Feature

OWL Exports from a Full Thesaurus

by Jay ven Eman

Jay ven Eman is CEO of Access Innovations in Albuquerque, New Mexico. He can be reached by email: j_ven_eman at accessinn.com

What do you make of "198"? You could assume a number. Computer applications make no reliable assumptions since it could be an integer and decimal but not octal, but it could also be something else, too. Neither you nor the computer could do anything useful with it. What if we added a period so "198" becomes "1.98"? Maybe it represents the value of something such as its price. If we found it embedded with additional information, we would know more. "It cost 1.98." The reader now knows that it is a price, but software applications still are unable to figure it out. There is much the reader still doesn't know. "It cost ¥1.98." "It cost £1.98." "It cost $1.98." There is even more information you would want. Wholesale? Retail? Discounted? Sale price? 1.98 for what?

Basic interpretation is something humans do very well, but software applications do not. Now imagine a software application trying to find the nearest gasoline station to your present location that has gas for $1.98 or less. Per gallon? Per liter? Diesel or regular? Such a request is theoretically possible using your location from your car's GPS and a wireless Internet connection, but it is beyond the most sophisticated software applications using Web resources. They cannot do the reasoning based upon the current state of information on the Web.

Trying to search the Web based upon conceptual search statements adds more complications. Looking for information about "lead" using just that term returns a mountain of unwanted information about leadership, your water and conditions at the Arctic Ocean. Refining the query to indicate your interest in "lead based soldering compounds" helps. Software applications still cannot reason or draw inferences from keywords found in context. At present, only humans are adept at interpreting within context.

Semantic Web

The Semantic Web is a series of initiatives to help make more of the vast resources found via the Web available to software applications and agents, so that these programs can perform at least rudimentary analysis and processing to help you find that cheaper gasoline. The Web Ontology Language (OWL) is one such initiative and will be described here in relation to thesauri and taxonomies.

At the heart of the Semantic Web are words and phrases that represent concepts that can be used for describing Web resources. Basic organizing principles for "concepts" exist in the present thesaurus standards (ANSI/NISO Z39.19 at www.niso.org and ISO 2788 and ISO 5964 at www.iso.org). They are being expanded and revised. Drafts of the revisions are available for review.

The reader is directed to the standards websites referenced above and to www.accessinn.com, www.dataharmony.com and www.willpowerinfo.co.uk/thesprin.htm for basic information on thesaurus and taxonomy concepts. It is assumed here that the reader will have a basic understanding of what a thesaurus is, what a taxonomy is and related concepts. Also, a basic understanding of the Web Ontology Language (OWL) is required. OWL is a W3C recommendation and is maintained at the W3C Web site. For an initial investigation of OWL, the best place to start is the guide found at www.w3.org/TR/ 2004/Rec-owl-guide-20040210/.

OWL

From the OWL guide, "OWL is intended to provide a language that can be used to describe the classes and relations between them that are inherent in Web documents and applications." OWL formalizes a domain by defining classes and properties of those classes; defining individuals and asserting properties about them; and reasoning about these classes and individuals. Ontology is borrowed from philosophy. In philosophy, ontology is the science of describing the kinds of entities in the world and how they relate.

An OWL ontology may include classes, properties and instances. Unlike ontology from philosophy, an OWL ontology includes instances – or members – of classes. Classes and members – or instances – can have properties and those properties have values. A class can also be a member of another class. OWL ontologies are meant to be distributed across the Web and to be related as needed. The normative OWL exchange syntax is RDF/XML (www.w3.org/RDF/).

Thesaurus

A thesaurus is not an ontology. It does not describe kinds of entities and how they are related in a way that a software agent could use. One could draw useful inferences about the domain of medicine by studying a medical thesaurus, but software cannot. You would discover important terms in the field, how terms are related, what terms have broader concepts and what terms encompass narrower concepts. An inference, or reasoning engine, would be unable to draw any inferences beyond a basic "broader term/narrower term" pairing like "nervous system/central nervous system," unless specifically articulated. Is it a whole/part, instance, parent/child or other kind of relationship?

Using OWL, more information about the classes represented by thesauri terms, the relationship between classes, subclasses and members can be described. In a typical thesaurus, the terms *nervous system* and *central nervous system* would have the labels BT and NT, respectfully. A software agent would not be able to make use of these labels and the relationship they describe unless the agent is custom coded. The purpose of OWL is to provide descriptive information using RDF/ XML syntax that would allow OWL parsers and inference engines, particularly those not within the control of the owners of the target thesaurus, to use the incredible intellectual value contained in a well developed thesaurus.

The levels of abstraction should be apparent at this point. At one level there are terms. At another level the relationships between groups of terms are described within a thesaurus structure. The thesauri standards do not dictate how to label thesaurus relationships. A term could be "USE Agriculture" or "Preferred Term Agriculture" or "PT Agriculture." Hard coding of software agents with all of the possible variations of thesaurus labels is impractical.

OWL then is used to describe labels such as BT (broader term), NT (narrower term), NPT (non-preferred term) and RT (related term) and to describe additional properties about classes and members such as the type of BT/NT relationship between two terms. Additional power can be derived when two or more thesauri OWL ontologies are mapped. This mapping would allow Web software agents to determine the meaning of subject terms (keywords) found in the metadata element of Web pages, to determine if other Web pages containing the same terms have the same meaning and to make additional inferences about those Web resources.

An OWL output from a full thesaurus provides semantic meaning to the basic classes and properties of a thesaurus. Such an output becomes a true Web resource and can be used more effectively by automated processes. Another layer of OWL wrapped around subject terms from an OWL-level thesaurus and the resources (such as Web pages) these subject terms are describing would be an order of magnitude more powerful, but also more complicated and difficult to implement.

OWL Thesaurus Output

An OWL thesaurus output contains two major parts. The first part articulates the basic definition of the structure of the thesaurus. It is an XML/RDF schema. As such, a software agent can use the resolving properties in the schema to locate resources that provide the necessary logic needed to use the thesaurus.

Without agonizing over the details, Figure 1 provides the

FIGURE 1 XML/RDF/OWL DECLARATIONS

```
<!DOCTYPE rdf:RDF [
<!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
<!ENTITY owl "http://www.w3.org/2002/07/owl#" >
<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" > ]>

<rdf:RDF
xmlns ="http://localhost/owlfiles/DHProject#"
xmlns:DHProject ="http://localhost/owlfiles/DHProject#"
xmlns:base ="http://localhost/owlfiles/DHProject#"
xmlns:owl ="http://www.w3.org/2002/07/owl#"
xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:xsd ="http://www.w3.org/2001/XMLSchema#">

<owl:Ontology rdf:about="">
<rdfs:comment>OWL export from MAIstro</rdfs:comment>
<rdfs:label>DHProject Ontology</rdfs:label>
</owl:Ontology>
```

FIGURE 2 SAMPLE TERM RECORD OUTPUT IN XML

```
<TermInfo>
<T>Agrotechnology</T>
<BT>Biotechnology</BT>
<NT>Animal management technologies</NT>
<NT>Controlled environment agriculture</NT>
<NT>Genetically modified crops</NT>
<RT>Agricultural science</RT>
<RT>Food technology</RT>
<UF>Plant engineering</UF>
<Scope_Note></Scope_Note>
<Editorial_Note></Editorial_Note>
<Facet></Facet>
<History></History>
</TermInfo>
```

necessary declarations in the form of URLs so that software agents can locate additional resources related to this thesaurus. The software agent would not have to have any of the W3C recommendations (XML, RDF, OWL) hard coded into its internal logic. It would have to have resolving logic such as, "if you encountered a URL, then do the following..."

Figure 2 shows a sample thesaurus term record output in XML for the term, *agrotechnology*. This term has BT, NT, RT, Status, UF, Scope_Note, Editorial_Note, Facet and History as a complex combination of classes, members and properties. Anyone familiar with thesauri can determine what the abbreviations such as BT, NT and RT mean, and, thus, they can infer the relationships among all of the terms in the term record. An OWL thesaurus output provides additional intelligence that helps software make the same inferences.

After the declarations portion, shown in Figure 1, the remaining portion of the first part of an OWL thesaurus output is the schema describing the classes, subclasses and members that comprise a thesaurus and all of the properties of each. Each of the XML elements (e.g., <RT>) in Figure 2 is defined in the schema, as are their properties and relationships. These definitions conform to the OWL W3C recommendation.

The first part of an OWL thesaurus output contains declarations and classes, subclasses and their properties. It contains all of the logic needed by a specialized agent to make sense of your thesaurus and other OWL thesaurus resources on the Web.

FIGURE 3 OWL Output of Term Record *Agrotechnology*

```
</PreferredTerm>
<PreferredTerm rdf:ID="T131">
        <rdfs:label xml:lang="en">Agrotechnology</rdfs:label>
<BroaderTerm rdf:resource="#T603" newsindexer:alpha="Biotechnology"/>
<NarrowerTerm rdf:resource="#T252" newsindexer:alpha="Animal management technologies"/>
<NarrowerTerm rdf:resource="#T1221" newsindexer:alpha="Controlled environment agriculture"/>
<NarrowerTerm rdf:resource="#T2166" newsindexer:alpha="Genetically modified crops"/>
<Related_Term rdf:resource="#T127" newsindexer:alpha="Agricultural science"/>
<Related_Term rdf:resource="#T2020" newsindexer:alpha="Food technology"/>
<Non-Preferred_Term rdf:resource="#T3898" newsindexer:alpha="Plant engineering"/>
</PreferredTerm>
```

Since it is designed to be distributed and referenced, a given base OWL thesaurus can grow as other thesaurus ontologies reference it.

The second part of an OWL thesaurus contains the terms of your thesaurus marked up according to the OWL recommendation. Figure 3 shows an OWL output for our sample term, *agrotechnology*. (Note: Since there are no values found in Figure 2 for Scope_ Note, Editorial_Note, Facet and History, these elements are not present in Figure 3.)

Now our infamous software agent could infer that *agrotechnology* is a Narrower-Term of *biotechnology. Agrotechnology* has three NarrowerTerms, two RelatedTerms and one NonPreferredTerm. From the OWL output, the software agent can resolve the meaning and use of BroaderTerm, NarrowerTerm, RelatedTerm and NonPreferredTerm by navigating to the various URLs. The agent can determine from the schema dictates that if a term has property value, NarrowerTerm, then it must have property type value, BroaderTerm. A term cannot be a narrower term, if it doesn't have a broader term. A term that is a BroaderTerm must also be a PreferredTerm and so on.

FIGURE 4 **OWL Output of Term Record *Plant engineering***

```
<NonPreferredTerm rdf:ID="T3898">
        <rdfs:label xml:lang="en">Plant engineering</rdfs:label>
        <USE rdf:resource="T131" newsindexer:alpha="Agrotechnology"/>
</NonPreferredTerm>
```

FIGURE 5 **OWL Output of Term Record *Machine-aided Indexing***

```
<PreferredTerm rdf:ID="T131">
     <rdfs:label xml:lang="en">Machine aided indexing</rdfs:label>
<BroaderTerm rdf:resource="#T603" newsindexer:alpha="Information technology"/>
<NarrowerTerm rdf:resource="#T1221" newsindexer:alpha="Concept extraction"/>
<NarrowerTerm rdf:resource="#T2166" newsindexer:alpha="Rule base techniques"/>
<Related_Term rdf:resource="#T127" newsindexer:alpha="Categorization systems"/>
<Related_Term rdf:resource="#T2020" newsindexer:alpha="Classification systems"/>
     <Non-Preferred_Term rdf:resource="#T3898" newsindexer:alpha="MAI"/>
</PreferredTerm>
```

Our thesaurus software agent can infer from Figure 3 that the thesaurus it is evaluating uses *agrotechnology* for *plant engineering*. Figure 4 identifies *plant engineering* as a NonPreferredTerm and identifies *agrotechnology* as the PreferredTerm. (The logic in the schema dictates that if you have a NonPreferredTerm, then it must have a PreferredTerm.)

Suppose our software agent encounters *plant engineering* at another website and uses it to locate resources there. Now the agent locates your website. The agent would first use *plant engineering*. From your OWL thesaurus output it would infer that at your site it should use your preferred term, *agrotechnology*, to locate similar resources.

All the terms and terms relationship in your thesaurus or taxonomy would be defined in part two of the OWL thesaurus output. It is now a Web resource that can be used by software agents. Since it is designed to be distributed and referenced, a given base OWL thesaurus can grow as other thesaurus ontologies reference it.

More Meaning Needed

Even a thesaurus wrapped in OWL falls short of the full potential of the Semantic Web. This first order output allows other thesaurus applications to make inferences about classes, subclasses and members of a thesaurus. By reading the OWL wrappings, any thesaurus OWL software agent can make useful infers. By using classes, subclasses, and members and their properties, Web software agents would be able to reproduce the hierarchical structure of a thesaurus outside of the application used to construct it.

However, a lot is still missing. For example, knowing a term's parent, children, other terms it is related to and terms it is used for does not tell you what the term means and what it might be trying to describe. Additional classes, subclasses and members – all with properties – are needed. How a term is supposed to be used and why one term is preferred over another would be enormously useful properties for improving the performance of software agents.

A more difficult layer of semantic meaning is the relationship between a thesaurus term and the entity, or object, it describes. An assignable thesaurus term is a member of class PreferredTerm. When it is assigned to an object, for example a research report or Web page, that term becomes a property of that object. For a Web page, descriptive terms become attributes of the Meta element:

```
<META NAME="KEYWORDS" CONTENT="content management software, xml thesaurus, concept extraction, information retrieval, knowledge extraction, machine-aided indexing, taxonomy management system, text management, xml">
```

None of the intelligence found in an OWL thesaurus output is found in the Meta element. Having that intelligence improves the likelihood that our software agent can make useful inferences about this Web resource.

This intelligence is not currently available because HTML does not allow for OWL markup of keywords in the Meta element. There are major challenges to doing this. To illustrate, the single keyword, *machine-aided indexing*, is rendered in Figure 5 as an OWL thesaurus output. This rendering is a very heavy overhead.

The entire rendering depicted in Figure 5 would not be necessary for each keyword assigned to the Meta element of a Web page. A shorthand version could be designed that would direct software agents to the OWL thesaurus output, but such a shorthand method is not available.

Even if HTML incorporates a shorthand OWL markup for Meta keywords, the intelligence required to apply the right keywords automatically, for example, making the determination, "Web page x is about machine-aided indexing," is not in the current OWL output. Automatic or semiautomatic indexing is the only way to handle volume and variety, especially dealing with Web pages.

Commercial applications such as Data Harmony's M.A.I. Concept Extractor and similar products provide machine-automated indexing solutions. Theoretically, the knowledge representation systems that drive machine-automated indexing and classification systems could incorporate OWL markup. When a machine indexing system assigned a preferred term to a Web page, it would write it into the Meta element along with its OWL markup.

However, to truly achieve the objectives of the Semantic Web the OWL W3C recommendation should be extended to include the decision algorithms used in the machine-automated indexing process. Or alternative W3C recommendations regarding the Semantic Web should be used in conjunction with OWL. If this enhancement could be accomplished, software agents could determine the logic used in assigning terms. Then the agent could compare the logic used at other Web sites and be able to make comparisons and draw conclusions about various Web resources – conclusions like, "Of the 18 websites your software agent reviewed that discussed selling gasoline, only eight were actual gas stations and only four of the eight had data the agent could determine was the retail price for unleaded."

We have moved closer to locating the least expensive gasoline within a five-mile radius of our current location. What has been described here is actually being done, but so far only in closed environments where all of the variables are controlled. For example, there are websites that specialize in price-comparison shopping.

Beyond these special cases, for the open Web the challenges are great. The sheer size of the Web and its speed of growth are obvious. More challenging is capturing meaning in knowledge representation systems like OWL and other Semantic Web initiatives at W3C such as SKOS and Topic Maps. How many OWL thesauri will there be? How many are needed? How much horsepower will be needed for an agent to resolve meaning when OWL thesauri are cross-referencing each other in potentially endless loops?

For these and other reasons, the Semantic Web may not live up to its full promise. The complexity and the magnitude of the effort may prove to be insurmountable. That said, there will be a Semantic Web and OWL will play an important role, but it will probably be a more simplified semantic architecture and more isolated, for example, to vertical markets or specific fields and disciplines.

For the reader, before you launch your own initiatives, assess your internal resources and measure the level of internal commitment, particularly at the upper levels of your organization. Know what is happening in your industry or field. If semantic initiatives are happening in your industry, then the effort needed to deploy a taxonomic strategy (OWL being one piece of the solution) should be seriously considered. If you don't make the effort, your Web resources and your vast internal, private resources risk being lost in the sea of meaninglessness, putting you at a tremendous competitive disadvantage.