A theoretical framework for ontology evaluation and validation

Aldo Gangemi, Carola Catenacci, Massimiliano Ciaramita, Jos Lehmann

Laboratory for Applied Ontology, ISTC-CNR, Roma (Italy)
{aldo.gangemi, carola.catenacci, m.ciaramita, jos.lehmann}@istc.cnr.it

Abstract. The need for evaluation-methodologies emerged very early in the field of ontology development and reuse and it has grown steadily. Yet, no comprehensive and global approach to this problem has been proposed to date. This situation may become a serious obstacle for the success of ontology-based Knowledge Technology, especially in the industrial and commercial sectors. In this paper we look at existing ontology-evaluation methods from the perspective of their integration in one single framework. Based on a catalogue of qualitative and quantitative measures for ontologies, we set up a formal model for ontology. The proposed formal model consists of a meta-ontology - O^2- that characterizes ontologies as semiotic objects. The meta-ontology is complemented with an ontology of ontology evaluation and validation - oQual. Based on O^2 and oQual, we identify three main types of measures for ontology evaluation: structural measures, that are typical of ontologies represented as graphs; functional measures, that are related to the intended use of an ontology and of its components, i.e. their function; usability-related measures, that depend on the level of annotation of the considered ontology.

1. Introduction

The need for evaluation-methodologies in the field of ontology development and reuse emerged as soon as 1994 [Sure 2004] and it has grown steadily ever since. Yet, no comprehensive and global approach to this problem has been proposed to date. This situation may become a serious obstacle for the success of ontology-based Knowledge Technology, especially in the industrial and commercial sectors. A typical example in this sense is the development of the Semantic Web. On the one hand, the idea of conveying semantics through ontologies arouses the interest of large parts of the Software Industry. Ontologies promise to be crucial components of web-like technologies that are able to cope with high interconnection, constant change and incompleteness. On the other hand, though, the lack of well-understood notions of ontology evaluation, validation and certification significantly slows down the transition of ontologies from esoteric symbolic structures into reliable industrial components.

In this paper we look at existing ontology-evaluation methods from the perspective of their integration in one single framework. We first provide a brief overview of the state of the art. Partially based on such catalogue of qualitative and quantitative measures for ontologies, we set up a formal model for ontology-evaluation, mainly focusing on theoretical issues. The proposed formal model consists of a meta-

ontology – called O^2 – which characterizes ontologies as semiotic objects. O^2 is meant to provide a meta-theoretical foundation to ontology evaluation and annotation. The meta-ontology is complemented with an ontology of ontology evaluation and validation – oQual – which allows to pick up ontology elements by means of O^2, and which provides quality-parameters and, when feasible, their ordering functions. In practice, we model ontology evaluation as a diagnostic task based on ontology-descriptions. Such descriptions make explicit knowledge that is key to ontology-evaluation, namely: roles and functions of the elements of the ontology that is subject to evaluation; parameters for the descriptions that typically denote the quality of an ontology; functions that compose those parameters according to a preferential ordering. Based on O^2 and oQual it is possible to identify three main types of measures for ontology evaluation: structural measures, that are typical of ontologies represented as graphs; functional measures, that are related to the intended use of an ontology and of its components, i.e. their function; usability-related measures, that depend on the level of annotation of the considered ontology. For each of these measure-types our paper provides (formal) definitions as well as examples of preferential orders. Finally, some conclusions are drawn.

2. State of the art in ontology evaluation

The available literature on ontology evaluation is rather complex and fragmentary. Any given approach may address more or less specific evaluation issues, and often more than one quality-criterion is discussed at the same time, therefore only partially clarifying the problems at stake.

As opposed to this situation, [Hartmann 2004] tries to systematically disentangle issues by providing a classification-grid for ontology evaluation methods. In this review, we present various existing approaches and, in order to allow the comparison between them, we often use the classification-grid proposed in [Hartmann 2004] as background structure. Such grid allows to present ontology evaluation methods in terms of answers to the following questions: what is the considered method/tool like? Subordinately: what is its goal (Goal)? What functions are supported by it (Function)? At which stage of development of an ontology may it be applied (Application)? Furthermore, how useful is the method? Subordinately: for which type of users is it conceived (Users types: Knowledge Engineers, Project Managers, Application Users, Ontology Developers)? How relevant is it to practice (Usefulness)? How usable is it (Usability)? For which type of uses was it conceived in the first place (Use cases)?

[Yao et al. 2005] defines a number of Cohesion Metrics that are specific to ontologies. There exists a number of mathematical theories of how to describe and measure (graphical) structures. The most general ones are Graph Theory and Metric Theory. These define notions that are certainly relevant to the problem of ontology evaluation, but their level of abstraction makes them unsuitable for direct application. In order to define ontology-specific metrics [Yao et al. 2005] propose to adapt software cohesion metrics, which traditionally refers to the degree to which the elements in a software module belong together. Since cohesion metrics usually are intended to measure modularity, metrics similar to the software cohesion metrics can be defined to measure relatedness of elements in ontologies. The authors propose to

see ontology cohesion metrics as part of a measure for ontology modularity: ontology cohesion refers to the degree of the relatedness of OWL classes, which are conceptually related by the properties. An ontology has a high cohesion value if its entities are strongly related. The idea behind this is that the concepts grouped in an ontology should be conceptually related for a particular domain or a sub-domain in order to achieve common goals. The paper proposes three main Coehesion Metrics, or *functions*: Number of Root Classes (NoR); Number of Leaf Classes (NoL); Average Depth of Inheritance Tree of Leaf Nodes.

Most of the literature on ontology evaluation focuses on functionality-related issues, rather than structural ones. The functionality of an ontology is mostly measured by evaluating its appropriateness as semantic backbone of either decision-support or information systems that operate in the domain represented by the ontology.

[Lozano-Tello et al. 2004] proposes OntoMetric, an adaptation of the Analytic Hierarchy Process, i.e. a mathematical method for scaling priorities in hierarchical structures. The main *goal* of this method is to help choose the appropriate ontology for a new project. The *functions* supported by OntoMetric are the ordering by importance of project objectives, the qualitative analysis of candidate ontologies for the project, the quantitative measure of the suitability of each candidate. The *application* of OntoMetric can only follow ontology release. The method is meant for *users types* like Engineers or Project Managers who need to look for ontologies over the Web at the purpose of incorporating them into their systems. Therefore, OntoMetric makes itself *useful* as a support to the evaluation of the relative advantages and risks of choosing an ontology over others. The main drawback of OntoMetric is related to its *usability*: specifying the characteristics of an ontology is complicated and takes time; assessing its characteristics is quite subjective. On top of this, the number of *use cases* is limited, which is an important obstacle to defining (inter-subjective or objective) parameters based on a large enough number of comparable cases.

[Welty et al., 2001] proposes OntoClean, which is meant for application at the pre-modelling and modelling stages, i.e. during ontology development. The main *goal* is to detect both formal and semantic inconsistencies in the properties defined by an ontology. The main *function* of OntoClean is the formal evaluation of the properties defined in the ontology by means of a predefined ideal taxonomical structure of meta-properties.

[Spyns 2005] presents EvaLexon which finds *application* at the pre-modelling/modeling stage. The main *goal* here is to evaluate at development time ontologies that are created by human beings from text. In sharp contrast with OntoClean, EvaLexon is meant for linguistic rather than conceptual evaluation. Its main *function* is the measurement of how appropriate are the terms (to be) used in an ontology. A term is judged more or less appropriate depending on its frequency both in the text from which the ontology is (being) derived and in a list of relevant domain-specific terms. Regression allows for direct and indirect measurement of the ontology's recall, precision, coverage and accuracy.

In [Porzel et al., 2004], partly building on [Brewster et al., 2004], a linguistics-based approach partly comparable to EvaLexon is defined. The *goal* of the proposal is to evaluate ontologies with respect to three basic levels: vocabulary, taxonomy and (non-taxonomic) semantic relations. The *functions* proposed in [Porzel et al. 2004] are

based on two key arguments: the task and the gold standard. The task needs to be sufficiently complex to constitute a suitable benchmark for examining a given ontology. The gold standard is a perfectly annotated corpus of part-of-speech tags, word senses, tag ontological relations, given sets of answers (so-called keys) used to evaluate the performance of algorithms that are run on the ontology to perform the task.

[Gómez-Pérez, 2003] draws a distinction between two main evaluation dimensions: content evaluation and ontology technology evaluation. *Content evaluation* is related to the Knowledge Representation (KR) paradigm that underlies the language in which the ontology is implemented (be it RDF schemas, description logic, first order logic, etc.). The *goal* of content evaluation is to detect inconsistencies or redundancies before these spread out in applications. The *application* of content evaluation techniques should take place during the entire ontology life-cycle, as well as during the entire ontology-building process. *Functions* should support the evaluation of concept taxonomies, properties, relations and axioms. On the other hand, *ontology technology*, i.e. ontology development tools like OILed and Protégé, should be subject to evaluation too. Here the *goal* is to ensure smooth and correct integration with industrial software environments. The *application* of such evaluation should be directed at the expressiveness of the KR model underlying the ontology editor; the tool's interoperability, in terms of quality of import/export functions (i.e. how much knowledge is lost with format transformation), scalability (i.e. how different building platforms scale when managing large ontologies with thousands of components, as well as time required to open and save, etc.), navigability (e.g. how easy it is to search for a component), usability (e.g. user interfaces' clarity and consistency), and available content evaluation functions.

[Daelemans et al., 2004] points out how recently developed NPL techniques can be - and currently are - used for evaluating ontologies' semantics (vs their syntax). NLP not only helps content collection from huge amounts of text and maintenance, but it also provides the means for showing that ontologies indeed represent "consensual conceptualizations and not just one person's ideas".

In [Noy, 2004] it is argued that, although most structural and functional evaluation methods are necessary, none are helpful to ontology consumers, who need to discover which ontologies exist and, more important, which ones would be suitable for their tasks at hand. Knowing whether an ontology is correct according to some specific formal criteria might help in the ultimate decision to use an ontology but will shed little light on whether or not it is good for a particular purpose or task. What is needed is not only a system for objectively evaluating ontologies from some generic viewpoint, but also practical ways (*function*) for ontology consumers to discover and evaluate ontologies. Information such as the number of concepts or even an ontology's complete formal correctness is probably not the most important criteria in this task (although it is often the easiest to obtain). Based on this considerations alternative techniques are proposed: *ontology summarization*, *e-pinions for ontologies*, *views and customization*.

3. O^2: a semiotic meta-ontology

We consider an ontology as a semiotic object, i.e. an object constituted by an *information object* and an *intended conceptualization* established within a *communication setting*.
The basic intuition is that information is any pattern that is used to represent another pattern, whereas that representation is interpretable by some rational agent (this intuition comes back at least to C.S. Peirce [Peirce 1931]) as an explanation, an instruction, a command, etc.

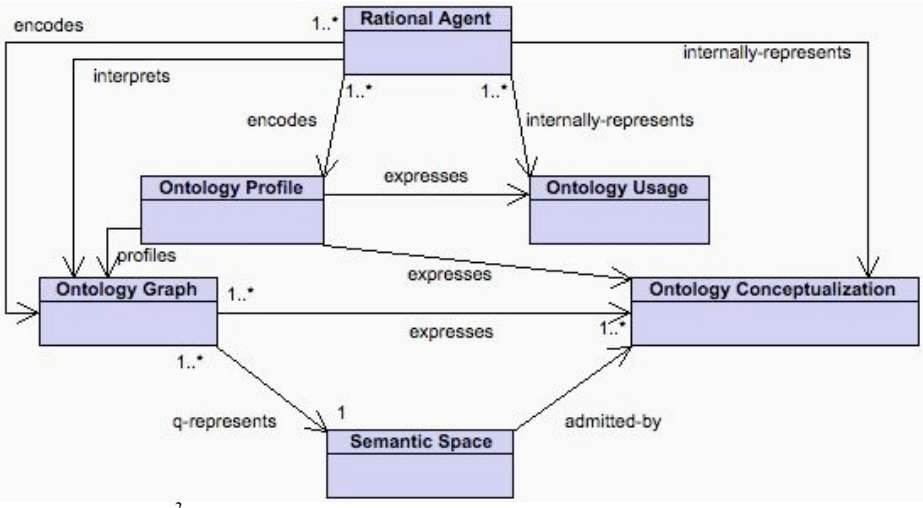

Figure 1. The O^2 design pattern, based on the Information↔Description pattern from the DOLCE-Lite-Plus ontology library [Masolo et al, 2004]. Ontologies are graphs that express a conceptualization and can be profiled by additional information that expresses their usage context. An ontology graph has ("q-represents") a formal semantics if it is admitted by the conceptualization. These constraints are the sensible part of ontology evaluation: does the formal semantics of a graph catch the intended conceptualization (the "cognitive" semantics)?

That intuition is formalized by applying an ontology design pattern called *Description↔Situation* [Gangemi, 2005], and originates a new pattern called O^2 (because it is a "meta-ontology"), which formalizes the following specification: an ontology is a special kind of information, whose patterns are graph-like structures, and whose represented patterns are intended conceptualizations, i.e. internal representations (for a rational agent) of *entity types*. For example, one can define an ontology for subways, but one will hardly consider the London Underground graph as an ontology (it would be eventually considered a *model* of an appropriate subway ontology).
The UML class diagram in Fig.1 summarizes O^2: an ontology graph has an intended conceptualization and a formal semantic space admitted by the conceptualization. The graph and the conceptualization are kept together by a rational agent who

encodes/interprets the graph. An agent can also provide a profile of the structural and functional properties of an ontology graph in order to enhance or to enforce its usability.

4. oQual: a model of ontology evaluation and validation

We model ontology evaluation and validation as a *diagnostic task* over ontology elements, processes, and attributes (Fig.s 2,3) involving:

- *Quality-Oriented Ontology Descriptions (*qoods*)*, which provide the *roles* and *tasks* of the elements resp. processes from/on an ontology, and have elementary qoods (called *principles*) as parts. For example, a type of qood is *retrieve*, which formalizes the requirement to be able to answer a certain competency question. In Fig. 3, the *retrieve* type is instantiated as a requirement to the ontology to be able to retrieve the "family history for a condition related to haemocancer", in an ontology project for "haemocancer information service".
- *Value spaces* ("attributes") of ontology elements. For example, the presence of a relation such as: R(*p,f,c,i*), where Patient(*p*), Family(*f*), Condition(*c*), Indicator(*i*).
- *Principles* for assessing the ontology fitness, which are modelled as elementary qoods, and are typically parts of a project-oriented qood. For example, "description of fitness to expertise" is a principle.
- *Parameters* (ranging over the attributes -*value spaces*- of ontologies or ontology elements), defined within a principle. For example, "relation fitness to competency question" is a parameter for the relation R(*p,f,c,i*).

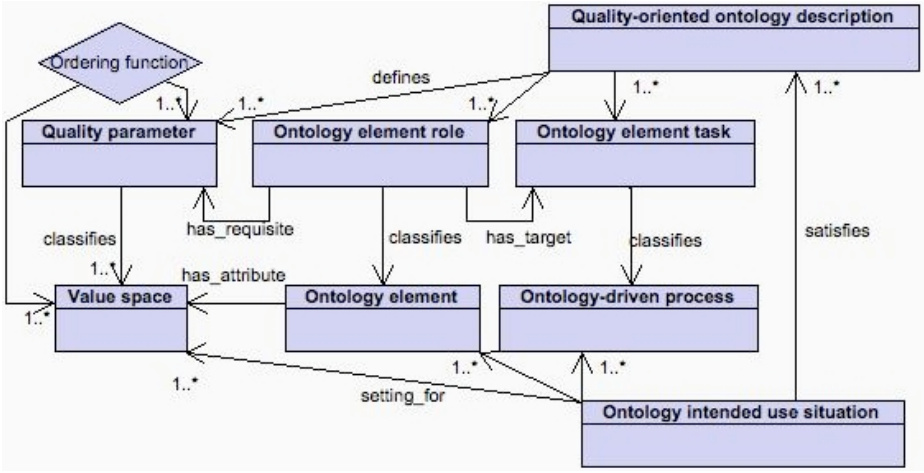

Figure 2. The ontology evaluation design pattern (oQual).

- *Parameter dependencies* occurring across principles because of the interdependencies between the value spaces of the measured ontology elements. For example, the "relation fitness to competency question" parameter is dependent on either "first-order expressiveness" or "presence of a relation reification method" parameters ranging on the logical language of the ontology, because the relation

R(*p,f,c,i*) has four arguments and it's not straightforwardly expressible in e.g. OWL(DL).

- *Preferential ordering* functions that compose parameters from different principles. For example, in a "haemocancer information service" project, the "relation fitness to competency question" parameter may be composed with the "computational complexity" parameter.
- *Trade-offs*, which provide a conflict resolution description when combining principles with conflicting parameters. For example, the two abovementioned parameters might be conflicting when the cost of the expressiveness or of the reification method are too high in terms of computational efficiency. A trade-off in this case describes a guideline to simplify the competency question, or a strategy to implement the relation differently.

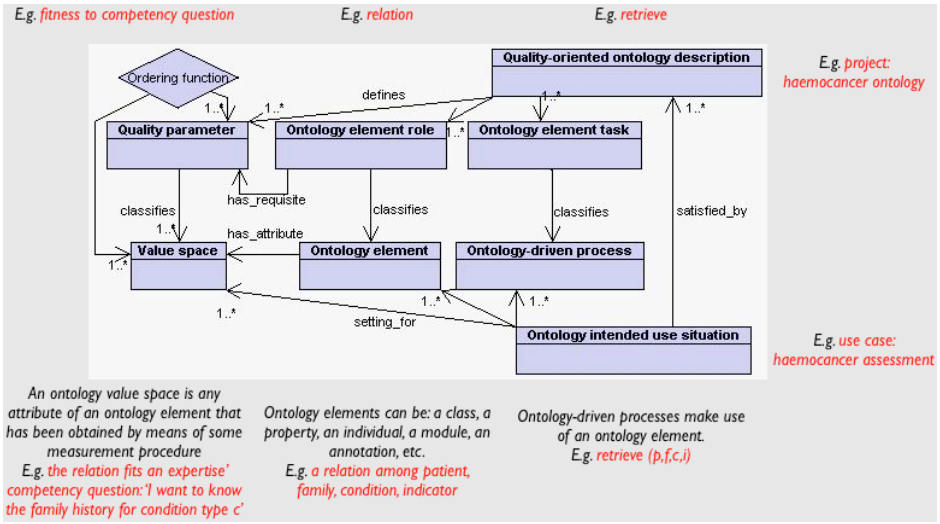

Figure 3. Applying oQual to a clinical use case.

The formal model for ontology diagnostic tasks, called *oQual*, is based on the the *Description↔Situation* pattern [Gangemi, 2005] from the DOLCE-Lite-Plus ontology library [Masolo et al., 2004], which is integrated with the *Information↔Description* pattern used for O^2.

Ontology descriptions, roles, parameters, and ordering functions are defined on the results of the measurement types that can be performed on an ontology graph, conceptualization, or profile. The results are represented as regions within value spaces). Quality parameters constrain those regions *within* a particular qood.

5. Measures

Based on the O^2 pattern, we have devised a large amount of possible measurement methods for ontologies. We introduce here the main distinctions among measure sets, and provide some examples of them.

5.1 Measure types

Ontology evaluation deals with establishing the measures for an ontology, and deploying them within a qood. We want to answer the following three questions.

What to measure in an ontology? and how? the quality of an ontology may be assessed relatively to various dimensions. As explained above (see Section 1), by ontology we mean any kind of graph of metadata, and we propose to measure its quality relatively to three main groups of dimensions: structural, functional and usability-related dimensions.

An ontology shows its *structural* dimensions when represented as a graph. In this form, the topological and logical properties of an ontology may be measured by means of a metric. The existence of these structural dimensions, however, can be considered independent from the metric is being used.

The *functional* dimensions are related to the intended use of a given ontology and of its components, i.e. their function. Functional dimensions include *agreement, task, topic, design*, etc. Such dimensions become apparent in an ontology depending on the context, which in turn is given by how the ontology is chosen, built, exploited, etc.

Finally, the *usability-related* dimensions depend on the level of annotation of a given ontology. How easy it is for users to recognize its properties? How easy is to find out which one is more (economically, computationally) suitable for a given (series of) task(s)?

Notice that those dimensions follow a partition into *logical types*: structurally, we look at an ontology as an (information) object; functionally, we look at it as a language (information object+intended conceptualization), and from the usability viewpoint, we look at its meta-language (the profile about the semiotic context of an ontology). Therefore, the dimension types correspond to the constituents of the O^2 pattern (Fig.1), and heterogeneous measurement methods are needed.

Which parameters for the quality of an ontology? Each measure can have more than one quality parameter, depending on other parameters/measures, and the overall composition for a given ontology project implies a non-linear procedure to quality assessment. For example, in an ontology project we may want to combine measures like *logical complexity* and *presence of dense areas* (e.g. of *design patterns*). If *high density* is chosen as a quality parameter, then the parameter associated with *high complexity* is chosen too, because usually dense areas involve a lot of restrictions, sometimes with indirect cycles; in other words, high-density parameter depends on the high-complexity parameter. On the other hand, if the quality parameter is *lower complexity*, then the parameter associated with *lower density* is chosen too, because the first depends on the second. Actually, this is an application of a general pattern of parameter composition ranging on mutually dependent scalar spaces: when we compose two parameters p_1 and p_2 ranging respectively on value spaces s_1 and s_2 with a scalar metrics, and p_1 ranges over the higher part of s_1, and also depends on p_2 ranging over the lower part of s_2, then the converse is true, i.e. that a parameter p_3 ranging on the higher part of s_2 depends on a parameter p_4 ranging on the lower part of s_1. Hence, different trade-offs denote good/bad quality according to which criterion is preferred. oQual formalizes the observation that quality parameters are defined according to some principle, e.g. in the example, “high” parameters could be defined with reference to a *transparency* principle, while the “low” parameters could be

defined with reference to a *computational efficiency* principle. When combining principles, the need for a trade-off typically arises, producing either a *preference ordering function*, or a *relaxation of parameters*. In our example project, the preference ordering function is: $pref(p_1^q, p_2^r, p_1^q, p_2^r, c) \mapsto p_i^x$, where *q a* and *r* are principles, p_i^x is a parameter defined by a principle, and *c* is a local constraint (a "meta-parameter"), e.g. *availability of resources*, *user overruling*, *good practice*, etc.
If no local constraint can be applied to create a preference ordering function, the trade-off can resort to a relaxation of parameters. In our example project, either high density can be relaxed to e.g. *medium-high density*, or low complexity can be increased to e.g. *medium-low complexity*.
Which examples? There are typical examples and patterns of good/bad quality for each measure. In the next future, examples will be provided after the analysis of a sample set of ontologies, and an appropriate processing of user feedback (e.g. from the KnowledgeZone initiative.[1]

5.2 Measuring the structural dimension

The structural dimension of ontologies focuses on syntax (e.g. graph structure), and formal semantics.

Here we propose our own treatment of structural dimensions. The idea is to define a general function like the following: $M = \langle D, S, mp, c \rangle$, where dimension *D* is a graph property or concept we want to measure: the intensional counterpart of the metric space; the set of graph elements *S* is a collection of elements in the graph (which may be seen as the ontology structure); *mp* is a measurement procedure; and *c* is a coefficient of measurement error.

The value of *M* is a real number obtained by applying a measurement procedure *mp* for a dimension *D* to a set *S* of graph elements, modulo a coefficient *c* (if any), i.e.: $mp_{D,c,S} \xrightarrow[yields]{} m \in \Re$

Within the possible sets of graph elements, we consider here the following sets:

- The set of graph nodes *G* from a graph *g*, $G \subseteq S$
- The set of root nodes $ROO \subseteq G$, where the root nodes are those having no outgoing *isa* arcs in a graph *g*.
- The set of leaf nodes $LEA \subseteq G$, where the leaf nodes are those having no ingoing *isa* arcs in a graph *g*.
- The sets of sibling nodes $SIB_{j \in G}$ connected to a same node *j* in a graph *g* through *isa* arcs.
- The set of paths *P* where $\forall j \in P \Rightarrow j \subseteq G$, where a path *j* is any sequence of directly connected nodes in a digraph *g* starting from a root node $x_{\in ROO}$ and ending at a leaf node $y_{\in LEA}$.

[1] http://smi-protege.stanford.edu:8080/KnowledgeZone.

- The set of levels ("generations") L where $\forall j \in L \Rightarrow j \subseteq G$, where a generation j is the set of all sibling node sets having the same distance from (one of) the root node(s) $r_{\in ROO}$ of a digraph g.
- The sets of graph nodes $N_{j \in P}$ from a same path j in a digraph g
- The sets of graph nodes $N_{j \in L}$ from a same level j in a digraph g
- The set M of modules from a graph g. A *module* is any subgraph sg of g, where the set of graph elements S' from sg is such that $S' \subseteq S$. Two modules sg_1 and sg_2 are *taxonomically disjoint* when only ≥0 *isa* arcs a_i connect sg_1 to sg_2, and each a_i has the same direction.

Several structural measures can be defined, involving dimensions such as *depth*, *breadth*, *tangledness*, *leaf and sibling distribution*, *density*, *modularity*, *consistency*, *complexity*, *logical elements distribution*, *etc.* An extended list of the measures definable within those dimensions is presented in [Gangemi et al., 2005]; here we include some of them as examples.

Depth is a graph property related to the cardinality of paths in a graph, where the arcs considered are only isa arcs. This measure type only applies to digraphs (directed graphs). E.g., **average depth**, where $N_{j \in P}$ is the cardinality of each path j from the set of paths P in a graph g, and $n_{P \subseteq g}$ is the cardinality of P: $m = \frac{1}{n_{P \subseteq g}} \sum_{j}^{P} N_{j \in P}$.

Breadth is a property related to the cardinality of levels ("generations") in a graph, where the arcs considered here are again only isa arcs. E.g. **average breadth**, where $N_{j \in L}$ is the cardinality of each generation j from the set of generations L in a digraph g, and $n_{L \subseteq g}$ is the cardinality of L: $m = \frac{1}{n_{L \subseteq g}} \sum_{j}^{L} N_{j \in L}$.

Tangledness is related to the multihierarchical nodes of a graph. In the **tagledness** measure n_G is the cardinality of G, and $t_{\in G \wedge \exists a_1, a_2 (isa(m,a_1) \wedge (isa(m,a_2)}$ is the cardinality of the set of nodes with more than one ingoing *isa* arc in g: $m = \frac{n_G}{t_{\in G \wedge \exists a_1, a_2 (isa(m,a_1) \wedge (isa(m,a_2)}}$.

Fan-outness is related to the "dispersion" of graph nodes, along isa arcs. We distinguish the fan-outness measures related to *leaf node sets*, and fan-outness measures related to *sibling node sets* ("internal dispersion"). For example, in the **ratio of leaf fan-outness** $n_{LEA \subseteq g}$ is the cardinality of the set LEA in the digraph g, and n_G is the cardinality of G: $m = \frac{n_{LEA \subseteq g}}{n_G}$

Density can be defined as the presence of clusters of classes with many non-taxonomical relations holding among them (wrt to overall ontology graph). For example, so-called *core ontology patterns* (e.g. including thematic roles in events,

contracts, diagnoses, etc.) usually constitute dense areas in an ontology. The following measures can be established.

1. Various clustering techniques can be used to detect dense areas, and the absolute size and number of them can be measured.
2. A measure of the relevance of those areas for the overall ontology can be obtained by calculating the proportion of classes and properties in the ontology, which logically depend on the dense areas.
3. Dense areas can be -explicitly or implicitly- a specialization of *content ontology design patterns* [Gangemi 2005].

Modularity is related to the asserted modules of a graph, where the arcs considered here are either *isa* or *non-isa* arcs. For example, the **modularity rate**, where n_M is the cardinality of *M*, and n_S is the cardinality of *S* (the set of graph elements) within a module: $m = \frac{n_M}{n_S}$

Logical adequacy is related to graphs having a formal semantics, where the arcs considered here are either isa or conceptual relation arcs. For example, the **consistency ratio**, where n_{Inc} is the cardinality of the set of consistent classes from *g*, and n_G is the cardinality of the set of (class) nodes from *g*: $m = \frac{n_{Cons}}{n_G}$

5.3 Measuring the functional dimension

The functional dimension is coincident with the main purpose of an ontology, i.e. specifying a given conceptualization, or a set of contextual assumptions about an area of interest. Such specifications, however, are always approximate, since the relationship between an ontology and a conceptualization is always dependent (Fig. 1) on a rational agent that conceives that conceptualization (the "cognitive" semantics) and on the formal encoding of that conceptualization (the "formal" semantics). Hence, an appropriate evaluation strategy should involve a measurement of the degree of how those dependencies are implemented. We refer to this as the *matching problem*.

The matching problem requires us to find ways of measuring the extent to which an ontology mirrors a given expertise [Steels, 1990], competency [Uschold, 1996], or task: something that is in the experience of a given community and that includes not only a corpus of documents, but also theories, practices and know-hows that are not necessarily represented in their entirety in the available documents. This seems to imply that no automatized method will ever suffice to the task and that intellectual judgement will always be needed. However, both automatic and semi-automatic techniques can be applied that make such evaluation easier, less subjective, more complete and faster (cf. [Daelemans et al. 2004]).

The functional measures provided in [Gangemi et al., 2005] are variants of the measures introduced by [Guarino 2004], which uses an analogy with *precision, recall,* and *accuracy* measures, which are widely used in information retrieval (cf. [Baeza-Yates & Ribeiro-Neto, 1999]). Due to the matching problem, the adaptation of precision, recall and accuracy to ontology evaluation is by no means straightforward. Since expertise is by default in the cognitive "black-box" of rational agents, ontology

engineers have to elicit it from the agents, or they can assume a set of data as a *qualified expression* of expertise and task, e.g. texts, pictures, diagrams, database records, terminologies, metadata schemas, etc.

Based on these assumptions, precision, recall and accuracy of an ontology can be measured against: a) experts' judgment, or b) a data set assumed as a qualified expression of experts' judgment. Therefore, we distinguish between *black-box* and *glass-box* measurement methods:

(1) *Agreement* (*black-box*). It is measured through the proportion of *agreement* that experts have with respect to ontology elements; when a group of experts is considered, we may also want to measure the *consensus* reached by the group's members.
(2) *User-satisfaction* (*black-box*). It can be measured by means of dedicated polls, or by means of provenance, popularity, and trust assessment.
(3) *Task*: what has to be supported by an ontology? (*glass-box*). It deals with measuring an ontology according to its fitness to some goals, preconditions, postconditions, constraints, options, etc. This makes the measurement very reliable at design-time, while it needs a reassessment at reuse-time.
(4) *Topic*: what are the boundaries of the knowledge domain addressed by an ontology? (*glass-box*). It deals with measuring an ontology according to its fitness to an existing knowledge repository. This makes the measurement reliable both at design-time, and at reuse-time, but is based on the availability of data that can be safely assumed as related to the (supposed) topic covered by an ontology. Natural Language Processing (NLP)-based methods fit into this category, and are currently the most reliable method for ontology evaluation, at least for lightweight ontologies.
(5) *Modularity*: what are the building blocks for the design of an ontology? (*glass-box*). It is based on the availability of data about the *design* of an ontology. Therefore, it deals with measuring an ontology according to its fitness to an existing repository of reusable components. This makes the measurement very reliable both at design-time, and at reuse-time. On the other hand, modularity can only be assessed easily on ontologies that have been designed with an appropriate methodology.

Black-box methods require rational agents, because they don't explicitly use knowledge of the internal structure of an expertise.

Glass-box methods require a data set that "samples" that knowledge, and, on this basis, we can treat the internal structure of those data *as if* it is the internal structure of an expertise.

[Gangemi et al., 2005] has extensive discussions on the abovementioned methods, which cannot be summarized here. The most automatized techniques are currently NLP-based, and here we present a summary of them.

5.4 NLP-driven evaluation

When the ontology is lexicalized; i.e., it defines, at least to some extent, what instances of classes and relations are called in natural language, and there exists a substantial amount of textual documents that contain information about the content of

the ontology, NLP can support ontology evaluation in several ways. A typical such case is when the ontology directly supports information retrieval or text mining applications and thus concerns objects mentioned in web-pages or other large repositories of texts (e.g., newswire, biomedical or legal literature, etc.). One of the simplest examples of lexicalized ontology is the kind used for newswire information extraction which is usually based on three classes: "person" (e.g., Kofi Annan), "location" (e.g., South East Asia), and "organization" (e.g., U.N.). Sometimes these classes are also associated with relations such as is-located-in(Djakarta, South_East_Asia) or works-for(Kofi_Annan,U.N.).

If there is a corpus of documents that contains the kind of information conceptualized in the ontology, NLP can be used to identify occurrences in text of classes and relations. A corpus-based analysis of the ontology can reveal important properties of the ontology that might not be discovered otherwise. Most importantly, it allows to estimate empirically the accuracy and the coverage of the ontology.

By identifying mentions of ontological elements in the corpus, it is possible to count the frequency of classes (or relations). The relative frequency of each class c (or relation r) is the proportion of mentions of ontology instances which are equal to c; i.e., $P(c) = count(c)/sum_i\ count(c_i)$. The relative frequency measures the importance of each class and provides a first simple measure of the ontology quality. For example, in newswire text the three classes above have somewhat similar frequencies, while if the corpus analysis reveals that one of the classes is much more unlikely than the others this means that there is something wrong with the instances of that class. this might indicate that the low frequency class is underrepresented in the ontology, at the lexical level. If the ontology has a hierarchical structure, it is also possible to estimate the frequencies of higher of superordinate concepts, the frequency of a class c then would be the sum of the frequencies of its descendants. The probability of a concept in a hierarchy can be computed as $P(c) = sum_i\{c_j$ is descendant of $c\}\ count(c_j)/sum_i\ count(c_i)$. Each class can be seen as a random variable, and this can be useful to estimate the information-theoretical measures such as entropy $H(c) = -sum_j\ \{c_j$ is descendant of $c\}\ P(c_j)logP(c_j)$. Entropy and other information theoretic measures can be used to identify classes that are particularly useful or "basic" [Gluck & Corter, 1985]. Thus for example in a general purpose ontology, a concept such as "tree", which has many descendants similar to each other, is likely more important than a concept such as "entity" which has very dissimilar descendants (e.g., organisms, artifacts, etc.).

One problem with trying to estimate distributional properties of the ontology directly is that the existing lexicon associated with the ontology might be insufficient because it contains only the names that the experts have listed. Notice that creating such "dictionaries" requires not only domain expertise but also lexicographic expertise and it is a slow and expensive process. Therefore typically the starting ontology lexicon is quite limited. This issue impacts on the matching problem, and the related measures of *precision* and *recall* for ontology.

Intuitively, precision measures the ability of a system in recognizing instances of a given class, while recall measures the coverage of the system, that is how many true instances were left out. Measuring precision and recall requires manual tagging of enough textual data to be able to compare the empirical lexicon so generated with the ontology lexicon. Typically, the lexicon that is defined by the experts has a good

precision because it is unlikely that wrong instances were placed in any class/relation lists. However, the lexicon defined by the experts can be limited on several aspects, because it can have very low coverage, thus miss important instances, or it is not a sample of the domain thus it can over-represent certain types of objects and under-represent others.

NLP can be used for assisting experts in populating the objects defined by the ontology. Machine learning methods for supervised and unsupervised classification can be applied to corpus data to retrieve unknown instances of ontology objects. So far most of the work in this area has concentrated on the problem of finding new members of a class of objects [Riloff, 1996][Roark & Charniak, 1998], and on finding examples of structural relations such as is-a [Hearst, 1992][Pantel & Ravichandran, 2004] or part-of [Berland & Charniak, 1999]. Recent work however has focused also on discovering class attributes [Almuhareb & Poesio, 2004] and arbitrary relation between classes [Ciaramita et al., 2005]. Automatic or semi-automatic population of ontology objects is valuable also in terms of evaluation. In fact, it is possible that new senses of already known instances are discovered, for example because the instance is polysemous/ambiguous (e.g., "Washington" is a person and a location).

5.5 Measuring the usability-profile of ontologies

Usability-profiling measures focus on the ontology profile, which typically addresses the communication context of an ontology (i.e. its pragmatics). An ontology profile is a set of ontology annotations, i.e. the metadata about an ontology and its elements. *Presence, amount*, *completeness*, and *reliability* are the usability measures ranging on annotations.

Annotations contain information about structural, functional, or user-oriented properties of an ontology. There are also purely user-oriented properties, e.g. *authorship, price, versioning, organizational deployment, interfacing*, etc.

Three basic levels of usability profiling have been singled out: **recognition, efficiency,** and **interfacing**.

The *recognition level* makes objects, actions, and options visible. Users need an easy access to the instructions for using ontologies in an effective way, and an efficient process to retrieve appropriate meta-information. That is, "give your users the information that they need and allow them to pick what they want". Hence recognition is about having a complete documentation and to be sure to guarantee an effective access. Recognition annotations include at least the following ones:

1. Annotations (of the overall ontology) about the ontology structure: graph measures; logic-type and computational complexity; meta-consistency; modularization (e.g. owl:imports)
2. Annotations about the ontology function (either at design-time or reuse-time): lexical annotation of ontology elements (incl. multilingual); glosses (e.g. rdfs:comment) about ontology elements; agreement status; user satisfaction (e.g. http://smi-protege.stanford.edu:8080/KnowledgeZone/) and trust rating; task of the overall ontology (both originally and during its lifecycle); topic (e.g. rdf:about) of the overall ontology; and modularization design of the overall ontology
3. Annotations about the ontology lifecycle (either of the overall ontology, or of its elements): provenance; methods employed; versioning (e.g. owl:versionInfo), compatibility (e.g. owl:incompatibleWith)

Notice that annotations can be *resident* in an ontology file, *linked* through a URI, *dynamically produced* when needed (e.g. by a local software component, or through a web service), or *retrieved* from an incrementally growing repository (e.g. from a portal that collects users' feedback).

The *efficiency level* includes organizational, commercial, and developmental annotations. Large organizations tend to be compartmentalized, with each group looking out for its own interests, sometimes to the detriment of the organization as a whole. Information resource departments often fall into the trap of creating or adopting ontologies that result in increased efficiency and lowered costs for the information resources department, but only at the cost of lowered productivity for the company as a whole. This managing-operating-balance principle boils down to some requisites (*parameters*) for the *organization-oriented design* of ontology libraries (or of distributed ontologies), which provide constraints to one or more of the following entities: *organization architecture, (complex) application middleware, trading properties, cost, accessibility*, *development effort*. These parameters are defined by the principle of organizational fitness, and are annotated as follows:

1. Annotations (either on the overall ontology, or on ontology elements) about the *organizational design* of a modularized ontology, and about the middleware that allows its deployment.
2. Annotations about the *commercial* (trading, pricing) and *legal* (policy, disclaimer) *semantics*.
3. Annotations about the *application history* -with reference to development effort (task- or topic-specificity applied to a token scenario) of an ontology.

The *interfacing level* concerns the process of matching an ontology to a user interface. As far as evaluation is concerned, we are only interested in the case when an ontology includes annotations to interfacing operations. For example, a *contract negotiation* ontology might contain annotations to allow an implementation of e.g. a *visual contract modelling language*. If such annotations exist, it is indeed an advantage for ontologies that are tightly bound to a certain (computational) service. On the other hand, such annotations may result unnecessary in those cases where an interface language exists that maps to the core elements of a core ontology e.g. for contract negotiation.

6. Conclusions

Current and future work is focusing on the empirical assessment of the framework by measuring existing ontologies, comparing the quality of distinct ontologies that represent the same domain, creating correlations between user-oriented and structural measures, and creating tools to assist ontology evaluation in large industry- and agency-scale projects.

Bibliography

Almuhareb A., Poesio M., 2004: “Attribute-based and value-based clustering: an evaluation”. In Proceedings of the Conference on Empirical Methods in Natural Language Processing.
Berland M. and Charniak E., 1999: "Finding parts in very large corpora." In Proceedings of ACL'99.
Brewster C., Alani H,, Dasmahapatra S. and Wilks Y.: “Data-driven ontology evaluation”. Proceedings of LREC 2004.
Ciaramita M., Gangemi A., Ratsch E., Saric J., and Rojas I., 2005: “Unsupervised Learning of Semantic Relations between Concepts of a Molecular Biology Ontology”. In Proceedings of the 19th International Joint Conference on Artificial Intelligence.
Daelemans W., Reinberger M.L. , 2004: “Shallow Text Understanding for Ontology Content Evaluation” . IEEE Intelligent Systems 1541-1672, 2004.
Gangemi A., Catenacci C., Ciaramita M. Gil R., and Lehmann J., “Ontology evaluation: A review of methods and an integrated model for the quality diagnostic task”, Technical Report available at http://www.loa-cnr.it/Publications.html, 2005.
Gangemi A.: “Ontology Design Patterns for Semantic Web Content”. In Motta E. and Gil Y., Proceedings of the Fourth International Semantic Web Conference, 2005.
Gluck M. and Corter J., 1985: “Information, Uncertainty and the Utility of Categories”. In Proceedings of the 7th Annual Conference of the Cognitive Science Society.
Gómez-Pérez A.: “Ontology Evaluation”, in Handbook on Ontologies, S. Staab and R. Studer, eds., Springer-Verlag, 2003,pp. 251–274.
Guarino N.: “Towards a Formal Evaluation of Ontology Quality”. IEEE Intelligent Systems 1541-1672, 2004.
Hartmann J., Spyns P., Giboin A., Maynard D., Cuel R., Suárez-Figueroa M.C., and Sure Y., “Methods for ontology evaluation”. Knowledge Web Deliverable D1.2.3, v. 0.1 (2004).
Kaakinen, J., Hyona, J., & Keenan, J.M. (2002). Individual differences in perspective effects on on-line text processing. Discourse Processes, 33, 159 - 173.
Lozano-Tello, A. and Gomez-Perez A., 2004: “ONTOMETRIC: A method to choose the appropriate ontology”, J. of Database Management, 15(2).
Masolo, C., A. Gangemi, N. Guarino, A. Oltramari and L. Schneider: WonderWeb Deliverable D18: The WonderWeb Library of Foundational Ontologies (2004).
Noy, N., “Evaluation by Ontology Consumers”. IEEE Intelligent Systems 1541-1672, 2004.
Pantel P. and Ravichandran D., 2004: “Automatically Labeling Semantic Classes”. In Proceedings of HLT-NAACL 2004.
Peirce, Charles (1931-1958). Collected Papers, vols. 1-8, C. Hartshorne, P. Weiss and A.W. Burks (eds). Cambridge, MA: Harvard University Press.
Porzel R. and Malaka R.: “A Task-based Approach for Ontology Evaluation”. Proc. of ECAI 2004.
Spyns P., EvaLexon: Assessing triples mined from texts. Technical Report 09, STAR Lab, Brussel, 2005.
Steels L.: “Components of Expertise”, AI Magazine, 11, 2, 1990, pp. 30-49.
Sure Y. (ed.), 2004: “Why Evaluate Ontology Technologies? Because It Works!”, IEEE Intelligent Systems 1541-1672.
Uschold U. and Gruninger M.,”Ontologies: Principles, Methods, and Applications,” Knowledge Eng. Rev.,vol. 11,no. 2,1996,pp. 93–155.
Welty C., Guarino N., "Supporting ontological analysis of taxonomic relationships", *Data and Knowledge Engineering* vol. 39, no. 1, pp. 51-74, 2001.
Welty C., Kalra R., and Chu-Carroll J., 2003: “Evaluating Ontological Analysis”. In Proceedings of the ISWC-03 Workshop on Semantic Integration.
Yao H., Orme A.M., and Etzkorn L., 2005: “Cohesion Metrics for Ontology Design and Application”, Journal of Computer Science, 1(1): 107-113, 2005.