Applied Ontology 8 (2013) 179–194
DOI 10.3233/AO-130125
IOS Press

Communiqué

Towards ontology evaluation across the life cycle

The Ontology Summit 2013

Fabian Neuhaus [*,‡], Amanda Vizedom[‡], Ken Baclawski, Mike Bennett, Mike Dean, Michael Denny, Michael Grüninger, Ali Hashemi, Terry Longstreth, Leo Obrst, Steve Ray, Ram Sriram, Todd Schneider, Marcela Vegetti, Matthew West and Peter Yim

Keywords: Ontology evaluation, ontology lifecycle, best practices
Accepted by: Nicola Guarino

Executive summary

Problem

Currently, there is no agreed on methodology for development of ontologies, and there is no consensus on how ontologies should be evaluated. Consequently, evaluation techniques and tools are not widely utilized in the development of ontologies. This can lead to ontologies of poor quality and is an obstacle to the successful deployment of ontologies as a technology.

Approach

The goal of the Ontology Summit 2013 was to create guidance for ontology developers and users on how to evaluate ontologies. Over a period of four months a variety of approaches were discussed by participants, who represented a broad spectrum of ontology, software, and system developers and users. We explored how established best practices in systems engineering and in software engineering can be utilized in ontology development.

Results

This document focuses on the evaluation of five aspects of the quality of ontologies: intelligibility, fidelity, craftsmanship, fitness, and deployability. A model for the ontology life cycle is presented, and

[*]Corresponding author. E-mail: fneuhaus@web.de.
[‡]These authors contributed equally to this work.

evaluation criteria are presented in the context of the phases of the life cycle. We discuss the availability of tools and the document ends with observations and recommendations. Given the current level of maturity of ontology as an engineering discipline, any results on how to best build and evaluate ontologies have to be considered as preliminary. However, the results achieved a broad consensus across the range of backgrounds, application foci, specialties and experience found in the Ontology Summit community.

Recommendations

For more reliable success in ontology development and use, ontology evaluation should be incorporated across all phases of the ontology life cycle. Evaluation should be conducted against carefully identified requirements; these requirements depend on the intended use of the ontology and its operational environment. For this reason, we recommend the development of integrated ontology development and management environments that support the tracking of requirements for, and the evaluation of, ontologies across all phases of their development and use.

1. Purpose of this document

The purpose of this document is to advance the understanding and adoption of ontology evaluation practices. Our focus is on the critical relationships between usage requirements, the life cycle of an ontology, evaluation, and the quality of the result.

This document is rooted in the 2013 Ontology Summit. Over four months, Summit participants prepared and presented materials, shared references, suggested resources, discussed issues and materials by email list, and met virtually each week for presentations and discussions. This Summit had the focal topic "Ontology Evaluation across the Ontology Lifecycle". This document represents a synthesis of a subset of ideas presented, discussed, and developed over the course of these four months, and reflects the contributions of the Summit's participants and the consensus of the Summit community.

The intended audience for this document is, first and foremost, anyone who is developing or using ontologies currently, or who is on the cusp of doing so. We believe that the adoption of ontology evaluation as presented here has the potential to greatly improve the effectiveness of ontology development and use, and to make these activities more successful. Thus, our primary audience is the developers of ontologies and ontology-based systems. A secondary audience for this document is the community of software, systems, and quality assurance engineers. When ontologies are used in information systems, success depends in part on incorporation of ontology evaluation (and related activities) into the engineering practices applied to those systems and their components.

2. Introduction

Ontologies are human-intelligible and machine-interpretable representations of some portions and aspects of a domain. Since an ontology contains terms and their definitions, it enables the standardization of a terminology across a community or enterprise; thus, ontologies can be used as a type of glossary. Since ontologies can capture key concepts and their relationships in a machine-interpretable form, they are similar to domain models in systems and software engineering. And since ontologies can be populated with or linked to instance data to create knowledge bases, and deployed as parts of information systems for query answering, ontologies resemble databases from an operational perspective.

This flexibility of ontologies is a major advantage of the technology. However, flexibility also contributes to the challenge of evaluating ontologies. *Ontology evaluation* consists of gathering information about some properties of an ontology, comparing the results with a set of requirements, and assessing the ontology's suitability for some specified purpose. Some properties of an ontology can be measured independently of usage; others involve relationships between an ontology and its intended domain, environment, or usage-specific activity, and thus can only be measured with reference to some usage context. The variety of the potential uses of ontologies means that there is no single list of relevant properties of ontologies and no single list of requirements. Therefore, there is no single, universally-applicable approach to evaluating ontologies.

However, we can identify some kinds of evaluation that are generally needed. To determine the quality of an ontology, we need to evaluate the ontology as a domain model for human consumption, the ontology as a domain model for machine consumption, and the ontology as deployed software that is part of a larger system. In this document, we focus on five high-level characteristics:[1]

(1) Can humans understand the ontology correctly? (*Intelligibility*)
(2) Does the ontology accurately represent its domain? (*Fidelity*)
(3) Is the ontology well-built and are design decisions followed consistently? (*Craftsmanship*)
(4) Does the representation of the domain fit the requirements for its intended use? (*Fitness*)
(5) Does the deployed ontology meet the requirements of the information system of which it is part? (*Deployability*)

For *intelligibility*, it is not sufficient that ontologists can understand the content of the ontology; all intended users need to be able understand the intended interpretation of the ontology elements (e.g., individuals, classes, relationships) that are relevant to their use-case. Intelligibility does not require that users view and understand the ontology directly. To enable intelligibility, the documentation of the ontology needs to be tailored to the different kinds of users. This may require multiple annotations of an element of the ontology suitable for different audiences (e.g., to accommodate language localization and polysemous use of terms across domains). Intelligibility is particularly important for ontologies that are used directly by humans as a controlled dictionary. But it is also desirable for ontologies that are intended to be used "under the hood" of an information system, because these ontologies need to be maintained and reviewed by people other than the original ontology developers. *Fidelity* is about whether the ontology represents the domain correctly, both in the axioms and in the annotations that document the ontology for humans. *Craftsmanship* is concerned with the question whether the ontology is built carefully; this covers aspects ranging from the syntactic correctness of the ontology to the question whether a philosophical choice (e.g., four-dimensionalism) has been implemented consistently. Both *fitness* and *deployability* are dependent on requirements for the intended usage. These requirements might be derived from the operational environment, in the case of an ontology that is deployed as part of an information system; alternatively, they may be derived from the goals for the knowledge representation project, if the ontology is deployed as a standalone reference ontology. While both characteristics are about meeting operational requirements, they are concerned with different aspects of the

[1]There are different approaches to clustering aspects of ontologies to be evaluated. For example, in the OQuaRE (http://ontolog.cim3.net/cgi-bin/wiki.pl?OQuaRE) framework, characteristics are broken down into sub-characteristics, which are linked to metrics. For more on OQuaRE and other approaches, see the Ontology Characteristics and Groupings reference collection, https://www.zotero.org/groups/ontologysummit2013/items/collectionKey/PMKFZPDA.

ontology: fitness is about the ontology as a domain model, deployability is about the ontology as a piece of software.[2]

Since fitness and deployability are evaluated with respect to requirements for the intended use-case, a comprehensive look at ontology evaluation needs to consider how the requirements for the ontology derive from the requirements of the system that the ontology is a part of. Furthermore, although "ontology evaluation" can be understood as the evaluation of a finished product, we consider ontology evaluation as an ongoing process during the life of an ontology. For these reasons, we embrace a broad view of ontology evaluation and discuss it in the context of expected usage and the various activities during ontology development and maintenance. In the next section we present a high-level breakdown of these activities, organized as goal-oriented phases in the ontology life cycle. Afterward we identify, for each phase, some of the activities that occur during that phase, its outputs, what should be evaluated at the stage. The document concludes with some observations about the current tool support for ontology evaluation and recommendations for future work.

3. An ontology life cycle model

The life of any given ontology consists of various types of activities in which the ontology is being conceived, specified, developed, adapted, deployed, used, and maintained. Whether these activities occur in a sequence or in parallel, and whether certain kinds of activities (e.g., requirements development) only happen once during the life of an ontology or are cycled through repeatedly depends partially on how the development process is managed. Furthermore, as discussed above, ontologies are used for diverse purposes; thus, there are certain kinds of activities (e.g., the adaptation of the ontology to improve computational performance of automatic reasoning) that are part of the development of some ontologies and not of others. For these reasons, there is no single ontology life cycle with a fixed sequences of phases. Any ontology life cycle model presents a simplified view that abstracts from the differences between the ways ontologies are developed, deployed and used.

In spite of presenting a simplified view, an *ontology life cycle* model is useful because it highlights recognizable, recurring patterns that are common across ontologies. The identification of *phases* allows the clustering of activities around goals, inputs, and outputs of recognizable types. Furthermore, a life cycle model emphasizes that some of the phases are interdependent, because the effectiveness of certain activities depends on the outputs of others. For example, effective ontology development depends on the existence of identified ontology requirements; if requirements identification has been omitted or done poorly, ontology development is very unlikely to result in useful outputs. While the development process may vary considerably between two ontologies, there are invariant dependencies between phases.

Figure 1 presents the phases of an ontology life cycle model. Typically, an ontology will go through each of these phases more than once during its life. In the following sections, each of these phases is discussed in more detail, including input, outputs and relationships between the phases. As the figure illustrates, evaluation should happen throughout the ontology life cycle, varying in focus, process, and intensity according to phase-appropriate requirements. In the following sections each of the phases is linked to phase-appropriate evaluation activities. The evaluation provides information about the degree to which requirements of the life cycle phase are being met. Requirements identification will be discussed in greater detail in the next section.

[2]Fidelity, craftsmanship, and fitness are discussed in more detail in the section on ontology development on page 187.

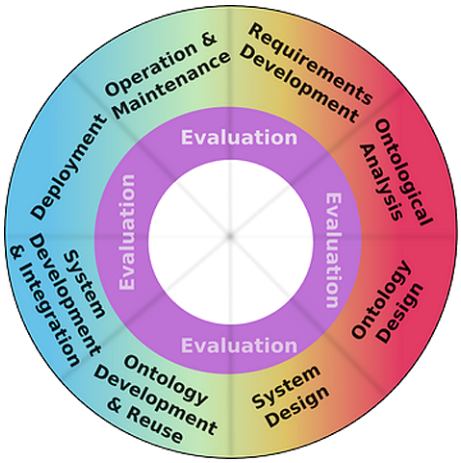

Fig. 1. An ontology life cycle model. (Colors are visible in the online version of the article; http://dx.doi.org/10.3233/AO-130125.)

This model applies to ontologies regardless of whether their use involves significant machine processing of ontology content. Because the systems into which ontologies are incorporated are information systems in a broad sense: systems of people, processes, hardware, software, and data that process information and make decisions.[3] For example, consider an ontology which is used by humans to curate documents. In this case the information system includes the ontology, the curators, and the tools they use to browse the ontology and to annotate the documents. The success of the whole system depends on the interaction of the ontology with both the other software components and the curators. For example, the whole system will be impaired if the ontology contains information that the browser cannot display properly, or if the definitions in the ontology are so ambiguous that different curators are not able to use the terms consistently in the curation process. Thus, the ontology needs to be evaluated for both deployability and intelligibility. This example illustrates that even if an ontology is not used for machine reasoning as part of a complex piece of software, its evaluation still depends on its intended use within a larger system.

4. Requirements development phase

The purpose of this phase is to establish understanding, context, scope, and initial requirements. Development of adequate requirements is critical to the success of any ontology development and usage. Most evaluation activities that are presented in the next sections depend on the results of this phase.

During the *requirements development* phase, expected or intended usages and interpretations are elicited and examined, and initial requirements are derived. Typically, an intended usage is initially understood from a business[4] perspective. The intended usage may be specified as use-cases or scenarios; at early stages, requirements may be captured only as brief statements of one or more business needs

[3] See the Broad view of Information Systems reference collection for more on this understanding, https://www.zotero.org/groups/ontologysummit2013/items/collectionKey/HU2MCEG4.

[4] "Business" here is meant in the broad sense, incorporating the activities of the organization or user that need the ontology and/or ontology-based system, regardless of whether those activities are commercial, governmental, educational, or other in nature.

and constraints. In many cases only some aspects of usage are addressed, and requirements development may include gathering information about other aspects that are significant for ontology analysis and design.[5]

One important way to specify requirements is by using *competency questions*: questions that the ontology must be able to answer.[6] These questions are formulated in a natural language, often as kinds of queries that the ontology should support in given scenarios.

The output of the requirements development phase is a document that should answer the following questions:

- Why is this ontology needed? (What is the rationale? What are the expected benefits)?
- What is the expected or intended usage (e.g., specified as use-cases, scenarios)?
- Which groups of users needs to understand which parts of the ontology?
- What is the scope of the ontology?
- Are there existing ontologies or standards that need to be reused or adopted?
- What are the competency questions?
- Are the competency questions representative of all expected or intended usages?
- What are the requirements from the operational environment?
- What resources need to be considered during the ontology and system design phases (e.g., legacy databases, test corpora, data models, glossaries, vocabularies, schemas, taxonomies, ontologies, standards, access to domain experts)?

5. Ontological analysis phase

The purpose of the *ontological analysis* 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 usually involves the resolution of ambiguity and the identification of entities that are denoted by different terms across different resources and communities. This activity requires close cooperation between domain experts and ontologists, because it requires both knowledge about the domain and knowledge about important ontological distinctions and patterns.

The results are usually captured in some informal way, understandable to both ontologists and domain experts. One way of specifying the output of ontological analysis is by a set of sentences in a natural language, which are interpreted in the same way by the involved subject matter experts and ontologists. The ontologists apply their knowledge of important ontological distinctions and relationships to elicit such sentences that capture the information needed to guide the ontology design.[7] Ontological analysis

[5] See the Ontology Usage reference collection for more about analyzing ontology usage, https://www.zotero.org/groups/ontologysummit2013/items/collectionKey/HJ6MK7W3.

[6] See the Competency Questions reference collection for more on capability questions, https://www.zotero.org/groups/ontologysummit2013/items/collectionKey/7B5TCZCZ.

[7] An example of such informal outputs is (phrases in italics indicate entities):

> Every *pick report* is also an *order status report*.
> Every *order* has a *shipping method*.
> Possible *shipping methods* include *ground*, and *air*.
> The *shipping method* for an individual *order* is determined by the *fulfillment software* after the *order* is *pack*ed.
> Every *order* has a *shipping speed*. Possible *shipping speeds* include *standard*, *two-day*, and *overnight*.
> The *shipping speed* for a specific *order* is *chosen by* the *buyer* when the *buyer places* the *order*.
> For the thing *people in the business* usually call *order*, the *fulfillment database* uses the word "sale."

outputs can also be captured in diagrams (e.g., concept maps, UML diagrams, trees, freehand drawings).

The output of the ontological analysis phase, whatever the method of capture, should include specification of:

- significant entities within the scope of the intended usage;
- important characteristics of the entities, including relationships between them, disambiguating characteristics, and properties important to the domain and activities within the scope of the intended usage;
- the terminology used to denote those entities, and provide enough contextual information to disambiguate polysemous terms.

These results provide input to ontology design and development. In addition, these results provide detail with which high-level requirements for ontology design and development phases can be turned into specific, evaluable requirements.

Evaluating ontological analysis results: *Questions to be answered*

The output of an ontological analysis phase should be evaluated according to the following high-level criteria, assisted in detail by the outputs of requirements development:

- Are all relevant terms from the use cases documented?
- Are all entities within the scope of the ontology captured?
- Do the domain experts agree with the ontological analysis?
- Is the documentation sufficiently unambiguous to enable a consistent use of the terminology?

6. Ontology design phase

In the *ontology design* phase, a design[8] is developed, based on the outputs from the requirements development and the ontological analysis. In particular, representation languages are chosen for ontology and for queries (these may be identical). Further, the structure of the ontology is determined. Structural choices include whether and how the ontology is separated into modules and how the modules are integrated. As part of the structural design, it may be decided that some existing ontologies are reused as modules. The intended behavior of the modules may be captured by competency questions. These module-specific competency questions are often derived from the ontology-wide competency questions.

Design phase activities include the determination of design principles and of top-level classes in the ontology. The *top-level classes* are the classes in the ontology that are at the highest level of the subsumption hierarchy. (In the case of OWL ontologies, these are the direct children of owl:thing.) These classes determine the basic ontological categories of the ontology. Together the top-level categories and the design principles determine whether and how some fundamental aspects of reality are represented (e.g., change over time). The design principles may also restrict representation choices by the developers (e.g., by enforcing single inheritance for subsumption).

One way to make these design decisions is to use an existing upper ontology. *Upper* or *foundational ontologies* (e.g., DOLCE, BFO or SUMO) are reusable, varyingly comprehensive ontology artifacts that specify the basic ontological categories, relationships between them, and some methodological decisions

[8]No distinction is made here between design and architecture. The design phase should be understood to encompass both.

about how to represent reality. Other approaches (e.g., OntoClean) rely on the systematic representation of logical and philosophical properties of classes and relationships. There are efforts (e.g., in the NeOn project) to capture design decisions in form of design patterns, and share them with the community.[9]

The results of the design decisions in this phase lead to additional requirements for the ontology. Some of these requirements concern characteristics entirely internal to the ontology itself (e.g., single inheritance for subsumption or distinction between rigid, anti-rigid, and non-rigid classes). Many of these requirements can be understood and evaluated using technical, ontological understanding, without further input of usage-specific or domain-specific information.

Note that there might be conflicting requirements for the expressivity of the ontology language and its performance (see system design phase). Such tension can be addressed by distinguishing between, and developing, separate reference and operational ontologies. A *reference ontology* is one which captures the domain faithfully, to the extent required by the intended or expected usage(s), and in a language that is expressive enough for that purpose. An *operational ontology* is one that is adapted from a reference ontology, potentially incorporating compromises in representation for the sake of performance. The two types of ontologies will be discussed further in the ontology development and reuse section.

Evaluating ontology design results: *Questions to be answered*

- Is the chosen ontology language expressive enough to capture the knowledge with sufficient detail in order to meet the ontology requirements?
- Is the chosen query language expressive enough to formalize the competency questions?
- Does the chosen language support all required ontology capabilities? (For example, if the ontology is to support probability reasoning, does the language enable the representation of probabilistic information?)
- Is every individual or class that has been identified in the ontological analysis phase either an instance or a subclass of some top-level class?
- Are naming conventions specified and followed?
- Does the design call for multiple, distinct ontology modules? If so, do the ontology modules together cover the whole scope of the ontology?
- Are all modules of the ontology associated with (informal) competency questions?
- Does the design avoid addition of features or content not relevant to satisfaction of the requirements?
- For each module, is it specified what type of entities are represented in the module (the intended domain of quantification)?
- For each module, is it specified how it will be evaluated and who will be responsible?
- Does the design specify whether and how existing ontologies will be reused?

7. System design phase

Information system design as a general activity is its own field of practice, and there is no need to re-invent or summarize it here. There is, however, a need to emphasize the interdependence of ontology design and system design for ontologies that are intended to be used as components of an information system. During system design, decisions are made that lead to requirements for the capabilities and

[9]See the Existing Methodologies and Upper Ontologies reference collection for some examples of upper ontologies and design methodologies, https://www.zotero.org/groups/ontologysummit2013/items/collectionKey/FVM3J9FJ.

implementation of the ontology and its integration within the larger information system. This interdependency is often underestimated, which leads to poor alignment between the ontology and the larger system it is part of, and thus, to greater risk of failure in ontology and system use.

The output of the system design phase should answer such questions as:

- What operations will be performed, using the ontology, by other system components? What components will perform those operations? How do the business requirements identified in the requirements development phase apply to those specific operations and components?
- What, if any, inputs or changes to the ontology will there be, once the system is deployed?
- What interfaces (between machines or between humans and machines) will enable those inputs? How will these interfaces be tested with respect to the resulting, modified ontology? What requirements will need to be met?
- What data sources will 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 the ontology be built, evaluated, and maintained? What tools are needed to enable the development, evaluation, configuration management, and maintenance of the ontology?
- If modularity and/or collaborative development of the ontology are indicated, how will they be supported?

Evaluating system design results: *Questions to be answered*

The bulk of system design requirements will derive from systems design principles and methodologies in general, and are thus out of the scope of this document. We emphasize here the often unmet need to explicitly recognize the ontology as a component of the system and to evaluate the system design accordingly:

- Does the system design answer the questions listed just above?

8. Ontology development phase

The *ontology development* phase consists of four major activities: informal modeling, formalization of competency questions, formal modeling, and operational adaptation (each of which is described below). These activities are typically cycled through repeatedly both for individual modules and for the ontology as whole. In practice, these activities are often performed without obvious transitions between them. 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.

The ontology development phase covers both new ontology development and ontology reuse, despite differences between these activities. We do not consider new development and reuse to be part of different phases, for the following reasons: the successful development, or selection and adaptation, of an ontology into an information system is possible only to the extent that the ontology meets the requirements of the expected or intended usage. Thus, whether an ontology is developed entirely from scratch, re-used from existing ontologies, or a combination of the two, good results depend on identification of ontology requirements, an ontological analysis, and the identification of ontology design requirements. Furthermore, the integration of the ontology into the broader information system, its deployment and its usage are not altered in substance by the ontology's status as new or reused. The ontology is evaluated against the same set of requirements, regardless of whether it is reused or newly developed. Therefore,

from a high-level perspective, both newly-developed and reused ontologies play the same role within the ontology life cycle.

8.1. Informal modeling

During *informal modeling*, 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 usually captured in some informal way (e.g., concept maps, UML diagrams, natural language text).

Evaluating informal modeling results: *Questions to be answered*

- All evaluation criteria from the ontological analysis phase apply to informal modeling, with the addition of the following:
- Does the model capture only entities within the specified scope of the ontology?
- Are the defined classes and relationships well-defined? (e.g., no formal definition of a term should use the term to define itself)
- 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?

8.2. Formalization of competency questions

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

Evaluating formal competency questions: *Questions to be answered*

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

8.3. Formal modeling

During *formal modeling*, the content of the informal model is captured in some ontology language (e.g., Common Logic, OWL 2 DL), and then fleshed out with axioms. The resulting reference ontology represents the domain appropriately (fidelity), adheres to the design decisions made in the ontology design phase (craftsmanship), and is supposed to meet the requirements for domain representation (fitness). This is either achieved by creating a new ontology module from scratch or by reusing an existing ontology and, if necessary, adapting it.

Evaluating formal modeling results: *Questions to be answered*

The ontology that is developed by the formal modeling activity or is considered for reuse is evaluated in three respects: whether the domain is represented appropriately (fidelity); whether the ontology is well-built and follows the decisions from the ontology design phase (craftsmanship); and whether the representation meets the requirements for its intended use (fitness).

Evaluating fidelity. Whether the domain is represented accurately in an ontology depends on three questions: Are the annotations of ontology elements (e.g., classes, properties, axioms) that document their intended interpretation for humans (e.g., definitions, explanations, examples, figures) correct? Are all axioms within the ontology true with respect to the intended level of granularity and frame of reference (universe of quantification)? Are the documentation and the axioms in agreement?

Since the evaluation of fidelity depends on some understanding of the domain, it ultimately requires review of the content of the ontology by domain experts.[10] However, there are some automated techniques that support the evaluation of fidelity. For example, one can evaluate the ontology for logical consistency, evaluate automatically generated models of the ontology on whether they meet the intended interpretations,[11] or compare the intrinsic structure of the ontology to other ontologies (or different versions of the same ontology) that are overlapping in scope.

Evaluating craftsmanship. In any engineering discipline, craftsmanship covers two separate, but related aspects. The first is whether a product is well-built in a way that adheres to established best practices. The second is whether design decisions that were made are followed in the development process. Typically, the design decisions are intended to lead to a well-built product, so the second aspect feeds into the first. Since ontology engineering is a relatively young discipline, there are relatively few examples of universally accepted criteria for a well-built ontology (e.g., syntactic well-formedness, logical consistency and the existence of documentation). Thus, the craftsmanship of an ontology needs to be evaluated largely in light of the ontological commitments, design decisions, and methodological choices that have been embraced within the ontology design phase.

One approach to evaluating craftsmanship relies on an established upper ontology or ontological meta-properties (such as rigidity, identity, unity, etc.), which are used to gauge the axioms in the ontology. Tools that support the evaluation of craftsmanship often examine the intrinsic structure of an ontology. This kind of evaluation technique draws upon mathematical and logical properties such as logical consistency, graph-theoretic connectivity, model-theoretic interpretation issues, inter-modularity mappings and preservations, etc. Structural metrics include branching factor, density, counts of ontology constructs, averages, and the like.[12]

Evaluating fitness. The formalized competency questions and scenarios are one source of evidence regarding fitness. These competency questions are used to query corresponding ontology modules and the whole ontology. Successful answers to competency questions provide evidence that the ontology meets the model requirements that derive from query-answering based functionalities of the ontology. The ability to successfully answer competency question queries is not the same as fitness, but, depending on the expected usage, it may be a large component of it.

Fitness can also be evaluated by performing a sample or approximation of system operations, using the ontology in a test environment and/or over a test corpora. For example, if the ontology is required to support automated indexing of documents with ontology terms, then fitness may be evaluated by running an approximation of the document analysis and indexing system, using

[10] See the Expert Review and Validation reference collection for more on expert evaluation of ontologies, https://www.zotero.org/groups/ontologysummit2013/items/collectionKey/6GGPKU3D.

[11] See the Evaluating Fidelity reference collection for more on this, including evaluation via simulation, https://www.zotero.org/groups/ontologysummit2013/items/collectionKey/929KF23Z.

[12] For more details, see the synthesis and community input pages for intrinsic ontology evaluation, http://ontolog.cim3.net/cgi-bin/wiki.pl?OntologySummit2013_Intrinsic_Aspects_Of_Ontology_Evaluation_Synthesis and http://ontolog.cim3.net/cgi-bin/wiki.pl?OntologySummit2013_Intrinsic_Aspects_Of_Ontology_Evaluation_CommunityInput.

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 by review of results by domain experts, and measured by some suitably defined notions of recall and precision. 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.

8.4. Operational adaptation

During *operational adaptation*, the reference ontology is adapted to the operational requirements, resulting in an operational ontology. One particular concern is whether the deployed ontology will be able to respond in a time-frame that meets its performance requirements. This may require a paring-down of the ontology and other optimization steps (e.g., restructuring of the ontology to improve performance). For example, it might be necessary to trim an OWL DL ontology to its OWL EL fragment to meet performance requirements.

In some cases the operational 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 instead interprets the negations in the ontology under a Closed World assumption).

Evaluating operational adaptation results: *Questions to be answered*

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

9. System development and integration 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 development phase. Of course, tools and components necessary to the activities in the ontology development phase should be in place as ontology development begins; e.g., ontology development environments, version control systems, collaboration and workflow tools. The system development and integration phase concerns the integration of the ontology and other components into subsystems as called for and into a system as specified in the system design phase.

The system development and integration phase is discussed as part of the ontology life cycle 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 full range of requirements can only be accurately evaluated once such interaction can be performed and results produced.[13]

Evaluating system development results: *Questions to be answered*

The bulk of system development requirements will derive from systems development principles and methodologies in general, and are thus out of the scope of this document. We emphasize here the often unmet need to explicitly recognize the ontology as a component of the system and to evaluate the system development results accordingly. Specifically:

[13]For more details, see the synthesis page on extrinsic aspects of ontology evaluation, http://ontolog.cim3.net/cgi-bin/wiki.pl?OntologySummit2013_Extrinsic_Aspects_Of_Ontology_Evaluation_Synthesis.

- Does the system achieve successful integration of the ontology, as specified in the system design?
- Does the system meet all requirements that specifically relate to the integrated functioning of the ontology within the system?

10. Deployment phase

In this phase, the ontology goes from the development and integration environment to an operational, live use environment. Deployment usually occurs after some development cycle(s) in which an initial ontology, or a version with some targeted improvement or extension, has been specified, designed, and developed. As described above, the ontology will have undergone evaluation repeatedly and throughout the process to this point. Nevertheless, there may be an additional round of testing once an ontology has passed through development and integration phases and deemed ready for deployment by developers, integrators, and others responsible for those phases. This additional, deployment-phase evaluation may or may not differ in nature from evaluation performed across other life cycle stages; it may be performed by independent parties (i.e., not involved in prior phases), or with more resources, or in a more complete testing environment (one that is as complete a copy or simulation of the operational environment as possible, but still isolated from that operational environment). The focus of such evaluation, however, is on establishing whether the ontology will function properly in the operational environment and will not interrupt or degrade operations in that environment. This deployment-phase testing typically iterates until results indicate that it is safe to deploy the ontology without disrupting business activities. 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 deployed, in-use system. If and when such evaluation criteria have been satisfied, the ontology and/or system version is incorporated into the operational environment, released, and becomes available for live use.

Evaluating deployment: *Questions to be answered*

- 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 succeeding competency questions been used to create regression tests?
- 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?

11. Operation and maintenance phase

This phase focuses on the sustainment of deployed capabilities, rather than the development of new ones. A particular system may have operation and maintenance and new ontology development phases going on at the same time, but these activities should be distinguished as they 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 an operation and maintenance phase, information is collected about the results of operational use of the ontology. 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 period of use should not be regarded as part of maintenance activity; rather, they are inputs to, or part of, exploration and possibly requirements development 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 forward-looking exploration and requirements development) while an ontology is in use, 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 activity consists of identifying and addressing bugs or problems.

Evaluating operation and maintenance: *Questions to be answered*

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, 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 in the ontology, can it be corrected before the next major development and deployment cycle? If so, is it being addressed?
- 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?

12. 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 effective software consideration beyond identifying where lexical definitions may be missing entirely. Another area of quality difficult for software determination is the fidelity of an ontology.

There are no tools for ontology development or to enable ontology evaluation across the whole life cycle. Existing tools support different life cycle phases, and for any given characteristic, some tools may perform better in one phase than in another phase where a different tool is better suited. However, significant new ontology evaluation tools are currently becoming available to users.[14] An overview is presented as part of the Ontology Quality Software Survey.[15]

13. Observations and recommendations

1. We still have a limited understanding of the ontology life cycle, ontology development methodologies, and how to make best use of evaluation practices. More research in these areas is needed. Thus, any recommendation in this area is provisional.

[14]See the Tool Support reference collection, https://www.zotero.org/groups/ontologysummit2013/items/collectionKey/DWNMSJ5S and the Ontology Summit synthesis page on software environments, OntologySummit2013_Software_Environments_For_Evaluating_Ontologies_Synthesis for more information about available tools.

[15]For example. For Survey and results, see the Software Support for Ontology Quality and Fitness Survey.

2. There is no single ontology life cycle with a fixed sequences of phases. However, there are recurring patterns of activities, with identifiable outcomes, which feed into each other. In order to ensure quality, these outcomes need to be evaluated. Thus, evaluation is not a singular event, but should happen across the whole life of an ontology.

3. The different outputs of the ontology life cycle phases need to be evaluated with respect to the appropriate criteria. In particular, different requirements apply to informal models, reference ontologies, and operational ontologies, even when implemented in the same language.

4. Ontologies are evaluated against requirements that derive both from design decisions and the intended use of the ontology. Thus, a comprehensive evaluation of an ontology needs to consider the system that the ontology is part of.

5. There is a shortage of tools that support the tracking of requirements for and the evaluation of ontologies across all stages of their development and use. These kinds of tools should be developed, and integrated in ontology development environments and repositories.

6. We strongly encourage ontology developers to integrate existing evaluation methodologies and tools as part of the development process.

14. Editorial remarks and endorsements

This document is the Communiqué of the Ontology Summit 2013. This Summit was organized by Ontolog, the National Institute of Standards and Technology (NIST), the National Center for Ontological Research (NCOR), the National Center for Biomedical Ontology (NCBO), the International Association for Ontology and its Application (IAOA), the National Coordination office for the Networking and Information Technology Research and Development program (NCO NITRD), and co-sponsored by 77 other organizations.[16]

This communiqué was endorsed by the following 162 members of the ontology community.[17]

Sameera Abar	Gary Berg-Cross	Peter Brown
Tara Athan	Julita Bermejo-Alonso	Pat Cassidy
Nathalie Aussenac-Gilles	Olivier Bodenreider	Francisco E. Castillo-Barrera
Ken Baclawski	Harold Boley	Vinay Chaudhri
Mary Balboni	Nikolay Borgest	Chris Chute
Michael Barnett	Stefano Borgo	Anthony Cohn
John Bateman	Michel van den Bossche	Mills Davis
Denise Bedford	Bruce Bray	Mike Dean
Joel Bender	Mathias Brochhausen	Michael Denny
Mike Bennett	Rex Brooks	Nicolau DePaula

[16]Certain commercial software systems may be identified in this paper. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology (NIST) or any other supporting U.S. government or corporate organizations; nor does it imply that the products identified are necessarily the best available for the purpose. Further, any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NIST or any other supporting U.S. government or corporate organizations. This article does not contain technical data as defined by the International Traffic in Arms Regulations, 22 CFR 120.10(a), and is therefore authorized for publication.

[17]Please note that these people made their endorsements as individuals and not as representatives of the organizations they are affiliated with.

Jim Disbrow
Ed Dodds
Michel Dumontier
Astrid Duque Ramos
Karen Engelbart
Doug Engelbart
Jesualdo T. Fernández Breis
Tim Finin
Michael Fitzmaurice
Elizabeth Florescu
Doug Foxvog
Gilberto Fragoso
Bart Gajderowicz
Aldo Gangemi
Gary Gannon
Katherine Goodier
Henson Graves
Tom Gruber
Michael Gruninger
Giancarlo Guizzardi
Melissa Haendel
Torsten Hahmann
Marc Halpern
Nurhamizah Hamka
Ali Hashemi
Brian Haugh
Martin Hepp
Matthew Hettinger
Scott Hills
Ralph Hodgson
Bill Hogan
Jeanne Holm
Doug Holmes
Ian Horrocks
Dosam Hwang
Kingsley Idehen
Megan Katsumi
Maria Keet
Elisa Kendall
Pavithra Kenjige
Kyoungsook Kim
Mitch Kokar
Pradeep Kumar
Oliver Kutz
Christoph Lange
Ken Laskey
David Leal
Kiyong Lee
Anatoly Levenchuk
Antonio Lieto
Laurent Liscia
Frank Loebe
Terry Longstreth
Joanne Luciano
Ernie Lucier
Mark Luker
Dickson Lukose
Deborah MacPherson
Diego Magro
Patrick Maroney
Richard Martin
Howard Mason
Bill McCarthy
John McClure
Tim McGrath
Chris Menzel
Riichiro Mizoguchi
Till Mossakowski
Enrico Motta
Regina Motz
Mark Musen
John Mylopoulos
Joel Natividad
Fabian Neuhaus
David Newman
Deborah Nichols
Duane Nickull
Pete Nielsen
Leo Obrst
Frank Olken
Alessandro Oltramari
Mary Parmelee
Chris Partridge
Vinod Pavangat
Lynne Plettenberg
Hans Polzer
María Poveda Villalón
David Price
Fouad Ramia
Steve Ray
Alan Rector
Michael Riben
Jack Ring
Carlos Rueda
Arturo Sanchez
Bob Schloss
Todd Schneider
James Schoening
Ravi Sharma
Bradley Shoebottom
Barry Smith
Bob Smith
Jerry Smith
Dagobert Soergel
Richard Mark Soley
John Sowa
Simon Spero
Jim Spohrer
Christopher Spottiswoode
Ram Sriram
Mark Starnes
George Strawn
Rudi Studer
Samir Tartir
Hans Teijgeler
Andreas Tolk
Meika Ungricht
Michael Uschold
Marcela Vegetti
Laure Vieu
Amanda Vizedom
Evan Wallace
James Warren
Chris Welty
Matthew West
Trish Whetzel
Nancy Wiegand
Tim Wilson
Dennis Wisnosky
Peter Yim
Cecilia Zanni-Merk
Marcia Zeng