Chapter 5
Ontological Evaluation and Validation

Samir Tartir, I. Budak Arpinar, and Amit P. Sheth

5.1 Introduction

Building an ontology for a specific domain can start from scratch (Cristani and Cuel, 2005) or by modifying an existing ontology (Gómez-Pérez and Rojas-Amaya, 1999). In both cases, techniques for evaluating the characteristics and the validity of the ontology are necessary. Not only such techniques might be useful during the ontology engineering process (Paslaru et al., 2006), they can also be useful to an end-user who is looking for an ontology that is suitable for her application domain. The user can select the best ontology according to her application needs among several ontologies (Sabou et al., 2005).

Ontology evaluation is an important task that is needed in many situations. For example, during the process of building of an ontology, ontology evaluation is important to guarantee that what is built meets the application requirement. Fernández et al. (1999) presents a life cycle for ontologies (Fig. 5.1). The life cycle is mainly based on Software Engineering processes. Their cycle includes three sets of activities: Management (that includes control and quality control), technical (that includes tasks for building an ontology), and support (that includes activities that are performed at the same time as the technical tasks). In this methodology, ontology evaluation was presented as an ongoing process throughout the ontology lifecycle in both the management and the support activities to illustrate its importance. Ontology evaluation is also important in cases where the ontology is automatically populated from different resources that might not be homogeneous, leading to duplicate instances, or instances that are clustered according to their sources in the same ontology, both of which may decrease the usefulness of the ontology. For example, the search for semantic associations (Anyanwu and Sheth, 2003) between entities in ontologies has been a major focus for the semantic web. These associations capture the complex relationships between entities that might be involve several other entities and can't be easily captured by human users in the midst of a large dataset. If a

S. Tartir (✉)
Faculty of Information Technology, Philadelphia University, Amman 19392, Jordan
e-mail: startir@philadelphia.edu.jo

R. Poli et al. (eds.), *Theory and Applications of Ontology: Computer Applications*,
DOI 10.1007/978-90-481-8847-5_5,

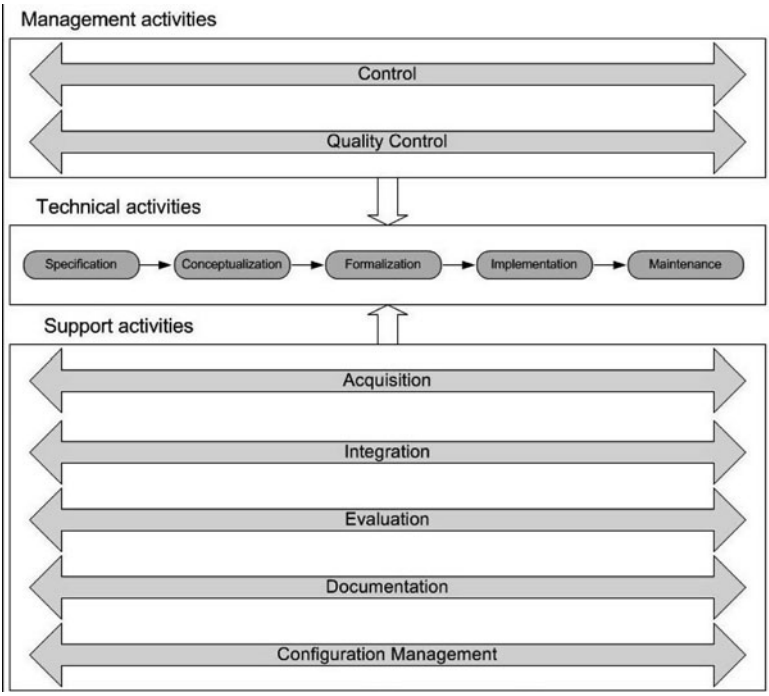

Fig. 5.1 Ontology life-cycle in meth ontology

user is interested in this type of search, she will also be interested to know about the presence of clusters of instances in the ontology, or the lack of a diverse set of relationnships that might be of importance to her, because this knowledge will directly affects results return by this type of search.

In addition to the need during the process of building an ontology, evaluation and validation of ontologies are also useful for end users in domains where several ontologies with similar areas of interest are common. For example, many ontologies have been created for bioinformatics, and a researcher building an application that utilizes ontologies about genes might use an ontology search engine [e.g. Swoogle (Finin et al., 2005)] or an ontology library (e.g. Protégé Ontologies Library [Protégé]) find an ontology that best fit his research (e.g. MGED [MGED], GO [GO], OBO [OBO]) but will often find several ontologies that cover genes and it will be difficult for the user to simply glance through the resulting ontologies to find the most suitable ontology. In this and similar domains, a tool that would provide an insight into the ontology and describe its features in a way that will allow such a researcher to make a well-informed decision on the ontology that best fits her needs is needed (Fig. 5.2).

The OntoQA (Tartir et al., 2005) technique we present in Section 5.4 for ontology analysis and evaluation was developed after facing some of the issues presented above in the continuing process of building SWETO (the Semantic Web Technology

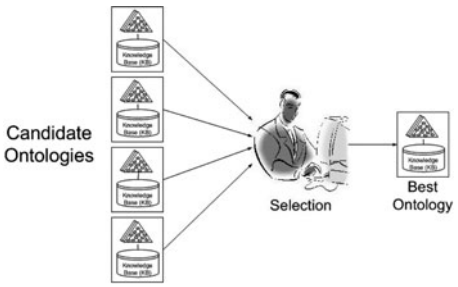

Fig. 5.2 Selecting the "best" ontology

Evaluation Ontology; Aleman-Meza et al., 2004). SWETO is a large-scale general purpose ontology that was populated with hundreds of thousands of instances that were mostly automatically extracted from different resources. This population process introduced a few problems. For example, SWETO includes some knowledge about geographical entities, like countries, states, and cities, and it was noticed that due to the nature of the sources instances were extracted from, that most of relationships extract were instances of the "located_in" relationship, or that most of the instances were about authors and publications. These and similar problems will prevent, for example, the discovery of interesting and useful relationships that connect people by their presence at the same location at the same time. Such problems would lower the efficiency and usefulness of some of the Semantic Web techniques such as the Semantic Association search mentioned above.

The rest of this chapter is organized as follows: Section 5.2 illustrates the need of ontology evaluation and validation. Section 5.3 introduces the current approaches in ontology evaluation and validation. Section 5.4 describes the OntoQA technique for ontology quality evaluation. Finally, Section 5.5 draws some conclusions and future recommendations.

5.2 Current Approaches in Ontology Evaluation and Validation

The increasing interest in the Semantic Web in recent years resulted in creating a large number of ontologies, and in increasing the amount of research on techniques to evaluate ontology quality and validity. With this growth in the number of ontologies, there have been some attempts to study the different approaches and tools for ontology evaluation and validation (Hartmann et al., 2004). Below is a description of the major current approaches currently in use for the evaluation and validation of ontologies.

5.2.1 Evolution-Based

This approach tracks an important characteristic of ontologies, change over time. Ontologies change over time by nature. More knowledge is always added to

domains, and it needs to be properly added to ontologies that model these domains. This approach tracks changes in the same ontology across different versions to get an indication of the quality of the ontology, and to detect (and possibly recover) any invalid changes made to the ontology. Ontologies change over time (evolve) due to three causes as proposed in Noy and Klein, (2004):

1. Changes in the domain,
2. Changes in conceptualization,
3. Changes in the explicit specification.

Changes in the domain are the most common, and are caused by change or addition in knowledge in the domains the ontology is modeling. For example, the more information about the genetic structure of a certain species are discovered, they need to be added to the ontology that models it.

Changes in conceptualization can result from a changing view of the world and from a change in usage perspective. Different tasks may imply different views on the domain and consequently a different conceptualization. For example, the view of a university from a faculty perspective is much different than the view from a student perspective, and the adoption of a certain perspective may result in a change of the ontology.

Changes in the explicit specification occur when an ontology is translated from one knowledge representation language to another. The languages differ not only in their syntax, but also (and more importantly) in their semantics and expressivity. Therefore, preserving the semantics of an ontology during translation is a non-trivial task.

An example of this approach is the technique presented in Plessers and De Troyer (2005). In this technique, when a change is needed on the ontology, a request is first added to the change log using CDL (Change Definition Language). Then, the change is implemented in the ontology. The technique finally matches the actual change with the change request from the log, if they are the same, the change is considered valid and it is propagated.

The technique in Haase et al. (2005) detects the two types of inconsistencies in evolving ontologies (user-defined and language-based) and repairs inconsistencies in ontologies across the different versions of the ontology by eliminating the statements that cause inconsistency.

5.2.2 Logical (Rule-Based)

Logical and rule-based approaches to ontology validation and quality evaluation use rules which are built in the ontology languages and rules users provided to detect conflicts in ontologies. Examples of first type are when two objects in an OWL ontology are said to be different from each other (owl:differentFrom), the ontology can't say that they are the same thing (owl:sameAs), or when two classes are said to

be disjoint of each other (owl:disjointWith) and the ontology can not have statements that mention an instances as being a member to both classes. Users can also identify properties that are considered in conflict in the domain. For example, a user can define that property motherOf conflicts with property marriedTo.

Several applications have adopted this approach. In Arpinar et al. (2006), a rule-based technique to conflict detection in ontologies is introduced. In this approach users identify conflicting rules using RuleML Boley et al. (2001) and the application will then list any cases were these rules are violated.

Authors of Parsia et al. (2005) use a logic model they call Swoop to detect unsatisfiable concepts in OWL ontologies. The technique is intended to be used by ontology designers to evaluate the quality of their work and to indicate any possible problems.

5.2.3 Metric-Based (Feature-Based)

Metric-based techniques to evaluate ontologies offer a quantitative perspective of ontology quality. These techniques scan through the ontology to gather different types of statistics about the knowledge presented in the ontology, or ask the user to input some information that is not included in the ontology itself. These techniques might consider classes' locations in the ontology schema graph as an indication of the type of knowledge the ontology focuses on. Some techniques also consider the instances of populated ontology in the measurement of quality metrics. The distribution of instances on the classes of the schema might also give an indication on the quality of the ontology.

Several techniques have adopted this approach. The authors of Lozano-Tello and Gomez-Perez (2004) propose a hierarchical framework they call OntoMetric that consists of 160 characteristics spread across five dimensions to evaluation the quality and suitability of ontologies to users' system requirements. The dimensions defined are: content of the ontology, language, development methodology, building tools, and usage costs. Users of OntoMetric will have the major task of supplying the application with several values that will be used to measure the suitability of an ontology for the given system requirements.

In Supekar et al. (2004) the authors propose a model for evaluating ontology schemas. The model contains two sets of features: quantifiable and non-quantifiable. Their technique is based on crawling the web to search for ontologies and store them locally, and then use information provided by the user, like the domain and weights for their proposed metrics to return the most suitable ontology.

Alani et al. (2006) presents a technique called AKTiveRank that finds a set of related ontologies to a set of terms the user enters. It uses an aggregation of the values of the four measures AKTiveRank includes to evaluation ontology schemas to select one of the ontologies to be the most suitable. The measures they developed are: class match, density, semantic similarity, and betweenness.

Corcho et al. (2004) introduce the ODEval tool that can be used for the automatic detection of possible syntactical problems in ontologies, such as the existence of

cycles in the inheritance tree of the ontology classes, inconsistency, incompleteness, and redundancy of classes and instances.

Mostowfi and Fatouhi (2006) define eight features they use to measure the quality of ontologies. These features are used to define a set of transformations to improve the quality of ontologies. For example, the authors suggest if a class (Student) has a property (Salary) that does not always have values (because it only holds for student assistants), then the class needs to be split into two: Student and Student Assistant. Other transformations attempt to make changes in properties or data types to make the ontology more consistent.

Another example technique is oQual (Gangemi et al., 2006), which evaluates ontologies on three dimensions: Structural: which uses a set of 32 features to study the syntax and formal semantics of the ontology. Functional: which uses a set of five qualitative measures to study the relationship between the ontology and its intended meaning. And finally, Usability profiling: which focuses on the communication (annotation) context of the ontology.

OntoClean (Guarino and Welty, 2004) also follows a feature-based approach to ontology evaluation and validation. A user of this technique would assign a set of four features to each of class in the ontology (Rigidity, Identity, Unity, and Dependence) and then use these features to identify problematic areas that needs to be reexamined. Based on these four features, classes might move up or down the taxonomy, and new classes might be added or removed to correct problems discovered through the detection of violations of a set of rules built using the four features.

The OntoQA framework we introduced in the abstract is one of the metric based approaches as well. In OntoQA we define the quality of a populated ontology as a set of five schema quality features and nine knowledgebase (or instance-base) quality features. An overview of OntoQA is presented in the next section.

Table 5.1 below provides a summary of the techniques mentioned above. The table compares the techniques on whether they target developers or end-users, whether users have to provide information to the technique (which might affect the training needed to be able to use the technique), whether it targets the schema or both the schema and the knowledgebase (KB), and whether users have to provide the ontologies or the application would crawl the internet for candidates.

It can be seen that among the techniques studied, most of them:

- Only work with schemas: This might miss problems and ignore knowledge available in the KB of a populated ontology.
- Require the user to provide the ontology: This might be problematic for a novice end-user who is not aware of ontologies available for his domain.
- Target developers (rather than end-users): Although evaluation and validation are important during the development process, it is important to provide end-users with tools they can use to select an error-free ontology that best fits their applications.
- Are feature-based: this is possibly due to the fact that a combination of metrics can provide insights about an ontology from different perspectives leading to a better understanding of the nature of the ontology.

Table 5.1 Comparison of different ontology evaluation techniques

Technique	Approach	Users	Automatic/ manual	Schema/KB	Ontology
Plessers and De Troyer (2005)	Evolution	Developers	Manual	Schema	Entered
Haase et al. (2005)	Evolution	Developers	Manual	Schema	Entered
Arpinar et al. (2006)	Logical	Developers	Manual	Schema + KB	Entered
Swoop	Logical	Developers	Automatic	Schema	Entered
OntoMetric	Metric	Developers	Manual	Schema	Entered
Supekar et al. (2004)	Metric	D + E	Automatic	Schema	Crawled
AKTiveRank	Metric	D + E	Automatic	Schema	Crawled
Mostowfi and Fatouhi (2006)	Metric	Developers	Automatic	Schema	Entered
oQual	Metric	D + E	Manual	Schema	Entered
OntoClean	Metric	Developers	Manual	Schema	Entered
OntoQA	Metric	D + E	Automatic	Schema + KB	Entered

Several researchers have studied the current approaches for ontology evaluation and validation. For example, Gómez-Pérez and Suarez-Figueroa (2003) compared several DAML/OIL and RDF(S) ontology checkers, validators, parsers and platforms (e.g. OilEd, OntoEdit, etc) and showed how most of the current tools were unable to find errors in ontologies. The authors also compared the tools with respect to three major problematic aspects: inconsistency, incompleteness, and redundancy. They concluded that tools that detect these errors are important for ontologies to be used more often.

5.3 OntoQA: Metric-Based Ontology Quality Analysis

In this section we describe OntoQA, our ontology evaluation tool. As mentioned in the previous section, OntoQA is a feature-based method for the evaluating ontologies (Fig. 5.3). OntoQA's main characteristic that distinguishes it from other

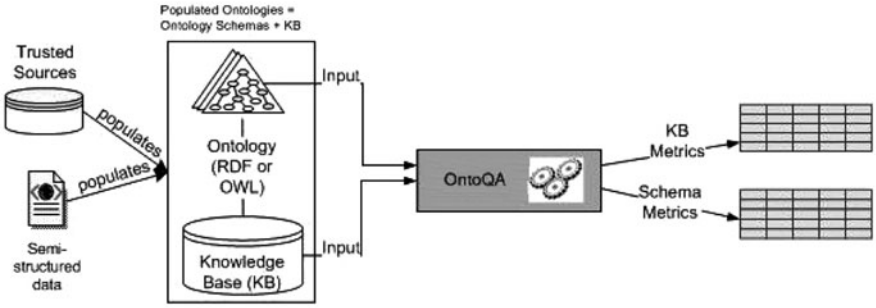

Fig. 5.3 OntoQA architecture

ontology quality tools is that it works on populated ontologies, thus enabling it from utilizing knowledge represented in the instances to gain a better measure of the quality of the ontology. OntoQA also uses much simpler techniques compared to others in that it doesn't require a lot of training as user involvement is minimal. In OntoQA, metrics (features) are divided into two groups: schema metrics that address the design of the ontology schema and instance metrics that address the way instances are organized within the ontology.

Metrics proposed in OntoQA describe certain aspects of the ontology rather than describing an ontology as merely "effective or ineffective" or "good or bad", because, in most cases, the way the ontology is built is largely dependent on the domain in which it is designed. For example, ontologies modeling human activities (e.g., travel or terrorism) will have distinctly different characteristics from those modeling the natural (or physical) world (e.g. genes or complex carbohydrates).

We divided the metrics into two related categories: schema metrics and knowledgebase (instance) metrics. The first category evaluates ontology design and its potential for rich knowledge representation. The second category evaluates the placement of instance data within the ontology and the effective utilization of the knowledge modeled in the schema. Below is a description of both categories of metrics.

5.3.1 Schema Metrics

Schema metrics address the design of the ontology. Although we cannot definitely know if the ontology design correctly models the domain knowledge, metrics in this category indicate the richness, width, depth, and inheritance of an ontology schema design. The most significant metrics in this category are described next.

5.3.1.1 Relationship Richness

This metric reflects the diversity of the types of relations in the ontology. An ontology that contains only inheritance relationships usually conveys less information than an ontology that contains a diverse set of relationships. The relationship richness is represented as the percentage of the (non-inheritance) relationships between classes compared to all of the possible connections that can include inheritance and non-inheritance relationships.

Definition 1: The relationship richness (*RR*) of a schema is defined as the ratio of the number of (non-inheritance) relationships (*P*), divided by the total number of relationships defined in the schema (the sum of the number of inheritance relationships (*H*) and non-inheritance relationships (*P*)).

$$RR = \frac{|P|}{|H| + |P|}$$

5.3.1.2 Inheritance Richness

Inheritance Richness (IR) measure describes the distribution of information across different levels of the ontology's inheritance tree or the fan-out of parent classes. This is a good indication of how well knowledge is grouped into different categories and subcategories in the ontology. This measure can distinguish a horizontal ontology (where classes have a large number of direct subclasses) from a vertical ontology (where classes have a small number of direct subclasses). An ontology with a low inheritance richness would be of a deep (or vertical) ontology, which indicates that the ontology covers a specific domain in a detailed manner, while an ontology with a high IR would be a shallow (or horizontal) ontology, which indicates that the ontology represents a wide range of general knowledge with a low level of detail.

Definition 2: The inheritance richness of the schema (*IR*) is defined as the average number of subclasses per class.

$$IR = \frac{|H|}{|C|}$$

5.3.1.3 Attribute Richness

The number of attributes (slots) that are defined for each class can indicate both the quality of ontology design and the amount of information pertaining to instance data. In general we assume that the more slots that are defined the more knowledge the ontology conveys.

Definition 3: The attribute richness (*AR*) is defined as the average number of attributes (slots) per class. It is computed as the number attributes for all classes (att) divided by the number of classes (*C*).

$$AR = \frac{|\text{att}|}{|C|}$$

5.3.2 Knowledgebase Metrics

The way data is placed within an ontology is also a very important measure of ontology quality because it can indicate the effectiveness of the ontology design and the amount of real-world knowledge represented by the ontology. Instance metrics include metrics that describe the KB (Knowledgebase) as a whole, and metrics that describe the way each schema class is being utilized in the KB.

5.3.2.1 Class Richness

This metric is related to how instances are distributed across classes. The number of classes that have instances in the KB is compared with the total number of classes, giving a general idea of how well the KB utilizes the knowledge modeled by the schema classes. Thus, if the KB has a very low Class Richness, then the KB does not have data that exemplifies all the class knowledge that exists in the schema. On the other hand, a KB that has a very high *CR* would indicate that the data in the KB represents most of the knowledge in the schema.

Definition 4: The class richness (*CR*) of a KB is defined as the percentage of the number of non-empty classes (classes with instances) (C') divided by the total number of classes defined in the ontology schema (C).

$$CR = \frac{|C'|}{|C|}$$

5.3.2.2 Class Connectivity

This metric is intended to give an indication of what classes are central in the ontology based on the instance relationship graph (where nodes represent instances and edges represent the relationships between them). This measure works in tandem with the importance metric mentioned next to create a better understanding of how focal some classes function. This measure can be used to understand the nature of the ontology by indicating which classes play a central role compared to other classes.

Definition 5: The connectivity of a class ($\text{Conn}(C_i)$) is defined as the total number of relationships instances of the class have with instances of other classes (NIREL).

$$\text{Conn}(C_i) = |\text{NIREL}(C_i)|$$

5.3.2.3 Class Importance

This metrics calculates the percentage of instances that belong to classes at the inheritance subtree rooted at the current class with respect to the total number of instances. This metric is important in that it will help in identifying which areas of the schema are in focus when the instances are added to the KB. Although this measure doesn't consider the domain characteristics, it can still be used to give an idea on what parts of the ontology are considered focal and what parts are on the edges.

Definition 6: The importance of a class ($\text{Imp}(C_i)$) is defined as the percentage of the number of instances that belong to the inheritance subtree rooted at C_i in the KB ($\text{inst}(C_i)$) compared to the total number of class instances in the KB (*CI*).

$$\text{Imp}(C_i) = \frac{|\text{Inst}(C_i)|}{|\text{KB}(CI)|}$$

5.3.2.4 Cohesion

In a semantic association discovery, relationships between instances are traced to discover how two instances are related. If the instances have disconnections among themselves, this may hinder such a search. This metric can be used to indicate the existence of such cases where the KB has more than one connected component (one being the ideal situation where all instances are connected to each other), indicating areas that need more instances in order to enable instances from one connect component to connect to instances in other connected components.

Definition 7: The cohesion (*Coh*) of a KB is defined as the number of connected components (*CC*) of the graph representing the KB.

□

5.3.2.5 Relationship Richness

This is an important metric reflecting how much of the relationships defined for the class in the schema are actually being used at the instances level. This is another good indication of the utilization of the knowledge modeled in the schema.

Definition 8: The relationship richness (*RR*) of a class C_i is defined as the percentage of the number of relationships that are being used by instances I_i that belong to C_i ($P(I_i,I_j)$) compared to the number of relationships that are defined for C_i at the schema level ($P(C_i,C_j)$).

In addition to these eight metrics (Tartir et al., 2005), includes other metrics that evaluate the ontology on other design aspects.

□

5.3.3 OntoQA Results

Figures 5.4 and 5.5 and Table 5.2 below show the OntoQA results when it is run on the three ontologies: SWETO (a general-purpose ontology with a focus on scientific publications), TAP (Guha and McCool 2003) (a general-purpose ontology) and GlycO (Sheth et al., 2004) (an ontology for the field of glycomics). It can be seen how different each one is by looking at the classes most instances in the ontology's KB fall into.

Figure 5.4 shows the most important classes in each of the ontologies. From the figure, it can be clearly seen that classes related to publications are the dominant classes in SWETO. While, with the exception of the Musician class, TAP gives consistent importance to most of its classes covering the different domains

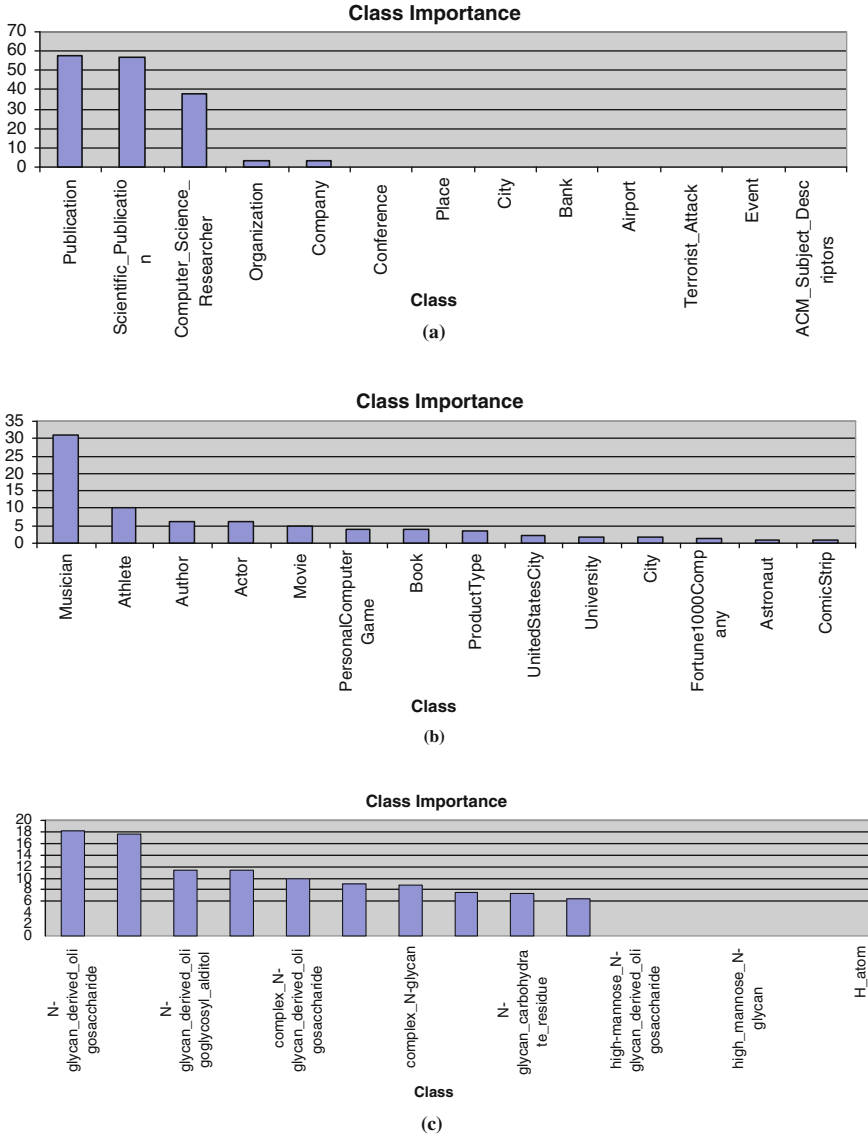

Fig. 5.4 Class importance in (**a**) SWETO (**b**) TAP and (**c**) GlycO

it includes. The nature of the GlycO ontology is reflected in the classes that are most important. The importance of the "N-glycan_residue" and the "alpha-D-mannopyranosyl_residue" and other classes show the narrow domain of GlycO is intended for, although the "glycan_moiety" class is the most important class covering about 90% of the instances in the KB.

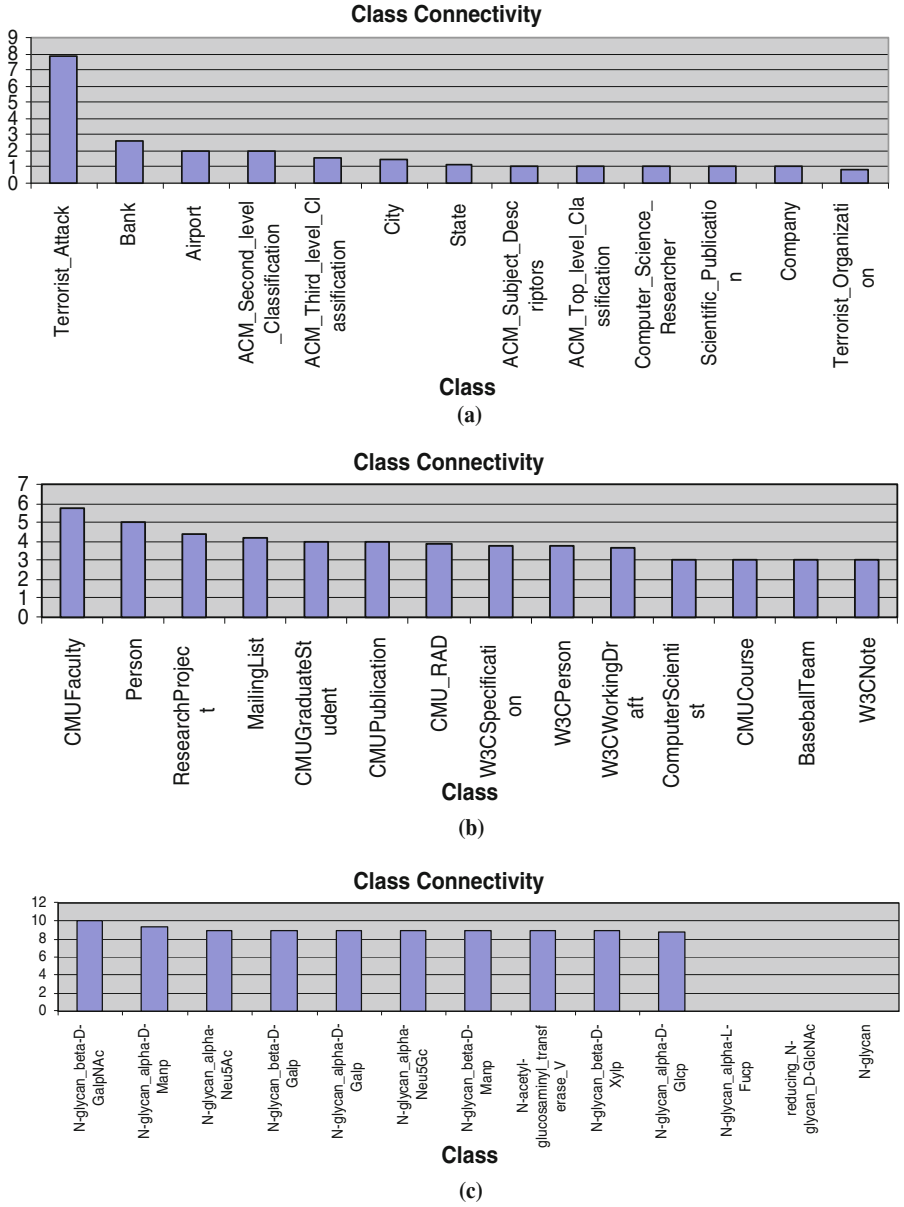

Fig. 5.5 Class connectivity in (**a**) SWETO (**b**) TAP and (**c**) GlycO

Figure 5.5 shows the most connected classes in the three ontologies. From the figure, it can be seen that SWETO also includes good information about domains other than publications, including the terrorism domain (Terrorist_Attack and Terrorist_Organization), the business domain (Bank and Company) and geographic

Table 5.2 Summary of SWETO, TAP, and GlycO

Ontology	Classes	Relations	Instances	Class richness
SWETO	44	101	813,217	59.1
TAP	6,959	25	85,637	0.24
GlycO	361	56	660	48.1

information (City and State). In a similar manner, TAP continues to show that it covers different domains, and its most connected classes cover the education domain (CMUCourse and CMUSCS_ResearchArea), the entertainment domain (TV and Movie), and other domains as well. GlycO's specific-purpose nature is evident from the Glycan related classes that are most connected.

Table 5.2 shows the differences between the three ontologies by the number classes, relationships, and instances, and by their class richness metric, which indicate that more of SWETO's classes are populated with instances compared to TAP or GlycO, which may indicate that the instance population process was carried out to cover resources that reflect the diversity of knowledge in the schema.

5.4 Conclusion

Ontologies form the cornerstone of the Semantic Web, and as the Semantic Web gains acceptance of the different scientific domains, more ontologies will be created to capture and share the knowledge in these domains. With this comes the need of being able to evaluate and validate these ontologies to ensure that they correctly represent the domain knowledge, and to be able to select the ontology among different ontologies that best fits a certain application. In this chapter we have summarized the current major trends in evaluating and validating ontologies and given examples techniques of each trends. We also presented our work in OntoQA and shown how it can be used to evaluate the ontology across different dimensions to give accurate metrics describing the ontology.

Still, more work is needed in ontology evaluation and validation to have techniques that can help the user by searching for ontologies instead of requiring the user to provide one, and have more techniques that target end-users in addition to developers.

References

Alani, H., C. Brewster, and N. Shadbolt. 2006. Ranking ontologies with aktiverank. In Proceedings of the 5th International Semantic Web Conference, Athens, GA, 5–9 Nov 2006.

Aleman-Meza, B., C. Halaschek, A. Sheth, I.P. Arpinar, and G. Sannapareddy. 2004. SWETO: Large-scale semantic web test-bed. In Proceedings of the 16th Seke 2004: Workshop on Ontology in Action, Banff, AB, 21–24 June 2004, 490–493.

Anyanwu, K., and A. Sheth. 2003. ρ-Queries: Enabling querying for semantic associations on the semantic web. In Proceedings of the 12th International. WWW Conference, Hungary.

Arpinar, I.B., K. Giriloganathan, and B. Aleman-Meza. 2006. Ontology quality by detection of conflicts in metadata. In Proceedings of the 4th International EON Workshop, Edinburgh, 22 May 2006. Edinburgh; International Conference Center.

Boley, H., S. Tabet, and G. Wagner. 2001. Design rationale of ruleml: A markup language for semantic web rules. In Proceeding of the 1st Semantic Web Working Symposium. Palo Alto, CA: Stanford University.

Corcho, O. et al. 2004. ODEval: A tool for evaluating RDF(S), DAML+OIL, and OWL concept taxonomies. In Proceedings of the 1st IFIP Conference on Artificial Intelligence Applications and Innovations (AIAI 2004), Toulouse, France, 369–382.

Cristani, M., and R.A. Cuel. 2005. Survey on ontology creation methodologies. *International Journal of Semantic Web and Information Systems* (*IJSWIS*) 1(2):49–69.

Fernández, M., A. Gómez-Pérez, J. Pazos, and A. Pazos. 1999. Building a chemical ontology using methontology and the ontology design environment. *IEEE Intelligent Systems Applications*. 4(1):37–45.

Finin, T., et al. 2005. Swoogle: Searching for knowledge on the semantic web. In Proceedings of the 20th National Conference on Artificial Intelligence (AAAI 05), Pittsburg, PA.

Gangemi, A., C. Catenacci, M. Ciaramita, and J. Lehmann. 2006 Modelling ontology evaluation and validation. In Proceedings of the 2006 European Semantic Web Conference. Berlin: Springer.

The Gene Ontology. http://www.geneontology.org

Gómez-Pérez, A., and M. Rojas-Amaya. 1999. Ontological reengineering for reuse. In Proceedings of the 11th European Workshop on Knowledge Acquisition, Modeling and Management, Dagstuhl Castle, Germany.

Gómez-Pérez, A., and M.C. Suarez-Figueroa. 2003. Results of taxonomic evaluation of RDF(S) and DAML+OIL ontologies using RDF(S) and DAML+OIL validation tools and ontology platforms import services. In Proceedings of the 2nd International Evaluation of Ontology-Based Tools Workshop, 20th Oct 2003. Sanibel Island, FL: Sundial Resort.

Guarino, N. and C. Welty. 2004. An overview of ontoclean. *Handbook on ontologie*, eds. S. Staab, and R. Studer, 151–159. Berlin: Springer.

Guha, R., and R. McCool. 2003. TAP: A semantic web test-bed. *Journal of Web Semantics* 1(1): 81–87.

Haase, P., F. van Harmelen, Z. Huang, H. Stuckenschmidt, and Y.A. Sure. 2005. Framework for handling inconsistency in changing ontologies. In Proceedings of ISWC2005, Galway, Ireland.

Hartmann, J., P. Spyns, A. Giboin, D. Maynard, R. Cuel, M. Carmen Suárez-Figueroa, and Y. Sure. 2004. Methods for ontology evaluation. *Knowledge Web Deliverable*, D1.2.3, v. 0.1.

Lozano-Tello, A., and A. Gomez-Perez. 2004. ONTOMETRIC: A method to choose the appropriate ontology. *Journal of Database Management* 15:1–18.

MGED. The MGED Ontology. http://mged.sourceforge.net/ontologies/MO_FAQ.htm

Mostowfi, F., and F. Fotouhi. 2006. Improving quality of ontology: An ontology transformation approach. In Proceedings of the 22nd International Conference on Data Engineering Workshops (ICDEW'06), Atlanta, GA.

Noy, N.F., and M. Klein. 2004. Ontology evolution: Not the same as schema evolution. In *Knowledge and information systems.*

Open Biomedical Ontologies. http://obo.sourceforge.net

Parsia, B., E. Sirin, and A. Kalyanpur. 2005. Debugging OWL ontologies. In Proceedings of WWW 2005, 10–14 May 2005, Chiba, Japan.

Paslaru, E., B. Simperl, C. Tempich, and Y. Sure. 2006. ONTOCOM: A cost estimation model for ontology engineering. In Proceedings of 5th International Semantic Web Conference (ISWC 2006), Athens, GA.

Plessers, P., and O. De Troyer. 2005. Ontology change detection using a version log. In Proceedings of the 4th International Semantic Web Conference (ISWC-05).

Protégé Ontologies Library. http://protege.cim3.net/cgi-bin/wiki.pl?ProtegeOntologiesLibrary.
Sabou, M., V. Lopez, E. Motta, and V. Uren. 2005. Ontology selection: Ontology evaluation on the real semantic web. In Proceedings of the 15th International World Wide Web Conference 2005, Edinburgh.
Sheth. A., et al. 2004. Semantic web technology in support of bioinformatics for glycan expression. W3C Workshop on Semantic Web for Life Sciences, 27–28 Oct 2004, Cambridge, MA.
Supekar, K., C. Patel, and Y. Lee. 2004. Characterizing quality of knowledge on semantic web. In Proceedings of AAAI Florida AI Research Symposium (FLAIRS-2004), 17–19 May 2004, Miami Beach, FL.
Tartir, S., I.B. Arpinar, M. Moore, A.P. Sheth, and B. Aleman-Meza. 2005. OntoQA: Metric-based ontology quality analysis. In IEEE ICDM 2005 Workshop on Knowledge Acquisition from Distributed, Autonomous, Semantically Heterogeneous Data and Knowledge Sources, 27 Nov 2005, Houston, TX.