some of the ramifications of such an open system? What does the AAA slogan imply for the content of an organically grown web?

For the network effect to take hold, we have to be prepared to cope with a wide range of variance in the information on the Web. Sometimes the differences will be minor details in an otherwise agreed-on area; at other times, differences may be essential disagreements that drive political and cultural discourse in our society. This phenomenon is apparent in the document web today; for just about any topic, it is possible to find webpages that express widely differing opinions about that topic. The ability to disagree, and at various levels, is an essential part of human discourse and a key aspect of the Web that makes it successful. Some people might want to put forth a very odd opinion on any topic; someone might even want to postulate that the world is round, while others insist that it is flat. The infrastructure of the Web must allow both of these (contradictory) opinions to have equal availability and access.

There are a number of ways in which two speakers on the Web may disagree. We will illustrate each of them with the example of the status of Pluto as a planet:

They may fundamentally disagree on some topic. While the IAU has changed its definition of *planet* in such a way that Pluto is no longer included, it is not necessarily the case that every astronomy club or even national body agrees with this categorization. Many astrologers, in particular, who have a vested interest in considering Pluto to be a planet, have decided to continue to consider Pluto as a planet. In such cases, different sources will simply disagree.

Someone might want to intentionally deceive. Someone who markets posters, models, or other works that depict nine planets has a good reason to delay reporting the result from the IAU and even to spreading uncertainty about the state of affairs.

Someone might simply be mistaken. Websites are built and maintained by human beings, and thus they are subject to human error. Some website might erroneously list Pluto as a planet or, indeed, might even erroneously fail to list one of the eight "nondwarf" planets as a planet.

Some information may be out of date. There are a number of displays around the world of scale models of the solar system, in which the status of the planets is literally carved in stone; these will continue to list Pluto as a

planet until such time as there is funding to carve a new description for the ninth object. Websites are not carved in stone, but it does take effort to update them; not everyone will rush to accomplish this.

While some of the reasons for disagreement might be, well, disagreeable (wouldn't it be nice if we could stop people from lying?), in practice there isn't any way to tell them apart. The infrastructure of the Web has to be able to cope with the fact that information on the Web will disagree from time to time and that this is not a temporary condition. It is in the very nature of the Web that there be variations and disagreement.

To Each Their Own

How can the Web infrastructure support this sort of variation of opinion? That is, how can two people say different things, about the same topic? There are two approaches to this issue. First, we have to talk a bit about how one can make any statement at all in a web context.

The IAU can make a statement in plain English about Pluto, such as "Pluto is a dwarf planet," but such a statement is fraught with all the ambiguities and contextual dependencies inherent in natural language. We think we know what "Pluto" refers to, but how about "dwarf planet"? Is there any possibility that someone might disagree on what a "dwarf planet" is? How can we even discuss such things?

The first requirement for making statements on a global web is to have a global way of identifying the entities we are talking about. We need to be able to refer to "the notion of Pluto as used by the IAU" and "the notion of Pluto as used by the American Federation of Astrologers" if we even want to be able to discuss whether the two organizations are referring to the same thing by these names.

In addition to Pluto, another object was also classified as a "dwarf planet." This object is sometimes known as UB313 and sometimes known by the name Xena. How can we say that the object known to the IAU as UB313 is the same object that its discoverer Michael Brown calls "Xena"?

One way to do this would be to have a global arbiter of names decide how to refer to the object. Then Brown and the IAU can both refer to that "official" name and say that they use a private "nickname" for it. Of course, the IAU itself is a good candidate for such a body, but the process to name the object has already taken over two years. Coming up with good, agreed-on global names is not always easy business.

In the absence of such an agreement, different Web authors will select different URIs for the same real-world resource. Brown's Xena is IAU's UB313. When information from these different sources is brought together in the distributed network of data, the Web infrastructure has no way of knowing that these need to be treated as the same entity. The flip side of this is that we

cannot assume that just because two URIs are distinct, they refer to distinct resources. This feature of the Semantic Web is called the *Nonunique Naming Assumption*; that is, we have to assume (until told otherwise) that some Web resource might be referred to using different names by different people.

There's Always One More

In a distributed network of information, as a rule we cannot assume at any time that we have seen all the information in the network, or even that we know everything that has been asserted about one single topic. This is evident in the history of Pluto and UB313. For many years, it was sufficient to say that a *planet* was defined as "any object orbiting the sun of a particular size." Given the information available during that time, it was easy to say that there were nine planets around the sun. But the new information about UB313 changed that; if a planet is defined to be any body that orbits the sun of a particular size, then UB313 had to be considered a planet, too. Careful speakers in the late twentieth century, of course, spoke of the "known" planets, since they were aware that another planet was not only possible but even suspected (the so-called "Planet X," which stood in for the unknown but suspected planet for many years).

The same situation holds for the Semantic Web. Not only might new information be discovered at any time (as is the case in solar system astronomy), but, because of the networked nature of the Web, at any one time a particular server that holds some unique information might be unavailable. For this reason, on the Semantic Web we can rarely conclude things like "there are nine planets," since we don't know what new information might come to light.

In general, this aspect of a Web has a subtle but profound impact on how we draw conclusions from the information we have. It forces us to consider the Web as an *Open World* and to treat it using the *Open World Assumption*. An open world in this sense is one in which we must assume at any time that new information could come to light, and we may draw no conclusions that rely on assuming that the information available at any one point is all the information available.

For many applications, the open world assumption makes no difference; if we draw a map of all the Mongotel hotels in Boston, we get a map of all the ones we know of at the time. The fact that Mongotel might have more hotels in Boston (or might open a new one) does not invalidate the fact that it has the ones it already lists. In fact, for a great deal of Semantic Web applications, we can ignore the open world assumption and simply understand that a semantic application, like any other webpage, is simply reporting on the information it was able to access at one time.

The openness of the Web only becomes an issue when we want to draw conclusions based on distributed data. If we want to place Boston in the list of cities that are not served by Mongotel (e.g., as part of a market study of new places to

target Mongotels), then we cannot assume that just because we haven't found a Mongotel listing in Boston, no such hotel exists.

As we shall see in the following chapters, the Semantic Web includes features that correspond to all the ways of working with open worlds that we have seen in the real world. We can draw conclusions about missing Mongotels if we say that some list is a comprehensive list of all Mongotels. We can have an anonymous "Planet X" stand in for an unknown but anticipated entity. These techniques allow us to cope with the open world assumption in the Semantic Web, just as they do in the open world of human knowledge.

SUMMARY

The aspects of the Web we have outlined here—the AAA slogan, the network effect, nonunique naming and the open world assumption—already hold for the document Web. As a result, the Web today is something of an unruly place, with a wide variety of different sources, organizations, and styles of information. Effective and creative use of search engines is something of a craft; efforts to make order from this include community efforts like social bookmarking and community encyclopedias to automated methods like statistical correlations and fuzzy similarity matches.

For the Semantic Web, which operates at the finer level of individual statements about data, the situation is even wilder. With a human in the loop, contradictions and inconsistencies in the document Web can be dealt with by the process of human observation and application of common sense. With a machine combining information, how do we bring any order to the chaos? How can one have any confidence in the information we merge from multiple sources? If the document Web is unruly, then surely the Semantic Web is a jungle—a rich mass of interconnected information, without any roadmap, index, or guidance.

How can such a mess become something useful? That is the challenge that faces the working ontologist. Their medium is the distributed web of data; their tools are the Semantic Web languages RDF, RDFS, and OWL. Their craft is to make sensible, usable, and durable information resources from this medium. We call that craft *modeling*, and it is the centerpiece of this book.

The cover of this book shows a system of channels with water coursing through them. If we think of the water as the data that are on the Web, the channels are the model. If not for the model, the water would not flow in any systematic way; there would simply be a vast, undistinguished expanse of water. Without the water, the channels would have no dynamism; they have no moving parts in and of themselves. Put the two together, and we have a dynamic system. The water flows in an orderly fashion, defined by the structure of the channels. This is the role that a model plays in the Semantic Web.

Without the model, there is an undifferentiated mass of data; there is no way to tell which data can or should interact with other data. The model itself has no significance without data to describe it. Put the two together, however, and you have a dynamic web of information, where data flow from one point to another in a principled, systematic fashion. This is the vision of the Semantic Web—an organized worldwide system where information flows from one place to another in a smooth but orderly way.

Fundamental Concepts

The following fundamental concepts were introduced in this chapter.

The AAA slogan—Anyone can say Anything about Any topic. One of the basic tenets of the Web in general and the Semantic Web in particular.

Open world/closed world—A consequence of the AAA slogan is that there could always be something new that someone will say; this means that we must assume that there is always more information that could be known.

Nonunique naming—Since the speakers on the Web won't necessarily coordinate their naming efforts, the same entity could be known by more than one name.

The network effect—The property of a web that makes it grow organically. The value of joining in increases with the number of people who have joined, resulting in a virtuous cycle of participation.