On the Impact of Ontological Commitment

Marian Nodine
Telcordia Technologies
106 E. 6th Street, Suite 415
Austin, TX 78733
(512)478-8923
nodine@research.telcordia.com

Jerry Fowler
setenv, LLC
Houston, TX
(713)526-8641
gfowler@acm.org

ABSTRACT

Ontological commitment, or the agreement to have your applications and users conform to a common domain understanding as encapsulated in one or more shared ontologies, is a noble goal and essential for open agent systems. Our experiences building ontology-based agent systems in multiple domains have shown us that the intention for a new application to locate and conform to some existing ontology or ontologies within its domain has many impediments to its success. For instance, the goals of the designer of a domain ontology include developing a complete and comprehensive domain description; however, the application developer may only require a small fragment of that ontology. Multiple applications that conform to the ontology may, in fact, use completely orthogonal fragments of the ontology, and not be able to interact at all. Users may insist on importing into the ontology sets of terms that are neither logically consistent nor easily modelable.

With these issues in mind, in this paper we propose some guidelines for ontology development and evolution paradigm that should facilitate ontology reuse. These guidelines could underpin a *usage model* for ontologies; one that enables the application designer to reuse ontological concepts from multiple ontologies in a more flexible manner, while retaining the essentially good properties of ontology sharing and reuse. These guidelines affect both the design and use of ontology-based applications, as well as the way applications advertise themselves to other agents with which they may interoperate.

1. INTRODUCTION

The goal of knowledge representation is to make explicit the semantics of a particular domain of interest for the purposes of sharing the knowledge among humans and computer artifacts. Sowa [11] subdivides knowledge representation into categories:

> "***Logic*** provides the formal structure and rules of inference.
>
> ***Ontology*** defines the kinds of things that exist in the application domain.
>
> ***Computation*** supports the applications that distinguish knowledge representation from pure philosophy."

There is a strong relationship between some specific ontology and the logical rules and computational artifacts that use that ontology, in that when they communicate among themselves, they have some level of assurance that the same terms have the same meanings to all. However, this use requires that the logical rules and the computational artifacts have explicit linkages with the ontology; often in the form of hard-coding the ontological terms into the rules and/or the application code itself.

In an agent-based system, common ontologies specify the ontological commitments of a set of participating agents [16]. An *ontological commitment* is an agreement to use a vocabulary in a way that is consistent with an ontology. An agent or human committed to an ontology understands (some subset of) the ontology and agrees to use it in a manner consistent with the semantics of the ontology. Agents and humans committed to the same ontology can share knowledge among themselves with some confidence that they share an underlying understanding of what is being said. Commitment to common, shared ontologies facilitates openness in an agent-based system.

In this paper, we examine the conflicting requirements and goals of ontology designers, ontology-committed applications, and ontology-aware users, and their respective impact on the problem of ontology commitment and reuse. As ontological sharing and reuse increases, the gap between the ontology designers and the ontology users grows larger. Our goal is to analyze what issues inhibit reuse and to propose strategies for facilitating reuse. In particular, we consider the problem of reuse of ontologies whose specification is complete, for applications whose requirements were not considered during the design of the ontology. This problem is not addressed in the ontology design methodologies summarized in [17]. We develop guidelines and approaches for agents to use existing ontologies in a more flexible manner.

Throughout the paper, we relate the issues to real issues we have encountered within the context of one of our applications, EDEN [3]. EDEN[1] is an agent-based system developed for the purpose of inter-organizational sharing of environmental data collected, stored and monitored by multiple government agencies and non-government scientists spread throughout the US and Europe, and relating information from these disparate data sources and schemas at a semantic level as needed by the users. EDEN uses ontologies to represent the semantics of the underlying information, and real and varied databases to populate those ontologies with instances.

2. ONTOLOGICAL COMMITMENT

Because ontologies are meant to facilitate sharing and reuse of knowledge, it is important that the ontology and its collection of users (both human and agent) align themselves to a shared view of the domain during the process of designing and evolving the ontology for that domain. However, many existing ontologies have been developed either by designers attempting to characterize a domain (with no real computational applications that use them) or by application developers to support individual applications (with no real sharing of the ontology with other

[1] EDEN was funded jointly by the DOD, DOE, EPA and EEA.

applications). The plethora of existing ontologies argues that many concepts are already represented within some ontology, so reuse of these ontologies, increasing the ontological commitment level, is now feasible. For example, some ontologies such as the Unified Medical Language System (UMLS) Metathesaurus [7] have achieved a higher level of commitment.

There are several issues that impede this sharing and knowledge. First are issues related to the conflicting goals and objectives of ontology definers, ontology-committed applications, and users of such applications. Second are issues of the mutual conflict between ontological commitment and ontological evolution. Third are issues of conceptual mismatches between ontologies and the applications and users that use them. Unfortunately, these issues stem from fundamental issues and characteristics of the problem of sharing ontologies so broadly.

2.1 Conflict: Definer, Application, User

The commitment to ontologies is hampered by the conflicting goals of ontology definers, developers of ontology-committed applications and ontology-committed application users, and often by the confusion of users and definers over the demands of ontology-committed applications. Here, we use the descriptor "ontology-committed" to mean that something purports to use the terms in the ontology(-ies) in a manner consistent with its (their) definitions.

The goal of the *ontology designer,* at least one working towards maximizing the usefulness of his ontology to a wide variety of applications, is to completely characterize a particular domain at the semantic level. The ontology designer needs expertise in knowledge representation and in the domain of the ontology. His intent is to develop a comprehensive and up-to-date ontology, with a broad set of acceptable terms. His job is hampered by the fact that the different domain experts have different viewpoints of the domain, and these viewpoints must be reconciled within the ontology itself, placing pressure on the designer to either take a commonly agreed-upon but consistent subset of the domain, or to attempt to placate everyone by including everything. The likely result is either the ontology will express concepts at a higher level than can be used effectively by an application that seeks to attach suitable labels to real data, or the terminology within the ontology will not be logically consistent and thus easily incorporatable into an application. Fortunately, our experience indicates that many terminology disagreements can be addressed by synonymy. For instance, what the EPA refers to as a "site" the DOD calls a "facility." What the EPA calls an "operable unit" (a specific location contained in an EPA "site") the DOD calls a "site."

The goal of the *application designer* is to use the ontology to support an application. The application designer has expertise in building applications, especially within given domains. His intent is to develop an efficient and useful application. It is likely that the application will not cover the whole domain, or may span multiple domains. Also, applications need to focus on some issues that are unimportant to the designer. For instance, a failure to take sufficient care of value representation issues in the ontology may force the application designer to code this information directly into the application, even though it is really domain-related knowledge. For example, within the Environmental Data Registry [4], it was difficult to relate measurements of chemicals between different representations because the measurement ontology was "well understood," and often explicit in the data representations themselves. Even terms such as "milligrams per kilogram" were sometimes represented inconsistently in text fields. This posed a problem in extracting values for computation in this "well understood" measurement system.

The desire of the *application user* is to be able to do his job well and easily. Application users have expertise in their own jobs and potentially in the domain of the application, but may have minimal understanding of knowledge representation or application design issues. Typical users want an understandable, natural, easy-to-use interface to the application that facilitates their work. This implies that they want the scope of the ontology restricted to the exact area within the domain that they are specifically concerned with. Another issue is that a user may have a comfortable vocabulary of domain-related terms that do not map well to the ontology representation model, to the domain model that the ontology developer had in mind or to the needs of the application itself.

2.2 Conflict: Commitment, Evolution

A second hampering factor when considering ontological commitment itself is the fact that ontological commitment impedes ontology evolution, and vice versa. Yet, both of these are necessary for an ontology to be successful. As Mark Tuttle et.al. say, "if you don't use it, it won't improve, if you don't improve it, it won't get used." [13]. In other words, use brings out the need to change, change is required for continued use.

As stated before, a measure of ontological commitment is the number of agents and humans that are committed to using the ontology. Each of these committed entities has made the effort to learn the ontology and to use the concepts therein in a manner consistent with their definition. However, the ontology itself may have problems, or may be incomplete in some aspect, or not reflect changes in the domain itself. In these situations, it becomes desirable at some point to evolve the ontology, providing an updated version. Initially, there is no real commitment to the updated version; rather, the entities committed to the old ontology must explicitly learn the new ontology, and adapt to the changes therein. This leads to the observation that, *as commitment to an ontology rises, so does the level of effort required to accommodate an evolutionary step.*

Development of an application represents a strong degree of ontological commitment. The incorporation of an ontology into an application may require specific coding with respect to the ontology, so application developers tend not to appreciate it when the ontology evolves. This tends to discourage users from employing the precision they need to express their views of the data in an application. In EDEN, the first version of our ontology easily evolved into the second, because there were only three small database fragments and a single user view in the initial demonstration. As the complexity of the user interface increased to make multiple views possible, and as we added larger, more complex resources, the task of evolving the ontology from version two to version three became more demanding due simply to the numbers of agents impacted.

2.3 Impedance Mismatch

There are several areas where mismatch and other problems seem to crop up. This problem has been studied extensively within the heterogeneous database community [19,20]. Some examples of these issues are as follows:

Uneven concept granularity: Some ontologies and other dictionaries were developed with a focus on one particular aspect of the domain. This aspect was well-developed and well-tested, but other aspects of the same domain are ignored. For instance, the ontology could have very strong and detailed concepts for contaminants and hazardous waste sites on land and in lakes, but not for ocean contaminants. However, a general application concerning toxic waste sites would need all of these concepts.

Concept modeling mismatch: Some applications and information sources model individual concepts differently from others, thus it was hard to develop a single ontological concept that would relate well to all the ways it could be represented. For instance, the concept of a set of measurements of contaminant levels taken daily could be represented in the ontology and/or the data itself either with a class of measurement vectors (one instance per day), or a class of measurements, with one instance per measurement type per day, specialized on measurement type.

Value representation mismatch: Many ontologies we have found do not even consider issues relating to the representation of the values of the attributes in the instances, a requirement for many applications. For instance, it would be easy to say that "area" was an integer; therefore setting its range as the range of positive integers. However, the Americans measured area in square feet, etc, while the Europeans measured it in square meters, etc. The application needed to know in which unit the integer in a particular instance of the area is represented, and how the units relate. These concepts should therefore exist in the domain ontology. In a more complex example, one information source in the EDEN application had "soil" as one of the media that could be polluted; but a second had "topsoil" and "subsoil" as separate concepts. A third site had multiple, more fine-grained classifications for soil. These did not all map easily onto any single ontological representation for soil, as mapping between the different concepts required additional knowledge (e.g. the depth of the soil), and sometimes was lossy (e.g. if everything was just mapped to "soil").

Terminology mismatch: Sometimes different sets of users use the same term for different concepts, or the term best suited to the user is already present in the ontology with a different meaning. This means that necessary concepts are sometimes omitted because the lexical terms needed to express them in human language are all present in other usages. For instance, the UMLS set of Semantic Relations was slow to develop the concept of genetic relationship because the terms parent and child existed for their use in logical tree structures.

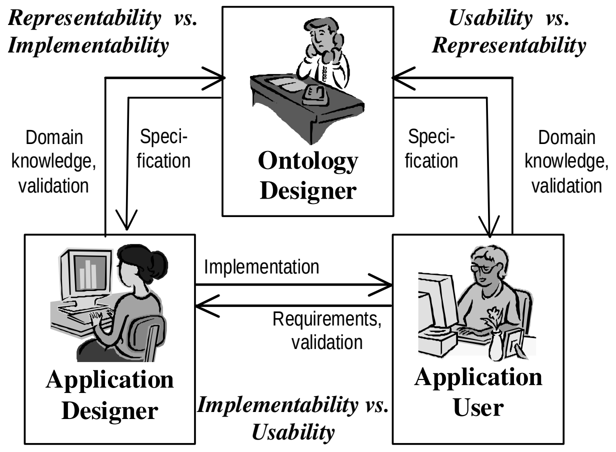
All of these issues conspire to make ontology sharing challenging.

3. ONTOLOGY DEVELOPMENT ISSUES

For ontological commitment to become real, ontology designers, application designers and users within a domain need all to be attracted to the same ontology. This affects not only the original design, but also the process of evolving the ontology.

3.1 Design and Representation

Ideally, the design and representation of an ontology should be done in collaboration with users and applications; the applications then can incorporate the ontology as designed, and the users can validate the concepts, relationships and axioms that they access in the ontology via the application. These interrelationships are shown in Figure 1.



Representability vs. Implementability
Usability vs. Representability
Domain knowledge, validation
Speci-fication
Ontology Designer
Speci-fication
Domain knowledge, validation
Implementation
Requirements, validation
Application Designer
Implementability vs. Usability
Application User

Figure 1. Ontological Interactions

Partly because of the desire to maximize ontological commitment, it is normally the goal of the ontology designer to represent the domain as completely as possible. Ontologies such as UMLS or environmental thesauri such as the General Multilingual Environmental Thesaurus (GEMET) [6] or the Terminology Reference System (TRS) [12] provide a broad base for discourse over their relative domains, medicine and the environment. Their definition has largely been driven, not by the implementers of the computer systems that use the ontologies, but rather by the practitioners within the domain itself. An example of the practical impact of this high-level design on the EDEN system was the design of value mapping across conceptual domains. The Environmental Data Register's (EDR) [4] implementation of its value domain model was fully normalized. This produced response times that were suitable for individual term lookups and mappings, but the performance scaled inadequately when several agents were performing simultaneous large-scale term conversions for entire tables. To obtain any tolerable performance at all, we were compelled to revise the ontology of the conceptual domain to support a flatter, non-normal data structure.

3.2 Evolution

As application developers develop applications using ontologies, and users use those applications, they identify weaknesses, missing concepts, and different issues that need to be addressed. For instance, when we developed our initial environmental ontology, some of the users who were used to the terminology in GEMET requested that we evolve our own ontology to be consistent with GEMET, or otherwise incorporate its terms. This request mandated an incorporation of that ontology into the environmental ontology. Once this incorporation was complete, all agents had to be adapted to the new ontology (this required a varying amount of effort depending on how closely the agent was hardwired to the ontology). Some of the users who were not used to the GEMET terminology also had to learn it. Thus, the evolution of the ontology engendered quite a bit of effort on many people. One could envision that the evolution of an ontology that has a higher level of ontological commitment, such as UMLS, could get very complex.

Clearly, the evolution of an ontology to a new version must be done carefully and collaboratively, involving the same types of as were involved in the original design. This indicates a slower, more coarse-grained type of evolution, which in turn affects the contents of the ontology. An ontology may contain several types of information useful to the domain that it represents. These include:

- Concepts/classes, their attributes, and the types of representations their attributes can take.
- Relationships between concepts, including subclassing, synonyms, and containment.
- Axioms and functions over those concepts.
- Instances of those concepts the relationships between them, and axioms over the instances.

The first three of these are referred to as the *schema information* (also known as the conceptual model or meta-knowledge) and the last is referred to the *instance knowledge*.

Experience has shown us that instances, because they represent changing objects in the real world, tend to evolve much faster than the concepts themselves. Because of this, it is easier to factor the ontology into two pieces, the schema and the instances. Within the EDEN application, the environmental ontology itself only contained schema-level domain information, and we populated the schema using information from different environmental databases. Because the agents themselves naturally were written only against the schema concepts, and the schema-level ontology information evolved more slowly, ontology evolution did not create as significant a problem as it could have. This experience has carried over into numerous other applications developed using the InfoSleuth agent system [8,3]; thus, we recommend this factoring unless the instance information is guaranteed to be stable (e.g., constants).

Ontological representation systems maintain this natural factoring between schema and instances. In OIL [9], the schema level is encapsulated within Standard OIL. OKBC clearly distinguishes between schema and instances in its own vocabulary. Therefore, there is no real barrier to this factoring approach.

4. COMPOSITIONAL ONTOLOGIES

Given that the best approach to ontology design and use is a collaborative one among ontology designers, application developers and users, let us now turn our attention to the issue of reuse. Consider an application developer, developing an application to fulfill some of the needs of a specific class of users in a given application domain. The developer has decided to reuse as much already-defined ontological information as he can within his application, tagging the imported concepts back to the ontologies from which they were taken to facilitate the process of evolving the application within the ontology. Unfortunately, up to this point the application developer has had no input whatsoever into the ontological design, though he has been working closely with the prospective application users. Thus, the necessary collaboration between the ontology designer, the application developer and the users is missing in this situation. Furthermore, the users' input into the application design is much more direct than that of the ontology designer. The result often is that there is no good existing ontological "match" to the application and/or its users.

Our experiences with our own ontology-based agent applications has led us to the following observations:

1. Reusing ontologies is necessary for acceptance by domain specialists, but many mature ontologies specify a lot more than current agent systems can actually use.
2. Most of the applications we have developed require concepts from multiple ontologies.
3. Most of the applications that we have developed require some concepts defined in no ontologies, because of mismatch issues.

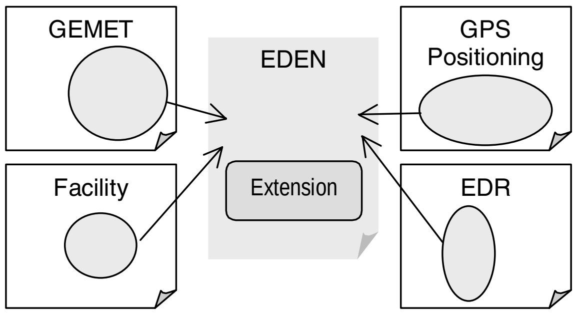


Figure 2. A Compositional Ontology

In order to facilitate reuse, we have designed our applications to use a compositional notion of ontologies, where we select concepts from different ontologies as needed, and supplement the result with any new concepts that we cannot locate elsewhere. This leads us to support three operations over ontologies themselves, which we term *subset*, *compose* and *extend*, respectively, as shown in Figure 2.

4.1 Subset

Agents are frequently coded as specialists, understanding a focused subset of the domain itself. For example, an agent that is interfacing with information repositories on environmental remediation techniques would not necessarily understand the information related to companies and their responsibilities for cleaning up specific toxic waste sites; yet the scope of the domain encompasses both areas. These agents need not incorporate entire ontologies; in fact, this may not even be possible for lightweight agents using large ontologies. Standards such as FIPA [5] allow agents to advertise the ontologies they understand, However, such an advertisement may be misleading to other agents if the agent only understands a *subset* of the ontology, in that two agents advertising the same ontology may in fact understand orthogonal subsets of the ontology, rendering communication impossible.

Furthermore, too much ontology information may also confuse users. As a practical matter, users of several of our agent applications including EDEN were confused by the presence of ontological terms for which no agent had advertised any knowledge. The presence of the term implied to them that they should be able to extract information (other than definition and ontological relationships) relevant to the term. That no agent had advertised the term was an unsatisfying explanation. Because of this, the agents themselves that implement the ontology must be careful how they present themselves to their users.

We define this subsetting using *ontological fragments*. An ontological fragment consists of:

- The name of the ontology
- The classes or entities in the ontology that are supported within the application, and their superclasses.
- Constraints on those classes including such things as ranges of values allowed for specific slots
- The axioms that reference the supported classes/ attributes.

The fragment can be expressed as a set of constraints over the classes and attributes. As an example from the EDEN system, each agent that provided a wrapper for a data resource advertised only a fraction of the domain ontology – precisely that portion for which it could provide data instances. In some cases, a wrapper agent advertised only a fraction of the slots of a class, and further placed semantic constraints on the content of the slots based on knowledge of the data instances about which it could report. In other cases, an agent might fill a slot with a static value from the canonical data representation for that slot. For example, because there was no record in a particular remediation database for land use, one of the attributes of a site, certain government database wrappers returned the generic value "Federal Facility" which was one of the values used in other databases.

Another example decision that needs to be made on any system is whether a request for everything matching a query should return an empty set if an agent does not advertise all elements of a class, or should return nulls for the unadvertised elements. This provoked strong disagreement on our team between those who felt that the former was algebraically preferable and those who felt that the latter was more intuitive to users already intimidated by the challenge of understanding the nature of the results produced by a dynamic distributed information system.

4.2 Compose

Another issue that is brought about by ontological commitment is the issue that an agent often requires access to terms from multiple ontologies. There exist a growing number of useful and public ontologies, and it often seems more appropriate to pick and choose terms from those ontologies than to build a new one from scratch, with just the terms that you need for your ontology. The result is a virtual ontology that imports *fragments* of other ontologies, into some sort of unified and useful whole. For example, within the EDEN application the diverse ways of representing values relating to various concepts such as chemical or geographic identifiers made it necessary to compose the domain ontology incorporating terminology from the field of hazardous waste pollution and remediation with the EDR ontology for value representation.

The ability to compose ontologies has an immediate benefit in modular ontology design. For instance, if you look within the EDR tags, there are tags relating to location (street address, postal code) as well as geographic (latitude and longitude). These tags are not specific to the environmental domain, but are shared among other domains such as address books and GPS positioning systems. Furthermore, such structures seem relatively stable. This indicates that it may be much more useful to have, say, a single ontology for geographic positions, their representations, and the relationships between them that can be incorporated into environmental applications as well as those in other domains.

4.3 Extend

Unfortunately, applications frequently need to incorporate concepts that are not represented in any ontology, or they may need to "glue" concepts from different ontologies together using new concepts. Thus, it is not sufficient to compose subsets of ontologies; frequently it is necessary also to add some new concepts. For example, within the setting of our EDEN application, the domain ontologies covered the concepts for which users wanted to retrieve information; for instance, they enabled the user to answer questions such as, "What toxic waste sites are within 10 miles of Houston, TX?" However, the agent-based system that supported the application also was able to ask higher-level, more abstract questions such as "Are the number of toxic waste sites within 10 miles of Houston increasing faster than they can be remediated?" These higher-level questions used concepts present neither in the domain ontology, nor in any other ontology available at the time.

The important issue with extension is to do it in a manner that will not create further problems later. There are two ways to extend an ontology; extensions that are *safe* and can easily be incorporated into other applications, and extensions that are *unsafe* and should eventually be agreed upon and then folded back by consensus into the ontology. We define safe and unsafe extensions with respect to the permissible definitional changes with respect to the underlying ontology(-ies). A safe extension can be characterized as follows:

Safe extension: an extension incorporating existing concepts from one or more ontologies, where any new concepts are defined axiomatically against the existing concepts in such a way that instances of the new concepts can be determined computationally from instances of the existing concepts. The transformation axioms must be invertible; the conversion of any instance between its view in the existing concept(s) and its view in the new concept must be lossless in both directions.

Safe extensions are "safe" primarily because they can be shared among the agents easily, simply by sharing the axioms that allow for conversion from the existing classes to the extended classes. We note that new concepts within such a safe extension cannot have new attributes that would need to be present in any instance of that concept, as this would violate the lossless conversion property. So, for example, "milliliters per liter" would be a safe extension for a concept if the concepts "water pollutant" and "parts per million" were already present in the ontology being extended.

In contrast, unsafe extensions are "unsafe" because they are harder to share with other agents, as the other agents may not maintain the information necessary to populate instances of the unsafe classes. For example, adding a new concept, "Water pollutant" with a new property, "concentration" would not be a safe extension if there were no concept of containing medium in the original ontology. Other agents, committed to the ontology, may not have the information necessary to provide a concentration for each pollutant they have, or may have different ways of modeling the concept of how badly the pollutant is contaminating the water.

Because unsafe extensions allow agents to instantiate classes that have been extended unsafely and thus cannot easily be shared; these extensions limit the openness of the agent. However, unsafe extensions are sometimes necessary; for instance, they are often required to deal with some of the impedance mismatch issues discussed earlier. Allowing unsafe extensions increases the possibility that different agents will create divergent extensions to the ontology, and become incompatible with one another. Therefore, unsafe extensions should be treated with caution. One approach is to limit such extensions to those that are *monotonic* [16] -- not requiring modifications to existing ontological elements. Idealistically, the need for specific unsafe extensions should be propagated to the ontology designers, who can determine whether or not there is a consensus on the need for the particular extension, and whether they should be incorporated into the next version.

4.4 Example

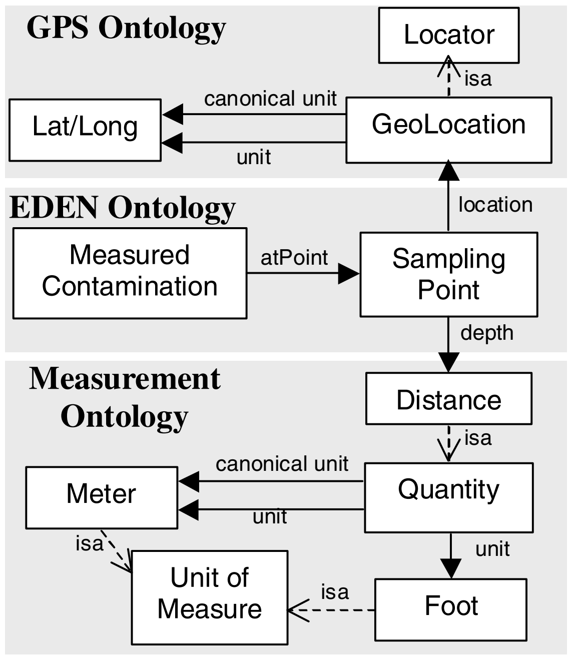


Figure 3. A part of the EDEN compositional ontology.

Figure 3 shows a fragment of an example compositional environmental ontology closely related to the one we used in the EDEN application. This ontology used subsets of the GPS and Measurement ontologies, composed together with a fragment of the environmental ontology. The necessary concepts for this particular application span all three ontologies. The gray boxes indicate the three ontologies, and the white boxes within indicate the classes taken from the individual ontology into the compositional ontology. Note that the attributes from the "Sampling Point" class refer to concepts in the other ontologies.

5. IMPLEMENTATION ISSUES

Given that we allow the operations of subsetting, composition and (safe) extension over ontologies within an agent system, the agents themselves need to be able to represent exactly what combination they are committed to, in order to ensure that they do not mislead others with respect to their capabilities.

There are three issues with representation and implementation. The first is that the agent needs to be able to represent internally the full set of details on how the ontology fragments it is using fit together. This can be done with a suitable abstract ontology representation. Secondly, the agent must be able to converse in some detail with other agents about the parts of the ontologies that they understand. This must be done using a suitable exchange representation. Thirdly, messages should be able to represent the ontology *fragments* that the message covers. This requires a more specific ontology field within the agent communication language.

5.1 Internal Representation

Internally, an agent needs to understand in detail the specific schema-level knowledge to which it is committed. This understanding may be complicated by the fact that ontological knowledge may be represented using many different representational models; for instance, there are subject-verb-object models such as DAML, frame-oriented models such and OKBC, and object-oriented models such as UML [14, 1]. These all differ in key issues such as whether or not attributes are first-class objects, whether or not multiple inheritance is allowed, and how strongly-typed classes must be. Further complicating this, if the operations of subsetting, composition and extension are used, a single application may look into a set of ontologies whose underlying ontological models may diverge.

Recent discussions have included the notion of developing abstract ontology representations for the internal representation of ontological knowledge. As with all applications that attempt to span multiple viewpoints, key choices in specifying an abstract ontology representation revolve around whether it should incorporate every modeling features present in some ontology representation, or it should only look at the modeling features used by every representation, or it should take some path in the middle. Wilmott et.al. [15] discuss in depth some of the issues associated with the specification and implementation of an abstract ontology representation from a theoretical standpoint. Fortunately, an abstract ontology representation is implemented within an ontology service, and thus only needs to incorporate the requirements of the agents that it serves. In the InfoSleuth system, our ontology service focused on frame-based ontologies, as they were most compatible with the databases with which we were working.

Internally, an agent that is using ontologies need only store the subsets of the ontology that it needs, and the glue that puts them together. So, for instance, in the ontology of Figure 2, the agent itself only needs to understand the concepts in the shaded areas of the GEMET, GPS Positioning, Facility and EDR ontologies, as well as the additional information associated with their composition and extension. Managing this information in an efficient way is the work of the ontology service.

Our experience has shown us that a good abstract ontology representation and a good ontology service are essential to cope with the incorporation of existing ontologies and their evolutionary steps. The abstract ontology representation engenders a unified methodology by which the agent can absorb existing ontology information and its particular use within the application. Agent implementers then use the ontologies in a more declarative and less hard-wired manner, which in turn facilitates the incorporation of new ontological information.

5.2 External Representation

Agents need to be able to share which parts of the different ontologies that they use or require, and how they put them together, with their other collaborating agents. This happens when the agent advertises its capabilities to another agent, when an agent is looking to locate another agent that can help it with some task, or when agents are negotiating over who is responsible for what subtasks. We have identified three basic ways that ontologies may be adapted dynamically: subsetting, composition, and extension:

Subsetting: The notion of subsetting an ontology is not reflected in the definition of the ontology itself, but rather in artifacts related to the *use* of the ontology. For example, fragments may be *advertised* from one agent to another, so that agents can inform other agents within their community of their exact capabilities. Alternatively, an application may be supported at the user end by an agent that interfaces with the user, ensuring that the user is presented only with the fragments of the ontology that are supported within the underlying agent community. Any user interfaces can then be tailored to the ontology fragment.

Composition: As with subsetting, composition should be specified at the time of use, however, there are some representational difficulties at this point. While subsetting can be defined as a set of axioms that overlays a single ontology, composition spans multiple ontologies and thus presents different representational challenges. For instance, a composition may wish to state that the *values* of the "LatitudeMeasure" slot for the "FacilityIdentification" in the EDR may be understood by the agent as represented in the "LatLongCoordinate" units defined in the "GPS Coordinates" ontology. In EDEN, we defined a geographic location as comprising a (selectable) coordinate scheme and a coordinate value.

OKBC and DAML present different challenges with respect to composition. Within OKBC, composition must be done by explicitly importing the component ontologies. Existing tools do not necessarily facilitate importing ontologies from remote locations, or ones that are represented in different languages (e.g., DAML). The resulting new ontology, even if it can be defined is a potentially huge superset of what is actually needed by the agent. Furthermore, it may not be possible to relate the terms in the new ontology back to the ontologies that it imported, so it may be computationally difficult to re-relate terms imported from the same ontology into two such composed ontologies. DAML+OIL, on the other hand, uses namespaces to facilitate the combination of ontologies and the relating of concepts among multiple ontologies. However, this ability to use namespaces relies on having the component ontologies' contents also within a namespace.

Extension: Safe extensions by definition should be able to be specified in terms of logical axioms over existing concepts in the ontology. While axioms are strongly considered among ontology designers, many ontology exchange languages hardly consider them at all; for instance, DAML+OIL at the moment has no good foundation for representing axioms and is thus unsuitable for exchanging knowledge concerning safe extensions.

In EDEN, we prepared a number of ad hoc lexical translation processes to translate between an old version and an extended version of the ontology. This did not yield a reusable method.

5.3 Representation in ACL Messages

The last area of application impact concerns the communication of ontological reference information in ACL messages. Currently, agent communication languages such as the FIPA ACL [5] carry a specification of the ontology from which the message content takes its concepts in an ":ontology" property. Compositional ontologies do not map directly to a single ontology, so the message format needs to change in one of the following ways:

1. Give the compositional ontology a name and make it explicitly available, and use this in the message's :ontology property.
2. Specify all ontologies using multiple :ontology properties. This may not cover concepts in safe extensions to the ontology.
3. Allow the ontology field to contain a description of the concepts in the in the ontologies, including extensions, that are used in the message. This could result in unnecessary performance impact on communicative actions.

6. CONCLUSIONS

Ontological commitment is the decision by a particular group of applications or users within an application domain to use the terms defined in a given ontology. High ontological commitment occurs when many users and/or applications within an application domain commit to sharing the same vocabulary of concepts, meanings and relationships defined within a specific ontology. Ontological commitment to a common set of ontologies is a key feature to provide for openness in an agent system, as it provides a common vocabulary over which agents can converse.

This paper focused mainly on two areas relating to ontological commitment and use. The first area is the clash of goals between ontology definers, application developers and users. Ontology definers are concerned with the completeness and purity of their ontological design. Application developers are often concerned only with specific subsets of the ontology that relate directly to the application, and with application-specific representational and computational issues. Users are concerned mainly with sticking with known and familiar terminology and vocabularies, which may not be logically structured and therefore may not be amenable to being reformulated into computationally-accessible concepts. This clash is complicated by the widening gap between these groups – developers, for instance, may be using ontologies that are already well-established. Thus, while several good design paradigms involving ontology designers, application developers, and users, exist for initial ontology development [16,17,18], none of these seem to encompass these more long-term concerns. We recommend the development of an extended ontology lifecycle that fosters evolution. Components required to support this lifecycle should include long-term ontology design support, a widely-available feedback channel from developers and users back to designers, and an open forum for discussing issues and extensions to the ontology.

An application committed to reuse existing ontologies may encounter several difficulties, as search for the perfect ontology for the application – one that is acceptable to both the application and its users – is not necessarily fruitful. We described a *compositional* approach to ontological use. Compositional ontologies base themselves in existing ontologies as much as possible, using the operations of ontological subsetting, ontological composition and ontological extension to tailor them to the specific needs of the applications. Compositional ontologies fit well with the needs of the application, and the approach has the potential to raise the level of ontological commitment to existing ontologies. However, they require more sophisticated ontology-related exchanges among collaborating agents. In order to incorporate these compositional ontologies, further work needs to be done on the development of a more formal algebra to support the subset, compose and extend operations over ontologies (in contrast to our current ad-hoc methods). Visser and Cui [19] attacked a related problem of heterogeneous ontology structures, and this may provide further insight into this formal algebra. As a second task, we need to develop a methodology for ensuring the correctness and consistency of these compositional ontologies.

Many approaches to the representation and specification of ontologies exist; some common ones for the exchange of ontological information include OKBC, DAML+OIL, OIL and its varieties, and UML. Each of these has features amenable to their adoption as a means to exchange information on ontological components and extensions, though none cover all of the issues addressed in this paper. Therefore, the exchange of such information using these standards is still problematic.

7. REFERENCES

[1] Cranefield, S., Haustein, S and Purvis, M. "UML-based ontology modeling for software agents". In *Proceedings of the workshop on Ontologies in Agent Systems*, Autonomous Agents 2001, Montreal, Canada, 2001.

[2] *DAML.org*. http://www.daml.org, 2002.

[3] Deschaine, L. M., Brice, R. S. and Nodine, M. H. "Use of InfoSleuth to Coordinate Information Acquisition, Tracking and Analysis in Complex Applications." In *Proceedings of Advanced Simulation Technologies Conference*, April, 2000.

[4] "EPA Environmental Data Registry". http://www.epa.gov/edr, 2002.

[5] FIPA. "Agent Communication Language Specifications". http://www.fipa.org/repository/aclspecs.html, 2001.

[6] "General Multilingual Environmental Thesaurus". http://www.nu.niedersachsen.de/cds/etc-cds_neu/library/select.html

[7] Humphreys, B.L. et.al. "Assessing and Enhancing the Value of the UMLS Knowledge Sources". In *Proceedings of the 15th Annual Symposium on Computer Applications in Medical Care*, pp. 78-82, Washington, D.C., November, 1991.

[8] Nodine, M. et.al. "Active Information Gathering in InfoSleuth." *International Journal of Cooperative Information Systems* 91/2, 2000, pp.3-28.

[9] "Welcome to the OIL Page". http://www.ontoknowledge.org/oil, 2002.

[10] "OKBC Home Page". http://www.ksl.stanford.edu/software/OKBC, 2002.

[11] Sowa, J. F. *Knowledge Representation.* Brooks/Cole, California, 2000, pp. xi-xii.

[12] "EPA Terminology Reference System." http://www.epa.gov/trs/index.htm , 2002.

[13] Tuttle, M. S. et al., "Merging terminologies". *Medinfo*. 1995;8 Pt 1:162-6.

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

[15] Willmott, S., Constantinescu, I. and Callisti, M. "Multilingual Agents: Ontologies, Languages and Abstractions." In *Proceedings of the First International Workshop on Ontologies in Agent Systems*, Autonomous Agents 2001, Montreal, Canada, May, 2001.

[16] Gruber, T.R. "A Translation Approach to Portable Ontology Specifications." *Knowledge Acquisition* 5(2), 1993.

[17] Gomez-Perez, A., "Ontological Engineering: A State of the Art." *Expert Update*, 1999.

[18] Gruninger, M and Fox, M. S. "Methodology for the Design and Evaluation of Ontologies." In *Proceedings of the Workshop on Basic Ontological Issues and Knowledge Sharing*, 1995.

[19] Hammer, J. and McLeod, D. "An Approach to Resolving Semantic Heterogeneity in a Federation of Autonomous, Heterogeneous Database Systems." In *International Journal of Intelligent and Cooperative Information Systems* 2(1), 1993.

[20] Kent, W. "The Many Forms of a Single Fact". In Proceedings of IEEE COMPCON, 1998.