# The semantic Web and its languages

Editor: Dieter Fensel
Vrije Universiteit Amsterdam
dieter@cs.vu.nl

The Web has drastically changed the availability of electronic information, but its success and exponential growth have made it increasingly difficult to find, access, present, and maintain such information for a wide variety of users. In reaction to this bottleneck, many new research initiatives and commercial enterprises have been set up to enrich available information with machine-processable semantics. Such support is essential for bringing the Web to its full potential in areas such as knowledge management and electronic commerce. This semantic Web will provide intelligent access to heterogeneous and distributed information, enabling software products (agents) to mediate between user needs and available information sources. Early steps in the direction of a semantic Web were SHOE[1] and later Ontobroker,[2] but now many more projects exist.

Originally, the Web grew mainly around HTML, which provided a standard for structuring documents so that browsers could translate them in a canonical way. On the one hand, it was HTML's simplicity that enabled the Web's fast growth, but on the other, its simplicity seriously hampered more advanced Web applications in many domains and for many tasks. This was the reason for XML (see Figure 1), which lets us define arbitrary domain- and task-specific extensions. Even HTML has been redefined as an XML application—XHTML. Consequently, we define the semantic Web as an XML application.

The Resource Description Framework took the first step toward defining the Web in XML terms. RDF defines a syntactical convention and a simple data model for representing data's machine-processable semantics. It is a standard for Web metadata developed by the World Wide Web Consortium. (See the essay by Ora Lassila.)

The RDF Schema candidate recommendation that defines basic ontological modeling primitives on top of RDF took the second step, followed by the Ontology Inference Layer, which uses the RDFS as a starting point and extends it to a full-fledged ontology modeling language. OIL unifies three important aspects provided by different communities: epistemologically rich modeling primitives, provided by the frame community; formal semantics and efficient reasoning support, provided by description logics; and a standard proposal for syntactical exchange notations, provided by the Web community. (See the essay by Frank van Harmelen and Ian Horrocks.)

Another candidate for such a Web-based ontology modeling language is DAML-ONT. The DARPA Agent Markup Language is a major, well-funded initiative, aimed at joining the many ongoing semantic Web efforts and focused on bringing ontologies to the Web. The DAML language inherits many aspects from OIL, and the capabilities of the two languages are relatively similar. Both initiatives cooperate in a Joint EU/US ad hoc Agent Markup Language Committee to achieve a joined language proposal. The next step will be DAML-Logic, a language with sufficient means for expressing axioms and rules. (See the essay by James Hendler and Deborah L. McGuiness.)

Defining languages for the semantic Web is just the first step. Developing new tools, architectures, and applications is the real challenge that will follow.

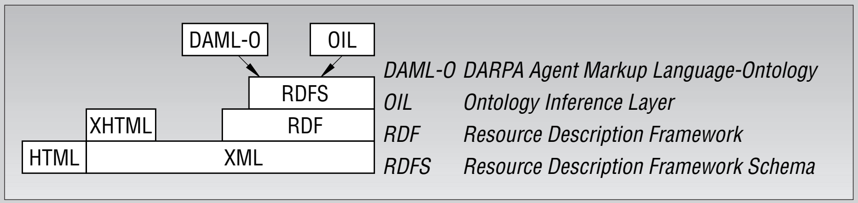

DAML-O
OIL
RDFS
XHTML
RDF
HTML
XML
DAML-O DARPA Agent Markup Language-Ontology
OIL Ontology Inference Layer
RDF Resource Description Framework
RDFS Resource Description Framework Schema

Figure 1. The layer language model for the Web.

### References

1. S. Luke, L. Spector, and D. Rager, "Ontology-Based Knowledge Discovery on the World-Wide Web," *Working Notes Workshop Internet-Based Information Systems at the 13th Nat'l Conf. Artificial Intelligence* (AAAI 96), 1996.
2. D. Fensel et al., "Ontobroker: The Very High Idea," *Proc. 11th Int'l Flairs Conf.* (FLAIRS 98), 1998, pp. 131–135.

**Dieter Fensel** is an associate professor at the Division of Mathematics and Computer Science, Vrije Universiteit, Amsterdam, and he is a new department editor for Trends & Controversies. After studying mathematics, sociology, and computer science in Berlin, he joined the Institute AIFB at the University of Karlsruhe. His major subject was knowledge engineering and his PhD thesis was about a formal specification language for knowledge-based systems. Currently, his focus is on the use of ontologies to mediate access to heterogeneous knowledge sources and to apply them in knowledge management and electronic commerce. Contact him at dieter@cs.vu.nl; www.cs.vu.nl/~dieter.

## The Resource Description Framework

*Ora Lassila, Nokia Research Center*

The Resource Description Framework is a standard for Web metadata that the World Wide Web Consortium (W3C) developed.[1–3] Expanding the traditional notion of document metadata (such as library catalog information), RDF is suitable for describing any Web resource, and as such it provides interoperability between applications that exchange machine-understandable information on the Web. Its goal is to add formal semantics to the Web, thus paving the way for the so-called semantic Web.

### Modeling

All items that RDF expressions describe are called *resources*, and, broadly speaking, anything a Universal Resource Identifier can name is also a resource.[4] Consequently, RDF can describe not just things on the Web (such as pages, parts of pages, or collections of pages) but also things not on the

Web—as long as they can be named using some URI scheme. (Hypothetically, we could name or describe any person with a "person" URI scheme—for example, "person:US:123-45-6789.") The RDF description model uses object–attribute–value triples: we can view instances of the model as directed or labeled graphs (which resemble semantic networks), or we can take a more object-centric view and think of RDF as a frame-based representation system. In RDF, these triples are known as *statements*.

Descriptions can be limited to one resource (for example, a library catalog card that names a document's author and publisher), in which case the values of resource attributes (or *properties*, as they are called in RDF) are typically strings. Descriptions can also span multiple resources: values of properties can be other resources—so we can thus describe arbitrary relationships between multiple resources. URIs name properties, which are also resources, and as such they can describe a property by asking, "What are a particular property's permitted values, which types of resources can it describe, and what is its relationship to other properties?" Meaning in RDF comes from specific terms and concepts that URIs define and then name. Because URIs can be unique, two systems can define some concept (say, "person") and each use a different URI to name it to avoid clashes; however, two systems agreeing on a common concept will use the same URI and effectively share semantics.

The RDF model also defines some meta-level constructs (such as container types for describing collections of resources) and higher-order statements (statements about other statements). Higher-order statements are modeled in RDF and allow the representation of modalities such as beliefs. Furthermore, an extensible, object-oriented type system (known as RDF Schema) is introduced as a layer on top of the basic RDF model.[5] The metaconstructs for the type system are terms and concepts that URIs name, so RDF effectively represents and defines classes and properties. Class definitions can be derived from multiple superclasses, and property definitions can specify domain and range constraints. We can also think of RDFS as a set of ontological modeling primitives on top of RDF.

Illustration by Sally Lee

## Syntax

RDF uses XML for the syntactic expression of model instances, which is a source of much controversy and confusion. RDF is essentially a data model and does not strive to replace XML. Instead, it builds a layer on top of it, making interoperable exchange of semantic information possible (for example, the object-oriented extensibility is intended to enable a partial understanding of data). RDF lacks primitive data types (such as integer, float, and so forth), so strings are essentially the only literals available; XML atomic datatypes will be used once W3C completes work on the XML Schema.[6]

We often hear claims such as, "You don't need RDF; you can do everything with XML." Sure, you could do that, but essentially you would end up reinventing the wheel by building a similar layer on top of XML that RDF already introduces. The XML developer community has focused on RDF's use (and abuse) of XML, sometimes forgetting that RDF is a data model whose syntax is largely irrelevant. (During development of the RDF standard, several syntaxes were proposed, some of which were not based on XML.)

RDF's object-oriented extensibility lets developers take pieces of existing RDF schemata and extend them as they see necessary. This might lead to some type of Darwinian evolution of metadata where the strong solutions will survive and evolve further (as opposed to the all-or-nothing situation that users of XML DTD [document type definitions] often face).

## Applications and future directions

In addition to the "syntax wars," controversy surrounds RDF within the knowledge representation community. RDF has been criticized for its low expressive power (it does not have variables, negation, or quantification—a far cry from first-order predicate calculus, for example). However, RDF is what it is by design. Sometimes you need to take baby steps before you can run—Web-related issues are important (such as simplicity, which enables wide adoption).

Several interesting RDF applications have already emerged. Mozilla (also known as Netscape 6) uses RDF internally as a representation format. In 1999, Netscape also introduced the RSS formalism (RDF Site Summary, www.egroups.com/group/rss-dev), which has now grown into a broader effort to build an extensible information description and syndication format.

Dublin Core is—at least initially—a metadata element set for describing cataloging information, such as that needed by digital libraries. The DC initiative early on embraced

## Correction

On page 30 of our September/October issue, the fifth sentence under the heading "IEA responses to some social and philosophical issues" called out the wrong figure. The sentence should read, "Similarly, as shown in *Figure 5*, on creative narratives, IEA scores agreed with highly trained expert grader scores as well as the latter agreed with each other." We apologize for this error, which occurred during final production of the issue.

RDF as the framework on which to build such metadata (www.dublincore.org).

RDF is also used as the basic building block for other W3C standards; for example, the Composite Capability/Preference Profile—or CC/PP—uses RDF to express profiles of mobile device characteristics to let servers better tailor content for these often restricted Web clients (www.w3.org/Mobile/CCPP).

As to what will happen in the future, RDF is playing an important role as a basis for the emerging DARPA Agent Markup Language (see the related essay by Hendler and McGuiness or visit www.daml.org), a large research program that will build more expressive layers of logic on top of the basic RDF framework. This is a large-scale effort to do real knowledge representation on the Web, and it is seen as a strong push toward building the semantic Web.

## References

1. O. Lassila and R. Swick, *Resource Description Framework (RDF) Model and Syntax Specification*, W3C Recommendation, World Wide Web Consortium, 1999; www.w3.org/TR/REC-rdf-syntax (current 6 Dec. 2000).

2. O. Lassila, *Introduction to RDF Metadata, W3C Note*, World Wide Web Consortium, 1997; www.w3.org/TR/NOTE-rdf-simple-intro (current 6 Dec. 2000).

3. O. Lassila, "Web Metadata: A Matter of Semantics," *IEEE Internet Computing*, vol. 2, no. 4, July/Aug. 1998, pp. 30–37.

4. T. Berners-Lee, R. Fielding, and L. Masinter, *Uniform Resource Identifiers (URI): Generic Syntax*, Internet Draft Standard RFC 2396, Aug. 1998; www.isi.edu/in-notes/rfc2396.txt (current 6 Dec. 2000).

5. D. Brickley and R.V.Guha, *Resource Description Framework (RDF) Schema Specification 1.0*, W3C Candidate Recommendation, World Wide Web Consortium, 2000; www.w3.org/TR/rdf-schema (current 6 Dec. 2000).

6. P.V. Biron and A. Malhotra, *XML Schema Part 2: Datatypes*, W3C Candidate Recommendation, World Wide Web Consortium, 2000; www.w3.org/TR/xmlschema-2 (current 6 Dec. 2000).

## FAQs on OIL: The Ontology Inference Layer

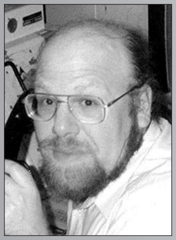
*Frank van Harmelen, Vrije Universiteit Amsterdam*
*Ian Horrocks, University of Manchester*

The Web is aimed at human readers. Machines are oblivious to its informational content: Web browsers, Web servers, and even search engines can't really distinguish weather forecasts from scientific papers, and they can't tell a personal home page from a major corporate Web site. This inability to process content seriously hampers the Web's functionality—computers are limited to transmitting and presenting information on it, and they can't really help us process that information. The semantic Web aims to create a Web in which both humans and machines can understand the information lurking in cyberspace. This, of course, requires representing information in such a way that its meaning (or "semantics") is machine-accessible. The Ontology Inference Layer is designed to be exactly such a

**Ora Lassila** is a research fellow at the Nokia Research Center and is a founder of the center's Agent Technology group. In his own research, he focuses on knowledge representation for the Web and its applications to ubiquitous computing. Previously, he held research positions at MIT, CMU, and Helsinki University of Technology. Contact him at ora.lassila@nokia.com.

**Frank van Harmelen** is a senior lecturer with the AI research group at the Vrije Universiteit, Amsterdam. His current interests include specification languages for KBS, using these languages for validation and verification of KBS, developing gradual notions of correctness for KBS, and verifying weakly structured data. He studied mathematics and computer science in Amsterdam and was awarded a PhD from the Department of AI in Edinburgh for his research on metalevel reasoning. Contact him at frank.van.harmelen@cs.vu.nl.

**Ian Horrocks** is a lecturer in computer science at the University of Manchester, UK. His research interests include knowledge representation; automated reasoning (particularly decision procedures for description and modal logics); ontological engineering, and optimizing, testing, and evaluating reasoning systems. Contact him at horrocks@cs.man.ac.uk.


**James Hendler** is a professor at the University of Maryland, where he has joint appointments in the Department of Computer Science, the Institute for Advanced Computer Studies, and the Institute for Systems Research; he is also an affiliate of the Electrical Engineering Department. He was the recipient of a 1995 Fulbright Foundation Fellowship, is a member of the US Air Force Science Advisory Board, and is a fellow of the AAAI. He is on leave from the university, working at the Defense Advanced Research Projects Agency as chief scientist of the Information Systems Office. Contact him at jhendler@darpa.mil.

**Deborah L. McGuinness** is the associate director and senior research scientist of the Knowledge Systems Laboratory at Stanford University. She has built and deployed numerous ontology environments and ontology applications, including some that have been in continuous use for over a decade at AT&T and Lucent. She is coauthor of the current ontology evolution environment from Stanford University and of one of the more widely used description logic systems—CLASSIC from Bell Laboratories. She is also coeditor of the recently released DARPA Agent Markup Language. She is on the advisory board for Ontology.Org and Powermarket, is on the executive council for the AAAI, and is on the executive steering board for the international organization for description logics, the Ontology Inference Layer effort. Contact her at dlm@ksl.stanford.edu.

```
class-def Product
slot-def Price
  domain Product
slot-def ManufacturedBy
  domain Product
class-def PrintingAndDigitalImagingProduct
  subclass-of Product
class-def HPProduct
  subclass-of Product
  slot-constraint ManufacturedBy
    has-value "Hewlett Packard"
class-def Printer
  subclass-of PrintingAndDigitalImagingProduct
slot-def PrinterTechnology
  domain Printer
slot-def Printing Speed
  domain Printer
slot-def PrintingResolution
  domain Printer
class-def PrinterForPersonalUse
  subclass-of Printer
class-def HPPrinter
  subclass-of HPProduct and Printer
class-def LaserJetPrinter
  subclass-of Printer
  slot-constraint PrintingTechnology
    has-value "Laser Jet"
class-def HPLaserJetPrinter
  subclass-of LaserJetPrinter and HPProduct
class-def HPLaserJet1100Series
  subclass-of HPLaserJetPrinter and
    PrinterForPersonalUse
  slot-constraint PrintingSpeed
    has-value "8 ppm"
  slot-constraint PrintingResolution
    has-value "600 dpi"
class-def HPLaserJet1100se
  subclass-of HPLaserJet1100Series
  slot-constraint Price
    has-value "$479"
class-def HPLaserJet1100xi
  subclass-of HPLaserJet1100Series
  slot-constraint Price
    has-value "$399"
```

Figure 2. A very simple example of an Ontology Interface Layer ontology.

representation of machine-accessible semantics of information on the Web.

## How is OIL trying to achieve this?

OIL synthesizes work from three different communities to achieve the ambitious aim of providing a general-purpose markup language for the semantic Web. It uses frame-based systems, description logics, and Web standards (XML and RDF).

*Frame-based languages* have a long history in AI. Their central modeling primitives are classes (known as *frames*) with properties (known as *slots*). A frame provides a context for modeling a class—which is generally defined as a subclass of one or more other classes—using slot-value pairs to specify additional constraints on instances of the new class. Many frame-based systems and languages with additional refinements of these modeling primitives have been developed, and, renamed as object orientation, they have been very successful in the software engineering community. OIL, based on a concept and the definition of their superclasses and slots, incorporates the essential modeling primitives of frame-based systems. It also treats slots as first-class objects that can have their own properties (such as domain and range) and be arranged in a hierarchy.

*Description logics* have been developed in knowledge-representation research, and they describe knowledge in terms of concepts (comparable to classes or frames) and roles (comparable to slots in frame systems). DLs have well-understood theoretical properties. In addition, the meaning of any expression in a DL can be described in a mathematically precise way, which enables reasoning with concept descriptions and the automatic derivation of classification taxonomies. There are now efficient implementations of DL reasoners that can perform these tasks. OIL inherits from DLs both their formal semantics and efficient reasoning support.

Besides modeling primitives (which frame systems provide) and their semantics (which description logics provide), we have to decide about the syntax of a markup language for the semantic Web. Any such syntax must be formulated using existing W3C standards for information representation. As a possible candidate, OIL has a well-defined syntax in XML based on a document type definition (DTD) and an XML schema definition. Second, OIL is defined as an extension of the RDF and its schema definition language RDFS. RDFS provides two important contributions: a standard set of modeling primitives such as instance-of and subclass-of relationships and a standardized syntax for writing such writing class hierarchies. OIL extends this approach to a full-blown modeling language.

## What does OIL look like?

Figure 2 gives a very simple example of an OIL ontology (it illustrates only the most basic constructs).

Figure 2 also defines a number of classes and organizes them in a class hierarchy (for example, `HPProduct` is a subclass of `Product`). Various properties (slots) are defined, together with the classes to which they apply (for example, `a Price` is a property of any `Product`, but a `PrintingResolution` can only be stated for a `Printer` (an indirect subclass of `Product`). For certain classes, these properties have restricted values (for example, the `Price` of any `HPLaserJet1100se` is restricted to $479). In OIL, classes can also be combined using logical expressions—for example, an `HPPrinter` is both an `HPProduct` and a `Printer` (and consequently inherits the properties from both these classes).

## What does the acronym OIL mean?

There are at least two possible meanings of the acronym—Ontology Inference Layer or Ontology Interchange Language—but all contain the word *ontology*. An ontology is a consensual, shared, and formal description of the important concepts in a given domain. Typically, an ontology identifies classes of objects that are important in a domain and organizes these classes in a subclass hierarchy. Each class is characterized by properties that all elements in that class share. Important relations between classes or elements of these classes are also part of an ontology. Ontologies are now an important notion in diverse areas such as knowledge representation, natural language processing, information retrieval, databases, knowledge management, and multiagent systems. They are widely considered to be a crucial ingredient for the semantic Web's infrastructure.

## Which applications will OIL enable?

Machine-processable representations of ontologies will be crucial to many applications of the semantic Web. We briefly mention a few:

- *Search engines*. Current search engines are seriously limited by their reliance on keyword matching. They can't find relevant information that is described in different terms, they often return information that uses the same words with a different meaning, and they can't combine information from multiple sources. We can alleviate these problems with search engines that search the semantic concepts underlying the information in Web pages rather than searching for matching keywords.
- *E-commerce*. Currently, consumers can only compare online shops by visiting each shop and doing the comparison. So-called "shopbots" that try to perform this task do this by *screen scraping*: retrieving the information by interpreting regularities in the layout of the Web pages of the various shops. They typically only retrieve limited information from the various shops (such as the price) and ignore information such as shipping conditions (which are harder to retrieve). In addition, they are cumbersome to construct and hard to maintain (they must be updated every time a Web shop changes the layout of its pages). Comparison shopping will become a reality only when Web shops offer their catalogs in machine-processable formats, with links to explicit and shared ontologies that agents can use to construct mappings between these catalogs.
- *Knowledge management*. An increasing number of companies rely on intranet technology as a knowledge repository for their employees. Traditional document-management systems provide insufficient means to structure and access the knowledge in such a repository. Explicit ontologies are the most promising technical vehicle for transforming document repositories into proper knowledge repositories.

## What are the design principles behind OIL?

The design of OIL was motivated by a desire to

- maximize compatibility with existing W3C standards—XML and RDF;
- maximize partial interpretability by less semantically aware processors;
- provide modeling primitives that have proven useful for large user communities;
- maximize expressiveness to model a wide variety of ontologies;
- provide a formal semantics (a mathematically precise description of the meaning of every expression) to facilitate machine interpretation of that semantics; and
- enable sound, complete, and efficient reasoning services by limiting the expressiveness of the language.

## What tools are available?

Ontology editors help human knowledge engineers develop and maintain ontologies. They support the definition and modification of concepts, slots, axioms, and constraints and enable the inspection, browsing, and codifying of the resulting ontologies. Currently, two editors for OIL are available and a third is under development:

- OntoEdit (http://ontoserver.aifb.uni-karlsruhe.de/ontoedit), developed at the Knowledge Management Group of the AIFB Institute at the University of Karlsruhe;
- OILedit (http://img.cs.man.ac.uk/oil), a freely available and customized editor for OIL, developed by the University of Manchester; and
- Protégé (www.smi.stanford.edu/projects/protege), an ontology editor built at the University of Stanford. Currently it supports only RDF, but work is starting on extending Protégé to OIL.

Inference engines can be used to reason about ontologies, helping build and use them for advanced information access and navigation. OIL uses the FaCT system (*fa*st *c*lassification of *t*erminologies, www.cs.man.ac.uk/fact) to provide reasoning support for ontology design, integration, and verification. FaCT is heavily optimized to deal with very large ontologies. It can check the consistency of thousands of classes and automatically derive their underlying class hierarchy in a matter of seconds, running on standard desktop hardware.

## How does OIL relate to RDF and RDFS?

The example in Figure 2 was stated in OIL's presentation syntax, which is intended for human readers and writers of OIL ontologies. For machines, OIL uses RDF as its syntax. OIL exploits as much as possible the modeling primitives of RDFS. This provides crucial backward compatibility, allowing OIL ontologies to be treated as extensions of RDF and RDFS ontologies and making OIL ontologies available not only to OIL aware applications, but also to applications that are only RDF-aware. Such RDF-aware applications can still process and reason with significant portions of OIL-ontologies. For illustration purposes, the last class of the example in Figure 2 in RDF syntax would look like the code presented in Figure 3.

```
<rdfs:Class rdf:ID=”HPLaserJet1100xi”>
  <rdfs:subClassOf rdf:resource=”#HPLaserJet1100Series”/>
  <oil:hasPropertyRestriction>
    <oil:HasValue>
        <oil:onProperty rdf:resource=”#Price”/>
        <oil:toConcreteType> 399 </oil:toConcreteType>>
    </oil:HasValue>
  </oil:hasPropertyRestriction>
</rdfs:Class>
```

Figure 3. The last class of the example in Figure 2 in Resourse Description Framework syntax.

To a program that is only RDF-aware (not OIL-aware), this would still be interpretable as saying that the 1100xi printers are a special type of the 1100 Series printers. The specific restriction that the 1100xi costs $399 would only be available to OIL-aware programs.

### How is OIL different from DAML?

The DAML language inherits many aspects from OIL, and the capabilities of the two languages are relatively similar. Both

- support hierarchies of classes and properties based on subclass and subproperty relations;
- allow classes to be built from other classes using arbitrary combinations of intersection (`AND`), union (`OR`), and complement (`NOT`);
- allow the domain, range, and cardinality of properties to be restricted;
- support transitive and inverse properties; and
- support concrete data types (integers, strings, and so forth).

However, there are also some important differences, which we only briefly discuss here. First, OIL achieves a greater backward compatibility with RDFS than DAML. Second, OIL was designed to enable reasoning services that are sound and complete as well as efficient. Some constructions in DAML make similar reasoning services impossible. Third, OIL can state either sufficient conditions for a class or conditions that are both sufficient and necessary. This last option makes it possible to perform automatic classification. Given a specific object in a domain, OIL can automatically decide to which classes this object belongs. In DAML, this distinction is not as well developed.

### Will OIL be a one-size-fits-all?

It is unlikely that a single ontology language can fulfill all the needs of the semantic Web's large range of users and applications. We have therefore organized OIL as a series of ever-increasing layers of sublanguages. Each additional layer will add functionality and complexity to the previous layer, so that agents (humans or machines) who can only process a lower layer can still partially understand ontologies expressed in any of the higher layers. A first and very important application of this principle is the relation between OIL and RDFS. Core OIL coincides largely with RDFS (with the exception of RDFS's reification features). This means that even simple RDFS agents can process the OIL ontologies and pick up as much of their meaning as possible with their limited capabilities.

***It is unlikely that a single ontology language can fulfill all the needs of the semantic Web's large range of users and applications.***

### Where can I find out more about OIL?

The European Union IST program for Information Society Technologies funds the OIL initiative under the On-To-Knowledge project (IST-1999-1013) and Ibrow (IST-1999-19005). OIL's homepage (www.ontoknowledge.org/oil) provides definitions of the OIL syntax, papers and presentations explaining OIL (ranging from the very introductory to the very formal), case studies using OIL, and tools that have been developed for OIL.

## The DARPA Agent Markup Language

*James Hendler, DARPA/ISO*
*Deborah L. McGuinness, Stanford University*

The DARPA Agent Markup Language (DAML) program is a US Government-sponsored endeavor aimed at providing the foundation for the next Web evolution—the semantic Web. The program is funding critical research to develop languages, tools, and techniques that will make considerably more Web content machine-understandable. This should lead to the next major generation of Web technology and enable considerably more machine-to-machine (agent-based) communication. Academic researchers, government agencies, software development companies, and industrial organizations such as the World Wide Web Consortium are participating in the program. The DAML project is also working closely with other efforts, including European Union-funded semantic Web projects such as On-To-Knowledge (www.ontoknowledge.org) and Ibrow (www.swi.psy.uva.nl/projects/ibrow/home.html) and the ongoing W3C RDF recommendation effort (www.w3.org/TR/REC-rdf-syntax).

### Motivation

The modern information technology world is a dynamically changing environment with an exponentially increasing ability to create and publish data that rapidly swamps human abilities to process that data into usable information. Agent-based computing can potentially help us recognize complex patterns in this widely distributed, heterogeneous, uncertain information environment. Unfortunately, this potential is hampered by the difficulty agents face in understanding and interacting with data that is either unprocessed or in natural languages. The inability of agents to understand the concepts on a Web page, their difficulty in handling the semantics inherent in a program's outputs, and the complexity of fusing information concepts from the outputs of sensors—to name but a few problems—truly keep the "agent revolution" from occurring.

One potential solution is for humans to meet the computer half way. By using tools to provide marked annotations attached to data sources, we can make information available to the agents in new and exciting ways. Beyond XML, the program's goal is to develop a language aimed at representing semantic relations in machine-readable ways compatible with current and future Internet technologies. Prototype tools are being developed to show the potential of such markups to provide revolutionary capabilities that will change the way humans interact with information. Deploying such tools to military and intelligence users and showing the incredible dual-use potential of such a technology caps off the program's goals.

### Description

To realize this solution, Internet markup languages must move beyond the implicit semantic agreements inherent in XML and community-specific controlled languages and move toward making semantic entities and markup a primary goal. To this end, DARPA is working with numerous partners and communities to create an eventual Web-standard semantic language and demonstrate the utility of such a language. We are doing this by developing an example language—DAML—and sample tools and applications. DAML will tie a page's information to machine-readable semantics (specifications of term meanings stored in ontologies) and eventually provide a logical language embedded on the Web. It will let communities extend simple ontologies for their own use, so that they can use a bottom-up design for meaning while sharing higher-level concepts.

In addition, DAML will provide mechanisms for explicitly representing services, processes, and business models, so that humans and programs running on the Web can recognize and understand nonexplicit information (such as that encapsulated in programs or sensors). Eventually, it will also supply a mechanism by which logical statements and proofs can become first-class Web entities, allowing a new set of capabilities in machine-to-machine communication. This will enable the development of a wide new range of software tools for industrial and government applications, with diverse uses ranging from business-to-business e-commerce to government efforts in combating the spread of weapons of mass destruction.

DAML will provide a number of advantages over current markup approaches. It will enable semantic interoperability—instead of enabling only syntactic interoperability as is done in XML. We will also be able to mark objects on the Web (manually or automatically) to include descriptions of information they encode, functions they provide, and data they can produce. Agents that use DAML will be able to link together Web pages, databases, programs, models, and sensors to recognize the concepts they need. Thus, Web-based information fusion from diverse sources will become a reality.

### Status report

DARPA kicked off its DAML program in the summer of 2000, and research is ongoing. It's delivering the language in two portions—an ontology language (DAML-ONT) and later DAML-Logic. It released DAML-ONT version 0.5 on 5 October 2000, and it maintains an ongoing discussion of the language and logic issues on the mailing list www-rrdf-logic@w3.org—archived at http://lists.w3.org/Archives/Public/www-rdf-logic, with the goal of reaching a more stable version.

The language extends W3C's Resource Description Framework[1,2] and its associated object-oriented type system.[3] It aims to add expressive power suited to agent and service interoperation. You can find the DAML-ONT specification and supporting documents at www.daml.org. Also available are early versions of DAML support tools and pointers to various Web resources marked up with DAML.

DAML-ONT aims to capture the commonly used modeling primitives used in object-oriented modeling, frame systems, and conceptual schemas and include them in an integrated language for the Web. It attempts to join the ease of modeling in frame systems, the ubiquity of the Web, and the formal foundations of knowledge representation in description logics to provide a sound language for representing and reasoning with term meanings. Currently, it is being used to mark up project pages used in the DAML project and to facilitate services and applications work. In addition, several other communities (including the OIL project described in the previous essay) are developing tools to translate markup to the DAML-ONT language and to provide pages in the DAML repository (www.daml.org/ontologies).

The project's next goal is to create an early version of a logic language—DAML-Logic. We expect DAML-Logic to include both a language for expressing constraints in DAML-ONT and for adding inference rules to the language. The first version should come out in the Spring of 2001. In addition, work continues on the evolution of DAML-ONT. We've published documents showing its intended meaning (http://www.ksl.stanford.edu/people/dlm/daml-semantics and http://www.cs.man.ac.uk/~horrocks/DAML-OIL/semantics.html)[4] and how it maps to current research in semantic Web languages such as OIL,[5] SHOE (www.cs.umd.edu/projects/plus/SHOE),[6] and KIF (http://logic.stanford.edu/kif). We're also exploring how it maps to emerging Web languages (such as XML and RDF) and to ongoing efforts in agent-based systems, particularly the FIPA standardization effort (www.fipa.org).

We're also trying to encourage use from a broad spectrum of users. Sometimes, users will use DAML as a unifying language for stating explicit information and won't connect to deductive engines that look for logical implications of the implicit information. Other times, users will connect to more or less complete inference engines that infer the logical completion of all the statements. This provides opportunities for a spectrum of users and does not require users to have extensive computational resources connected to their systems. Ultimately, the language should express the meaning of information on Web pages or in applications and lead to a Web-standard language for expressing semantic content. ■

### Acknowledgments

The US Defense Advanced Research Projects Agency funds the DAML project. The program is the result of many people's work, and the evolving list of major contributors is maintained on www.daml.org.

### References

1. O. Lassila and R. Swick, *Resource Description Framework (RDF) Model and Syntax Specification*, W3C Recommendation, World Wide Web Consortium, 1999; www.w3.org/TR/REC-rdf-syntax (current 6 Dec. 2000).
2. O. Lassila, "Web Metadata: A Matter of Semantics," *IEEE Internet Computing*, vol. 2, no. 4, July/Aug. 1998, pp. 30–37.
3. D. Brickley and R.V. Guha, *Resource Description Framework (RDF) Schema Specification 1.0*, W3C Candidate Recommendation, World Wide Web Consortium, 2000; www.w3.org/TR/rdf-schema (current 6 Dec. 2000).
4. R. Fikes and D.L. McGuinness, "An Axiomatic Semantics for DAML-ONT," 10 Dec. 2000, www.ksl.stanford.edu/people/dlm/daml-semantics (current 6 Dec. 2000).
5. S. Bechhofer et al., "An Informal Description of OIL-Core and Standard OIL: A Layered Proposal for DAML-O," www.ontoknowledge.org/oil/downl/dialects.pdf (current 6 Dec. 2000).
6. J. Heflin and J. Hendler, "Semantic Interoperability on the Web," *Proc. Extreme Markup Languages 2000*, Graphic Communications Assoc., Montreal, 2000, pp. 111–120.

## Next in Trends & Controversies

## Ontologies and Electronic Commerce