Where are the Semantics in the Semantic Web?[⌘]

Michael Uschold
The Boeing Company
PO Box 3707 MS 7L-40
Seattle, WA 98124 USA
+1 425 865-3605
michael.f.uschold@boeing.com

ABSTRACT

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. But 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. This in turn, requires the machine to “know” what the content means (i.e. its semantics). The challenge of developing the Semantic Web is how to put this knowledge into the machine. The manner in which this is done is at the heart of the confusion about the Semantic Web. The goal of this paper is to clear up some of this confusion.

We proceed by describing a variety of meanings of the term “semantics”, noting various things that can be said to have semantics of various kinds. We introduce a semantic continuum ranging from implicit semantics, which are only in the heads of the people who use the terms, to formal semantics for machine processing. We list some core requirements for enabling machines to use Web content, and we consider various issues such as hardwiring, agreements, clarity of semantics specifications, and public declarations of semantics. In light of these requirements and issues in conjunction with our semantic continuum, it is useful to collectively regard shopping agents as a degenerate case of the Semantic Web. 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*.

We note various shortcomings of this approach, which give rise to some ideas about how the Semantic Web should evolve. We argue that this evolution will take place by (1) moving along the semantic continuum from implicit semantics to formal semantics for machine processing, (2) reducing the amount of Web content semantics that is hardwired, (3) increasing the amount of agreements and standards, and (4) developing semantic mapping and translation capabilities where differences remain.

Keywords
Semantic Web, Software Agents, Semantic Heterogeneity, Ontologies

[⌘] The content of this paper was first presented as an invited talk at the Ontologies in Agent Systems workshop held at the Autonomous Agents Conference in Montreal, June 2001. This paper is a significantly revised and extended version of a short paper that appeared in a special issue of the Knowledge Engineering Review for papers from that workshop.

1 Introduction

The current evolution of the Web can be characterized from various perspectives [Jasper & Uschold 2001]:

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 being 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].

All of these new capabilities for the Web depend in a fundamental way on the idea of semantics. This gives rise to a fourth perspective along which the Web evolution may 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, mainly in research labs. In the 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.*[W3C 2001] " [emphasis mine]

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 *et al* 2001] [emphasis mine]
>
> *"[S]omething has semantics when it can be 'processed and understood by a computer,' such as how a bill can be processed by a package such as Quicken."* [Trippe 2001]

There is no widespread agreement on exactly what the Semantic Web is, nor exactly what it is for. From the above descriptions, there is clear emphasis on the information content of the Web being:

- *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 *sofbots* and are found in Web applications.

Machine usable content presumes that the machine knows what to do with information on the Web. One way for this to happen is for the machine to read and process a machine-sensible specification of the semantics of the information. This is a robust and very challenging approach, 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," to name a few. 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 air fare between two given cities. So, we still lack an adequate characterization of what distinguishes the future Semantic Web from what exists today.

Because RDF (Resource Description Framework) [W3C 1999] is hailed by the W3C as a Semantic Web language, 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 Artificial Intelligence. 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[1] is one. This is true only in the same sense that KEE and ART were AI applications. They were certainly generating income for the vendors, but that 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], [DAML 2001], [W3C 2001]). It will be layered, extensible, and composable. A major part of this will entail representing and reasoning with semantic metadata, and/or 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 entails, emphasizing the interdependence of agent technology and ontologies [Hendler 2001]. In order 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 principal, 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 will usually be an assumption 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 pre-defined manner. This 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 assumed that the terms will mean the same thing.

The moment we accept the problem and grant that agents may not use the same terms to mean the same things, we need a way for an agent to discover what another agent means when it communicates. In order for this to happen, every agent will need to publicly declare exactly what terms it is using and what they 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 be just a glossary. However, meaning must be accessible to other software agents. This requires that the meaning be encoded in some kind of formal language. This 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 theory. In practice there is a plethora of difficulties. The holy grail is for this to happen consistently, reliably, and fully automatically. Most of these difficulties arise from various sources of heterogeneity. For example, there are many different ontology representation languages, different modeling styles and inconsistent use of terminology, to name a few. This is explored further in section 3.

[1] Closed World Machine http://infomesh.net/2001/cwm/

2 Semantics: A Many-Splendored Thing

The core meaning of the word "semantics" is: *meaning* itself. Yet there is no agreement as to how this applies to the term "Semantic Web." In what follows, we characterize the many things that one might mean when talking about semantics as it pertains to the Semantic Web. It is not our 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, we are 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 in order 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[2] are concerned with the "*mapping of objects in the model or computational world onto the real world …* [and] *issues that involve human interpretation, or meaning and use of data or information.*" [Ouksel & Sheth 1999] In this context, we talk about the semantics of an "item", which might be a tag or a term, or possibly a complex expression in some language. We may 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 items or sets of items refer to.

Agent Communication Language Performatives—In the context of the Semantic Web, there are special items that require semantics to ensure that agents communicate effectively. These are performatives such as *request* or *inform* in agent communication languages [Smith *et al.* 98].

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.*" 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*" [Fikes & McGuinness 2001]. Axiomatic semantics have been given for the Resource Description Framework (RDF), RDF Schema (RDF-S), and DAML+OIL. The axiomatic semantics for a language helps to ascribe a *real world semantics* to expressions in that language, in that it limits the possible models or interpretations that the set of axioms may 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*". [W3C 2002a] It is used as a technical tool for determining when proposed operations on the language preserve meaning. In particular, it characterizes what conclusions can validly be drawn from a given set of expressions, independently from what the symbols mean.

Intended vs. 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 that content. In the case of shopping agents, the meaning of terms like "price" is conveyed based on human consensus. However, mistakes are always possible, due to 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 we 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, etc., in the real world that

[2] This term is commonly used in the literature on semantic integration of databases.

are consistent with the axioms. The goal is to create a set of axioms such that the actual models only include the intended model(s).

We believe that the idea of real world semantics, as described above 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 (e.g., semantic markup);
2. Terms or expressions in an agent communication language (e.g., *inform*);
3. A language for representing the above information (e.g., the semantics of DAML+OIL or RDF).

2.1 A semantic continuum

We ask three questions about how semantics may be specified:

1. *Are the semantics explicit or implicit?*
2. *Are the semantics expressed informally or formally?*
3. *Are the semantics intended for human processing, or machine processing?*

These give rise to four kinds of semantics:

1. *Implicit;*
2. *Explicit and informal;*
3. *Explicit and formal for human processing;*
4. *Explicit and formal for machine processing.*

We define these to be four somewhat arbitrary points along a semantic continuum (see 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, we have formal and explicit semantics that are fully automated. The further we move along the continuum, the less ambiguity there is and the more likely we are to have robust, correctly functioning and easy to maintain Web applications. We consider these four points on our semantic continuum, in turn. Note that there are likely to be many cases that are not clear cut and thus arguably may fall somewhere in between.

2.1.1 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 DTD or Schema, does it say what these tags mean [Cover 98]. However, if there is an implicit shared consensus about what the tags mean, then people can hardwire this implicit semantics into web application programs, using screen-scrapers and wrappers. This is 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 is at or near the current state of the art. The disadvantage of implicit semantics is that they are rife with ambiguity. People often *do* disagree about the meaning of a term. For example, prices come in different currencies and they may or may not include various taxes or shipping costs. The removal of ambiguity is the major motivation for the use of specialized language used in legal contracts. The costs of identifying and removing ambiguity are very high.

2.1.2 Informal Semantics

At a further point along the continuum, the semantics are explicit and are expressed in an *informal* manner, e.g., a glossary or a text specification document. Given the complexities of natural language, machines have an extremely limited ability to make direct use of informally expressed semantics. This is mainly for humans. There are many examples of informal semantics, usually found in text specification documents.

- The meaning of tags in HTML such as <h2>, which means second level header;

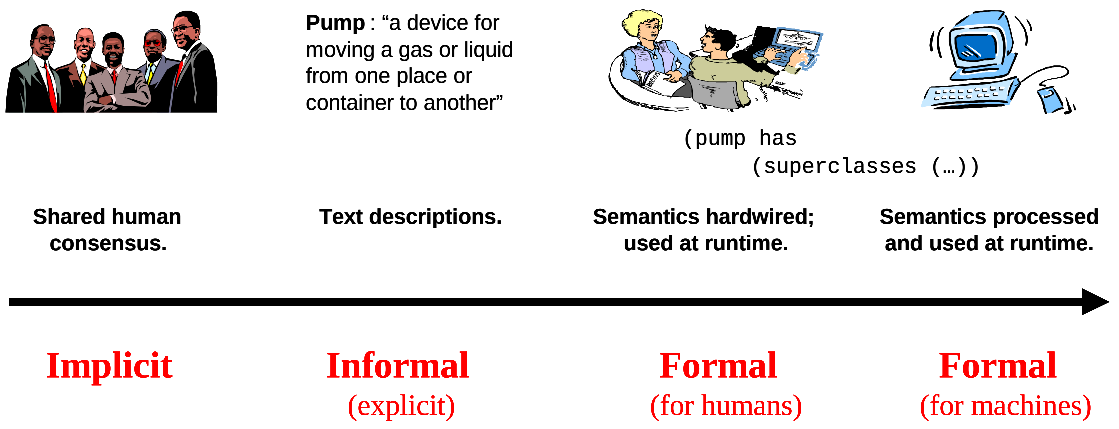

Pump : "a device for moving a gas or liquid from one place or container to another"
(pump has (superclasses (…))
Shared human consensus.
Text descriptions.
Semantics hardwired; used at runtime.
Semantics processed and used at runtime.
Implicit
Informal (explicit)
Formal (for humans)
Formal (for machines)

Figure 1: Semantic Continuum —*Semantics may be implicit, existing only in the minds of the humans who communicate and build Web applications. They may also be explicit and informal, or they may be formal. The further we move along the continuum, the less ambiguity there is and the more likely it is to have robust correctly functioning Web applications. For implicit and informal semantics, there is no alternative to hardwiring the semantics into Web application software. In the case of formal semantics, hardwiring remains an option, in which case the formal semantics serve the important role in 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 runtime. This would allow for much more robust Web applications, in which agents automatically learn something about the meaning of terms at runtime.*

- The meaning of expressions in modeling languages such as UML (Unified Modeling Language) [OMG 2000], and the original specification of RDF Schema [W3C 1999];
- The meaning of terms in the Dublin Core [Weible & Miller 2000]

Typically, the semantics expressed in informal documents are hardwired *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.

The main disadvantage of implicit semantics is that there is still much room for ambiguity. This decreases one's confidence that two different implementations (say of RDF Schema) will be consistent and compatible. Implementations may differ in subtle ways. Users may notice "features" and start depending on them. This can result in problems if interoperability is required or if implementations change. For these and other reasons, informal specifications are sometimes inadequate. This motivates efforts to create formal semantics, e.g., for UML [Evans et al 1998], RDF [W3C 2002a] and DAML+OIL [van Harmlen et.al. 2001].

2.1.3 Formal Semantics for Human Processing

Yet further along the continuum, we have explicit semantics expressed in a formal language. However, they are intended for human processing only. We can think of this as formal documentation, or as formal specifications of meaning. Some examples of this are:

1. Modal logic is used to define the semantics of ontological categories such as rigidity and identity [Guarino *et al.* 1994]. These are for the benefit of humans, to reduce or eliminate ambiguity in what is meant by these ideas.
2. Modal logic is used to define the semantics of performatives such as *inform* and *request* in agent communication languages (ACL) [Smith *et al.* 98]. 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.

3. 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 that 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 to eliminating ambiguity, but because there is still a human in the loop, there is ample scope for errors.

2.1.4 Formal 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 how to use them. Inference engines can be used to derive new information for a wide variety of purposes. We will explore this topic in depth in the next section.

3 Machine Processible Semantics

The defining feature of the Semantic Web is machine usable *content*. This implies that the machine knows what to do with the Web content it encounters. This does not imply 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—we call this *machine processible semantics*. This may be an impossible goal to achieve in its full generality, so we will restrict this discussion to the following specific question: *How can a machine (i.e., software agent) learn* something *about the meaning of a term that it has never before encountered?*

One way to look at this is from a procedural perspective. For example, how does a compiler know how to interpret a symbol like "+" in a computer language? Or, how does an agent system know what to do when it encounters the perfomative "inform"? The possibly informal semantics of these symbols are hardwired into a procedure by a human beforehand, 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 this may be viewed as machine processible semantics.

We are instead focusing on a declarative view. From this perspective, we ask how an agent can learn the meaning of a new term from a formal, declarative specification of its semantics. Ideally, we would like to do this 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. We have no knowledge of the meaning of the symbols, the rules of syntax for the language, nor do we have any information on the semantics of the language. This general case is the most challenging kind of cryptography. It is extremely difficult for humans, never mind machines. So, we have to start making some assumptions.

3.1 Issues and Assumptions

3.1.1 Language Heterogeneity

Different ontology languages are often based on different underlying paradigms (e.g., 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 we are to allow all these different languages, then we are faced with the very challenging problem of translating between them. For simplicity then, we will assume that the expressions encountered by our agent are from *a single language whose syntax and semantics are already known to the agent*, e.g., RDF Schema, or DAML+OIL.

3.1.2 Incompatible Conceptualizations

Even with a uniform language, there may still be incompatible assumptions in the conceptualization. For example, in [Hayes 96] 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. So, we shall further assume that *the conceptualizations are compatible*.

3.1.3 Term Heterogeneity and Different Modeling Styles

Even if we 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 may have the same meaning and the same term may have two different meanings. The same concept may be modeled at different levels of detail. A given idea may 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 modeled as a class called something like RedThings? Or 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?

Even if the exact same language is used, and if 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 at best, and may be impossible.

In section 2, we spoke of the intended vs. actual models of a logical theory. Respectively, these correspond to what the author of the theory *wanted* to represent, vs. what they actually *did* represent. The actual models consist of all the objects and relationships, etc., in the real world that are consistent with the axioms. Because the machine has access to the axioms, it may 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. That would be true, for example, if the two theories had the same actual models. However, to compute this is, in general, *intractable*.

For a computer to automatically determine the *intended* meaning of a given term in an ontology is an impossible task, in principle. This 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 is analogous to formal specifications for software. The specification is what the author actually *said* he or she wanted the program to do. It may 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. A much more detailed discussion of these formal issues may be found in [Gruninger and Uschold 2002].

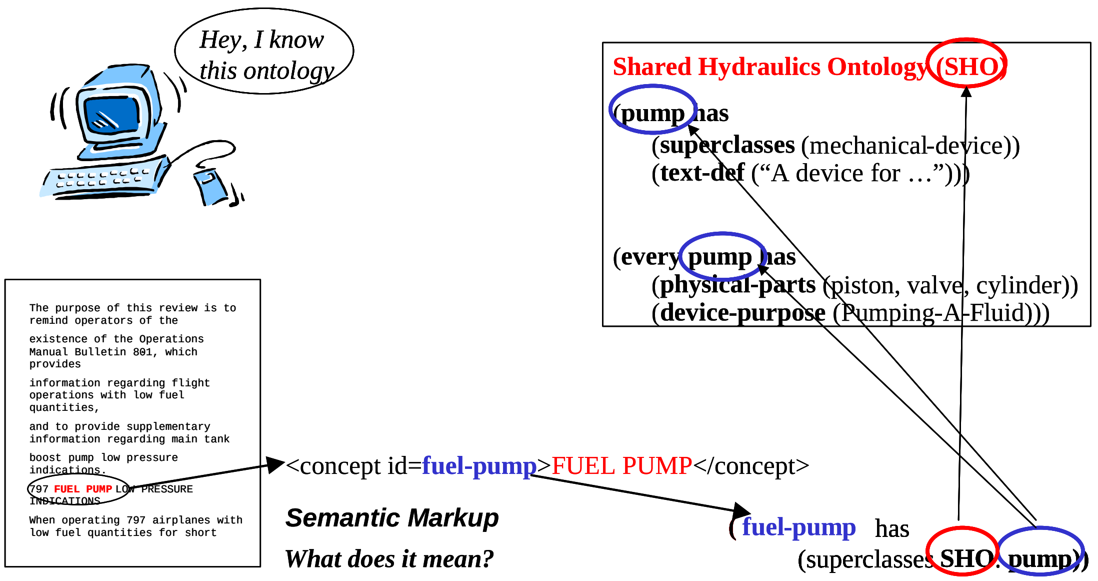
To reduce the problems of term heterogeneity and different modeling styles, we further assume that the *agent encounters a term that explicitly corresponds to a publicly declared concept* that it already knows about (e.g. via markup).

3.2 An Example

We now consider a simple example of how we can use machine processing of formal semantics to do something practical using today's technology. As we can see, automatic machine processing of formal semantics is fraught with difficulties. We have made the following simplifying assumptions:

1. All parties agree to use the same representation language;
2. The conceptualizations are logically compatible;
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*" (see Figure 2). Lacking natural language understanding capability, the term is completely ambiguous. We can reduce the ambiguity by associating the text "*FUEL PUMP*" with a formally defined term *fuel-pump* (this is called *semantic markup*). The agent may never have encountered this concept before. In this case, the definition for the new term is defined in terms of the term *pump* that in turn is defined in an external 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


Hey, I know this ontology
Shared Hydraulics Ontology (SHO)
(pump has
(superclasses (mechanical-device))
(text-def ("A device for …")))
(every pump has
(physical-parts (piston, valve, cylinder))
(device-purpose (Pumping-A-Fluid)))
<concept id=fuel-pump>FUEL PUMP</concept>
Semantic Markup
What does it mean?
(fuel-pump has
(superclasses SHO:pump))

The purpose of this review is to remind operators of the existence of the Operations Manual Bulletin 801, which provides information regarding flight operations with low fuel quantities, and to provide supplementary information regarding main tank boost pump low pressure indications.

797 FUEL PUMP LOW PRESSURE INDICATIONS

When operating 797 airplanes with low fuel quantities for short

Figure 2: Formal Semantics for Machine Processing —*An agent is searching for information about mechanical devices, as defined in a public 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.*

agent now knows that *fuel-pump* is not a typewriter or a space ship, 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 is sufficient to allow the agent to return this document as being relevant to mechanical devices, even though it has never before heard of the term *fuel-pump.* It is possible to do this with today's technology using research tools that have been developed [Decker *et al*. 99; Jasper & Uschold 2001]. There are also attempts to commercialize this technology, e.g., [Ontoprise 2001]. Scale remains a huge 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 use theorem proving and consistency checking techniques to determine whether or not 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 language such as DAML+OIL. The language semantics also allow the agent to infer the meaning 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 approaches. Overall it remains an unproven conjecture that such approaches will enhance search capabilities, or have significant impact anywhere else 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 to drive the requirements for a standard web ontology language [W3C 2002b]. For further discussion of inference on the Semantic Web, see ([Horrocks 2002] ,[Jasper & Tyler 2001]).

4 Why do Web Shopping Agents Work?

We 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, we argue that by this definition there is an important sense in which the Semantic Web *already exists*. The best examples of this are travel and bookseller shopping agents that automatically access Web pages looking for good deals. We shall not quibble about whether or not this should "count," nor how the definition of Semantic Web might need to be modified accordingly. It is more useful to regard these examples collectively as *a degenerate case of the Semantic Web*. In this section, we examine why Web shopping agents work, what their limitations are, and what we can expect in the future.

4.1 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. So:

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

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

In determining what makes the Web shopping agent examples work, we consider the following questions:

Question 1: What is *hardwired* and what isn't?

Question 2: How much *agreement* is there among different Web sites in their use of terminology and in the similarity of the concepts being referred to?

Question 3: To what extent are the *semantics of the content clearly specified*? Is it implicit, explicit and informal, or formal?

Question 4: Are agreements and/or semantics *publicly declared*?

4.1.1 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? The 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 to hardwiring is allowing the machine to process the semantics specifications directly. In our simple fuel pump example, we have an additional degree of flexibility because we need not hardwire the meaning of every term. Instead, we hardwire the semantics of the representation language inference procedures. To make this work, we made many assumptions. For example, by assuming (1) that only one representation language is used, (2) that the conceptualizations are logically compatible, and (3) that there are some publicly declared ontologies that can be shared among Web applications and that Web content sites can point to, we have seen that machines use process formal semantics specifications directly to do useful things (figure 2). In these cases, instead of hardwiring the semantics of the terms representing Web content, the semantics of the *representation languages* are made public and hardwired into the inference engines used by the applications.

4.1.2 Agreements and Semantics Specification

The more agreement there is, the better. For example, there are emerging standards for XML DTDs in specific domains. Agreements can also lessen the amount of change, alleviating some maintenance issues. Agreements about semantics should be clearly specified so that humans can build Web applications that do the right thing.

If there is *not agreement* (e.g., only some Web sites include taxes in the price information), then effort is required to make sure that you have the right concepts at a given Web site. This creates more work in programming, e.g., one may need to create separate Web application modules for each site. This is made easier if the semantics of the terms and concepts at a given Web site *are clearly specified* (possibly informally).

When there is *not agreement* and if the semantics of the terms are *not clearly specified*, 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 [Gruninger & Uschold 2002].

4.1.3 Publicly Declared Concepts

The assumption that there will be terms whose meaning is publicly declared and thus sharable is critical to making the Semantic Web work. Although we brought up this issue 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 Set (DCMES) [Weible & Miller 2000] a set of 15 terms for describing resources. 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, that application can make use of Web content that is marked up and points to the Dublin Core elements. Because XML is widely used, DTDs and XML Schema are convenient ways to express standards. There are many such standards emerging in a variety of sectors. For example, the news and magazine publishing industries have developed NewsML [ITPC 2000] and PRISM (Publishing Requirements for Industry Standard Metadata) [PRISM 2001]. However, XML is merely convenient, it is not necessary. The DTDs and XML Schema say nothing about the semantics of the terms. Semantic agreements are implicit or informally captured in text documentation.

4.2 Web shopping agents work because…

We 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. We do this by showing that all of the above requirements are met. To this end, we first provide answers to the four questions in section 4.1:

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.

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

Question 4: We are aware of no public standards, although there could well be standard XML DTDs for the travel and bookseller industries, as there are for other industries.

We now consider the above requirements for machine usable content in reverse order.

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

Requirement 2: *Humans know what to do with the content.* This follows from understanding what content means, and from the specifications of the functionality of the Web agents.

Requirement 1: *The machine "knows" what to do with the content.* This is due to the human programmers hardwiring the semantics of the content and creating appropriate procedures to be executed.

Shopping agents can work even if there is no automatic processing of semantics; it can be done without any formal representation of semantics; it can even be done without any 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.*

5 Summary and Conclusions

There are many different views of what the Semantic Web is and how it may or should evolve. We have 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 requires that machines must "know" how to recognize the content they are looking for, and they must "know" what to do when they encounter it. This "knowledge" requires access to the meaning (i.e., semantics) of the content, one way or the other. But what does that mean? The manner in which the machine can access the semantics of Web content is at the heart of the confusion about the Semantic Web. The main objective of this paper was to shed light on that issue and remove some of this confusion.

We began by introducing the Semantic Web and noting some of the major challenges in making it a reality. Perhaps the most important issue is the pervasive heterogeneity that exists on the Web. We discussed a variety of meanings for the term "semantics", including real world semantics, axiomatic semantics and model theoretic semantics and how they are related. We stated that the key to the success of the Semantic Web is the ability of Web applications to correctly determine the intended semantics of Web content. We noted various things that have semantics. This includes terms and expressions referring to subject matter, performatives in agent communication languages, and representation languages such as RDF or DAML+OIL.

To get at the heart of the matter of how machines can access the semantics of Web content, we introduced a semantic continuum. At one end we have implicit semantics, which are only in the heads of the people who use the terms; next there are explicit informal semantics, followed by formal semantics intended for human processing only. In all of these cases the semantics must be *hardwired* into the Web application software. The last point on the semantic continuum corresponds to the use of formal semantics that are intended for machine processing. In this case we can reduce the amount of hardwiring of Web content semantics and begin to dynamically infer something about the meaning. This happens by hardwiring the semantics of the representation languages that are used to perform inference.

We gave a simple example where using formal semantics, a machine can infer something about the meaning of the term *fuel-pump,* that it had never encountered before. This example can be implemented using today's technology. But to do this, we were forced to make many simplifying assumptions to remove various heterogeneities (language, terms, modeling style, etc.). A crucial aspect of this example is the reliance on publicly declared semantics of terms that can be shared among many software agents and Web sites. We also showed that such public declarations need not be formal to be useful; the Dublin Core is an example of informal public declarations which are widely used.

We argued that today's Web shopping agents satisfy the above definition for the Semantic Web. We also acknowledged that most people would say that these examples do *not* satisfy their vision of the Semantic Web. We resolved this conflict by regarding these applications collectively as a degenerate case of the Semantic Web, and we explored what it is about these applications that enables them to make use of today's Web content. We listed fundamental requirements for enabling machines to use Web content, and we considered various issues such as hardwiring, agreements, semantics specifications, and public declarations of semantics. Using these requirements and issues in conjunction with our semantic continuum, we explained why the Web shopping agents work and what their shortcomings are.

We hope that this work will inspire people to generate "genuine" examples of the Semantic Web. We 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, and/or changing which parts are hardwired and which are not. This will entail a corresponding increase in the amount of automated inference to infer the meaning of Web content, thus enabling agents on the Semantic Web to do correctly perform their tasks. The importance of compelling use cases to drive the demand for this cannot be underestimated.
3. Increasing the amount of public standards and agreements, thus reducing the negative impact of today's pervasive heterogeneities.
4. Developing technologies for semantic mapping and translation for the many cases where integration is necessary, but it is not possible to reach agreements.

Finally, we stress that there is nothing inherently good about being further along the semantic continuum. In some cases there will be advantages; in other cases there will not. What is good is what works. For many applications there is no 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 what <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. We believe that in the short and possibly 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. Inspired by this analysis we conjecture that the following is a law of the semantic web:

"The more agreement there is, the less it is necessary to have machine processable semantics."

Eventually there will be a need for automated semantics processing. Relying too heavily on hardwiring semantics can result in different implementations having different functionality. At best this means inter-operation is difficult; at worst there maybe incorrect functionality. Another disadvantage of the hardwiring approach is brittleness and consequent maintenance difficulties.
In closing, the answers to the question "Where are the Semantics in the Semantic Web?" are many:

1. Often just in the human as unstated assumptions derived from implicit consensus (e.g., "price" on a travel or bookseller Web site).
2. In informal specification documents (e.g., the semantics of UML or RDF Schema).
3. Hardwired in implemented code (e.g., in UML and RDF tools; and in Web shopping agents)
4. In formal specifications to help humans understand and/or write code. (e.g., a modal logic specification of the meaning of "inform" in an agent communication language).
5. Formally encoded for machine processing
   e.g.,(`fuel-pump has (superclasses SHO: pump)`)
6. In the axiomatic and model-theoretic semantics of representation languages (e.g., the formal semantics of RDF).

Finally, we wish to note that there are many other important issues for the Semantic Web that we have merely touched on or failed to address in this paper. These include Web services, semantic markup, semantic integration, and how natural language processing techniques may be used to glean the semantics of natural language documents.

Acknowledgements

This paper was inspired by a stimulating e-mail debate with Dieter Fensel on how and whether XML has the ability to carry semantic information. I am grateful for valuable feedback from Peter Clark, John Thompson, Pat Hayes and an anonymous referee on a prior draft. This paper benefits from many discussions with Peter Clark, Rob Jasper, Michael Gruninger, John Thompson and Frank van Harmlen.

6 References

[Berners-Lee *et al.* 2001] Berners-Lee, T., Hendler, J. and Lassila, O. (2001) "The Semantic Web," *Scientific American*, May 2001. See http://www.scientificamerican.com/2001/0501issue/0501berners-lee.html

[Cover 1998] Cover, R. "XML and Semantic Transparency," The XML Cover Pages http://www.oasis-open.org/cover/xmlAndSemantics.html

[DAML 2001]. See http://www.daml.org/.

[Decker *et al.* 1999] Stefan Decker, Michael Erdmann, Dieter Fensel, and Rudi Studer; "Ontobroker: Ontology Based Access to Distributed and Semi-Structured Information." In R. Meersman et al. (eds.): *Semantic Issues in Multimedia Systems. Proceedings of DS-8*. Kluwer Academic Publisher, Boston, 1999, 351-369. See also: http://ontobroker.aifb.uni-karlsruhe.de/index_ob.html.

[Evans et al 1998] A.S.Evans, R.B.France, K.C.Lano, B.Rumpe "The UML as a formal modelling notation" In: *UML'98 - Beyond the notation*, LNCS. Springer, 1998.

[Fikes and McGuinness 2001] R.Fikes and D. McGuinness. *An Axiomatic Semantics for RDF, RDF Schema, and DAML+OIL*, KSL Technical Report KSL-01-01, 2001. http://www.ksl.Stanford.EDU/people/dlm/daml-semantics/abstract-axiomatic-semantics.html

[Gruninger & Uschold 2002] M. Gruninger and M. Uschold; "Ontologies and Semantic Integration," to appear in *Software Agents for the Warfighter*, the first in a series of reports sponsored by the US Government Information Technology Assessment Consortium (ITAC). Edited by Jeff Bradshaw, Institute for Human and Machine Cognition (IHMC), University of West Florida

[Gruber 1993] Gruber, T.R. A translation approach to portable ontology specifications. *Knowledge Acquisition* 5:199-220.

[Guarino *et al.* 1994] Guarino, N., Carrara, M., and Giaretta, P. 1994. "An Ontology of Meta-Level Categories." In E. Sandewall and P. Torasso (eds.), *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourth International Conference (KR94*). Morgan Kaufmann, San Mateo, CA: 270-280.

[van Harmlen et.al. 2001] F. van Harmlen, I. Horrocks and P Patel-Schneider, "Model-Theoretic Semantics for DAML+OIL;" W3C Note 18, December 2001 http://www.w3.org/TR/daml+oil-model

[Hayes 1996] Hayes, P. *A Catalog of Temporal Theories*.; Technical Report UIUC-BI-AI-96-01, University of Illinois 1996

[Hendler 2001] Hendler, J. "Agents on the Semantic Web," *IEEE Intelligent Systems*, Vol. 16, No. 2, March/April 2001

[Horrocks 2002] Horrocks, I. *DAML+OIL: a Reason-able Web Ontology Language* To appear in the proceedings of the Eighth Conference on Extending Database Technology (EDBT 2002) March 24-28 2002, Prague

[ITPC 2000] International Press Telecommunications Council (IPTC) *NewsML Version 1.0 Functional Specification*, 24 October 2000 http://citeseer.nj.nec.com/452173.html

[Jasper & Tyler 2001] Jasper, R and Tyler, A. "The Role of Semantics and Inference in the Semantic Web: A Commercial Challenge." In *Proceedings of the International Semantic Web Working Symposium* held at Stanford University, July 30 - August 1, 2001

[Jasper & Uschold 2001] Jasper, R and Uschold, M. (2001) "Enabling Task-Centered Knowledge Support through Semantic Metadata." In *Semantic Web Technolog*y, D. Fensel, J. Hendler, H. Lieberman, and W. Wahlster (eds.): MIT Press, Boston, to appear.

[OMG 2000] Object Management Group. OMG Unified Modeling Language Specification, version 1.3. http://www.omg.org/technology/documents/formal/unified modeling language.htm, 2000.

[Ontoprise 2001] Ontoprise. "Ontoprise: Semantics for the Web" http://www.ontoprise.de/com/

[Ouksel & Sheth 1999] A. Ouksel and A. Sheth "A Brief Introduction to the Research Area and the Special Section" *Special Section on Semantic Interoperability in Global Information Systems,* SIGMOD Record Vol 28(1), March 1999 http://www.acm.org/sigmod/record/issues/9903/

[Precise UML Working Group 2001] Precise UML Group. The Precise UML Group home page. http://www.puml.org , 2001.

[PRISM 2001] PRISM Working Group *PRISM: Publishing Requirements for Industry Standard Metadata Version 1.0*, April 9, 2001 http://www.prismstandard.org/techdev/prismspec1.asp

[Smith 2001] R. Smith, "What's Required in Knowledge Technologies - A Practical View", *Proceedings of Knowledge Technologies 2001*. See http://www.gca.org/attend/2001_conferences/kt_2001/default.htm

[Smith *et al.* 1998] I. Smith, P. Cohen, J. Bradshaw, M. Greaves, and H. Holmback. "Designing Conversation Policies using Joint Intention Theory". *Proceedings of the Third International Conference on Multi Agent Systems (ICMAS-98),* 3 - 7 July, 1998, Paris, France, IEEE Press, pp. 269-276.

[Trippe 2001] Bill Trippe, "Taxonomies and Topic Maps: Categorization Steps Forward." *Econtent Magazine*, August 2001. http://www.ecmag.net/Magazine/Features/trippe8_01.html

[Uschold *et al.*1998] M. Uschold, M. King, S. Moralee and Y. Zorgios. (1998) "The Enterprise Ontology" *The Knowledge Engineering Review*, Vol. 13, Special Issue on Putting Ontologies to Use.

[Weible & Miller 2000] Stuart Weibel, Eric Miller An Introduction to Dublin Core October 25, 2000 http://www.xml.com/pub/a/2000/10/25/dublincore/

[W3C 2001] World Wide Web Consortium: "Semantic Web Activity Statement." http://www.w3.org/2001/sw/Activity

[W3C 2002a] W3C *RDF Model Theory W3C Working Draft* Editor, P Hayes 14 February 2002 http://www.w3.org/TR/rdf-mt/

[W3C 2002b] W3C Web Ontology Working Group; Jan 2002 Face to Face Meeting http://www.w3.org/2001/sw/WebOnt/ftf1.html

[W3C 1999] World Wide Web Consortium "Resource Description Framework (RDF) Schema Specification" Editors: D. Brickly, R. Guha http://www.w3.org/TR/1998/WD-rdf-schema/