

Where Are the Semantics in the Semantic Web?

Michael Uschold

■ The most widely accepted defining feature of the semantic web is machine-usable content. By this definition, the semantic web is already manifest in shopping agents that automatically access and use web content to find the lowest air fares or book prices. However, where are the semantics? Most people regard the semantic web as a vision, not a reality—so shopping agents should not “count.” To use web content, machines need to know what to do when they encounter it, which, in turn, requires the machine to know what the content means (that is, its semantics). The challenge of developing the semantic web is how to put this knowledge into the machine. The manner in which it is done is at the heart of the confusion about the semantic web. The goal of this article is to clear up some of this confusion.

I explain that shopping agents work in the complete absence of any explicit account of the semantics of web content because the meaning of the web content that the agents are expected to encounter can be determined by the human programmers who hardwire it into the web application software. I therefore regard shopping agents as a degenerate case of the semantic web. I note various shortcomings of this approach. I conclude by presenting some ideas about how the semantic web will likely evolve.

The current evolution of the web can be characterized from various perspectives (Jasper and Uschold 2003):

Locating resources: The way people find things on the web is evolving from simple free text and keyword search to more sophisticated semantic techniques both for search and navigation.

Users: Web resources are evolving from be-

ing primarily intended for human consumption to being intended for use both by humans and machines.

Web tasks and services: The web is evolving from being primarily a place to find things to being a place to do things as well (Smith 2001).¹

All these new capabilities for the web depend in a fundamental way on the idea of semantics, giving rise to another perspective from which the web evolution can be viewed:

Semantics: The web is evolving from containing information resources that have little or no explicit semantics to having a rich semantic infrastructure.

Despite the widespread use of the term *semantic web*, it does not yet exist except in isolated environments, primarily in research labs. In the World Wide Web Consortium (W3C) Semantic Web Activity Statement, we are told that

the Semantic Web is a vision: the idea of having data on the Web defined and linked in a way that it can be **used by machines** not just for display purposes, but for automation, integration and reuse of data across various applications (emphasis mine).²

As envisioned by Tim Berners-Lee:

the Semantic Web is an extension of the current Web in which information is given well-defined **meaning**, better enabling computers and people to work in cooperation (Berners-Lee, Hendler, and Lassila 2001, p. 35) (emphasis mine).

[S]omething has semantics when it can

The web is evolving from containing information resources that have little or no explicit semantics to having a rich semantic infrastructure.

be ‘processed and understood by a computer,’ such as how a bill can be processed by a package such as QUICKEN (Trippe 2001, p. 1).

There is no widespread agreement on exactly what the semantic web is for or exactly what it is. Some good ideas about what the semantic web will be used for have emerged from the W3C effort to define a standard ontology language.³ From the previous descriptions, there is clear emphasis on the information content of the web as machine usable and associated with more meaning.

Note that *machine* refers to computers (or computer programs) that perform tasks on the web. These programs are commonly referred to as *software agents*, or softbots, and are found in web applications.

Machine-usable content presumes that the machine knows what to do with information on the web. For this to happen, the machine reads and processes a machine-sensible specification of the semantics of the information. This approach is robust and very challenging and largely beyond the current state of the art. A much simpler alternative is for the human web application developers to hardwire the knowledge into the software so that when the machine runs the software, it does the correct thing with the information. In this second situation, machines already use information on the web. There are electronic broker agents in routine use that make use of the meaning associated with web content words, such as *price*, *weight*, *destination*, and *airport*. Armed with a built-in understanding of these terms, these so-called shopping agents automatically peruse the web to find sites with the lowest price for a book or the lowest airfare between two given cities. Thus, we still lack an adequate characterization of what distinguishes the future semantic web from what exists today.

Because the RESOURCE DESCRIPTION FRAMEWORK (RDF) is hailed by the W3C as a semantic web language,⁴ some people seem to have the view that if an application uses RDF, then it is a semantic web application. This is reminiscent of the “if it is programmed in Lisp or Prolog, then it must be AI” sentiment that was sometimes evident in the early days of AI. There is also confusion about what constitutes a legitimate semantic web application. Some seem to have the view that an RDF tool such as CWM is one.⁵ This is true only in the same sense that KEE and ART were AI applications. They were certainly generating income for the vendors, which is different from the companies using the tools to develop applications that help their bottom line.

The lack of an adequate definition of the semantic web, however, is no reason to stop pursuing its development any more than an inadequate definition of AI was a reason to cease AI research. Quite the opposite, new ideas always need an incubation period.

The research community, industrial participants, and software vendors are working with the W3C to make the semantic web vision a reality (Berners-Lee et al 2001).⁶ It will be layered, extensible, and composable. A major part will entail representing and reasoning with semantic metadata and providing semantic markup in the information resources. Fundamental to the semantic infrastructure are ontologies, knowledge bases, and agents along with inference, proof, and sophisticated semantic querying capability.

The main intent of the semantic web is to give machines much better access to information resources so they can be information intermediaries in support of humans. According to the vision described in Berners-Lee et al. (2001), agents will be pervasive on the web, carrying out a multitude of everyday tasks. Hendler describes many of the important technical issues that this approach entails, emphasizing the interdependence of agent technology and ontologies (Hendler 2001). To carry out their required tasks, intelligent agents must communicate and understand meaning. They must advertise their capabilities and recognize the capabilities of other agents. They must locate meaningful information resources on the web and combine them in meaningful ways to perform tasks. They need to recognize, interpret, and respond to communication acts from other agents.

In other words, when agents communicate with each other, there needs to be some way to ensure that the meaning of what one agent “says” is accurately conveyed to the other agent. There are two extremes, in principle, for handling this problem. The simplest (and perhaps the most common) approach is to ignore the problem altogether. That is, just assume that all agents are using the same terms to mean the same things. In practice, this assumption will usually be built into the application. The assumption could be implicit and informal, or it could be an explicit agreement among all parties to commit to using the same terms in a predefined manner. This approach only works, however, when one has full control over what agents exist and what they might communicate. In reality, agents need to interact in a much wider world, where it cannot be assumed that other agents will use the same terms, or if they do, it cannot be as-

sumed that the terms will mean the same thing.

The moment one accepts the problem and grants that agents might not use the same terms to mean the same things, one needs a way for an agent to discover what another agent means when it communicates. Thus, every agent needs to publicly declare exactly what terms it is using and what the terms mean. This specification is commonly referred to as the agent's ontology (Gruber 1993). If it were written only for people to understand, this specification could just be a glossary. However, meaning must be accessible to other software agents, requiring the meaning to be encoded in some kind of formal language. This approach will enable a given agent to use automated reasoning to accurately determine the meaning of other agents' terms. For example, suppose agent 1 sends a message to agent 2 and in this message is a pointer to agent 1's ontology. Agent 2 can then look in agent 1's ontology to see what the terms mean, the message is successfully communicated, and the agent's task is successfully performed. At least this is the goal. The holy grail is for this processing to happen consistently, reliably, and fully automatically. In practice, there is a plethora of difficulties, most arising from various sources of heterogeneity. For example, there are many different ontology representation languages, different modeling styles, and an inconsistent use of terminology. For further discussion, see the section entitled Why Do Web Shopping Agents Work?

Semantics: A Many-Splendored Thing

The core meaning of the word *semantics* is meaning itself. However, there is no agreement about how this definition applies to the term *semantic web*. In what follows, I characterize the many things that one might mean when talking about semantics as it pertains to the semantic web. It is not my intention to define the term but, rather, to make some important distinctions that people can use to communicate more clearly when talking about the semantic web.

In the context of achieving successful communication among agents on the web, I am talking about the need for agents to understand the meaning of the information being exchanged between agents and the meaning of the content of various information sources that agents require to perform their tasks. We focus attention on the questions of what kinds of semantics there are, what kinds of things

have semantics, where the semantics are, and how they are used. We identify a kind of semantic continuum ranging from the kind of semantics that exist on the web today to a rich semantic infrastructure on the semantic web of the future.

Real-world semantics: *Real-world semantics*⁷ are concerned with the "mapping of objects in the model or computational world on to the real world ... [and] issues that involve human interpretation, or meaning and use of data or information" (Ouksel and Sheth 1999). An object in the model might be a tag or a term or, possibly, a complex expression in some language. We might also speak of the semantics of a possibly large set of expressions, which collectively are intended to represent some real-world domain. The real-world semantics correspond to the concepts in the real world that the objects in the model refer to.

Agent communication language performatives: In the context of the semantic web, *performatives* such as request or inform in agent communication languages (Smith et al. 1998) require semantics to ensure that agents communicate effectively.

Axiomatic semantics: An *axiomatic semantics* for a language specifies "a mapping of a set of descriptions in [that] language into a logical theory expressed in first-order predicate calculus" (p. 4). The basic idea is that "the logical theory produced by the mapping ... of a set of such descriptions is logically equivalent to the intended meaning of that set of descriptions" (p. 1) (Fikes and McGuinness 2001). Axiomatic semantics have been given for RDF, RDF SCHEMA (RDF-S), and DAML + OIL. The axiomatic semantics for a language helps to ascribe a real-world semantics to expressions in this language, in that it limits the possible models or interpretations that the set of axioms might have.

Model-theoretic semantics: "A model-theoretic semantics for a language assumes that the language refers to a 'world,' and describes the minimal conditions that a world must satisfy in order to assign an appropriate meaning for every expression in the language."⁸ Model-theoretic semantics are used as a technical tool for determining when proposed operations on the language preserve meaning. In particular, they characterize what conclusions can validly be drawn from a given set of expressions independent of what the symbols mean.

Intended versus actual meaning: A key to the successful operation of the semantic web is that the intended meaning of web content be accurately conveyed to potential users of this content. In the case of shopping agents, the meaning of terms such as *price* is conveyed

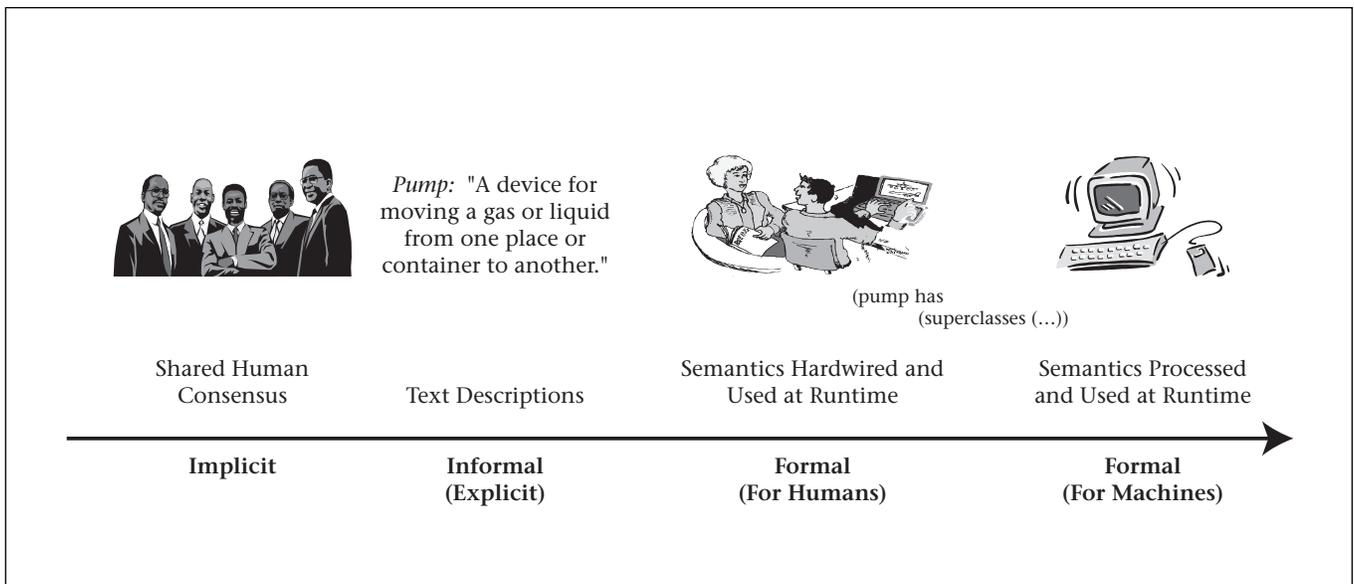


Figure 1. Semantic Continuum.

Semantics can be implicit, existing only in the minds of the humans who communicate and build web applications. They can also be explicit and informal, or they can be formal. The further I move along the continuum, the less ambiguity there is, and the more likely it is to have interoperable, robust, and correctly functioning web applications.

based on human consensus. However, mistakes are always possible because of inconsistency of natural language usage. When formal languages are used, an author attempts to communicate meaning by specifying axioms in a logical theory. In this case, one can talk about intended versus actual models of the theory. There is normally just one intended model. It corresponds to what the author wanted the axioms to represent. The actual models correspond to what the author actually has represented. They consist of all the objects and relationships, and so on, in the real world that are consistent with the axioms. The goal is to create a set of axioms such that the actual models include only the intended model(s).

I believe that the idea of real-world semantics, as described earlier, captures the essence of the main use of the term *semantics* in a semantic web context. However, it is only loosely defined. The ideas of axiomatic and model-theoretic semantics are being used to make the idea of real-world semantics for the semantic web more concrete.

From this discussion, it is clear that several things have semantics: (1) terms or expressions, referring to the real-world subject matter of web content (for example, semantic markup); (2) terms or expressions in an agent communication language (for example, inform); and (3) a language for representing the previous information (for example, the semantics of DAML = OIL of RDF).⁹

A Semantic Continuum

Three questions can be asked about how semantics can be specified: First, are the semantics explicit or implicit? Second, are the semantics expressed informally or formally? Third, are the semantics intended for human processing or machine processing?

These questions give rise to four kinds of semantics: (1) implicit, (2) explicit and informal, (3) explicit and formal for human processing, and (4) explicit and formal for machine processing.

For implicit and informal semantics, there is no alternative to hard wiring the semantics into web application software. In the case of formal semantics, hard wiring remains an option; in which case, the formal semantics serve the important role of reducing ambiguity in specifying web application behavior compared to implicit or informal semantics. There is also the new possibility of using automated inference to process the semantics at run time, thus allowing for much more robust web applications in which agents automatically learn something about the meaning of terms at run time.

I define these four kinds of semantics to be four somewhat arbitrary points along a semantic continuum (figure 1). At one extreme, there are no semantics at all, except what is in the minds of the people who use the terms. At the other extreme are formal and explicit seman-

tics that are fully automated. The further you move along the continuum, the less ambiguity there is, and the more likely the web applications are to be robust, correctly functioning, easy to maintain, and interoperable. I consider these four points on the semantic continuum, in turn. Note that there are likely to be many cases that are not clear cut and, thus, arguably might fall somewhere in between.

Implicit Semantics In the simplest case, the semantics are implicit only. Meaning is conveyed based on a shared understanding derived from human consensus. A common example of this case is the typical use of XML tags, such as price, address, or delivery date. Nowhere in an XML document, or DOCUMENT-TYPE DEFINITION (DTD) or SCHEMA, does it say what these tags mean.¹⁰ However, if there is an implicit shared consensus about what the tags mean, then people can hardwire these implicit semantics into web application programs, using screen scrapers and wrappers. This example illustrates how one implements shopping agents that search web sites for the best deals. From the perspective of mature commercial applications that automatically use web content as conceived by semantic web visionaries, this approach is at or near the current state of the art. The disadvantage of implicit semantics is that they are rife with ambiguity. People often don't agree about the meaning of a term. For example, prices come in different currencies, and they might or might not include various taxes or shipping costs. The removal of ambiguity is the major motivation for the use of specialized language in legal contracts. The costs of identifying and removing ambiguity are very high.

Informally Expressed Semantics At a further point along the continuum, the semantics are explicit and are expressed in an informal notation or language, for example, a glossary or a text specification document. They are mainly for humans. Given the complexities of natural language, machines have an extremely limited ability to make direct use of informally expressed semantics. There are many examples of informally expressed semantics, usually found in specification documents written in (often highly technical) natural language: (1) the meaning of terms in the Dublin core;¹¹ (2) the meaning of tags in HTML such as <h2>, which means second-level header; (3) the meaning of expressions in modeling languages such as the UNIFIED MODELING LANGUAGE (UML);¹² (4) the informal semantics in the original specification of RDF SCHEMA;¹³ (5) the model theory (formal semantics) of RDF SCHEMA that is developed subsequently (does not include the axiomatic se-

mantics that is expressed in a formal language).¹⁴

Typically, the semantics expressed in informal documents are hard wired by humans in working software. Compiler writers use language-definition specifications to write compilers. The specifications for RDF and UML are used to develop modeling tools such as CWM and RATIONAL ROSE.

I characterized this (somewhat arbitrary) point on the semantic continuum by having the semantics expressed in an informal language. However, we can further distinguish whether the semantics are informal or formal. The latter case reduces ambiguity and can be seen as further along the semantic continuum,¹⁵ which helps to avoid inconsistent and incompatible implementations. Users might notice features and start depending on them, resulting in problems if interoperability is required or implementations change. For these and other reasons, informal semantics are sometimes inadequate, which motivates efforts to create formal semantics, for example, for UML (Evans et al. 1998),¹⁶ RDF, and DAML + OIL.¹⁷

Formal semantics are not just playthings for logicians and academics. Given the emerging importance of RDF and the semantic web, vendors were demanding to know what the RDF specification actually meant.¹⁸

Although the formal semantics are a big help, if they are expressed informally, they are not amenable for machine processing. Next, I consider the case of semantics expressed in a formal language. I distinguish between whether they are intended for human processing only or for machine processing as well.

Formally Expressed Semantics for Human Processing Further along the continuum, there are explicit semantics expressed in a formal language. However, they are intended for human processing only. These semantics can be thought of as formal documentation or as formal specifications of meaning. Some examples of this include the following:

Modal logic is utilized to define the semantics of ontological categories such as rigidity and identity (Guarino et al. 1994). These descriptions are for the benefit of humans, to reduce or eliminate ambiguity in what is meant by these ideas.

Modal logic is used to define the semantics of performatives such as inform and request in agent communication languages (ACLs) (Smith et al. 1998). Humans use the formal definitions to understand, evaluate, and compare alternative ACLs. They are also used to implement agent software systems that support these notions.

... how [can]
an agent ...
learn the
meaning of a
new term
from a
formal,
declarative
specification
of its
semantics[?]

Many axioms and definitions in the ENTERPRISE ontology (Uschold et al. 1998) were created without the expectation that they would be used for automated inferencing (although this remained a possibility). The primary purpose was to help communicate the intended meaning to people.

Formal semantics for human processing can go a long way toward eliminating ambiguity, but because there is still a human in the loop, there is ample room for error.

Note that in all practicality, there is little difference between formal semantics expressed informally and formal semantics expressed formally but intended only for human processing. In both cases, the processing is intended for humans, and the formality serves to reduce ambiguity.

Formally Expressed Semantics for Machine Processing Finally, there is the possibility of explicit, formally specified semantics that are intended for machines to directly process using automated inference. The idea is that when new terms are encountered, it is possible to automatically infer something about their meaning and, thus, their use. Inference engines can be used to derive new information for a wide variety of purposes. I explore this topic in depth in the next section.

Machine-Processible Semantics

The defining feature of the semantic web is machine-usable content, which implies that the machine knows what to do with the web content it encounters. This does not mean that there is any explicit account of the semantics. Instead, the semantics (whether implicit, informal, or formal) can be hardwired into the web applications. A more robust approach is to formally represent the semantics and allow the machine to process it to dynamically discover what the content means and how to use it—I call this *machine-processible semantics*. This goal might be impossible to achieve in its full generality, so I restrict this discussion to the following specific question: How can a machine (that is, software agent) learn something about the meaning of a term that it has never encountered before enough to accomplish its task?

One way to look at this questions is from a procedural perspective. For example, how does a compiler know how to interpret a symbol such as + in a computer language, or how does an agent system know what to do when it encounters the performative *inform*? The possibly informal semantics of these symbols are hardwired into a procedure by a human before-

hand, and it is intended for machine processing. When the compiler encounters the symbol, it places a call to the appropriate procedure. The meaning of the symbol is what happens when the procedure is executed. The agent determines the meaning of the symbol by calling the appropriate procedure, so in some sense, these symbols can be viewed as having machine-processible semantics.

I am instead focusing on a declarative view. From this perspective, I ask how an agent can learn the meaning of a new term from a formal, declarative specification of its semantics. Ideally, this learning should occur without making any assumptions at all. In this case, all symbols might as well be in a never-before-seen script from a long-extinct intelligent species on Mars. There is no knowledge of the meaning of the symbols and the rules of syntax for the language, nor is there any information on the semantics of the language. This general case is the most challenging kind of cryptography.

Issues and Assumptions

Cryptography is extremely difficult for humans, never mind machines, so we have to start making some assumptions. Here, I consider some key issues and assumptions.

Language heterogeneity: Different ontology languages are often based on different underlying paradigms (for example, description logic, first-order logic, frame-based representation, taxonomy, semantic net, and thesaurus). Some ontology languages are very expressive, and some are not. Some ontology languages have a formally defined semantics, and some do not. Some ontology languages have inference support, and some do not. If all these different languages are to be allowed, then there is the challenging problem of translating between them. For simplicity then, I assume that the expressions encountered by our agent are from a single language whose syntax and semantics are already known to the agent, for example, RDF SCHEMA OF OWL.

Incompatible conceptualizations: Even with a uniform language, there can still be incompatible assumptions in the conceptualization. For example, in Hayes (1996), it is shown that two representations for time, one based on time intervals and another based on time points, are fundamentally incompatible. That is, an agent whose time ontology is based on time points can never incorporate the axioms of another agent whose ontology for time is based on time intervals. From a logic perspective, the two representations are like oil and water. Thus, I further assume that the conceptualizations are compatible.

Term heterogeneity and different modeling styles: Even if I assume a shared language and compatible conceptualizations, it is still possible, indeed likely, that different people will build different ontologies for the same domain. Two different terms can have the same meaning, and the same term can have two different meanings. The same concept can be modeled at different levels of detail. A given idea can be modeled using different primitives in the language. For example, is the idea of being red modeled by having the attribute color with value red, or is the idea modeled as a class called something like RedThings? Is it both, where either (1) they are independent or (2) RedThings is a derived class defined in terms of the attribute color and the value red?

In the section entitled Machine-Processible Semantics, I spoke of the intended versus the actual models of a logical theory. The former correspond to what the author of the theory wanted to represent. The actual models correspond to what the author actually did represent. The actual models consist of all the objects, relationships, and so on, in the real world that are consistent with the axioms. Because the machine has access to the axioms, it might, in principle, be possible for a computer to determine whether two logical theories are equivalent and, thus, whether the semantics of two terms are identical. This determination would be true, for example, if the two theories had the same actual models. However, even if the exact same language is used, and there is substantial similarity in the underlying conceptualizations and assumptions, the inference required to determine whether two terms actually mean the same thing is intractable.

For a computer to automatically determine the intended meaning of a given term in an ontology is an impossible task, in principle; it would require seeing into the mind of the author. Therefore, a computer cannot determine whether the intended meaning of two terms is the same. This situation is analogous to formal specifications for software. The specification is what the author actually said he/she wanted the program to do. It might be possible to verify that a computer program conforms to this specification, but it will never be possible to verify that a program does what the author actually wanted it to do.¹⁹

To reduce the problems of term heterogeneity and different modeling styles, I further assume that the agent encounters a term that explicitly corresponds to a publicly declared concept that it already knows about (for example, using markup).

An Example

I now consider a simple example of how machine processing of formal semantics can be utilized to do something practical with today's technology. As you can see, automatic machine processing of formal semantics is fraught with difficulties. I have made the following simplifying assumptions: (1) all parties agree to use the same representation language, (2) the conceptualizations are logically compatible, and (3) there are publicly declared concepts that different agents can use to agree on meaning.

Suppose that an agent is tasked with discovering information about a variety of mechanical devices. It encounters a web page with the text *FUEL PUMP* (figure 2). Lacking natural language-understanding capability, the term is completely ambiguous. You can reduce the ambiguity by associating the text *FUEL PUMP* with a formally defined term *fuel-pump* (this is called *semantic markup*). The agent might never have encountered this concept before. In this case, the definition for the new term is defined in terms of the term *pump*, which, in turn, is defined in an externally shared hydraulics ontology. The agent can learn that *fuel-pump* is a subclass of *pump*, which, in turn, is a subclass of *mechanical-device*. The agent now knows that *fuel-pump* is not a typewriter or a spaceship because they are not kinds of pumps. The agent has no knowledge of what kind of pump it is, only that it is some kind of pump. However, this information is sufficient to allow the agent to return this document as relevant to mechanical devices, even though it has never before heard of the term *fuel-pump*. It is possible to do this processing with today's technology using research tools that have been developed (Decker et al. 1999).²⁰ This technology is also being commercialized (Staab and Maedche 2001).²¹ Scale remains a significant barrier to commercial success.

This example illustrates the importance of semantic markup and the sharing of ontologies. It also demonstrates the importance of formal ontologies and automated inference. Inference engines can be used to derive new information for a wide variety of purposes; in particular, a formally specified ontology allows agents to utilize theorem-proving and consistency-checking techniques to determine whether they have agreement on the semantics of their terminology.

The ability of the agent to infer something about the meaning of *fuel-pump* depends on the existence of a formal semantics for ontology languages such as OWL. The language semantics also allow the agent to infer the mean-

For a computer to automatically determine the intended meaning of a given term in an ontology is an impossible task....

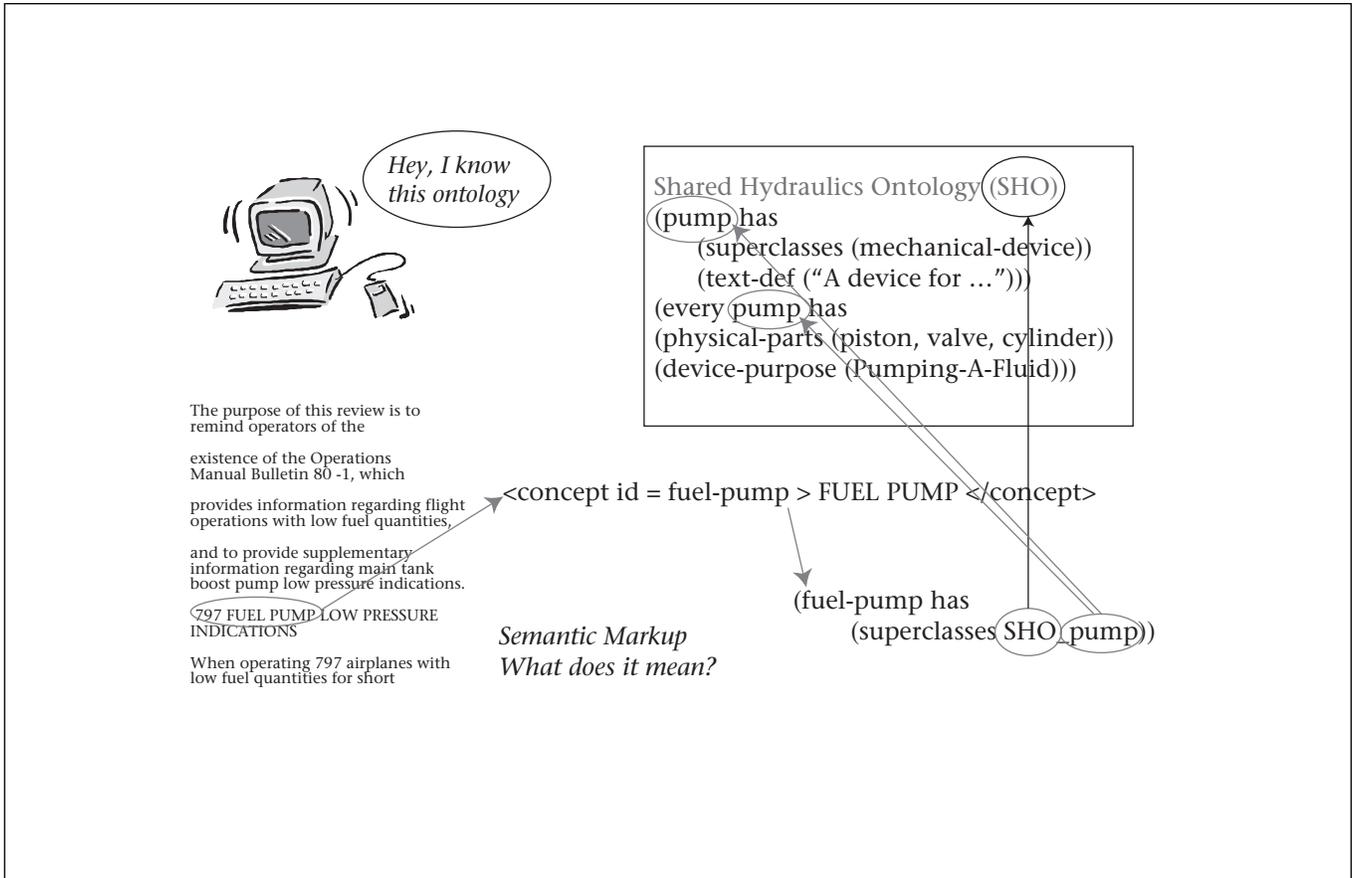


Figure 2. Formal Semantics for Machine Processing.

An agent is searching for information about mechanical devices, as defined in a shared hydraulics ontology (SHO). A document contains the term *FUEL PUMP*, which the agent has never encountered. Semantic markup reveals that it refers to the concept *FUEL PUMP*, which is a kind of *pump*, which is, in turn, defined in SHO as a kind of mechanical device. The agent infers that the document is relevant.

ing of complex expressions built up using language primitives. The semantics of the language are not machine processible; they are written for humans only. People use them to write inference engines or other software to correctly interpret and manipulate expressions in the language.

Note that today's spectacularly impressive search engines by and large do not use formal semantics at all. Overall it remains an unproven conjecture that semantic approaches will have significant impact anywhere on the web. For example, there appear to be insufficient business drivers to motivate venture capitalists to heavily invest in semantic web companies. Fortunately, the W3C is moving forward on this issue by identifying a wide variety of use cases (going well beyond search) to drive the requirements for a standard web ontology language (OWL).²² Also, a strong business case for the semantic web is put forward in a re-

cent book on the semantic web (Daconta et. al. 2003). For further discussion of inference on the semantic web, see Horrocks (2002) and Jasper and Tyler (2001).

Why Do Web Shopping Agents Work?

I have taken some time to consider what people might mean when they talk about the semantic web. There appears to be consensus that the key defining feature is machine-usable web content. However, I argue that by this definition there is an important sense in which the semantic web already exists. For example, travel and bookseller shopping agents automatically access web pages looking for good deals. I don't quibble about whether this should "count" or how the definition of *semantic web* might need to be modified accord-

ingly. It is more useful to regard these examples collectively as a degenerate case of the semantic web. In this section, I examine why web shopping agents work, what their limitations are, and what you can expect in the future.

Requirements for Machine-Usable Content

The following requirements are fundamental for enabling machines to make use of web content.

Requirement 1: The machine needs to know what to do with the content that it encounters.

For example, it needs to recognize that it has found the content it is looking for and to execute the appropriate procedures when it has been found. Ultimately, it is humans that write the programs that enable the machines to do the right thing.

Requirement 2: Humans must know what to do with the content that the program is expected to encounter.

Requirement 3: Humans know the meaning of the expected content or are able to encode a procedure that can learn the meaning.

In determining what makes the web shopping agent examples work, consider the following questions: (1) Hardwiring: What is hardwired and what isn't? (2) Agreements and public declarations: How much agreement is there among different web sites in their use of terminology and in the similarity of the concepts being referred to? Are agreements publicly declared? (3) Specification of semantics: To what extent are the semantics of the content clearly specified? Is it implicit, explicit and informal, or formal?

Hardwiring The general case of automatically determining the meaning of web content is somewhere between intractable and impossible. Thus, a human will always be hardwiring some of the semantics into web applications. The question is what is hardwired and what is not? Shopping agent applications essentially hardwire the meaning of all the terms and procedures. The hardwiring enables the machine to know how to use the content. The hardwiring approach is not robust to changes in web content.

The alternative is allowing the machine to process the semantics specifications directly. Thus, the semantics of the representation languages are made public and hardwired into the inference engines used by the applications. This approach gives an additional degree of flexibility because you do not hardwire the meaning of every term. By making various assumptions regarding languages, conceptualiza-

tions, and shared ontologies, one can get the machine to process formal semantics specifications directly and do useful things (figure 2).

Agreements and Public Declarations In general, the more agreement there is, the better. Making agreements public is critical for the semantic web to take off. For example, the news and magazine publishing industries have developed NEWSML and PRISM (publishing requirements for industry standard metadata).^{23,24} Agreements can also lessen the amount of change, alleviating some maintenance issues.

Although I brought up the issue of public agreements in the context of machine-processible semantics, it is equally important when the semantics are hardwired by the human. For example, consider the Dublin core metadata element `dc:terms` (DCMES), a set of 15 terms for describing resources.²⁵ The elements include such things as title, subject, and date and are designed to facilitate search across different subject areas. The meaning for these elements is defined in English, not a formal language. Nevertheless, if this meaning is hardwired into a web application, this application can make use of web content that is marked up and points to the Dublin core elements.

Semantics Specification Agreements about semantics should clearly be specified so that humans can build reliable and correctly functioning web applications. In the absence of agreements, effort is required to make sure that your application will do the right thing at each web site that uses terms in different ways (for example, only some web sites include taxes in the price information), thus creating more work in programming because a different version is needed for every web site.

Even in the absence of agreements, it is very important for the semantics of web site content to clearly be specified (possibly informally). Otherwise, there will be a lot of guesswork, thus undermining the reliability of applications. When information from different web sites needs to be integrated, there must be some way to map the different meanings to each other. Ontologies in conjunction with semantic mapping and translation techniques play a key role in semantic integration (Bradshaw et al. 2003).

Web Shopping Agents Work because...

I regard the web shopping agents as degenerate examples of the semantic web. It is important to understand why they are able to make use of today's web content in the apparent absence of semantics.

Here I show that they are able to make use of

By making various assumptions regarding languages, conceptualizations, and shared ontologies, one can get the machine to process formal semantics specifications directly and do useful things....

Making agreements public is critical for the semantic web to take off.

today's web content because all three requirements from the section Requirements for Machine-Usable Content are met. I first provide answers to the three questions in that section:

Question 1: Everything is hardwired.

Question 2: There is no agreement among different web sites in their use of terminology, although there is very strong overlap in the underlying concepts that are relevant. We are aware of no public standards, although there could well be standard XML SCHEMA for the travel and bookseller industries, as there are for other industries.

Question 3: I assume that the semantics of the terms and concepts are not specified at all, or if so, they are specified informally.

I now consider the three requirements for machine-usable content in reverse order.

Requirement 3: Humans know the meaning of the expected content, which seems surprising, given the lack of specification of terms and any public standards. The meaning is available instead because there is sufficient human consensus on the use of terms such as price and destination. One can think of this consensus as an *implicit shared semantic repository*, which enables web application developers to make educated guesses and develop useful software.

Requirement 2: Humans know what to do with the content, which follows from understanding what content means and knowing the specifications of the functions of the web agents.

Requirement 1: The machine knows what to do with the content because the human programmers hardwired the semantics of the content and created appropriate procedures to be executed.

Shopping agents can work even if there is no automatic processing of semantics; they can work without any formal representation of semantics; they can even work with no explicit representation of semantics at all. The key to enabling shopping agents to automatically use web content is that the meaning of the web content that the agents are expected to encounter can be determined by the human programmers who hardwire it into the web application software.

Summary and Conclusions

There are many different views of what the semantic web is and how it can or should evolve. I attempted to show a variety of perspectives and possibilities. The most frequently quoted defining feature of the semantic web is machine-usable web content. Fundamentally, this definition requires that machines must know

how to recognize the content they are looking for, and they must know what to do when they find it. This knowledge requires access to the meaning (that is, semantics) of the content, one way or the other. The manner in which the machine can access the semantics of web content is at the heart of much confusion about the semantic web. The main objective of this article was to shed light on this issue.

I argued that paradoxically, today's web shopping agents demonstrate the defining feature of a semantic web application without any explicit representation of semantics. Furthermore, no one would seriously regard them as examples of the semantic web. I resolved the paradox by regarding shopping agents as degenerate cases of the semantic web. I hope that this work will inspire people to generate "genuine" examples of the semantic web. I anticipate that progress in development of the semantic web will take place by (1) moving along the semantic continuum from less clearly specified (implicit) semantics to more clearly specified (formal) semantics; (2) reducing the amount of hardwiring that is necessary or changing which parts are hardwired and which are not (This approach will entail a corresponding increase in the amount of automated inference to determine the meaning of web content, thus enabling agents on the semantic web to correctly perform their tasks. The importance of compelling use cases to drive the demand for this approach cannot be underestimated.); (3) increasing the amount of public standards and agreements, thus reducing the negative impact of today's pervasive heterogeneities; and (4) developing technologies for semantic mapping and translation for the many cases where integration is necessary but where it is not possible to reach agreements.

Finally, I want to stress that there is no need for semantics envy. The needs of the application dictate the appropriate place to be along the semantic continuum. What is right is what works. For many applications, there is no need for rigorous formal approaches because the cost is too high. There might not be a need for machines to automatically determine the meaning of terms; the human can simply hardwire this meaning into the software. Web shopping agents know how to find the fare for a given trip or the price of a book. Every browser knows that `<h2>` means it is a second-level header. There is no need to do inference; it is sufficient to hardwire the meaning of `<h2>` into the browser. I believe that in the short and, possibly, the medium term, approaches that do not make use of machine-processible semantics are likely to have the most impact on the

development of the semantic web. Based on this analysis, I conjecture that the following is a law of the semantic web:

The more agreement there is, the less it is necessary to have machine-processible semantics.

Eventually, there will be a need for automated semantics processing. Relying too heavily on hardwiring semantics can result in different implementations having different functions, which, at best, means interoperability is difficult; at worst, there might be incorrect functions. Another disadvantage of the hardwiring approach is brittleness and consequent maintenance difficulties.

In closing, there are many answers to the question, Where are the semantics in the semantic web? First, they are often just in the human-as-unstated assumptions derived from implicit consensus (for example, price on a travel or bookseller web site). Second, they are in informal specification documents, for example, the semantics of UML or RDF SCHEMA. Third, they are hardwired in implemented code (for example, in UML and RDF tools and in web shopping agents). Fourth, they are in formal specifications to help humans understand or write code (for example, a modal logic specification of the meaning of *inform* in an agent communication language). Fifth, they are formally encoded for machine processing, for example, (fuel-pump has (superclasses SHO: pump)). Sixth, they are in the axiomatic and model-theoretic semantics of representation languages (for example, the formal semantics of RDF).

Finally, I want to note that there are many other important issues for the semantic web that I merely touched on or failed to address in this article. These issues include web services, semantic markup, semantic integration, and use of natural language-processing techniques to glean the semantics of natural language documents.

Acknowledgments

This article was inspired by a stimulating e-mail debate with Dieter Fensel on how and whether XML has the ability to carry semantic information. This article benefited from many discussions with Peter Clark, Rob Jasper, Michael Gruninger, John Thompson, and Frank van Harmelen. I am grateful for valuable feedback from Peter Clark, John Thompson, Pat Hayes, and an anonymous referee on a prior draft. The content of this article was first presented as an invited talk at the Ontologies in Agent Systems Workshop held at the Autonomous Agents Conference in Montreal,

Canada, in June 2001. This article is a significantly revised and extended version of a short article that appeared in a special issue of the *Knowledge Engineering Review* that contained papers from this workshop.

Notes

1. Semantic Web Services Initiative. www.swsi.org.
2. Also see World Wide Web Consortium. 2001. Semantic Web Activity Statement. www.w3.org/2001/sw/Activity.
3. www.w3.org/TR/webont-req/.
2. World Wide Web Consortium. 2003. RDF Semantics, ed. P. Hayes. www.w3.org/TR/rdf-mt/.
4. World Wide Web Consortium. 1999. RESOURCE DESCRIPTION FRAMEWORK (RDF) Schema Specification, eds. D. Brickly, R. Guha. www.w3.org/TR/1998/WD-rdf-schema/.
5. Closed-World Machine. infomesh.net/2001/cwm/.
6. DAML. 2001. See www.daml.org/.
7. This term is commonly used in the literature on semantic integration of databases.
8. World Wide Web Consortium. 2003. LBASE: Semantics for Languages of the Semantic Web, eds. P. Hayes and R. V. Guha. www.w3.org/TR/lbase/.
9. Recently, DAML + OIL has evolved into a W3C standard called the WEB ONTOLOGY LANGUAGE (OWL). www.w3.org/TR/OWL-REF/.
10. R. Cover. 1998. XML and Semantic Transparency, The XML Cover Pages. www.oasis-open.org/cover/xmlAndSemantics.html.
11. S. Weibel and E. Miller. 2000. An Introduction to Dublin Core. www.xml.com/pub/a/2000/10/25-dublincore/.
12. Object Management Group. 2000. OMG Unified Modeling Language Specification, version 1.3. www.omg.org/technology/documents/formal/unified_modeling_language.htm.
13. World Wide Web Consortium. 1999. RESOURCE DESCRIPTION FRAMEWORK (RDF) Schema Specification, eds. D. Brickly, R. Guha. www.w3.org/TR/1998/WD-rdf-schema/.
14. World Wide Web Consortium. 2003. LBASE: Semantics for Languages of the Semantic Web, eds. P. Hayes and R. V. Guha. www.w3.org/TR/lbase/.
15. For simplicity, this distinction is not shown as two separate points on the continuum in figure 1.
16. UML Working Group. 2001. The Precise UML Group home page. www.puml.org.
17. F. van Harmelen, I. Horrocks, and P. Patel-Schneider. 2001. Model-Theoretic Semantics for DAML + OIL. World Wide Web Consortium Note 18. www.w3.org/TR/daml+oil-model.
18. Pat Hayes, personal communication, 2002.
19. A much more detailed discussion of these formal issues can be found in *Complete Semantic Integration*, 2003, M. Gruninger and M. Uschold (forthcoming).
20. ONTOPRISE. 2001. ONTOPRISE: Semantics for the Web. www.ontoprise.de/com/.
21. ONTOPRISE. 2001. ONTOPRISE: Semantics for the Web. www.ontoprise.de/com/.

22. World Wide Web Consortium. 2003. Web Ontology Language (owl) Use Cases and Requirements. www.w3.org/TR/webont-req/.
23. International Press Telecommunications Council (IPTC). 2000. NewsML, Version 1.0 Functional Specification. citeseer.nj.nec.com/452173.html.
24. PRISM Working Group. 2001. PRISM: Publishing Requirements for Industry Standard Metadata Version 1.0. www.prismstandard.org/techdev/prism-spec1.asp.
25. S. Weibel and E. Miller. 2000. An Introduction to Dublin Core. www.xml.com/pub/a/2000/10/25/dublincore/.

References

- Berners-Lee, T.; Hendler, J.; and Lassila, O. 2001. The Semantic Web. *Scientific American* 284(5): 34–43.
- Bradshaw, J. M.; Boy, G.; Durfee, E.; Gruninger, M.; Hexmoor, H.; Suri, N.; Tambe, M.; Uschold, M.; and Vitek, J., eds. 2003. *Software Agents for the Warfighter*. Menlo Park, Calif.: AAAI Press. Forthcoming.
- Daconta, M. C.; Obrst, L. J.; and Smith, K. T. 2003. *The Semantic Web—A Guide to the Future of XML, Web Services, and Knowledge Management*. New York: Wiley.
- Decker, S.; Erdmann, M.; Fensel, D.; and Studer, R. 1999. ONTOBROKER: Ontology-Based Access to Distributed and Semistructured Information. In *Semantic Issues in Multimedia Systems, Proceedings of DS-8*, ed. R. Meersman, Z. Tari, and S. Stevens, 351–369. Boston: Kluwer Academic. See also ontobroker.aifb.uni-karlsruhe.de/index_ob.html.
- Evans, A. S.; France, R. B.; Lano, K. C.; and Rump, B. 1998. The UML as a Formal Modeling Notation. In *UML'98—Beyond the Notation*, 297–307. Lecture Notes in Computer Science 1723. New York: Springer.
- Fikes, R., and McGuinness, D. 2001. An Axiomatic Semantics for RDF, RDF SCHEMA, and DAML + OIL, KSL Technical Report KSL-01-01, Knowledge Systems Laboratory, Stanford University. See www.ksl.Stanford.EDU/people/dlm/daml-semantic/abstract-axiomatic-semantic.html.
- Gruninger, M., and Uschold, M. 2003. Complete Semantic Integration. Forthcoming.
- Gruber, T. R. 1993. A Translation Approach to Portable Ontology Specifications. *Knowledge Acquisition* 6(2): 199–221.
- Guarino, N.; Carrara, M.; and Giarretta, P. 1994. An Ontology of Meta-Level Categories." In *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourth International Conference (KR94)*, eds. E. Sandewall and P. Torasso, 270–280. San Francisco, Calif.: Morgan Kaufmann.
- Hayes, P. 1996. A Catalog of Temporal Theories. Technical Report, UIUC-BI-AI-96-01, University of Illinois.
- Hendler, J. 2001. Agents on the Semantic Web. *IEEE Intelligent Systems* 16(2): 30–37.
- Horrocks, I. 2002. DAML + OIL: A Reasonable Web Ontology Language. Paper presented at the Eighth Conference on Extending Database Technology (EDBT 2002), 24–28 March, Prague, The Czech Republic.
- Jasper, R., and Tyler, A. 2001. The Role of Semantics and Inference in the Semantic Web: A Commercial Challenge. Paper presented at the International Semantic Web Working Symposium, 30 July–1 August, Stanford, California.
- Jasper, R., and Uschold, M. 2003. Enabling Task-Centered Knowledge Support through Semantic Metadata. In *Spinning the Semantic Web*, eds. D. Fensel, J. Hendler, H. Lieberman, and W. Wahlster, 223–251. Cambridge, Mass.: MIT Press.
- Ouksel, A., and Sheth, A. 1999. A Brief Introduction to the Research Area and the Special Section. *SIGMOD Record* (Special Section on Semantic Interoperability in Global Information Systems) 28(1): 5–12. Also see www.acm.org/sigmod/record/issues/9903/.
- Smith, R. 2001. What's Required in Knowledge Technologies—A Practical View. Paper presented at Knowledge Technologies Conference, 4–7 March, Austin, Texas. Also see www.gca.org/attend/2001_conferences/kt_2001/default.htm.
- Smith, I.; Cohen, P.; Bradshaw, J.; Greaves, M.; and Holmback, H. 1998. Designing Conversation Policies using Joint Intention Theory. Paper presented at the Third International Conference on Multiagent Systems (ICMAS-98), 3–7 July, Paris, France.
- Staab, S., and Maedche, A. 2001. Knowledge Portals—Ontologies at Work. *AI Magazine* 21(2): 63–75.
- Trippe, B. 2001. Taxonomies and Topic Maps: Categorization Steps Forward. *Econtent Magazine*, August. www.econtentmag.com/Articles/ArticleReader.aspx.
- Uschold, M.; King, M.; Moralee, S.; and Zorgios, Y. 1998. The Enterprise Ontology. *The Knowledge Engineering Review* (Special Issue on Putting Ontologies to Use) 13(1): 31–89.



Michael Uschold is a research scientist at Boeing, Phantom Works. His interests center on the core area of developing and applying ontologies, including the semantic web, semantic integration, and world modeling for autonomous systems. Uschold is on the industrial advisory boards of various international projects and initiatives related to these areas. He received his B.S. in mathematics and physics, a masters in computer science, and a Ph.D. in artificial intelligence. From 1983 to 1987, he worked at the University of Edinburgh.