Ontologies Come of Age

Deborah L. McGuinness

Associate Director and Senior Research Scientist

Knowledge Systems Laboratory

Stanford University

Stanford, CA 94305

dlm@ksl.stanford.edu

Abstract

Ontologies have moved beyond the domains of library science, philosophy, and knowledge representation. They are now the concerns of marketing departments, CEOs, and mainstream business. Research analyst companies such as Forrester Research report on the critical roles of ontologies in support of browsing and search for e-commerce and in support of interoperability for facilitation of knowledge management and configuration. One now sees ontologies used as central controlled vocabularies that are integrated into catalogues, databases, web publications, knowledge management applications, etc. Large ontologies are essential components in many online applications including search (such as Yahoo and Lycos), e-commerce (such as Amazon and eBay), configuration (such as Dell and PC-Order), etc. One also sees ontologies that have long life spans, sometimes in multiple projects (such as UMLS, SIC codes, etc.). Such diverse usage generates many implications for ontology environments.

In this paper, we will discuss ontologies and requirements in their current instantiations on the web today. We will describe some desirable properties of ontologies. We will also discuss how both simple and complex ontologies are being and may be used to support varied applications. We will conclude with a discussion of emerging trends in ontologies and their environments and briefly mention our evolving ontology evolution environment.

Introduction: The web's growing needs

We may be poised for the next major evolution of online environments. In the early days of the web, HTML pages were generated by hand. The pages contained information about how to present information on a page. Early adopters took to the web quickly since it provided a convenient method for information sharing. Arguably, the generation of tools for machine generation and management of web pages allowed the web to really take off. Tool platforms allowed non-technical people to generate and publish web pages quickly and easily. The resulting pages typically included content and display information and targeted human readers (rather than targeting programs or automatic readers).

The web continues to grow at an astounding rate with web pages ubiquitously integrated into many aspects of business and personal life. However, web pages still preserve much of their character of being aimed at human consumption. Thus, applications such as search still require humans to review results pages in order to find the right answer to their queries. While search engine advances such as Google [Google 2000] improve the situation, most people agree that finding the exact information one is seeking on the

Deborah L. McGuinness. "Ontologies Come of Age". To appear in Dieter Fensel, Jim Hendler, Henry Lieberman, and Wolfgang Wahlster, editors. The Semantic Web: Why, What, and How, MIT Press, 2001.

web today is not as easy as one would hope. One reason for this is that answers to search queries typically are a rank ordered list of pages that may contain the answer to the query. The answers rarely are just the portion of the page that the search engine "thought" contained the answer to the query. Additionally, web pages typically do not contain markup information about the contents of the page. If pages were marked up with information concerning what information or services could be obtained (and how that information or service could be obtained), then a page could be used more effectively by programs to return the portion of the page (or the answer from a service) that contains a specific answer to a question. Once web pages are aimed for machine or program consumption, instead of human consumption, the next generation of the web can be realized. The proliferation of markup languages aimed at marking up content and services instead of just presentation information can be viewed as support for this position. Markup languages such as XML [XML 2000], RDF [Lassila 1998, Lassila-Swick 1999], RDFS [Brickley-Guha 2000], DAML [Hendler-McGuinness 2000], etc are becoming more accepted as users and application developers see the need for more understanding of what is available from web pages.

The view presented in this paper is consistent with the vision being put forward by Tim Berners-Lee of the W3C consortium. In a widely cited presentation [Berners-Lee 2000] at XML 2000 conference, Berners-Lee presented his vision of the semantic web as being machine processable. We support this view as well. We believe that the next web evolution requires machines to understand the content of pages – both what can be obtained from pages and what that information means. Markup languages allow specification of this information. Berners-Lee offered an architecture diagram[1] in his presentation that provides a nice foundation. We include it here in Figure 1.

[1] http://www.w3.org/2000/Talks/1206-xml2k-tbl/slide10-0.html

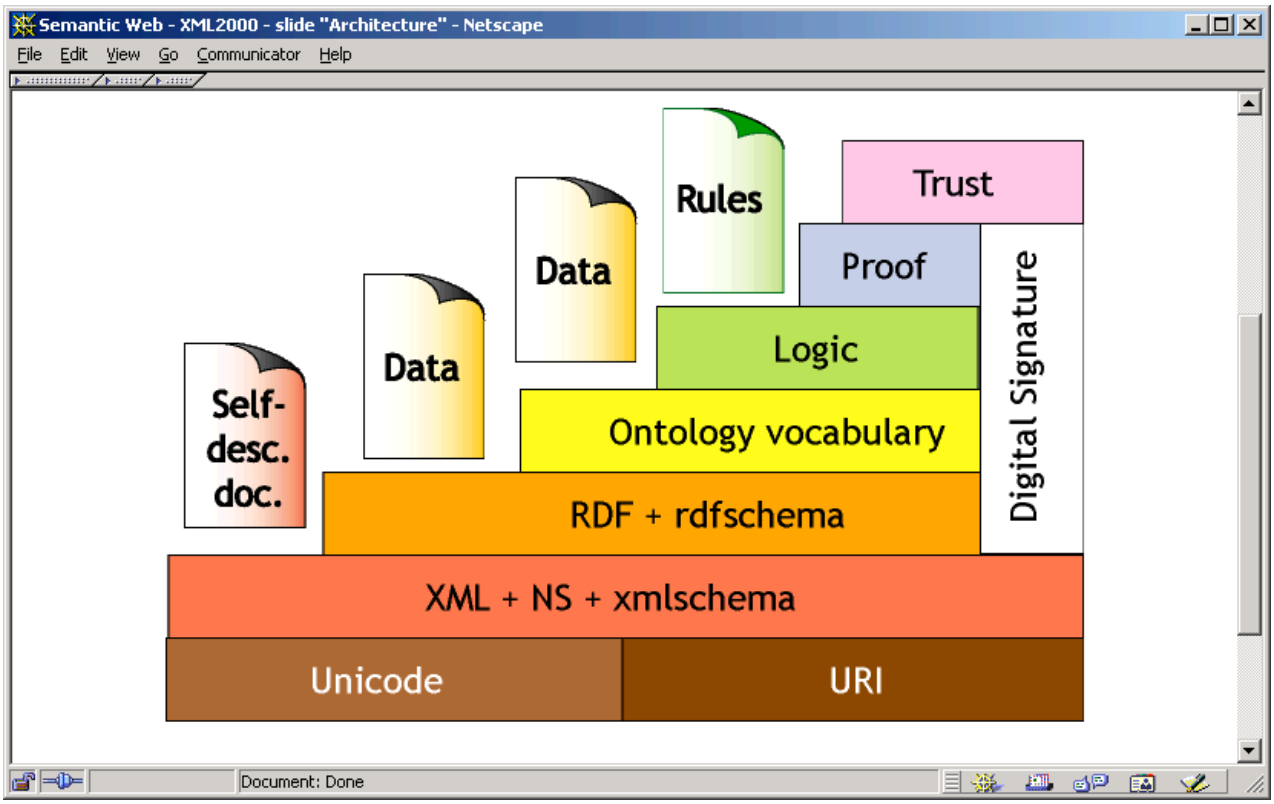

Figure 1: Berners-Lee's Architecture

He shows the markup languages at the base (just above Unicode) for use in term specification (or in web speak, "resource" definition). The next layer and the one we will consider here, is the ontology layer. In this layer, we can define terms and their relationships to other terms. The next layer is the logic layer. In this layer, we can deduce information, thereby allowing us to deduce implications of the term definitions and relationships. In the rest of this paper, we will discuss the ontology and logic layers, what they have come to mean on the web, and how one might generate ontologies and use them in applications.

Ontologies

The term ontology has been in use for many years. Merriam Webster, for example, dates ontology circa 1721 and provides two definitions (1) a branch of metaphysics concerned with the nature and relations of being and (2) a particular theory about the nature of being or the kinds of existents. These definitions provide an abstract philosophical notion of ontology. Mathematical or formal ontologies have also been written about for many years. Smith [Smith 1998] points out that at least as early as 1900, the notion of a formal ontology has been distinguished from formal logic by the philosopher Husserl. While ontologies (even formal ontologies) have had a long history, they remained largely the topic of academic interest among philosophers, linguists, librarians, and knowledge representation researchers until somewhat recently.

Ontologies have been gaining interest and acceptance in computational audiences (in addition to philosophical audiences). Guarino [Guarino 1998] provides a nice collection of fields that embrace ontologies including knowledge engineering, knowledge representation, qualitative modeling, language engineering, database design, information retrieval and extraction, and knowledge management and organization. That collection put together in early 1998 did not include nearly the web emphasis that is seen today. We would also include areas of library science [Dublin Core 1999], ontology-enhanced search (e.g., eCyc (http://www.e-Cyc.com/) and FindUR [McGuinness 1998]), possibly the largest one, e-commerce (e.g., Amazon.com, Yahoo Shopping, etc.), and configuration.

In this paper, we will be restricting our sense of ontologies to those we see emerging on the web. Today's use of ontology on the web has a different slant from the previous philosophical notions. One widely cited definition of an ontology is Gruber's [Gruber 1993] "A specification of a conceptualization". We will use this notion and expand upon it in our use of the term.

People (and computational agents) typically have some notion or conceptualization of the meaning of terms. Software programs sometimes provide a specification of the inputs and outputs of a program, which could be used as a specification of the program. Similarly ontologies can be used to provide a concrete specification of term names and term meanings. Within the line of thought where an ontology is a specification of the conceptualization of a term, there are still a number of potential interpretations. Web ontologies may be viewed as a spectrum of detail in their specification. One might visualize a simple (linear) spectrum of definitions in Figure 2[2] below.

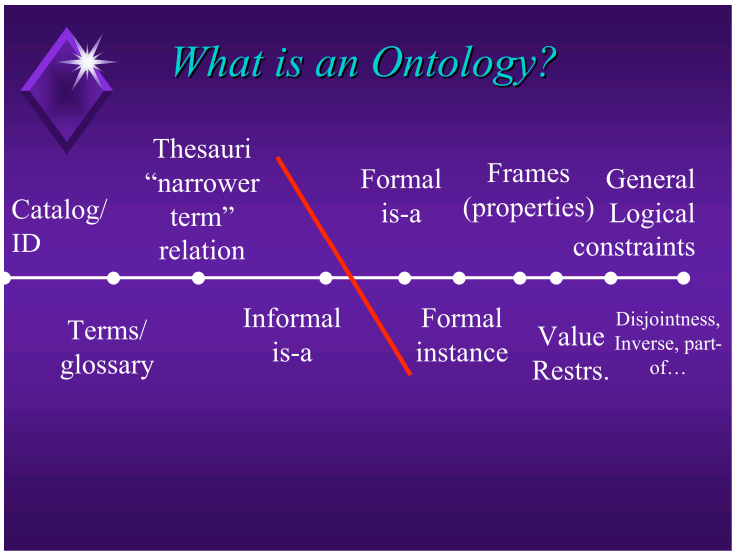

Figure 2: An Ontology Spectrum

[2] This spectrum arose out of a conversation in preparation for an ontology panel at AAAI '99. The panelists (Lehman, McGuinness, Ushold, and Welty), chosen because of their years of experience in ontologies found that they encountered many forms of specifications that different people termed ontologies. McGuinness refined the picture to the one included here.

One of the simplest notions of a possible ontology may be a controlled vocabulary – i.e., a finite list of terms. Catalogs are an example of this category. Catalogs can provide an unambiguous interpretation of terms – for example, every use of a term, say car – will denote exactly the same identifier – say 25.

Another potential ontology specification is a glossary (a list of terms and meanings). The meanings are specified typically as natural language statements. This provides a kind of semantics or meaning since humans can read the natural language statements and interpret them. Typically, interpretations are not unambiguous and thus these specifications are not adequate for computer agents, thus this would not meet the criteria of being machine processable.

Thesauri provide some additional semantics in their relations between terms. They provide information such as synonym relationships. In many cases their relationships may be interpreted unambiguously by agents. Typically thesauri do not provide an explicit hierarchy (although with narrower and broader term specifications, one could deduce a simple hierarchy).

Early web specifications of term hierarchies, such as Yahoo's, provide a basic notion of generalization and specialization. Yahoo, for example, provides a small number of top-level categories such as apparel and the category dresses as a kind of (women's) apparel. A small number of people consider the previous categories (of catalogues, glossaries, and thesauri) to be ontologies but many prefer to have an explicit hierarchy included before something is considered an ontology. Yahoo, for example, does provide an explicit hierarchy. Its hierarchy is not a strict subclass or "isa" [Brachman: 1983] hierarchy however. This point was distinguished on the spectrum slide since it seems to capture many of the naturally occurring taxonomies on the web. In these organization schemes, it is typically the case that an instance of a more specific class is also an instance of the more general class but that is not enforced 100% of the time. For example, the general category apparel includes a subcategory women (which should more accurately be titled women's apparel) which then includes subcategories accessories and dresses. While it is the case that every instance of a dress is an instance of apparel (and probably an instance of women's dress), it is not the case that a dress is a woman and it is also not the case that a fragrance (an instance of a women's accessory) is an instance of apparel. This mixing of categories such as accessories in web classification schemes is not unique to Yahoo – it appears in many web classification schemes[3]. Without true subclass (or true "isa") relationships, we will see that certain kinds of deductive uses of ontologies become problematic.

The next point on the figure includes strict subclass hierarchies. In these systems if A is a superclass of B, then if an object is an instance of B it necessarily follows that the object is an instance of A. For example, if "Dress" is a subclass of "Apparel" and "MyFavoriteDress" is an instance of "Dress", then it follows that "MyFavoriteDress" is an instance of "Apparel". Strict subclass hierarchies are necessary for exploitation of inheritance. The next point on the ontology spectrum includes formal instance relationships. Some classification schemes only include class names while others include ground individual content. This point includes instances as well.

---

[3] Some prominent hierarchies such as Yahoo have renamed their classes to broad disjunctive categories such as "Apparel, Accessories, and Shoes" presumably in order to provide for more strict subclass relationships. Disjunctive categories make inheritance more problematic however with class-specific properties.

The next point includes frames[4]. Here classes include property information. For example, the "Apparel" class may include properties of "price" and "isMadeFrom". My specific dress may have a price of $100 and may be made from cotton. Properties become more useful when they are specified at a general class level and then inherited consistently by subclasses and instances. In a consumer hierarchy, a general category like consumer product might have a "price" property associated with it. Possibly apparel would be the general category to which the property "isMadeFrom" is associated. This would mean that the domain of "isMadeFrom" is apparel. All subclasses of these categories would inherit these properties.

A more expressive point in the ontology spectrum includes value restrictions. Here we may place restrictions on what can fill a property. For example, a "price" property might be restricted to have a filler that is a number (or a number in a certain range) and "isMadeFrom" may be restricted have fillers that are a kind of material. Here we can see a possible problem with a classification scheme that does not support strict "isa" or subclass relationships. For example, if "Fragrance" were a subclass of "Apparel", it would inherit the property "isMadeFrom" and the value restriction of material that was stated.

As ontologies need to express more information, their expressive requirements grow. For example, we may want to fill in the value of one property based on a mathematical equation using values from other properties. Some languages allow ontologists to state arbitrary logical statements. Very expressive ontology languages such as that seen in Ontolingua [Farquhar et al 1997] or CycL allow ontologists to specify first order logic constraints between terms and more detailed relationships such as disjoint classes, disjoint coverings, inverse relationships, part-whole relationships, etc.

In this paper, we will require the following properties to hold in order to consider something an ontology. Specifications meeting these properties will be referred to as simple ontologies.

- Finite controlled (extensible) vocabulary
- Unambiguous interpretation of classes and term relationships
- Strict hierarchical subclass relationships between classes

We consider the following properties typical but not mandatory:

- Property specification on a per-class basis
- Individual inclusion in the ontology
- Value restriction specification on a per-class basis

Finally, the following properties may be desirable but not mandatory nor typical:

- Specification of disjoint classes
- Specification of arbitrary logical relationships between terms
- Distinguished relationships such as inverse and part-whole

[4] Frames were introduced by Minsky [Minsky 1975] and have been widely adopted, see for example [Fikes & Kehler 1985, Karp 1992, and Chaudhri-et-al 1998].

The line in our chart is drawn such that everything to the right of it will be called an ontology and meet at least the first three conditions stated above. Additionally, everything to the right of it can be used as a basis for inference.

Simple Ontologies and Their Uses

We will now consider ontologies and their impact on applications. We break this section into two parts: uses of simple ontologies and uses for more sophisticated ontologies. We do this because we acknowledge that building the more complicated ontologies may be cost prohibitive for certain applications.

Simple ontologies however are not as costly to build and potentially more importantly, many are available. Simple ontologies are available in many forms – many exist as freeware on the web today and also many exist as internal information organization structures within companies, universities, etc. Some collaborative efforts exist such as DMOZ (www.dmoz.com) that are generating large simple ontologies. DMOZ, for example, leverages over 35,000 volunteer editors and at publication time, had over 360,000 classes in a taxonomy. Additionally, some more sophisticated ontologies are available today. For example, the unified medical language system (UMLS - http://www.nlm.nih.gov/research/umls/ and [Humphreys & Lindberg 1993]) developed by the national library of medicine is a large sophisticated ontology about medical terminology. Some companies such as Cycorp (www.cyc.com) are making available portions of large, detailed ontologies. We will further address the issue of ontology acquisition and maintenance later, but for now, we just wanted to make the point that many simple and some sophisticated ontologies are easily available today.

Now lets consider some of the ways that simple ontologies may be used in practice.

First, they provide a ***controlled vocabulary***. This by itself can provide great leverage since users, authors, and databases can all use terms from the same vocabulary. In addition programs can generate interfaces that encourage usage of the controlled terms. The result is that people use the same set of terms. Of course, some of the terms may still be used with different senses, but common term usage is a start for interoperability.

Second, a simple taxonomy may be used for ***site organization and navigation support***. Many web sites today expose on the left hand side of a page the top levels of a generalization hierarchy of terms. The categories are typically hot and a user may click on them to expand the subcategories.

Third, taxonomies may be used to support *expectation setting*. It is an important user interface feature that users be able to have realistic expectations of a site. If they may explore even the top level categories of the hierarchy, they can quickly determine if the site might have content (and/or services) of interest to them.

Fourth, taxonomies may be used as ***"umbrella" structures from which to extend content***. Some freely available ontologies are attempting to provide the high level taxonomic organization from which many efforts may inherit terms. The UNSPSC (Universal Standard Products and Services Classification www.unspsc.org) is one such categorization scheme. It was jointly done by the United Nations Development Program and Dun & Bradstreet and was aimed at providing the infrastructure for interoperability of terms in the domains of products and services. It provides a classification scheme (with associated numbers). For example, Category 50 – Food, Beverage, and Tobacco Products has a subclass

family 5010 called “Fruits and vegetables and nuts and seeds[5]” which in turn contains a subclass 501015 called Vegetables, which in turn has a subclass commodity 50101538 called fresh vegetables. A number of e-commerce applications today are looking for such umbrella organization structures and in fact a number have chosen to be compliant with the UNSPSC. Most applications will need to extend these ontologies, but if applications need to communicate between a number of content providers, it is convenient to use a shared upper level ontology.

Fifth, taxonomies may provide ***browsing support***. Content on a site may be tagged with terms from the taxonomy. This may be done manually in the style of Yahoo or it may be done automatically (possibly using a clustering approach). Once a page (or service) is meta-tagged with a term chosen from a controlled vocabulary, then search engines may exploit the tagging and provide enhanced search capabilities.

Sixth, taxonomies may be used to provide ***search support***. A query expansion method may be used in order to expand a user query with terms from more specific categories in the hierarchy. We exploited this approach in our work on FindUR[McGuinness:1998] and found that under certain conditions (such as short document length and limited content areas), query expansion can radically improve search.

Seventh, taxonomies may be used to ***sense disambiguation support***. If the same term appears in multiple places in a taxonomy, an application may move to a more general level in the taxonomy in order to find the sense of the word. For example, if an ontology contains the information that Jordan is an instance of a Basketball-player and also an instance of a country, an application may choose to query a user searching for Jordan if she is interested in basketball-players or countries. Sense disambiguation using ontologies may be seen in the work of eCyc along with Hotbot and Lycos.

Structured Ontologies and Their Uses

Up to this point, we have focused on simple taxonomies for usage in applications. Once ontologies begin to have more structure however, they can provide more power in applications. Once the ontologies have more structure than simple generalization links, property information can be used in many forms.

First, they can be used for simple kinds of ***consistency checking***. If ontologies contain information about properties and value restrictions on the properties, then type checking can be done within applications. For example, if a class called “Goods” has a property called “price” that has a value restriction of number, then something that is known to be of type “Goods” that has its “price” property filled in with a value that is not a number can be caught as an error. This just exploits simple value restrictions that are types. A value restriction might include a range, for example, a number between 10 and 100. Then if the ”price” is 10,000, it is out of the range and can be determined to be an error.

Second ontologies may be used to provide ***completion***. An application may obtain a small amount of information from a user, such as the fact that she is looking for a high-resolution screen on a pc, and then have the ontology expand the exact pixel range that is to be expected. This can be done simply by defining what the term “HighResolutionPc” is with respect to a particular pixel range on two roles – “verticalResolution” and “horizontalResolution”. Similarly, information may interact. For example, a

[5] Note, if one is using the common logical meanings of connectives, this class should really be named “Fruits or vegetables or nuts or seeds”.

medical system may obtain information from an ontology that if a patient is stated to be a man, then the gender of the patient is "male" and that information may be used to determine that a question concerning whether or not the patient is pregnant should not be asked since there could be information in the system that things whose gender is male are disjoint from things that are pregnant.

Third, ontologies may be able to provide ***interoperability support***. In the simple case of considering controlled vocabularies, there is enhanced interoperability support since different users/applications are using the same set of terms. In simple taxonomies, we can recognize when one application is using a term that is more general or more specific than another term and greater facilitate interoperability. In more expressive ontologies, we may have a complete operational definition for how one term relates to another term and thus, we can use equality axioms or mappings to express one term precisely in terms of another and thereby support more "intelligent" interoperability. For example, an ontology may include a definition that a "StanfordEmployee" is equal to a "Person" whose "employer" property is filled with the individual "Stanford University". This definition may be used to expand the term "StanfordEmployee" in an application that does not understand either "StanfordEmployee" or "Employee" but does understand the terms "Person", "employer", and "Stanford".

Fourth, ontologies may be used to ***support validation and verification testing*** of data (and schemas). If an ontology contains class descriptions, such as "StanfordEmployee", these definitions may be used as queries to databases to discover what kind of coverage currently exists in datasets. For example, if one was going to expose the class "StanfordEmployee" on an interface to some application, it would be useful to know first if the dataset contained any instances of "Person" whose "employer" property was filled with the value "Stanford". Additionally, if in a simple data model, we stated that a "Person" had at most one "employer", then we could use that information to check to see if any current information on "Person"s in the dataset contained more than one "employer" value. Similarly, checks could be done to see if there were currently "Person"s in the dataset that were known to be "Employee"s yet did not have a value for the "employer" property (thereby showing that the dataset is not complete). Chimaera [McGuinness-et-al: 2000] is an example ontology evolution environment that provides a set of diagnostics tests for checking ontologies for both problems in the ontology definitions as well as problems with the instance data. It looks for provable inconsistencies as well as conditions that "typically" reflect situations where an ontology or the data may need to be fixed.

Fifth, ontologies containing markup information may ***encode entire test suites***. An ontology may contain a number of definitions of terms, some instance definitions, and then include a term definition that is considered to be a query – find all terms that meet the following conditions. Markup information could be encoded with this query to include what the answer should be, thus providing enough information to encode regression testing data. We provide one such example ontology in http://ksl.stanford.edu/projects/DAML/chimaera-jtp-cardinality-test1.daml. The ontology contains a regression test suite for checking cardinality inferences (such as persons having two employers yet being stated to have at most one employer) in a Stanford Theorem prover (http://www.ksl.Stanford.EDU/software/jtp/).

Sixth, ontologies can provide the foundation for ***configuration support***. Class terms may be defined so that they contain descriptions of what kinds of parts may be in a system. Additionally interactions between properties can be defined so that filling in a value for one property can cause another value to be filled in for another slot. For example, one may generate an ontology of information about home theatre products as is done in a small configurator example using a simple description logic-based system [McGuinness-et-al 1995]. Terms such as television, amplifier, tuner, etc are defined. Additionally, information connecting

the terms together is included. A class of HighQualityTelevisions is defined so that users may choose from this class and the configurator will automatically fill in limited sets of manufacturers to choose from, minimum diagonal values, minimum price ranges etc. Also, information is encoded that propagates restrictions from one component to another. For example, some of the components in this system were meant to be sold in pairs. If one buys one particular kind speaker (which is only sold in pairs, thus two speakers are added to the parts list), then restrictions on particular speaker stands appear in the configuration specification. There are many such configuration examples using ontologies, some of which are described in a special configuration issue of Artificial Intelligence for Engineering Design, Analysis, and Manufacturing Journal [Darr-et-al 1998].

Seventh, ontologies can ***support structured, comparative, and customized search***. For example, if one is looking for televisions, a class description for television may be obtained from an ontology, its properties may be obtained (such as diagonal, price, manufacturer, etc), and then a comparative presentation may be made of televisions by presenting the values of each of the properties. Those properties can also be used to provide a form for users to fill in so that they may provide a detailed set of specifications about the items they are looking to find. This also provides the foundation for providing a number of different search interfaces – the simple text box along with search interfaces exposing important properties of products. More sophisticated ontologies may be generated that mark which properties are most useful to present in comparative analyses so that users may have concise descriptions of the products instead of comparisons in complete detail. Thus, ontologies with markup information may also be used to prune comparative searches.

Eighth, ontologies may be used to ***exploit generalization/specialization information***. If a search application finds that a user's query generates too many answers, one may dissect the query to see if any terms in it appear in an ontology, and if so, then the search application may suggest specializing that term. For example, if one did a search for concerts in the San Francisco Bay area and got too many answers, a search engine might look up concert in an ontology and discover that there are subclasses of concert (and it may also discover that there are specific concert locations in the Bay area).

The search engine could then choose to present the user with the option of looking for a particular kind of concert (say rock concert). Further the search engine could proactively run queries in the background while waiting for user input or also cache information from popular queries. Then the search engine could also present a list of subclasses of concerts and provide the user with the approximate number of retrievals the user would get if they specialized their query in the different manners. These are just some of the ways in which ontologies may be used to refine search queries. We could also look at the ontology to provide alternative values (by looking at siblings in the ontology) for terms specified in the search query.

We have not claimed to present an exhaustive list of the ways in which ontologies may be used in applications. The above lists are just illustrative of some ways that ontologies have been used to support intelligent applications.

Ontology Acquisition

Now that at least some readers may be convinced that ontologies are useful components in applications, we will look at some sources of ontologies. First, as we mentioned previously, many ontologies exist in the public domain. It may be possible to start with an existing industry standard and use that as the ontology

starting point. Most likely application developers will need to modify and/or extend ontologies that are available and were developed for other uses. Still, one methodology for obtaining ontologies is to begin with an industry standard ontology and then modify or extend it.

Another methodology is to semi-automatically generate a starting point for an ontology. Many taxonomic structures exist on the web or in the table of contents of documents. One might crawl certain sites to obtain a starting taxonomic structure and then analyze, modify, and extend that.

One question is where to look for existing ontologies or sources of information to be crawled. Many controlled vocabularies are being made available today. Sometimes standards organizations, such as NIST (the National Institute of Standards and Technology - http://www.nist.gov/), support efforts in producing controlled vocabularies and ontologies. Some consortiums are forming to generate ontologies. See, for example, RosettaNet (http://www.rosettanet.org) in the area of information technology, electronic technology, electronic components, and semiconductor manufacturing. They are creating industry-wide open e-business standards and providing a language for business processes. Sometimes trade organizations provide class hierarchies on their sites that can also be used as a standard structured controlled vocabulary. There are also broad sources of class structures. Essentially every e-commerce site today encodes at least a taxonomic organization of terms. Sites like Amazon in organizing their book and music information provides a very broad organization of information.

Another emerging trend is the use of markup languages. Some pages are being annotated using markup languages such as XML, RDF, DAML, etc. The pages including the annotations may be using markup terms from controlled vocabularies. Some libraries are emerging of ontologies potentially of use for markup. For example, the DAML program maintains a library of DAML ontologies in http://www.daml.org/ontologies/.

Much of this section has introduced the idea of obtaining either a simple or complex ontology as a starting point and then analyzing, modifying, and maintaining it over time. In the next section, we will address the issue of implications and needs from ontology-based applications.

Ontology-related Implications and Needs

When starting an ontology-based application, the two major concerns will be language and environment.

Language: When considering ontology-related applications, inevitably the issue of *ontology language* will arise. An ontology must be encoded in some language. If one is using a simple ontology, few issues arise. However, if one is considering a more complex ontology, expressive power of a representation and reasoning language needs to be considered. As with any problem where a language is being chosen, it must be epistemologically adequate -- the language must be able to express the concepts in the domain.

For example, if one wants to do range checking in an e-commerce application, then it would be unwise to choose just a simple language that only contains subclass and instance relationships and does not include property specification with value restrictions. There are a number of candidate ontology languages – in fact there are so many that some research efforts arose in the last decade in order to produce standard specification languages (such as the KRSS effort – the Knowledge Representation System Specification effort [Patel-Schneider-Swartout 1992]) interchange formats (such as KIF -the Knowledge Interchange Format which is now a proposed ANSI standard [KIF]), and common application programming interface standards (such as OKBC – Open Knowledge Base Connectivity [Chaudhri-et-al, 1997]).

One does not just want to consider representational constructs in a language; one also wants to consider the reasoning that may be supported in the language. Some fields such as description logics (www.dl.kr.org), make this a central focus in language design. They look for tradeoffs that maintain expressive power needed by applications and also consider what it takes to provide inference engines that can provide deductions based on the constructs represented in the language. For example, if a language supports the notion of stating that two classes are disjoint, then a reasoning engine should be able to be built that enforces the constraint that the classes are disjoint. Thus, an inference engine should be able to warn a user if she is creating an instance or subclass of two disjoint classes.

Also, a language should be usable with existing platforms and should be something that non-experts can use to do their conceptual modeling. The web is clearly the most important platform with which to be compatible today, thus any language choice should be able to leverage the web. Additionally, frame-based systems have had a long history of being thought of as conceptually easy to use, thus a frame paradigm may be worth considering.

Language efforts seen today attempt to take the best of the research on expressive power along with reasoning power and provide representationally powerful languages that have known reasoning properties. The DARPA Agent Markup Language program, for example, attempted to take the emerging web languages of today such as XML and RDF and create a language that is web compatible but draws on the 20 year history of description logics in choosing language constructs along with reasoning paradigms. The resulting language –DAML+OIL – attempts to merge the best of existing web languages, description logics, and frame reasoning systems. OIL [Bechhofer-et-al 2000] attempts to provide a layered approach to language design.

Environment: Another consideration is how to analyze, modify, and maintain an ontology over time. If the ontology is to be maintained by subject matter experts (and not by knowledge experts), most likely some ontology tools will be needed. There are a number of simple ontology tools available commercially. Some information retrieval companies such as Verity have provided simple editors for generating and browsing simple generalization hierarchies. Verity, for example, has provided a “topic editor” for years which will support users in generating taxonomies and utilizing them in search queries. Research efforts have existed for many years in producing ontology toolkits. Stanford University’s previously mentioned tools of Ontolingua [Farquhar-et-al 1997] and Chimaera [McGuinness-et-al. 2000] are just two examples, however examples abound including OilEd (http://img.cs.man.ac.uk/oil/) from Manchester University and Protégé [Protégé 2000] from Stanford Medical Informatics, just to name a few. Application developers may choose commercial vendors as their toolkit provider, sophisticated research applications as the base, or somewhere in between. Some companies with extensive ontology needs such as VerticalNet

(http://www.verticalnet.com/) have or are developing their own ontology tools in order to build ontologies that meet the needs of a sophisticated commercial ontologist. Their tools were built after analyzing existing research prototypes and were then designed to meet the commercial standards required in diverse, collaborative, e-commerce applications of today.

When choosing to use or build an ontology environment, there are a number of issues that should be considered including the following:

- *Collaboration and distributed workforce support*. Some ontology environments allow users to share a session –i.e., see each other's work environments. This can be particularly useful for debugging. Ontolingua, for example, supports this notion. Additionally, when workers are distributed in location, it becomes important to have an environment that allows access from multiple places. This is becoming much more typical today with server/client architectures. Finally, collaboration may require concurrency control, locking, and a kind of versioning and permission system.
- *Platform interconnectivity*. As applications become embedded in more complex platforms, it becomes important for environments to be able to read and write compatible formats, be able to be integrated with multiple hardware/software environments, etc. Java-based applications provide a convenient approach to this problem but other systems that support multiple input and output formats, understand common standards, and provide translation and mapping services may help.
- *Scale*. Many ontology applications today may need to scale a few orders of magnitude larger than past applications. It is important to look at scaling in terms of size of ontologies as well as numbers of simultaneous users.
- *Versioning*. As applications become long-lived and also are deployed in different environments possibly internationally, it becomes important to be able to support many versions of ontologies. In typical software engineering environments, there are source code control systems and versioning.
- *Security*. Some applications will have needs for differing access to portions of the ontology. Thus, it is important to have an environment that can expose portions of the ontology based on a security model. The security model may need to support both read and write access.
- *Analysis*. Environments are expected to support acquisition, evolution, and maintenance of ontologies. Thus, it would be common to expect ontologies to have periods when they are incomplete and incorrect. Analysis support that can focus the user's attention in areas that are likely to need modification can be quite useful. The Chimaera ontology environment, for example, supports a number of diagnostic tests aimed at helping users identify provably incorrect ontologies as well as possible problems.
- *Lifecycle issues*. As ontologies become larger and longer lived, it would be expected that application developers might be maintaining ontologies over many years. Additionally, they may be constantly merging new ontologies into their system as their applications interconnect with more diverse systems. Thus, it becomes important to consider support for ontology evolution issues

such as merging terms, breaking apart terms, multiple name spaces, source code control systems, truth maintenance systems, regression testing systems, etc.

- *Ease of use.* Even if an environment has everything an application developer may need, if it is difficult for the user to decide how to use parts of the environment, they may not get used. Thus training materials, tutorials, conceptual modeling support, graphical browsing tools, etc. all may be important. We have written separately on some of the issues required to make description logic-based systems usable in mainstream use. [McGuinness-Patel-Schneider 1998, Brachman-et-al, 1999].
- *Diverse user support.* Some environments are made for power users, some for naïve users, and some have settings that allow users to customize environments as appropriate to the type of user. It is important to determine if the environment can support all of the types of users anticipated.
- *Presentation Style*. Possibly closely related to user type is presentation style. Some users need to see extensive detail, some need pruned information, and some need abstractions. Presentation of information may be textual, graphical, or other. While no one environment needs to support all presentation styles, it is important that the environment is at least extensible enough to have new presentation methods added when needed.
- *Extensibility*. It will be impossible to anticipate all of the needs an application will have. Thus, it is important to use an environment that can adapt along with the needs of the users and the projects.

Conclusions

In this paper, we have noted the emergence of ontologies from academic obscurity into mainstream business and practice on the web. We have introduced the term ontology along with a spectrum of properties that ontologies may exhibit. We have provided criteria necessary, prototypical, and desirable for simple and complex ontologies. We have also identified ways that ontologies (both simple and complex) are being and may be used to provide value in many types of applications. We have addressed the issue of acquiring ontologies and then maintaining and evolving ontologies. Finally, we have identified a number of ontology-related issues that arise from the emergence of ontologies focusing on ontology language and environment. Finally, we concluded with issues that are gaining importance as ontologies grow in their importance and centrality in diverse applications.

References

[Bechhofer et al 2000] Sean Bechhofer, Jeen Broekstra, Stefan Decker, Michael Erdmann, Dieter Fensel, Carole Goble, Frank van Harmelen, Ian Horrocks, Michel Klein, Deborah McGuinness, Enrico Motta, Peter Patel-Schneider, Steffen Staab, and Rudi Studer, “An informal description of Standard Oil and Instance OIL”, available on-line as http://www.ontoknowledge.org/oil/downl/oil-whitepaper.pdf

[Berners-Lee 2000] Tim Berners-Lee, "Semantic Web on XML”, Keynote presentation for XML 2000. Slides available at: http://www.w3.org/2000/Talks/1206-xml2k-tbl/slide1-0.html. Reporting available at: http://www.xml.com/pub/a/2000/12/xml2000/timbl.html

[Berners-Lee 1999] Tim Berners-Lee, "Weaving the Web", Harper, San Francisco, 1999. http://www.harpercollins.com/hc/bookpage/index.asp?isbn=0062515861

[Berners-Lee et al 1998] Tim Berners-Lee, Roy Fielding, and Larry Masinter, "Uniform Resource Identifiers (URI): Generic Syntax", Internet Draft Standard RFC 2396, August 1998; available on-line as http://www.isi.edu/in-notes/rfc2396.txt.

[Brachman 1983] Ronald J. Brachman, R. J. *What ISA Is and Isn't: An Analysis of Taxonomic Links in Semantic Networks*. IEEE Computer, 16 (10), 30--6. 1983.

[Brachman et al. 1999] Ronald J. Brachman, Alex Borgida, Deborah L. McGuinness, and Peter F. Patel-Schneider. "Reducing" CLASSIC to Practice: Knowledge Representation Theory Meets Reality. In Artificial Intelligence 114(1-2) pages 203-237, October, 1999.

[Brickley & Guha 2000] Dan Brickley & R.V.Guha, "Resource Description Framework (RDF) Schema Specification 1.0", W3C Candidate Recommendation 27 March 2000, World Wide Web Consortium, Cambridge (MA); available on-line as http://www.w3.org/TR/rdf-schema/.

[Broekstra et. al. 2001] J. Broekstra, M. Klein, S. Decker, D. Fensel, F. van Harmelen, and I. Horrocks. "Enabling knowledge representation on the Web by Extending RDF Schema", Proceedings of the International Conference on the World Wide Web (WWW10), May 2001.

[Chaudhri et. al. 1998] Vinay Chaudhri, Adam Farquhar, Richard Fikes, Peter Karp, and James Rice; "*OKBC: A Programmatic Foundation for Knowledge Base Interoperability*", AAAI 1998.

[Darr et. al. 1998] Tim Darr, Mark Fox, and Deborah L. McGuinness, editors. Special Configuration Issue of the Artificial Intelligence for Engineering Design, Analysis, and Manufacturing Journal 1998.

[Dublin Core 1999] "Dublin Core Metadata Element Set, Version 1.1: Reference Description", Dublin Core Metadata Initiative, 1999; available on-line as http://purl.org/dc/documents/rec-dces-19990702.htm

[Farquhar et al 1997] Adam Farquhar, Richard Fikes, and James Rice; "*The Ontolingua Server: a Tool for Collaborative Ontology Construction*", Intl. Journal of Human-Computer Studies **46**, 1997

[Fikes & Kehler 1985] Richard Fikes & Tom Kehler, "The Role of Frame-Based Representation in Reasoning", CACM 28(9): 904-920 (1985).

[Fikes & McGuinness 2001] Richard Fikes & Deborah L. McGuinness, "An Axiomatic Semantics for RDF, RDF Schema, and DAML+OIL", KSL Technical Report KSL-01-01, Stanford University, 2001; available on-line as http://www.ksl.stanford.edu/people/dlm/daml-semantics/abstract-axiomatic-semantics.html.

[Google, 2000] Connie Guglielmo and Charles Babcock, "Gaga over Google" , Interactive Week, Nov 6, 2000. http://www.zdnet.com/intweek/stories/news/0,4164,2651081,00.html, and http://www.google.com/about.html

[Guarino 1998] Nicola Guarino, "Formal Ontology and Information Systems". In the Proceedings of Formal Ontology in Information Systems, June 1998. Also in Frontiers in Artificial Intelligence and Applications, IOS-Press, Washington, DC, 1998.

[Gruber 1993] Tom R. Gruber, "A translation approach to portable ontologies". Knowledge Acquisition, 5(2):199-220, 1993.

[Hendler & McGuinness 2000] James Hendler and Deborah McGuinness. ``The DARPA Agent Markup Language". In IEEE Intelligent Systems Trends and Controversies, November/December 2000. Available from http://www.ksl.stanford.edu/people/dlm/papers/ieee-daml01-abstract.html .

[Humphreys & Lindberg 1993] B. L. Humphreys and D. A. B. Lindberg. "The UMLS project: making the conceptual connection between users and the information they need. Bulletin of the Medical Library Association 81(2): 170.

[Husserl 1900] Edmund Husserl, Logische Untersuchungen, First edition Halle: Niemeyer, 1900/01.

[KIF] Knowledge Interchange Format, Language Description, draft proposed national standard. NCITS.T2/98-004. http://logic.stanford.edu/kif/kif.html.

[Karp 1992] Peter D. Karp, "The design space of frame knowledge representation systems", Technical Report 520, SRI International AI Center; available on line as ftp://www.ai.sri.com/pub/papers/karp-freview.ps.Z

[Lassila 1998] Ora Lassila, "Web Metadata: A Matter of Semantics", IEEE Internet Computing 2(4): 30-37 (1998).

[Lassila & Swick 1999] Ora Lassila & Ralph Swick, "Resource Description Framework (RDF) Model and Syntax Specification", W3C Recommendation 22 February 1999, World Wide Web Consortium, Cambridge (MA); available on-line as http://www.w3.org/TR/REC-rdf-syntax/.

[McGuinness-et-al 1995] Deborah L. McGuinness, Lori Alperin Resnick, and Charles Isbell. ``Description Logic in Practice: A CLASSIC: Application." In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, Montreal, Canada, August, 1995.

[McGuinness 1998] Deborah L. McGuinness ``Ontological Issues for Knowledge-Enhanced Search". In the Proceedings of Formal Ontology in Information Systems, June 1998. Also in Frontiers in Artificial Intelligence and Applications, IOS-Press, Washington, DC, 1998.

[McGuinness-et-al 2000] Deborah L. McGuinness, Richard Fikes, James Rice, and Steve Wilder. *An Environment for Merging and Testing Large Ontologies*. In the Proceedings of the Seventh International Conference on Principles of Knowledge Representation and Reasoning (KR2000), Breckenridge, Colorado, USA. April 12-15, 2000.

[McGuinness-Patel-Schneider] Deborah L. McGuinness and Peter F. Patel-Schneider. ``Usability Issues in Knowledge Representation Systems". In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, Madison, Wisconsin, July, 1998. This is an updated version of ``Usability Issues in Description Logic Systems" published in *Proceedings of International Workshop on Description Logics*, Gif sur Yvette, (Paris), France, September, 1997.

[Minsky 1975] Marvin Minsky, "A Framework for Representing Knowledge", in Patrick Henry Winston (ed.), *The Psychology of Computer Vision*, McGraw-Hill, New York, 1975.

[Patel-Schneider-Swartout] Peter F. Patel-Schneider and Bill Swartout; "Description-Logic Knowledge Representation System Specification"; KRSS Group of the ARPA Knowledge Sharing Effort. http://www-db.research.bell-labs.com/user/pfps/papers/krss-spec.ps

[Protégé 2000] The Protege Project. http://protege.stanford.edu

[Smith 1998] Barry Smith, "Basic Concepts of Formal Ontologies", in N. Guarino (Ed.) Formal Ontology in Information Systems, IOS Press, 1998.

[Woods 1975] William A. Woods, "What's in a Link: Foundations for Semantic Networks", in D.G.Bobrow & A.M.Collins (eds.), *Representation and Understanding: Studies in Cognitive Science*, 35-82, Academic Press, New York, 1975.

[XML 2000] Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, and Eve Maler, editors. Extensible Markup Language (XML) 1.0 (Second Edition). W3C Recommendation 6 October 2000. http://www.w3.org/TR/2000/REC-xml-20001006