Ontology Quality and Its Evaluation

Executive Summary

[To be added later. The intended audience of the executive summary is wider than the intended audience for the rest of the text.]

Introduction

Ontologies are human-intelligible and machine-interpretable representations of some portions and aspects of a domain. Ontologies are most often used as part of some larger IT system (itself part of some organizational or activity-based system), and in this document we focus on such use. Since they contain terms and their definitions, and enable the standardization of a terminology across a community or enterprise, ontologies can be considered as a type of glossary. Since ontologies capture key concepts and their relationships in a machine interpretable form, they bear similarities to domain models (in the software engineering sense). And since ontologies can be populated with or linked to instance data to create knowledge bases, and deployed as parts of IT systems for query answering, ontologies are resembling databases.

This versatility of ontologies is a big advantage of the technology. However, versatility also contributes to the challenge of evaluating ontologies. Ontology evaluation consists of gathering information about some characteristics of an ontology, comparison of the results with some set of ontology requirements, and assessment of the ontology's suitability for some purpose. Some ontology characteristics can be measured independent of usage; others involve how an ontology relates to some domain, environment, or usage-specific activity, and thus can only be measured with reference to some usage information. Moreover, measurement alone does not make an evaluation. Evaluation of an ontology requires: identifying which possible ontology characteristics are relevant to some usage and what requirements must therefore be met by the ontology; measuring these characteristics in particular; and determining, on this basis, to what degree the ontology fits the requirements for that usage. The variety in the potential uses of ontologies means that there is no single list of such requirements and no single approach to evaluating ontologies against them.

For use of an ontology in an IT system, however, we can identify some kinds of evaluation that are generally needed. To determine the quality of an ontology, we need to evaluate three facets of an ontology: the ontology as domain model for human consumption, the ontology as domain model for machine consumption, and the ontology as deployed software that is part of a larger system. In this document we will focus on four high-level questions:

(1) Can humans understand the ontology correctly? (Intelligibility)
(2) Does the ontology represent the domain appropriately? (Model quality)
(3) Does the representation of the domain fit the requirements for its intended use? (Model fitness)
(4) Does the deployed ontology meet the requirements of the IT system it is part of? (System fitness)

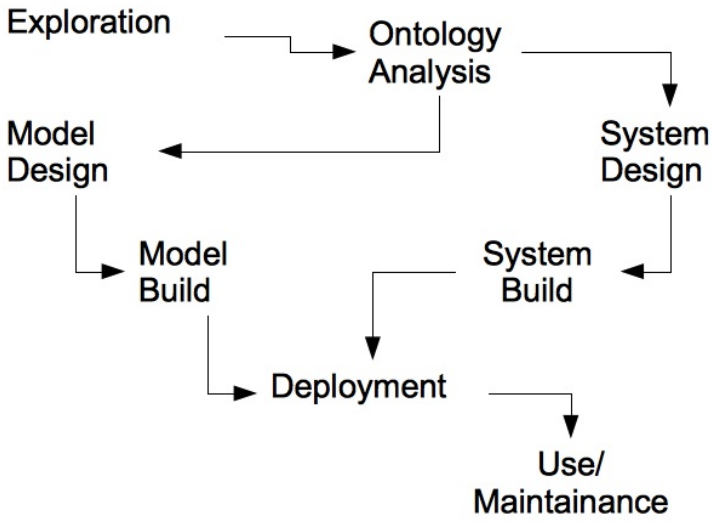
The purpose of this document is to provide an overview of the activities that need to occur during ontology development, deployment, and usage; and the critical relationship between evaluating these activities and the quality of the results. In the next section we present a breakdown of the ontology development lifecycle. In the following sections we identify the activities that happen during

each phase, the characteristics of the ontology that should be evaluated at the stage, and the ontology evaluation methods that are applicable. The documents ends with some observations about the current tool support for ontology evaluation, and recommendations for future work.

The Ontology Lifecycle -- an Overview

The lifecycle of an ontology is the succession of phases in which the ontology is being circumscribed, specified, developed, deployed, and used. While there is no single sequence that all ontologies follow, there are identifiable phases through which ontologies pass, usually iteratively. There are also phases, sequences, and iterations through which an ontology should go, for best quality and results in use.

The following diagram presents an overview over the major phases that occur within the lifecycle of an ontology. Each of these phases is discussed individually, in more detail, in one of the following sections.



The figure above presents a schematic view of the ontology lifecycle. While dependencies exist between some pairs of stages, there is not a single, necessary sequence that characterizes the ontology lifecycle. Moreover, ontology lifecycles may vary, as software lifecycles do, according to particular project methodologies that incorporate specific sequences, iterations or other process-organizing features.

Evaluation should happen throughout the ontology lifecycle; it plays two major roles: feedback-oriented and transition-oriented evaluation. The incorporation of evaluation processes into development stages is essential to provide feedback. This can prevent serious error accumulation and amplification by keeping development on-track. Feedback-oriented evaluation can be provided by tools for on-demand automated evaluation of products against subsets of requirements (e.g., consistency checking, small-corpus runs of application operations); the availability of domain and/or operational experts for consultation; tools for rapid, automated or semi-automated production of various visualization or other presentations of work product (for technical, domain, or operational expert review), and the adoption of methodologies that explicitly incorporate such small and frequent evaluative steps into development.

Transition-oriented evaluation have the purpose to determine whether the product of the current

phase meets applicable requirements sufficiently that this current phase can end and the next begin. There may be no qualitative difference between the evaluation techniques that distinguishes feedback-oriented and transition-oriented evaluation, but the transition-oriented evaluation process is more comprehensive.

The specific evaluations performed during each phase are dependent on the requirements that apply to that phase and to its outputs. Identification of requirements starts with the intended usage and continues in subsequent stages. Requirements identified during exploration are refined during the design process. Business requirements are transformed into technical requirements. Furthermore, the results of each stage may lead to discovery of additional requirements and revision of requirements previously specified. The continuing refinement of requirements may also reveal the need for a return to the earlier work products, including additional iterations of previous phases. For example, evaluations conducted during model building may reveal that the design has some major flaw; this discovery may lead to a design phase iteration that incorporates the new information. Information and process flows between phases can, in this way, occur in directions other than what is shown on the schematic diagram above. Furthermore, an ontology may undergo a full sequence of exploration-design-build-deploy-use, and the information and experience gathered during that sequence may serve as input to a following sequence in which a new release version, including improvement, fixes, and extensions to the ontology is designed, developed, and deployed. These sequences may themselves be overlapping; for example, exploration of a later release may begin while an earlier release is being built. Information may then flow not only in more than one direction across phases, but across entire sequences and cycles.

Some will concern the relationship of the ontology and aspects of the operational environment. For example, the adequacy of the scope of ontology coverage; inclusion of ontology content or logical features that support particular system operations; relationship to applicable standards and to other ontologies; and intelligibility of the ontology to the intended users. These characteristics can be understood and evaluated only with the benefit of both technical, ontological understanding and understanding of the domain and operational environment.

Some other ontology requirements, often derived from operational requirements and design decisions, concern characteristics entirely internal to the ontology itself. Once the requirements are determined, these characteristics (such as logical consistency, modularity, complexity, and formal relations support) can be understood and evaluated using technical, ontological understanding, without further input of usage-specific or domain-specific information.

Exploration will also identify requirements of a third kind: those that are entirely independent of the content or internal characteristics of the ontology. These characteristics, (such as access and security thereof, licensing conditions, cost) can be evaluated without technical ontological understanding. They are better understood as within the realm and expertise of project managers and technical experts of non-ontological sorts.

Evaluating whether an ontology meets requirements of this kind is not substantially different from evaluating whether any other entity meets those requirements. An ontologist's perspective has nothing particular to add to existing techniques for such evaluation. General engineering and project management best practices apply, without need for ontology-specific modification. For this reason, the remainder of this document focuses on evaluation of the first two types, in which specifically ontological considerations apply.

Exploration

The purpose of this phase is to establish rationale and requirements. The output of this phase is a document that answers the following questions: During the exploration phase, the intended usage is considered in detail, and requirements are derived from that usage. Typically, an intended usage is initially understood from a business or operational perspective. The intended usage may be specified as use-cases, scenarios; at early stages of exploration, it may be captured only as a brief statement of one or more operational needs and constraints. The exploration phase involves extending and clarifying initial information until the intended usage is sufficiently captured and understood to effectively guide technical decisions. This process involves an interplay of technical, operational, and project-sponsor understanding. Adequate exploration is critical to the success of any ontology development or usage. It is here that the grounding requirements for the ontology are identified. Without it, even the best possible execution of each following stage is unlikely to result in an ontology and/or system that meets the original need.

- Why do we need an ontology?
- What is the intended usage (e.g., specified as use-cases, scenarios)?
- What is the scope of the ontology?
- What are the requirement for domain representation (model fitness)?
- What are the competency questions?
- What are the requirements from the operational environment (system fitness)?
- What are the available resources for development and testing? (e.g., legacy databases, test corpora, data models, glossaries, vocabularies, schemas, taxonomies, ontologies, standards, access to domain experts)

Ontological Analysis

The purpose of this phase is to identify the key entities of the ontology (individuals, classes, and the relationships between them), as well as to link them to the terminology that is used in the domain. This involves usually the resolution of ambiguity and the identification of entities that are shared across different resources.

Evaluation

- Are all relevant terms from the use cases documented?
- Are all entities within the scope of the ontology captured?
- Are no entities outside the scope of the ontology captured?
- Are the defined classes and relationships well-defined? (e.g., no circularity.)
- Is the intended interpretation of the undefined individuals, classes, and relationships well-documented?
- Are the individuals, classes, and relationships documented in a way that is easily reviewable by domain experts?
- Do the domain experts agree with the ontological analysis?
- Is the documentation sufficiently unambiguous to enable a consistent use of the terminology? (Measurable by comparing text annotations by domain experts.)

Model Design Phase

Based on the requirements from the ontology analysis phase, in this phase where the basic modelling design decisions are made. These include the choice of an ontology language, the structure of the ontology, and the upper ontology and design principles.

The upper ontology and design principles determine the high level ontological categories and how some fundamental aspects of reality are represented (e.g. change over time). The structure of the ontologies determines how the ontology is separated into modules, and how the modules are supposed to interact.
The competency questions that are formulated in the exploration phase capture in natural language the kinds of queries that the ontology should support in given scenarios. The role of the individual ontology modules is clarified by formulating using the ontology-wide competency questions to formulate for each ontology module scenarios and competency questions that capture the intended behavior of the module.
Note that there might be a conflicting requirements for the expressivity of the ontology language and its performance (see system design phase). In this case one needs to distinguish between the reference ontology, which represents the domain faithfully in a language that is expressive enough for that purpose, and the implementation ontology, which might compromise the representation of the domain for the sake of performance.

Evaluation

- Is the chosen ontology language expressive enough to capture the knowledge sufficiently detailed to meet the requirements identified in the Ontology Analysis phase?
- Do the upper ontology and design principles meet established best practices?
- Do the ontology modules together cover the whole scope of the ontology?
- Are the competency questions representative for all intended usages?
- Are all modules of the ontology associated with competency questions?

System Design Phase

This phase involves basic design decisions about how the ontology is implemented and integrated within the larger IT system. The intended usage of the ontology (through all lifecycle stages), the means by which users interact with it, the operations to be performed using it, and the outputs of these operations all entail requirements for not only the ontology model but the characteristics of the ontology as a software artifact, and the larger system(s) within which the ontology is incorporated.The output of this phase should answer such questions as:
- What, if any, inputs to the ontology will there be, once the system is deployed? What interfaces (between machines or between humans and machines) will enable those inputs?
- What, if any, data sources, will be the ontology be used with? How will the ontology be connected to the data sources? What separate interfaces, if any, are needed to enable access to those connections?
- How will ontology be built, evaluated, and maintained? What tools are needed to enable the development, evaluation, and maintenance of the ontology?
- If modularity and/or collaborative development of the ontology are indicated, how will they be supported?
- What operations will be performed, using the ontology, by other system components? What components will perform those operations? How do the operational requirements identified in the exploration phase apply to those specific operations and components?

Evaluation

While the design of ontology-enabled systems has some specific steps or considerations, noted above, the evaluation of resulting systems designs should follow best practices for evaluation of IT system design in general.

Model Build Phase

The build phase consists of four major activities (Informal Modeling, Formalizing Competency Questions, Formal Modeling, Serialization) are typically cycled through repeatedly; both for individual modules and for the ontology as whole. In practice, these activities are often performed seamlessly. Nevertheless, it is important to separate them conceptually, since they have different prerequisites, depend on different types of expertise, and lead to different outputs, which are evaluated in different ways.

Informal Modelling

The result of the ontological analysis is refined. Thus, for each module the relevant entities (individuals, classes, and their relationships) are identified and the terminology used in the domain is mapped to them. Important characteristics of the entities might be documented (e.g., the transitivity of a relationship, or a subsumption between two classes). The results are captured in some informal way (e.g., concept maps, UML diagrams, natural language text).

Evaluation

see ontological analysis

Formalizing Competency Questions

Based on the results of the informal modelling, the scenarios and competency questions are formalized. This might involve revising the old competency questions and adding new ones.

Evaluation

- Are the competency questions representative for all intended usages?
- Does the formalization capture the intent of the competency question appropriately?

Formal Modeling

The content of the informal model is captured in some ontology language (e.g., Common Logic, OWL Full), and then fleshed out with axioms. The resulting reference ontology represents the domain appropriately and is supposed to meet the requirements for domain representation (model fitness). This is either achieved by creating a new ontology module from scratch or by reusing an existing ontology and, if necessary, adapting it.

Evaluation

The ontology that is developed by the formal modelling activity is evaluated in two respects: Is the domain represented appropriately (model quality) and does the representation meet the requirements for its intended use (model fitness).

Evaluating Model Quality

Model quality is determined by two questions: Does the ontology follow best practices; in particular does it implement the upper ontology and the design principles decisions made in the model design phase? And is the domain represented accurately? (The domain is represented accurately if all axioms within the ontology are true.)

Model quality is often evaluated by examining the intrinsic structure of one ontology or comparing the intrinsic structure of several ontologies that are overlapping in scope. These kind of evaluation techniques draw upon mathematical and logical properties such as logical consistency,

graph-theoretic connectivity, model-theoretic interpretation issues, inter-modularity mappings and preservations, etc. Structural properties include branching factor, density, counts of ontology constructs, averages, and the like are intrinsic.

Another set of techniques for the evaluation of model quality involves some understanding of the domain. This is often required to determine whether a particular axiom is in alignment with the reality it is supposed to model, whether the model captures the distinctions and properties important to the domain, or which ontological design principles to apply in a given situation. Some ontological meta-properties (such as rigidity, identity, unity, etc.) can be used to gauge the quality of the axioms of the ontology.

Evaluating Model Fitness

The formalized competency questions and scenarios are used to query the ontology modules and the whole ontology. If all competency questions are answered successfully, the ontology meets the requirements for domain representation, which derive from functionalities of the ontology that rely on query-answering.

If the ontology is expected to support other kinds of operations, model fitness may be tested by executing a sample or approximation of those operation in a test environment. For example, if the ontology is to support automated indexing of documents with terms from the ontology, then fitness may be evaluated by running an approximation of the document analysis and indexing system, using the ontology in question, over a test corpus. There are various ways of assessing the results, for example, by comparison to a gold standard or review of results by domain experts. The extent to which the results are attributable to the ontology, versus other aspects of the system, can be identified to a certain extent by comparison of results using the same indexing system but different ontologies.

Operational Adaptation

The implementation ontology is the result of adapting the reference ontology to the operational requirements. One particular concern is whether the deployed ontology will be able to respond in a time-frame that meets its performance requirements (system fitness). This often requires a paring down of the ontology and other optimization steps.
In some cases the implementation ontology uses a different ontology language with a different semantics. E.g., if the application-specific reasoning does not observe the full first-order logic or description logic Open World Assumption, but insteads the negations in the ontology under a Closed World assumption.

Evaluation

Does model support operational requirements (e.g., performance, precision, recall)?

System Build Phase

In this phase the system is built according to the design specified in the design phase. If system components other than the ontology need to be built or otherwise acquired, processes for doing so can occur more or less in parallel to the ontology build phase. (Of course, tools and components necessary to the activities in the model building phase should be in place as model building begins; e.g., ontology development environments, version control systems, collaboration and workflow tools.) The system build phase concerns the integration of the ontology and other independent

components into subsystems as called for and into a system as specified in the system design phase.

The system build phase is discussed as part of the ontology lifecycle because in a typical application, the functionalities supported by the ontology are realizable not by interaction with the ontology alone, but by processes carried out by some combination of the ontology and other components and/or subsystems. Thus, whether the ontology meets the operational requirements can only be accurately evaluated once such interaction can be performed and results produced.

Evaluation

The building of ontology-enabled systems has some specific steps or considerations, noted above. However, the evaluation of built systems should follow best practices for evaluation of IT system in general.

Deployment Phase

In this phase, the ontology goes from the development environment and status to a production or live-use environment. Deployment usually occurs after a development cycle in which some targeted improvement or extension has been created. The deployment phase begins with thorough, comprehensive testing to ensure that the ontology will function properly in the operational environment and will not interrupt or degrade operations in that environment. This pre-deployment testing typically iterates until results indicate that it is safe to deploy the ontology without disrupting operations. In order to provide such assurance, pre-deployment testing often includes testing in an "offline" (separate from the production system) environment that includes a copy, partial copy, or simulation of as much of the operational environment as is feasible. In cases featuring ongoing system usage and iterative ontology development and deployment cycles, this phase is often especially rigorous and protective of existing functionality in the production, operational system.

Evaluation

- Does the ontology meet all requirements addressed and evaluated in the development phases?
- Are sufficient (new) capabilities provided to warrant deployment of the ontology?
- Are there outstanding problems that raise the risk of disruptions if the ontology is deployed?
- Have regression tests been run to identify any existing capabilities that may be degraded if the ontology is deployed? If some regression is expected, is it acceptable in light of the expected benefits of deployment?

Use and Maintenance Phase

This phase accompanies a use phase and involves the sustainment of deployed capabilities, rather than the development of new ones. A particular system may have ontology maintenance and new ontology development phases going on at the same time, but these activities should be distinguished as have different goals (improvement vs sustainment) and they operate on at least different versions of an ontology, if not different ontologies or different modules of an ontology. When an ontology (or version thereof) is in a maintenance phase, information is collected about the results of operational use of the ontology. Any problems or sub-optimal results are identified and micro-scale development cycles may be conducted to correct those problems. Simultaneous identification of new use cases, desired improvements, and new requirements that may happen during the same use phase should not be regarded as part of maintenance phase; rather, they are inputs to, or part of, exploration activities for a future version, extension, new ontology or new module. A single set of tools may be used to collect information of both sorts (for maintenance and

for exploration and new development) during a use phase, but the information belongs to different activities. This distinction is manifested, for example, in the distinction between “bug reports” (or “problem reports”) and “feature requests” (or “requested improvements”) made by bug-tracking tools. The maintenance phase consists of identifying and addressing bugs or problems.

Evaluation

The evaluation should be continuous, e.g., open problem reporting and regular, e.g., nightly, automated regression testing:
- Are any regression tests failing? If so, how are they being addressed?
- Is any functionality claimed for the most recent deployment failing? If so, can the problem be tracked to the ontology, or is the problem elsewhere?
- If the problem is located with the ontology, can it be corrected before the next major development and deployment cycle? If so, what is being done to address it?
- If a problem occurs and cannot be addressed without a large development cycle effort, is the problem severe enough to warrant backing out of the deployment in which it was introduced?

Support and Recommendations

Tools for Ontology Evaluation

There are central aspects of ontology that may not be amenable to software control or assessment. For example, the need for clear, complete, and consistent lexical definitions of ontology terms is not presently subject to software consideration beyond identifying where lexical definitions may be missing entirely. Another area of quality difficult for software determination is the semantic fitness of an ontology to its world domain (reality) or to its application domain.
Generally, appreciation of the full cycle of life of an ontology is not well established within the ontology community. Thus, there are not tools that are aiming to guide the ontology development or enable ontology evaluation across the whole lifecycle. Thus, the existing tools enable functions at different points along the ontology lifecycle, and for a given characteristic, some tools may perform better in one lifecycle phase than in another phase where a different tool is better suited.

Significant new ontology evaluation tools are currently becoming available to users. An overview will be presented as part of the Ontology Quality Software Survey (add link). Carving a link between such tools and existing IT architecture and design tools (e.g., Enterprise Architect and Solution Architect) remains a future possibility in order to integrate ontology into mainstream application software development within enterprise or more focused IT environments. This capability could offer a definitive means of connecting ontology quality/fitness characteristics and measures to use case and application software requirements.