

# Usage Scenarios and Goals for Ontology Definition Metamodel<sup>1</sup>

Lewis Hart<sup>1</sup>, Patrick Emery<sup>1</sup>, Robert Colomb<sup>2</sup>, Kerry Raymond<sup>3</sup>, Dan Chang<sup>4</sup>,  
Yiming Ye<sup>5</sup>, Elisa Kendall<sup>6</sup>, and Mark Dutra<sup>6</sup>

<sup>1</sup> AT&T Government Solutions, Inc., {lewishart, patemery}@att.com

<sup>2</sup> School of Information Technology and Electrical Engineering, The University of Queensland,  
colomb@itee.uq.edu.au

<sup>3</sup> DSTC Pty Ltd, kerry@dstc.com

<sup>4</sup> IBM Silicon Valley Lab, Santa Jose California, dtchang@us.ibm.com

<sup>5</sup> IBM Watson Research Center, Hawthorne, New York

<sup>6</sup> Sandpiper Software, Inc., {ekendall, mdutra}@sandsoft.com

**Abstract.** This paper contains a taxonomy of the uses of ontologies, intended as motivation for the Ontology Definition Metamodel development effort by the Object Management Group. It describes several usage scenarios for ontologies and proposes example applications for use in these scenarios. Many of the scenarios and applications are based on efforts currently underway in industry and academia. The scenarios descriptions are followed by goals for the Ontology Definition Metamodel.

## 1 Introduction

Ontology is a philosophical concept which was introduced into computing by the Artificial Intelligence community to describe data models which were conceptually independent of specific applications. Over the past decade the term has introduced into several other branches of computing where there is a need to model data independently of applications. With the advent of the semantic web movement [1] and the consequent development of ontology modeling languages like OWL by the W3C, the development of ontologies has become mainstream. Consequently, in 2003 the Object Management Group issued a Request for Proposal for an Ontology Development Metamodel, for a Meta-Object Facility (MOF-2) metamodel intended to support

- development of ontologies using UML modeling tools
- Implementation of ontologies in the W3C Web Ontology language OWL
- forward and reverse engineering for ontologies

---

<sup>1</sup> The work reported in this paper has been funded in part by the Cooperative Research Centres Program through the Department of the Prime Minister and Cabinet of the Commonwealth Government of Australia, and funded in part through the United States Government Defense Advanced Research Program Office's DAML program.

The authors of this paper were the original team established by the original submitters to the RFP, who are working together to develop a draft standard scheduled for delivery to the OMG in late 2004.

Early in the process, the team realized that there was not a comprehensive analysis of what ontologies were and what they were used for. Such an analysis is essential in development of any software, so our first step was to develop a usage scenarios and goals document (to use OMG terminology).

The usage scenarios presented herein highlight characteristics of ontologies that represent important design considerations for ontology-based applications. They also motivate some of the features and functions of the ODM and provide insight into when users can limit the expressivity of their ontologies to a description logics based approach, as well as when additional expressivity, for example from first order logic, might be needed. This set of examples is not intended to be exhaustive. Rather, the goal is to provide sufficiently broad coverage of the kinds of applications the ODM is intended to support that ODM users can make informed decisions when choosing what parts of the ODM to implement to meet their development requirements and goals.

This analysis can be compared with a similar analysis performed by the W3C Web Ontology Working Group [3]. We believe that the six use cases and eight goals considered in [3] provide additional, and in some cases overlapping, examples, usage scenarios and goals for the ODM.

**Table 1.** Perspectives of applications that use ontologies that are considered in this analysis.

<b>Perspective</b>	<i>One Extreme</i>	<i>Other Extreme</i>
Level of Authoritativeness	Least authoritative, broader, shallowly defined ontologies	Most authoritative, narrower, more deeply defined ontologies
Source of Structure	Passive (Transcendent) – structure originates outside the system	Active (Immanent) – structure emerges from data or application
Degree of Formality	Informal, or primarily taxonomic	Formal, having rigorously defined types, relations, and theories or axioms
Model Dynamics	Read-only, ontologies are static	Volatile, ontologies are fluid and changing.
Instance Dynamics	Read-only, resource instances are static	Volatile, resource instances change continuously
Control / Degree of Manageability	Externally focused, public (little or no control)	Internally focused, private (full control)
Application Changeability	Static (with periodic updates)	Dynamic
Coupling	Loosely-coupled	Tightly-coupled
Integration Focus	Information integration	Application integration
Lifecycle Usage	Design Time	Run Time

## 2 Perspectives

In order to ensure a relatively complete representation of usage scenarios and their associated example applications, we evaluated the coverage by using a set of perspectives that characterize the domain. Table 1 provides an overview of these perspectives.

We found that these perspectives could be divided into two general categories, those that are model centric and those that are application centric. The model centric perspectives characterize the ontologies themselves and are concerned with the structure, formalism and dynamics of the ontologies, they are:

- Level of Authoritativeness – Least authoritative ontologies define a broad set of concepts, but to a limited level of detail while the most authoritative ontologies are likely to be the narrowest, defining limited numbers of concepts to a greater depth of detail. More authoritative ontologies will represent safer long term investments and thus are likely to be developed to a greater depth.
- SNOMED<sup>2</sup> is a very large and authoritative ontology. The periodic table of the elements is very authoritative, but small. However, it can be safely used as a component of larger ontologies in physics or chemistry. Ontologies used for demonstration or pedagogic purposes, like the Wine Ontology<sup>3</sup>, are not very authoritative. Table 1 can be seen as an ontology which at present is not very authoritative.
- Source of Structure – The source of an ontologies structure can be defined by external sources (*transcendent*), or it can be defined by information internal to the data and using applications (*immanent*). SNOMED is a transcendent ontology defined by the various governing bodies of medicine. E-commerce exchanges are generally supported by transcendent ontologies. The set of topics used for searching a newsfeed are immanent, since they change as the news does.
- Degree of Formality – refers to the level of formality from a knowledge representation perspective, ranging from highly informal or taxonomic in nature, where the ontologies may be tree-like, involving inheritance relations, to semantic networks, which may include complex lattice relations but no formal axiom expressions, to ontologies containing both lattice relations and highly formal axioms that explicitly define concepts. SNOMED is taxonomic, while engineering ontologies like [2] are highly formal.
- Model Dynamics – Some ontologies tend to be stable, while others are likely to be modified dynamically by the agents or applications that use them. The periodic table of the elements is pretty stable, but an ontology supporting tax accounting in Australia would be pretty volatile at the model level.
- Instance Dynamics—refers to the degree that instances of classes in the information resources or knowledge bases that use the ontology change as a result of some action the application takes as it is running. The periodic table of the elements is stable at the instance level (eg particular elements) as well as the model level (eg classes like noble gasses or rare earths), while an ontology supporting an e-

---

<sup>2</sup> <http://www.snomed.org>

<sup>3</sup> <http://www.w3.org/2001/sw/WebOnt/guide-src/wine.owl>

commerce exchange would be volatile at the instance level but not at the model level.

Application centric perspectives are concerned with how application use and manipulate the ontologies, they are:

- **Control / Degree of Manageability** – refers to the scope of control of the application using one or more ontologies, and also of control over changes made in the ontologies or knowledge bases. The ontology evolution control may span organizations or operate inside a private firewall or VPN, For public ontologies there may be little to no control from an ontology evolution perspective. An e-commerce exchange may have a high degree of control over the product catalogs and terms of trade, but a low degree of control over value-added tax categories and payment regimes.
- **Application Changeability** – The ontologies may be applied statically, as they might be if used for database schema mapping, with periodic updates to support evolution in the schemas, or they may be applied dynamically, as in an application that composes web services at run time.
- **Coupling** – refers to the degree that the information resources or applications using the ontologies are coupled. In an e-commerce exchange the players are tightly coupled using the ontology, while different applications using the engineering mathematics ontology may have nothing at all in common at run time.
- **Integration Focus** – refers to the degree that the ontology is focused on the structure of messages implementing interoperability without regard for content (for example Electronic Data Interchange or EDI), application interoperability without regard to content (eg a shared database) or both (eg booksellers using publishers' product catalogs as ontologies focus on content as well as message structure).
- **Lifecycle Usage** – refers to the phase of a project life cycle in which the ontologies are used. This ranges from early design and analysis phases to being an active part of the application at run time. The engineering mathematics ontology would be used mainly in the design phase, while an e-commerce exchange may need to validate messages dynamically.

### 3 Usage Scenarios

As might be expected, some of these perspectives tend to correlate across different applications, forming application areas with similar characteristics. Our analysis, summarized in Table 2, has identified three major clusters of application types that share some set of perspective values:

- **Business Applications** are characterized by having transcendent source of structure, a high degree of formality and external control relative to nearly all users.
- **Analytic Applications** are characterized by highly changeable and flexible ontologies, using large collections of mostly read-only instance data.
- **Engineering Applications** are characterized by again having transcendent source of structure, but as opposed to business applications their users control them primarily internally and they are considered more authoritative.

**Table 2.** Usage scenario perspective values  
Characteristic Perspective Values

Use Case Clusters		Characteristic Perspective Values									
		Model Centric					Application Centric				
Description	Authoritativeness	Structure From	Formality	Model Dynamics	Instance Dynamics	Control	Changeability	Coupling	Focus	Life Cycle	
2.1	<b>Business Applications</b>	<b>Outside</b>	<b>Formal</b>			<b>External</b>					
2.1.1	<b>Run-time Interoperation</b>	Outside	Formal	Read-Only	Volatile	External	Static	Tight	Information	Real Time	
2.1.2	<b>Application Generation</b>	Outside	Formal	Read-Only	Read-Only	External	Static	Loose	Application	All	
2.1.3	<b>Ontology Lifecycle</b>	Outside	Semi-Formal Formal	Read-Only	Read-Only	External	Static	Tight	Information	Real Time	
2.2	<b>Analytic Applications</b>			<b>Volatile</b>	<b>Read-Only</b>		<b>Dynamic</b>	<b>Flexible</b>			
2.2.1	<b>Emergent Property Discovery</b>	Inside	Informal	Volatile	Read-Only	Internal & External	Dynamic	Flexible	Information	Real Time	
2.2.2	<b>Exchange of Complex Data Sets</b>	Inside	Informal	Volatile	Read-Only/Volatile	Internal & External	Dynamic	Flexible	Information	Real Time	
2.3	<b>Engineering Application</b>	<b>Outside</b>				<b>Internal</b>					
2.3.1	<b>Information System Development</b>	Outside	Semi-Formal Formal	Read-Only	Volatile	Internal	Changeable	Tight	Information	Design Time	
2.3.2	<b>Ontology Engineering</b>	Outside	Semi-Formal Formal	Volatile	Volatile	Internal	Changeable	Flexible	???	Design Time	

## 4 Business Applications

### 4.1 Run Time Interoperation

Externally focused information interoperability applications are typically characterized by strong de-coupling of the components realizing the applications. They are focused specifically on information rather than application integration (and here we include some semantic web service applications, which may involve composition of vocabularies, services and processes but not necessarily APIs or database schemas). Because the community using them must agree upon the ontologies in advance, their application tends to be static in nature rather than dynamic.

Perspectives that drive characterization of these scenarios include:

- The level of authoritativeness of the ontologies and information resources.
- The amount of control that community members have on the ontology and knowledge base evolution
- Whether or not there is a design time component to ontology development and usage
- Whether or not the knowledge bases and information resources that implement the ontologies are modified at run time (since the source of structure remains relatively unchanged in these cases, or the ontologies are only changed in a highly controlled, limited manner).

These applications may require mediation middleware that leverages the ontologies and knowledge bases that implement them, potentially on either side of the firewall – in next generation web services and electronic commerce architectures as well as in other cross-organizational applications, for example:

- For semantically grounded information interoperability, supporting highly distributed, intra- and inter-organizational environments with dynamic participation of potential community members, (as when multiple emergency services organizations come together to address a specific crisis), with diverse and often conflicting organizational goals.
- For semantically grounded discovery and composition of information and computing resources, including Web services (applicable in business process integration and grid computing).
- In electronic commerce exchange applications based on stateful protocols such as EDI or Z39.50, where there are multiple players taking roles performing acts by sending and receiving messages whose content refers to a common world.

In these cases, we envision a number of agents and/or applications interoperating with one another using fully specified ontologies. Support for query interoperation across multiple, heterogeneous databases is considered a part of this scenario.

While the requirements for ontologies to support these kinds of applications are extensive, key features include:

- the ability to represent situational concepts, such as player/actor – role – action – object – state,
- the necessity for multiple representations and/or views of the same concepts and relations, and

- separation of concerns, such as separating the vocabularies and semantics relevant to particular interfaces, protocols, processes, and services from the semantics of the domain.
- Service checking that messages commit to the ontology at run time. These communities can have thousands of autonomous players, so that no player can trust any other to send messages properly committed to the ontology.

## 4.2 Application Generation

A common worldview, universe of discourse, or domain is described by a set of ontologies, providing the context or situational environment required for use by some set of agents, services, and/or applications. These applications might be internally focused in very large organizations, such as within a specific hospital with multiple, loosely coupled clinics, but are more likely multi- or cross-organizational applications. Characteristics include:

- Authoritative environments, with tighter coupling between resources and applications than in cases that are less authoritative or involve broader domains, though likely on the “looser side” of the overall continuum.
- Ontologies shared among organizations are highly controlled from a standards perspective, but may be specialized by the individual organizations that use them within agreed parameters.
- The knowledge bases implementing the ontologies are likely to be dynamically modified, augmented at run time by new metadata, gathered or inferred by the applications using them.
- The ontologies themselves are likely to be deeper and narrower, with a high degree of formality in their definition, focused on the specific domain of interest or concepts and perspectives related to those domains.

For example:

- Dynamic regulatory compliance and policy administration applications for security, logistics, manufacturing, financial services, or other industries.
- Applications that support sharing clinical observation, test results, medical imagery, prescription and non-prescription drug information (with resolution support for interaction), relevant insurance coverage information, and so forth across clinical environments, enabling true continuity of patient care.

Requirements:

- The ontologies used by the applications may be fully specified where they interoperate with external organizations and components, but not necessarily fully specified where the interaction is internal.
- Conceptual knowledge representing priorities and precedence operations, time and temporal relevance, bulk domains where individuals don't make sense, rich manufacturing processes, and other complex notions may be required, depending on the domain and application requirements.

## 4.3 Ontology Lifecycle

In this scenario we are concerned with activity, which has as its principle objectives conceptual knowledge analysis, capture, representation, and maintenance. Ontology

repositories should be able to support rich ontologies suitable for use in knowledge-based applications, intelligent agents, and semantic web services. Examples include:

- Maintenance, storage and archiving of ontologies for legal, administrative and historical purposes,
- Test suite generation, and
- Audits and controllability analysis.

Ontological information will be included in a standard repository for management, storage and archiving. This may be to satisfy legal or operations requirements to maintain version histories.

These types of applications require that Knowledge Engineers interact with Subject Matter Experts to collect knowledge to be captured. UML models provide a visual representation of ontologies facilitating interaction. The existence of meta-data standards, such as XMI and ODM, will support the development of tools specifically for Quality Assurance Engineers and Repository Librarians.

Requirements include:

- Full life-cycle support will be needed to provide managed and controlled progression from analysis, through design, implementation, test and deployment, continuing on through the supported systems maintenance period.
- Part of the lifecycle of ontologies must include collaboration with development teams and their tools, specifically in this case configuration and requirements management tools. Ideally, any ontology management tool will also be ontology aware.
- It will provide an inherent quality assurance capability by providing consistency checking and validation.
- It will also provide mappings and similarity analysis support to integrate multiple internal and external ontologies into a federated web.

## 5 Analytic Applications

### 5.1 Emergent Property Discovery

By this we mean applications that analyze, observe, learn from and evolve as a result of, or manage other applications and environments. The ontologies required to support such applications include ontologies that express properties of these external applications or the resources they use. The environments may or may not be authoritative; the ontologies they use may be specific to the application or may be standard or utility ontologies used by a broader community. The knowledge bases that implement the ontologies are likely to be dynamically augmented with metadata gathered as a part of the work performed by these applications. External information resources and applications are accessed in a read-only mode.

- Semantically grounded knowledge discovery and analysis (*e.g.*, financial, market research, intelligence operations)
- Semantics assisted search of data stored in databases or content stored on the Web (*e.g.*, using domain ontologies to assist database search, using linguistic ontologies to assist Web content search)
- Semantically assisted systems, network, and / or applications management.

- Conflict discovery and prediction in information resources for self-service and manned support operations (*e.g.*, technology call center operations, clinical response centers, drug interaction)

What these have in common is that the ontology is typically not directly expressed in the data of interest, but represents theories about the processes generating the data or emergent properties of the data. Requirements include representation of the objects in the ontology as rules, predicates, queries or patterns in the underlying primary data.

## 5.2 Exchange of Complex Data Sets

Applications in this class are primarily interested in the exchange of complex (multi-media) data in scientific, engineering or other cooperative work. The ontologies are typically used to describe the often complex multimedia containers for data, but typically not the contents or interpretation of the data, which is often either at issue or proprietary to particular players. (The OMG standards development process is an example of this kind of application.)

Here the ontology functions more like a rich type system. It would often be combined with ontologies of other kinds (for example an ontology of radiological images might be linked to SNOMED for medical records and insurance reimbursement purposes).

Requirements include

- Representation of complex objects (aggregations of parts)
- Multiple inheritance where each semantic dimension or facet can have complex structure.
- Tools to assemble and disassemble complex sets of scientific and multi-media data.
- Facilities for mapping ontologies to create a cross reference. These do not need to be at the same level of granularity. For the purposes of information exchange, the lower levels of two ontologies may be mapped to a higher level common abstraction of a third, creating a sort of index

## 5.3 Engineering Applications

The requirements for ontology development environments need to consider both externally and internally focused applications, as externally focused but authoritative environments may require collaborative ontology development.

## 5.4 Information Systems Development

The kinds of applications considered here are those that use ontologies and knowledge bases to support enterprise systems design and interoperation. They may include:

- methodology and tooling, where an application actually composes various components and/or creates software to implement a world that is described by one or more component ontologies.
- Semantic integration of heterogeneous data sources and applications (involving diverse types of data schema formats and structures, applicable in information integration, data warehousing and enterprise application integration).

- Application development for knowledge based systems, in general.

In the case of model-based applications, extent-descriptive predicates are needed to provide enough meta-information to exercise design options in the generated software (*e.g.*, describing class size, probability of realization of optional classes). An example paradigm might reflect how an SQL query optimizer uses system catalog information to generate a query plan to satisfy the specification provided by an SQL query. Similar sorts of predicates are needed to represent quality-type meta-attributes in semantic web type applications (comprehensiveness, authoritativeness, currency).

## 5.5 Ontology Engineering

Applications in this class are intended for use by an information systems development team, for utilization in the development and exploitation of ontologies that make implicit design artifacts explicit, such as ontologies representing process or service vocabularies relevant to some set of components. Examples include:

- Tools for ontology analysis, visualization, and interface generation.
- Reverse engineering and design recovery applications.

The ontologies are used throughout the enterprise system development life cycle process to augment and enhance the target system as well as to support validation and maintenance. Such ontologies should be complementary to and augment other UML modeling artifacts developed as part of the enterprise software development process. Knowledge engineering requirements may include some ontology development for traditional domain, process, or service ontologies, but may also include:

- Generation of standard ontology descriptions (*e.g.*, OWL) from UML models.
- Generation of UML models from standard ontology descriptions (*e.g.*, OWL).
- Integration of standard ontology descriptions (*e.g.*, OWL) with UML models.

Key requirements for ontology development environments supporting such activities include:

- Collaborative development
- Concurrent access and ontology sharing capabilities, including configuration management and version control of ontologies in conjunction with other software models and artifacts at the atomic level within a given ontology, including deprecated and deleted ontology elements
- Forward and reverse engineering of ontologies throughout all phases of the software development lifecycle
- Ease of use, with as much transparency with respect to the knowledge engineering details as possible from the user perspective
- Interoperation with other tools in the software development environment; integrated development environments
- Localization support
- Cross-language support (ontology languages as opposed to natural or software languages, such as generation of ontologies in the XML/RDF(S)/OWL family of description logics languages, or in the Knowledge Interchange Format (KIF) where first or higher order logics are required)
- Support for ontology analysis, including deductive closure; ontology comparison, merging, alignment and transformation

**Table 3.** Summary of Requirements

<b>Requirement</b>	<b>Section</b>
<i>Structural</i>	
Support ontologies expressed in existing description logic, (e.g. OWL/DL) and higher order logic languages (e.g. OWL Full and KIF), as well as emerging and new formalisms.	2.1.2 2.2.1 2.3.2
Represent complex objects as aggregations of parts	2.2.2
Multiple inheritance of complex types	2.2.2
Separation of concerns	2.1.1
Full or partial specification	2.1.2
Model-based architectures require extent-descriptive predicates to provide a description of a resource in an ontology, then generating a specific instantiation of that resource.	2.3.1
Efficient mechanisms will be needed to represent large numbers of similar classes or instances.	2.1.1
<i>Generic concepts</i>	
Support physical world concepts, including time, space, bulk or mass nouns like ‘water’, and things that do not have identifiable instances.	2.1.2
Support object concepts that have multiple facets of representations, e.g., conceptual versus representational classes.	2.1.1
Provide a basis for describing stateful representations, such as finite state automaton to support an autonomous agent’s world representation.	2.1.1
Provide a basis for information systems process descriptions to support interoperability, including such concepts as player, role, action, and object.	2.1.1
Other generic concepts supporting particular kinds of domains	2.1.2
<i>Run-time tools</i>	
Tools to assemble and disassemble complex sets of scientific and multi-media data.	2.2.2
Service to check message commitment to ontology	2.1.1
<i>Design-time tools</i>	
Full life-cycle support	2.1.3 2.3.2
Support for collaborative teams	2.1.3 2.3.2
Ease of use, transparency with respect to details	2.3.2
Support for modules and version control.	2.1.3
Consistency checking and validation, deductive closure	2.1.3 2.3.2
Mappings and similarity analysis	2.1.3 2.2.2 2.3.2

- Support for import/reverse engineering of RDBMS schemas, XML schemas and other semi-structured resources as a basis for ontology development

**5.6 Goals for Generic Ontologies and Tools**

The diversity of the usage scenarios illustrates the wide applicability of ontologies within many domains. Table 3 brings these requirements together. The table classifies the requirements into

- structural features – knowledge representation abstract syntax
- generic content – aspects of the world common to many applications
- run-time tools – use of the ontology during interoperation
- design-time tools – needed for the design of ontologies

Associated with each requirement is the usage scenario from which it arises.

To address all of these requirements would be an enormous task, beyond the capacity of the ODM development team. The team is therefore concentrating on the most widely applicable and most readily achievable goals. In particular

- ODM is based on the OMG Meta-Object Facility (MOF), so can benefit from a wide variety of MOF-based design-time tools.
- ODM supports several widely-used modeling languages, including UML, RDFS/OWL, Entity-Relationship and Topic Maps, with inter-metamodel mapping, as well as the logic language Simple Common Logic (successor to KIF).
- MOF supports a package structure allowing optional compliance points, so that elements needed for a subclass of applications such as e-commerce can be provided, perhaps by third parties.

The resulting ODM will be not a final solution to the problem, but will be intended as a solid start which will be refined as experience accumulates.

**Acknowledgement.** The authors would like to thank John Kling and John Poole for review and comments.

## References

1. T. Berners-Lee and M. Fischetti. *Weaving the Web : the past, present and future of the World Wide Web by its inventor* London : Orion Business 1999
2. T.R. Gruber, and G.R. Olsen. An ontology for engineering mathematics, in Jon Doyle, Piero Torasso and Erik Sandewell, eds. *Fourth International Conference on Principles of Knowledge Representation and Reasoning*, Morgan Kaufmann, 1994.
3. W3C. *OWL Web Ontology Language Usage Scenarios and Requirements*, W3C Candidate Recommendation, 18 August 2003, <http://www.w3.org/TR/webont-req/>