OQuaRE: A SQuaRE-based Approach for Evaluating the Quality of Ontologies

Astrid Duque-Ramos and Jesualdo Tomás Fernández-Breis

Departamento de Informática y Sistemas, Universidad de Murcia. 30071 Campus de Espinardo, Murcia, Spain
{astrid.duque, jfernand}@um.es

Robert Stevens

School of Computer Science, University of Manchester, Oxford Road, Manchester United Kingdom, M13 9PL
robert.stevens@manchester.ac.uk

Nathalie Aussenac-Gilles

Institute de Recherche en Informatique de Toulouse, Université Paul Sabatier, 118 Route de Narbonne, F-31062, Toulouse, France
aussenac@irit.fr

The development of the Semantic Web has provoked an increasing interest in the development of ontologies. There are, however, few mechanisms for guiding users in making informed decisions on which ontology to use under given circumstances. In this paper, we propose a framework for evaluating the quality of ontologies based on the SQuaRE standard for software quality evaluation. This method requires the definition of both a quality model and quality metrics for evaluating the quality of the ontology. The quality model is divided into a series of quality dimensions or characteristics, such as structure or functional adequacy, which are organized into subcharacteristics, such as cohesion or tangledness. Thus, each subcharacteristic is evaluated by applying a series of quality metrics, which are automatically measured. Finally, each characteristic is evaluated by combining values of its subcharacteristics. This work also includes the application of this framework for the evaluation of ontologies in two application domains.

Keywords: Semantic Web, Ontology Quality, Software Quality, ISO 25000

ACM Computing Classification System: D.2.13 Reusable Software, I.2.4 Knowledge Representation Formalisms and Methods

1. INTRODUCTION

The use of Semantic Web technologies is growing and this has provoked an increasing interest in the development of ontologies, including medicine (Stearns, Price, Spackman and Wang, 2000), mathematics (Gruber and Olsen, 1994), information security (Vorobiev and Bekmamedova, 2010), agents (Guan and Zhu, 2004), etc.. Different communities are producing different ontologies for similar purposes. As a result of this, users and developers have to decide which ontology to use for their particular purposes. Given the different backgrounds of the ontology builders, ontologies with

Manuscript received: 16 April 2011
Communicating Editor: Rafael Valencia García

different structure and content are produced, making it harder to choose for users and developers alike. Moreover, users and developers lack support for making an informed decision. Hence, there is a clear need for mechanisms and methods for evaluating the quality of the available ontologies.

There has been a great interest in applying Software Engineering (SE) methods to Knowledge Engineering (KE) and vice versa. There is an increasing convergence in terms of methodology, language, and tool support between traditional SE and KE. Traditionally, SE has produced results that have improved KE due to the higher maturity of the field. In the Semantic Web in general, and Ontology Engineering in particular, SE-based methods have been applied to improve the methods used for KE. One example is Methontology (López, Gómez-Pérez, Sierra and Sierra, 1999), whose development process is based on the activities identified in the IEEE standard for software development (IEEE, 1997). Moreover, the Reverse Engineering approach presented in Chikofsky and Cross (1990) was adapted to Ontology Reengineering in Gómez-Perez and Rojas-Amaya (1999). More recently, the development of ontologies has been addressed from a Model Driven Architecture perspective (Gasevic, Djuric, Devedzic and Selic, 2006; Gasevic, Kaviani and Milanovic, 2008). Besides, software design patterns have been used by the SE community for the last twenty years (Beck and Cunningham, 1987), and the same principle has been applied in KE in the form of Ontology Design Patterns (ODPs) (Gangemi and Presutti, 2007; Egaña, Antezana, Kuiper, and Stevens, 2008). Another example of SE techniques applied to KE is ontology performance (Wang and Parsia, 2007). On the other hand, some KE methods have been applied to improve SE processes (see for instance Lasheras, Valencia-Garcia, Fernandez-Breis, and Toval, 2009; Blanco, Fernandez-Medina, Valencia-Garcia and Toval, 2011). We have, however, not found any attempt to adapt SE approaches for evaluating ontology quality.

1.1 Related Work in Ontology Evaluation

Related work in ontology evaluation can be classified according to the particular evaluation aim: ranking, correctness, or quality. The ranking category includes approaches for ranking and selecting ontologies. These approaches range from generic ontology rankings to selection of the most appropriate ontology for a particular task.

Ontometric (Lozano-Tello and Gómez-Pérez, 2004) is an attempt to perform a systematic ontology selection, but it is considered to have limited usability due to its complexity. Its aim is to suggest the best ontology for a particular project, so ontologies are valued on the basis of 160 properties organised in five dimensions: content, language, methodology, tool and costs. In Netzer, Gabay, Adler, Goldberg, and Elhadad (2009), a method based in a corpus to evaluate the functional adequacy of the ontology is provided. On the other hand, an ontology selection and ranking model consisting of selection standards (semantic similarity, topic coverage, and richness) and metrics based on better semantic matching capabilities is proposed in Park, Oh, and Ahn (2011).

The correctness category includes the approaches accounting for the formal correctness of the ontological knowledge and primitives used. In this category, the most relevant approach is Ontoclean (Guarino and Welty, 2004), which checks for the correctness of the taxonomy, based on principles of rigidity, identity, unity and dependence. In Völker, Vrandecic and Sure (2005), a tool for evaluating real-world ontologies based on Ontoclean is presented. The framework presented in Corcho, Gómez-Pérez, González-Cabero and Suárez-Figueroa (2004) looks for taxonomic aspects such as circularity and redundancy, as well as errors in disjoint groups. Correctness is also evaluated in Wang (2006), where an internal evaluation is performed, based on the correct usage of OWL primitives.

The final category addresses the evaluation of the global quality of an ontology. In Vrandecic (2010), the author presents a theoretical framework for assessing the quality of an ontology for the Web. The framework summarizes ontology evaluation methods in two dimensions: ontology quality criteria (accuracy, adaptability, clarity, completeness, computational efficiency, conciseness, consistency, and organizational fitness) and ontology aspects (vocabulary, syntax, structure, semantics, representation, and context). To assess the quality of evolving ontologies, Ma, Ma, Liu and Jin (2009) propose a set of cohesion metrics, which are considered stable. This means that they consider that the further addition of new axioms for an ontology and the results of the metrics do not depend on the semantic or structural ontology representation. In Gangemi, Catenacci, Ciaramita, and Lehmann (2006), ontology evaluation is approached as a diagnostic task based on ontology descriptions, using three categories of criteria: structural (depth, breadth, tangledness, dispersion, consistency, anonymous classes, cycles, and density), functional (competence adequacy, functional modularity, precision, recall and accuracy), and usability profiling (documentation, efficiency, interfacing). By combining the different measurable criteria for each category, nine quality principles are defined. Not all the criteria can always be optimized simultaneously, so preference and trade-off criteria can be used, and this makes its application complex. Rogers (2006) proposes four qualitative criteria: philosophical rigour, ontological commitment, content correctness, and fitness for purpose.

In summary, ontology evaluation has to date been a disperse field that includes ontology ranking, ontology selection, ontology comparison, and ontology quality issues. These subproblems have been addressed in different heterogeneous ways, none having become standard. Thus, the need for standardised methods for evaluating the quality of an ontology remains unfulfilled.

1.2 Why Adapting a Software Engineering Standard for Ontology Evaluation?

We approach ontology evaluation as a tool for "*helping developers to evaluate their ontologies in order to build trust for sharing and reusing ontologies*", which is one of the main objectives of ontology evaluation according to Brank, Grobelnik, and Mladenic (2005). In terms of discipline maturity, the Software Engineering discipline is more advanced than Ontology Engineering, due to a longer existence. In Software Engineering, software quality measures the quality of software design, and to which extent the software conforms to that design. The application of a software quality standard is recommended because: (1) it provides a comprehensive specification and evaluation model for software product quality; (2) it addresses user needs of a product by allowing for a common language for specifying user requirements that is understandable by users, developers and evaluators; (3) it objectively evaluates the quality of software products based on observation, not opinion; and (4) it makes quality evaluation reproducible. All these properties are desirable for an ontology quality evaluation approach, and hence they represent a potentially useful tool to define such framework.

This would, however, only make sense if we can consider ontologies as software artefacts. In this work, ontologies are viewed as the result of the application of a construction process, and they are evaluated as products, independently of the particular development process. Ontology evaluation is here then approached from a double perspective: (1) software artefact; and (2) tool. On the one hand, an ontology is a software artefact, and it can be evaluated as a part of an information system. On the other hand, an ontology is also a tool that has to be useful. Hence, metrics for measuring the technical quality of the software artefact and metrics for the usefulness are needed. In summary, according to our understanding, the adaptation and application of Software Quality standards to ontology evaluation seems sensible. Our recent efforts have been put on investigating how such software quality standards can be adapted to measure ontology quality. In Fernandez-Breis, Egaña, and Stevens (2009), we presented an approach based on the ISO 9126

(ISO9126, 2001), which is an international standard for software product evaluation which was applied to two versions of the same biomedical ontology. The partial results obtained in that work showed that this evaluation process had the problem of the excessive workload and dependence on human judgement, since the method did not provide computer-support to the evaluator. Given these points, we believe that automatic support could be of great use to formal ontology evaluation processes, helping to make the evaluation of ontologies a proper engineering task, obtaining a more objective evaluation ontology and helping humans to perform evaluation tasks more easily.

In this work, we extend the number of methods applied for the evaluation of the quality of the ontologies, which are no longer manually calculated, but obtained by a home-made software tool. In this paper, the ISO 9126 will not be used but the ISO/IEC 25000:2005 (ISO25000, 2005), which is the standard for Software product Quality Requirements and Evaluation known as SQuaRE, that defines a complete evaluation process for a software product. One of the components of SQuaRE is the ISO 9126 although the quality model is extended with new quality characteristics. Our quality approach, called OQuaRE, is based on defining a series of quality characteristics for ontologies according to SQuaRE. The quality score for a particular characteristic will depend on the scores obtained for its quality subcharacteristics and the metrics calculated for the ontology. The novelty of the approach is not given by the definition of a series of new metrics, but in the adaptation of a quality standard, the standardization of the evaluation process, and the way the metrics are associated with the quality characteristics. Moreover, most ontology evaluation approaches have been designed for identifying the best ontology, but the main goal of our method is the identification of strengths and flaws of the ontologies, so the users and developers can make informed decisions according to their needs.

2. THE ONTOLOGY QUALITY EVALUATION FRAMEWORK (OQuaRE)

The ISO/IEC 25000:2005 is the standard for Software product Quality Requirements and Evaluation known as SQuaRE that defines a complete evaluation process of a software product. It combines ISO 9126 and ISO 14598 (ISO14598, 1999). SQuaRE covers two main processes: software quality requirements specifications and software quality evaluation. SQuaRE defines all the elements required for the evaluation of software products: evaluation support, evaluation process and metrics. Besides, the usage of SQuaRE requires the definition of the following components: quality model, quality metrics, quality requirements and quality evaluation. SQuaRE permits the definition of the quality model in terms of quality characteristics. In this way, this standard suggests a series of quality characteristics that should be used for measuring quality: functional adequacy, reliability, operability, maintainability, compatibility, and transferability. In addition to this, each quality characteristic has a set of quality subcharacteristics associated, which are measured through quality metrics. The quality metrics are the units of measurement of quality evaluation.

Our adaptation to the evaluation of ontologies is OQuaRE, which aims to define all the elements required for ontology evaluation: evaluation support, evaluation process and metrics. This adaptation is possible because, as described in the introduction section, ontologies are considered in this work as software artefacts. The current version of our framework includes, so far, the quality model and the quality metrics, because they are the basic modules for the technical evaluation of the quality of the ontology, and which are explained next.

2.1 The OQuaRE Quality Model

This model reuses and adapts the following SQuaRE characteristics to ontologies: reliability, operability, maintainability, compatibility, transferability and functional adequacy. This adaptation

process required analysing whether such quality characteristics were applicable to ontologies. This is reasonable as we have argued above that ontologies are software artefacts. Moreover, according to requirements, principles and characteristics of ontologies and to the state of the art of ontology evaluation, structural features of ontologies are important to evaluate their quality, but they are not considered in the standard. Consequently, we added such characteristic to our framework. In order to determine the quality subcharacteristics, we combined the ones suggested in SQuaRE with the ones suggested by state-of-the-art methods from the ontology evaluation community. In this way, we use structural subcharacteristics like cohesion, consistency or formal relations support, and the functional adequacy subcharacteristics are the intended uses for ontologies identified in Stevens and Lord (2009). The complete description of the categories is available at http://miuras.inf.um.es/evaluation/oquare.

Characteristics

- **Structural:** Formal and semantic important ontological properties that are widely used in state-of-the-art evaluation approaches. Some subcharacteristics are formalisation, formal relations support, cohesion, tangledness, redundancy and consistency.
- **Functional adequacy:** An ontology is evaluated for this criterion according to the degree of accomplishment of functional requirements, that is, the appropriateness for its intended purpose according to Stevens and Lord (2009): reference ontology, controlled vocabulary, schema and value reconciliation, consistent search and query, knowledge acquisition, clustering and similarity, indexing and linking, results representation, classifying instances, text analysis, guidance and decision trees, knowledge reuse, inferencing, and precision.
- **Reliability:** Capability of an ontology to maintain its level of performance under stated conditions for a given period of time. Recoverability and availability are some of its subcharacteristics.
- **Operability:** Effort needed for using an ontology, and in the individual assessment of such use, by a stated or implied set of users, and it is measured through subcharacteristics such as learnability.
- **Maintainability:** The capability of ontologies to be modified for changes in environments, in requirements or in functional specifications. Some subcharacteristics are modularity, reusability, analysability, changeability, modification stability and testability.

The evaluation of a particular characteristic depends on the evaluation of its quality subcharacteristics (see some examples in Table 1).

2.2 The Ontology Quality Metrics

The Software Engineering community has developed and proposed many metrics applicable to software programs in prior decades. A compendium with 23 well-known software metrics from different authors and its association with software quality standards can be seen in Lincke and Lowe (2007). We studied the quantitative evaluation metrics defined in Software Engineering and, in particular, Object-oriented Programming (OOP). We selected a set of well known software metrics like Coupling Between Objects (CBO), Depth of Inheritance Tree (DIT), Number Of Children (NOC), Response For a Class (RFC), Weighted Method Count (WMC), (Chidamber and Kemerer, 1994) and Number Of local Methods (NOM) (Li and Henry, 1993). Despite ontologies and object-oriented design having different properties, there are a series of shared notions as the existence of classes, individuals and properties that can be exploited to adapt OOP metrics to ontologies. We also

Structural
Formalisation: An efficient ontology has to be built on top of a formal model to support reasoning.
Formal relations support: Most ontologies only have formal support for taxonomy. The usage of additional formal theories would be a positive indicator.
Cohesion: An ontology has a high cohesion if the classes are strongly related.
Tangledness: This measures the distribution of multiple parent categories, so that it is related to the existence of multiple inheritance, which is usually a sign of suboptimal design.
Functional adequacy
Schema and value reconciliation: An ontology can provide a common data model that can be applied to particular views for their reconciliation and integration. Ontologies facilitate the achievement of semantic interoperability if they are able to provide the semantic context for data and information.
Consistent search and query: The formal model of the ontology allows for better querying and searching methods. The ontology structure can guide search processes if they provide a semantic context to evaluate the data wanted by the users. This semantic context is not just provided by the concepts, but also by all the machine computable properties and axioms.
Knowledge reuse: Degree to which the knowledge of an ontology can be used to build other ontologies.
Knowledge acquisition: Ontologies are templates for generating the forms by which instances are acquired.
Maintainability
Modularity: The degree to which the ontology is composed of discrete components such that a change to one component has a minimal impact on other components. Changes in ontologies may lead to unexpected effects, like inconsistencies, in the ontology.
Reusability: The degree to which a part of the ontology can be reused in more than one ontology, or to build other ontologies.
Analysability: The degree to which the ontology can be diagnosed for deficiencies or causes of inconsistencies.

Table 1: Definition of some subcharacteristics

reused metrics developed by the ontology engineering community, especially for the structural properties from, for instance, Yao, Orme, and Etzkorn (2005) or Tartir and Arpinar (2007). Next, we define some metrics included in our approach. In such definitions, the following notation has been adopted:

$C_1; C_2; \ldots; C_n$: Classes of the ontology.
$R_{C1}; R_{C2}; \ldots; R_{Ck}$: Relationships of the class C_i.
$P_{C1}; P_{C2}; \ldots; P_{Cz}$: Properties of the class C_i.
$I_{C1}; I_{C2}; \ldots; I_{Cm}$: Individuals of the class C_i.
$Sup_{C1}; Sup_{C2}; \ldots; Sup_{Cm}$: Direct superclasses of a given class C.
Thing: Root class of the ontology.

We can associate these concepts with the OWL modeling primitives: classes refer to owl:Class, relationships refer to owl:ObjectProperty, properties refer to owl:DatatypeProperty and individuals refers to owl:Individual.

- Lack of Cohesion in Methods (LCOMOnto): The semantic and conceptual relatedness of classes can be used to measure the separation of responsibilities and independence of components of ontologies. It is calculated as follows:
  LCOMOnto=$\sum path(|C(leaf)_i|)/m$, where $path|C(leaf)_i|$ is the length of the path from the leaf *class i* to *Thing*, and *m* is the total number of paths in the ontology.
- Weighted Method Count (WMCOnto): Mean number of properties and relationships per class. It is calculated as follows:
  WMCOnto=$(\sum|P_{Ci}|+\sum| RC_i|) / \sum|C_i|$, where C_i is the *i-th class* in the ontology.
- Depth of subsumption hierarchy (DITOnto): Length of the largest path from Thing to a leaf class. It is calculated as follows:
  DITOnto=*Max* $(\sum D|C_i|)$, where C_i are the classes and $D|C_i|$ is the length of the path from the *i-th leaf class* of the ontology to *Thing*.
- Number of Ancestor Classes (NACOnto): Mean number of ancestor classes per leaf class. It is the number of direct superclasses per leaf class, and calculated as follows: ***NACOnto***=$\sum|Sup_{C(Leaf)i}|/\sum|C(leaf)_i)|$
- Number of Children (NOCOnto): Mean number of direct subclasses. It is the number of relationships divided by the number of classes minus the relationships of Thing, and calculated as follows:
  NOCOnto=$\sum| R_{Ci}| / (\sum|C_i| - | R_{Thing}|)$
- Coupling between Objects (CBOOnto): Number of related classes. It is the average number of the direct parents per class minus the relationships of Thing, and calculated as follows: ***CBOOnto***=$\sum|Sup_{Ci}|/(\sum|C_i| - | R_{Thing}|)$
- Response for a class (RFCOnto): Number of properties that can be directly accessed from the class. It is calculated as follows:
  RFCOnto=$(\sum|P_{Ci}|+\sum|Sup_{Ci}|/(\sum|_{Ci}| - | R_{Thing}|)$
- Number of properties (NOMOnto): Number of properties per class. It is calculated as follows: ***NOMOnto***=$\sum| P_{Ci}| / \sum|C_i|$
- Properties Richness (RROnto): Number of properties defined in the ontology divided by the number of relationships and properties. It is calculated as follows:
  RROnto=$\sum| P_{Ci}| / \sum(| R_{Ci}| + \sum|_{Ci}|)$
- Attribute Richness (AROnto): Mean number of attributes per class. It is calculated as follows: ***AROnto***=$\sum|Att_{Ci}| / \sum|C_i|$
- Relationships per class (INROnto): Mean number of relationships per class. It is calculated as follows: ***INROnto***=$\sum| R_{Ci}| / \sum|C_i|$
- Class Richness (CROnto): Mean number of instances per class. It is calculated as follows: ***CROnto***=$\sum| I_{Ci}| / \sum|C_i|$; where I_{Ci}, is the set of *individuals* of the C_i class.
- Annotation Richness (ANOnto): Mean number of annotations per class. It is calculated as follows: ***ANOnto***=$\sum| A_{Ci}| / \sum|C_i|$; where C_i is the *i-th* class in the ontology.
- Tangledness (TMOnto): Mean number of parents per class. It is calculated as follows:
  TMOnto=$\sum| R_{Ci}| / \sum|C_i|-\sum|C(DP)_i|$; where C_i is the *i-th* class in the ontology and $C(DP)_i$ is the *i-th* class in the ontology with more than one direct parent.

2.3 Associating Metrics with Subcharacteristics

As mentioned, the evaluation of an ontology for a particular characteristic depends on the evaluation of its set of associated subcharacteristics. Likewise, the evaluation of a particular subcharacteristic depends on its associated metrics. Consequently, knowing the association between metrics and subcharacteristics is fundamental to understanding the framework. For this purpose, the recommendations and best practices included in the Compendium of Software Quality Standards and Metrics (Lincke and Lowe, 2007) have been followed. Given that such a compendium does not include associations for functional adequacy and structural subcharacteristics, we provided our own metrics.

The associations for maintainability are shown in Table 2, whereas the complete association list can be found at http://miuras.inf.um.es/evaluation/oquare/. In such table, "+" means that a higher value of the metric contributes to a higher score for the subcharacteristic, and "-" means that a lower value of the metric contributes to a higher score for the subcharacteristic. It should be mentioned that one quality metric could be associated with several quality subcharacteristics. In that case, a particular indicator contributes to multiple quality properties. For instance, the metric "mean number of properties per class" contributes to knowledge acquisition (of functional adequacy) because the fact that an ontology has more properties usually means that this ontology is potentially more useful. The number of properties also impacts on reusability (of maintainability) because having a more precisely defined ontology makes its knowledge more reusable. Another example is WMCOnto (see Table 2), which is used for all the subcharacteristics of maintainability.

Once the quality characteristics, subcharacteristics and metrics have been presented, the focus moves to how the quantitative values of the metrics are turned into quantitative values of subcharacteristics and characteristics. First, it should be noted that the different metrics generate quantitative values in different ranges. Metrics such as LCOMOnto or WMCOnto produce an absolute value, whereas metrics such as RROnto and AROnto generate relative ones. Second, the SQuaRE scores of the quality characteristics and subcharacteristics are in the range 1 to 5, where "*1 means not acceptable, 3 is minimally acceptable, and 5 is exceeds the requirements"* (ISO25000, 2005). Consequently, in OQuaRE, a mapping between the range of values of the metrics and the range 1 to 5 was needed, and such mapping must take into account that high values in the metrics might not correspond to a high quality score. For this purpose, the recommendations and best practices of the Software Engineering community for software metrics and ontology evaluation metrics were applied. Consequently, the mapping shown in Table 3 was proposed. There, it can be

Metric / Subcharacteristic	WMCOnto	DITOnto	NOCOnto	RFCOnto	NOMOnto	LCOMOnto	CBOOnto
Modularity	+						+
Reusability	+	+	-	+	+		+
Analysability	+	+		+	+	+	+
Changeability	+	+	+	+	+	+	+
Modification stability	+		+	+		+	+
Testability	+	+		+	+	+	+

Table 2: Association between metrics and quality subcharacteristics for maintainability

Score / Metric	1	2	3	4	5
LCOMOnto	> 8	(6-8]	(4,6]	(2, 4]	<=2
WMCOnto	> 15	(11,15]	(8,11]	(5,8]	<=5
DITOnto	> 8	(6-8]	(4,6]	(2, 4]	[1,2]
NACOnto	> 8	(6-8]	(4,6]	(2, 4]	[1,2]
NOCOnto	> 12	(8-12]	(6,8]	(3,6]	[1,3]
CBOOnto	> 8	(6-8]	(4,6]	(2, 4]	[1,2]
RFCOnto	> 12	(8-12]	(6-8]	(3-6]	[1-3]
NOMOnto	> 8	(6-8]	(4,6]	(2, 4]	< =2
RROnto	[0,20]%	(20-40]%	(40-60]%	(60-80]%	> 80%
AROnto	[0,20]%	(20-40]%	(40-60]%	(60-80]%	> 80%
INROnto	[0,20]%	(20-40]%	(40-60]%	(60-80]%	> 80%
CROnto	[0,20]%	(20-40]%	(40-60]%	(60-80]%	> 80%
ANOnto	[0,20]%	(20-40]%	(40-60]%	(60-80]%	> 80%
TMOnto	> 8	(6-8]	(4,6]	(2, 4]	(1,2]

Table 3: Evaluation criteria for the measurement primitives

observed that, for those metrics that generate absolute values, mappings based on the meaning of the metrics were defined. On the other hand, for those primitives whose result is a relative value, one unit in the OQuaRE scale corresponds to 20%.

2.4 Calculating the Quality Scores

Once the scores are obtained for each quality metric, the process followed for calculating the quality scores for a particular ontology is described next. First, the scores for each metric are transformed into the range 1–5. Second, the score for each quality subcharacteristic is obtained. In our current framework, the score is the mean of all the metrics associated with a particular quality subcharacteristic. Third, the score for each quality characteristic is obtained, and the score is the mean of the scores of its quality subcharacteristics. The set of scores for each quality characteristic is the intended result for our framework, since we do not intend to rank ontologies but offer information about the strengths and weaknesses of the ontologies. However, in case of needing a global score for the ontology, that could be calculated by obtaining the weighted average of the scores of all the quality characteristics.

3. THE TWO CASE STUDIES

In this section, we describe the application of OQuaRE to the evaluation of two versions of the ontology of types of cells and to a set of measurement units ontologies.

3.1 The Process

We have run a double evaluation for the sets of ontologies. First, they were evaluated manually by a set of experts in ontologies. Second, they were evaluated by using a home-made software tool that

implements the quality framework described in the previous sections. This tool takes an OWL ontology as input and generates a numeric report as output. This report includes the scores for all the characteristics, subcharacteristics and metrics involved in the evaluation process. The process followed in both experiments was similar. The ontologies were processed with the tool and the scores were then obtained and analysed. This double evaluation will allow us to evaluate the results of the automatic evaluation.

3.2 Ontologies of Units of Measurement

There are more than one hundred ontologies of units of measurement, thus we did an initial selection based on the ontology language used and in the modeling and design style. Next, we describe the ontologies used in this study:

- Measurement Units Ontology (MUOVOCAB): this ontology defines the classes and properties providing the essential vocabulary to define the semantics of measurements in domain ontologies (for instance, in the delivery context ontology). http://idi.fundacionctic.org/muo/muo-vocab.html
- Unified Code for Units of Measure (UCUM): the main objective of this ontology is to coin URIs for the most common units of measure, physical qualities and prefixes that can be shared and reused. Every unit of measurement is linked to a physical quality. http://idi.fundacionctic.org/muo/ucum-instances.owl
- Gist Units of Measure Ontology (GISTUM) minimalist upper ontology, expressed in OWL, and designed primarily for business use, including units of measurement. http://www.gist-ont.com/
- SWEET 2.0 Scientific Units Ontology (SCIUNITS): ontology for scientific units that is suitable for most physical science applications. This ontology includes complex unit expressions defined by combining base unit concepts via terms from the mathematics ontology. http://sweet.jpl.nasa.gov/2.0/sciUnits.owl
- Quantities-Units-Dimensions-Values(QUDV_SI): this is an ontology for quantities, units, dimensions and values, and it is divided into two main parts. On the one hand, SysML-QUDV contains the basic ontology, i.e. it defines the classes, object properties and data properties. On the other hand, SysML-QUDV-SI contains 38 individuals that represent some quantities, units, scales and a prefix for some international classification systems. http://www.omgwiki.org/OMGSysML/doku.php?id= sysml-qudv:qudv_owl
- OpenMath (OPENMATH): semantics of mathematical objects, including units of measurement. http://www.openmath.org/index.html
- OBO units of measurement (UNITPATO): this is the ontology developed by the OBO Foundry (http://www.obofoundry.org/) for representing metrical units, and that is used in conjunction with the quality properties defined in PATO (http://www.berkeleybop.org/ontologies/owl/PATO). http://purl.org/obo/owl/UO.
- Ontology of units of measure and related concepts (WURVOC): this ontology models concepts and relations important to scientific research, having a strong focus on units and quantities, measurements, and dimensions. In this ontology, the units are defined as individuals. http://www.wurvoc.org/vocabularies/om-1.6/.

3.3 Ontologies of Cell Types

The Cell Type Ontology was designed as a structured controlled vocabulary for cell types. It was constructed to be used by the model organism databases and other bioinformatics databases, where there is a need for a controlled vocabulary of cell types for integration. Two versions of this ontology have been evaluated. The original version ontology, written oCTO, is an axiomatically lean version with just subsumption and develops from, converted from OBO to OWL. On the other hand, the normalized version, written nCTO, is an axiomatically rich version created by normalisation (http://odps.sourceforge.net/odp/html/Normalisation.html) where many characteristics are made explicit. In a normalised ontology, the reasoner maintains the multiple subsumption hierarchy. In order to create such an ontology, an axis of primitive classes is defined, with at most one superclass per class, and disjoint siblings. The rest of the ontology is codified as defined modules. Both ontologies are available at http://dis.um.es/~jfernand/icbo/, since they are the same ontologies used in Fernández-Breis *et al* (2009).

4. RESULTS

In this section, we describe the main results of the application of OQuaRE to the ontologies included in both case studies. The complete ones can be found at http://miuras.inf.um.es/evaluation/oquare.

4.1 Ontologies of Units of Measurement

First, we manually evaluated the ontologies, whose results are shown in Figure 1. It can be observed that the global score for each ontology is greater than 3, with some of them over 4. Therefore, the general quality of these ontologies is acceptable according to the judgement of the human evaluators, although they can be improved.

Figure 2 contains the global automatic score for the eight ontologies and their score for each quality characteristic. It can be observed that the global score for each ontology is greater than 3, ranging from 3.20 to 3.97. Therefore, the general quality of those ontologies quality is acceptable according to the evaluation criteria, although they can be improved. Provided that we do not intend

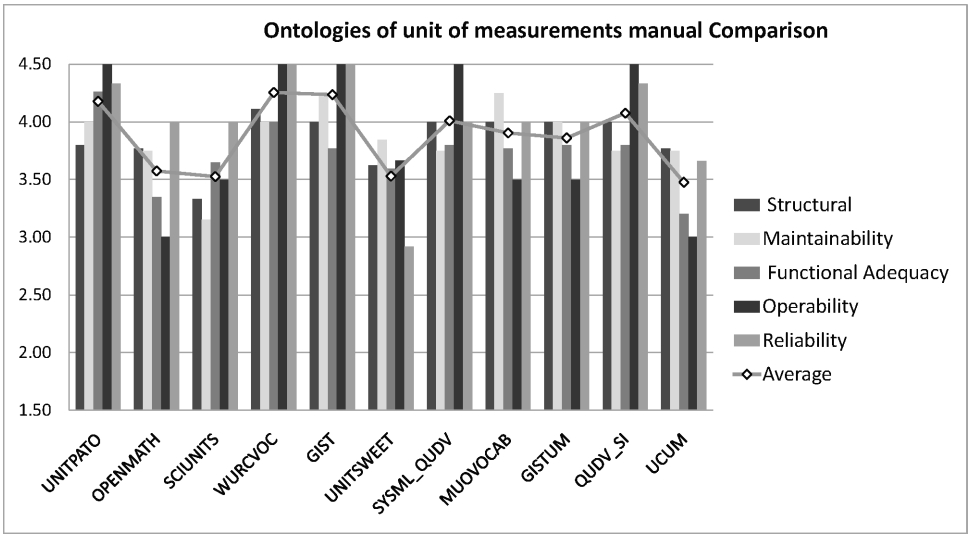


Figure 1: Manual score for ontologies of unit of measurements

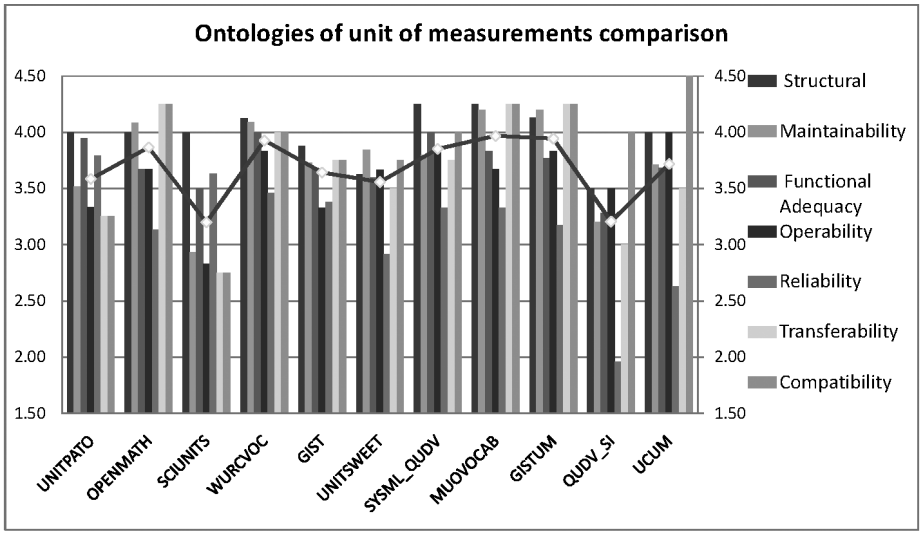


Figure 2: Complete automatic score for ontologies of unit of measurements

to rank the ontologies but to identify weaknesses and strengths we will not make a direct comparison about particular ontologies. It can be observed in Figure 2 that SCIUNITS is the ontology with more scores below 3 and that SCIUNITS, UNISTSWEET and UCUM are the three ontologies with at least one score below 3. These are indicators of the weaknesses of such ontologies. On the other hand, MUOVOCAB and GISTUM are the ontologies scoring over 4 in more categories, which indicate the strengths of those ontologies for particular characteristics. This intuitive comparison is already providing information to the users about the properties of the ontologies.

Let us identify some strengths and weaknesses for the evaluated ontologies according to our results.

- UCUM: This ontology could be difficult to re-establish in case wrong knowledge is identified, but this could be easily used in place of another specified ontology for the same purpose in the same language for creating, querying, exploiting and managing its knowledge, and its knowledge is suitable to be used for building other ontologies.
- WURCVOC: The components of this ontology could be modified with minimal effects over the rest of the components, but is not very useful for building other ontologies. The ontology structure can guide search processes and provides a semantic context to evaluate which data are wanted by the users.
- SCIUNITS: Changes (new axioms) in this ontology have minimal impact on other components of this ontology, but it is difficult to diagnose deficiencies or causes of failures (inconsistences), or for the parts to be modified, to be identified.
- QUDV_SI: Changes in parts of the ontology are likely to cause unexpected effects, like inconsistencies, in the rest of the ontology. This ontology could also present problems for detecting flaws and maintaining its level of performance for a given period of time, but its knowledge is good, so it could be used to replace another specified ontology.
- GISTUM: This ontology could be easily tested and validated, its knowledge could be effectively reused and adapted for different specified environments.

4.2 Types of Cells

The manual evaluation of these ontologies was not performed as part of this work, but the ones presented in Fernandez-Breis *et al* (2009) will be used. There, eight MSc students of the Semantic Web course at the University of Murcia evaluated the ontologies, thus providing a quantitative evaluation for each quality metric included in the framework. The results are based on an ISO 9126 framework and are shown in Figure 3. Most categories are shared with our OQuaRE approach, so a comparative analysis can be performed. One of the conclusions of the study was that the humans were too severe with the ontologies, and this will be taken into account in the comparative discussion.

The automatic scores obtained by each ontology for each quality dimension are shown in Figure 4. The global score for these ontologies range between 3.05 and 3.82, and the mean values for each characteristic for all the ontologies are: structural (4.00), maintainability (3.21), functional adequacy (3.85), operability (2.92), reliability (3.86), transferability (3.13), and compatibility (3.13). However, if we analyse the results for each ontology, oCTO gets scores below 3 for four categories, which is not a good result, whereas the scores for nCTO are in the positive range for all categories.

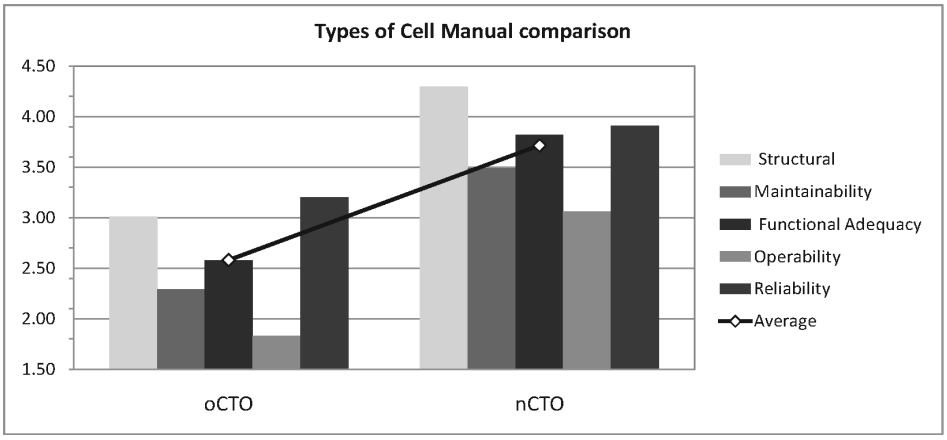


Figure 3: Manual scores for Types of Cell Ontologies

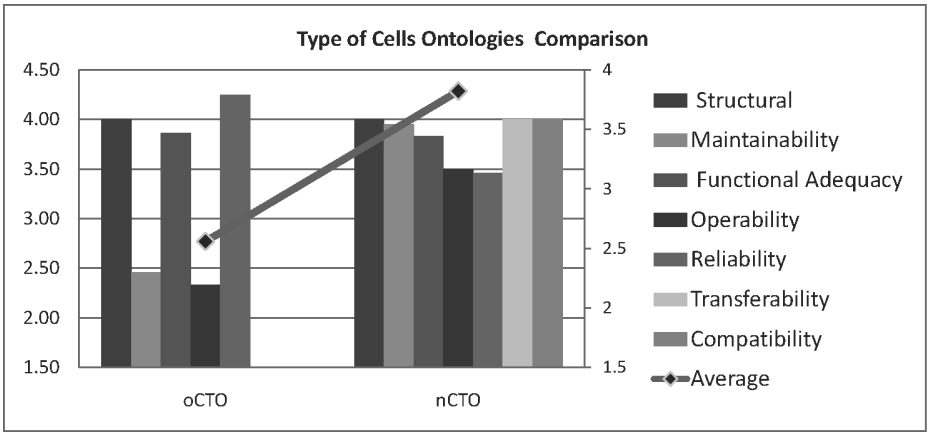


Figure 4: Automatic scores for Types of Cell Ontologies

Let us identify some strengths and weaknesses for the evaluated ontologies according to our results for the particular metrics:

- oCTO has high reliability (4.0), because its axioms and better formal support allowing for flaw detection earlier, the ratio of multiple parents per class is not high, which is good. The structure of the ontology can guide search processes, since it provides a semantic context to evaluate which data are wanted by the users. The components of this ontology can be hardly used in place of another ontology for the same purpose in the same environment.
- nCTO gets a high score for modularity. This means that its components could be modified with minimal effects over the rest of the components. However, its low modification stability means that changes in one part of the ontology may generate inconsistencies in other parts of the same ontology and the risk of non-detection this unexpected modifications effects is higher. The knowledge is codified in a precise and rich semantic context and therefore it is reusable by other ontologies. However, it could be difficult to re-establish a specified level of performance in the case of a failure due to the low recoverability, which is related with the distribution of the concepts in the ontology.

5. DISCUSSION AND CONCLUSIONS

In this section we will provide a comparative discussion of the results obtained in both case studies, and this will allow us to provide an initial evaluation and to draw some conclusions about the approach presented in this paper.

First, we must compare the results obtained by the manual and automatic evaluations. In the units of measurement case, we can see that the automatic scores obtained are, in general, lower than the manual ones, although they are usually in the range between 3.5 and 4. In the case of the cell types, the humans provide lower scores, although the results are, in general, consistent. In this way, we can say that the automatic scores provide expected results, that is, results similar to the manually obtained ones. An initial finding is that, if we consider absolute values, the human judgement produces higher differences in scores, but we do not know to date whether this might be caused by the need of adjustment of the evaluation method or the subjective nature of human judgement.

Second, we must compare the results of both automatic evaluation processes. In both case studies, the highest score has been obtained for the structural characteristic: 4.03 in units of measurements, and 4.00 in cell types. A similar global score is obtained for functional adequacy: 3.77 in units of measurements, and 3.85 in cell types. However, the results for the rest of the categories are different in the two scenarios, which means that the framework is able to identify the differences between domains and ontologies, which is a necessary property for a good evaluation framework. For instance, the ontologies of units of measurement get the lowest score for reliability 3.16, although this value is positive. On the other hand, the lowest score for the cell type ontologies is for operability. Such value is 2.92, which is not positive. This means that these cell type ontologies require more effort to be used, understood and learnt.

The results show that, from the perspective of our framework, the ontologies that have been evaluated in both experiments have an acceptable quality, although they can be improved in several directions in order to get higher scores and to increase, in this way, their quality.

The current version of OQuaRE provides a series of automatically calculated quantitative metrics; what makes it able to provide numeric scores as a result of the evaluation process. The quality model includes a series of the characteristics and subcharacteristics that have been defined, reused or adapted to ontology evaluation. Most of them have been used by state-of-the-art ontology

Approach \ Charact.	Structure	Functional adequacy	Maintain-ability	Operability	Reliability	Transfer-ability	Compat-ibility
Ontometric	X				X		
Netzer *et al*		X					
Park *et al*	X						
Ontoclean	X						
Völker *et al*	X						
Corcho *et al*	X						
Wang	X						
Vrandecic	X	X				X	
Ma *et al*		X					
Gangemi *et al*	X	X		X	X		
Rogers	X	X		X			

Table 4: How other approaches deal with the OQuaRE characteristics

evaluation methods, but they have not been combined in a standardised way as we do in OQuaRE. Table 4 shows how the related approaches match our framework. In the table each cell is marked in case the particular approach includes such quality characteristics. Most of the approaches make use of structural criteria, whereas no one uses maintainability and compatibility. This makes our approach the most complete among those available, so far, for ontology evaluation.

OQuaRE is a general framework that defines families of quality approaches, because it can be adapted to the needs of each particular domain by selecting the characteristics, subcharacteristics, metrics and specific mappings. In addition to this, it permits to give priority to some characteristics or subcharacteristics by weighting the importance of each subcharacteristic and characteristics.

In this work, our interest was put on evaluating the feasibility of adapting the software quality standards and metrics for evaluating the quality of ontologies. Our results show that this is possible and that the automatic methods proposed in OQuaRE are able to detect the main strengths and weaknesses in the evaluated ontologies. In addition to this, the framework is able to identify the differences between ontologies in the same domain, which is useful for supporting users in making informed decisions. In this sense, we do not pursue assigning a global score to the ontologies but to identify their strengths and weaknesses. This is why we have not concluded the experiments saying which ontologies is the best in each case study; instead, we have described the main features of each ontology. All the evaluated ontologies have an acceptable quality for most criteria, and the decision has to be made by the users or developers based on their needs.

The novelty and contribution of the approach is not in the definition of the categories, subcategories and metrics, but in how they are organized by using the guidelines of an ISO standard, and how the approach tends to convert the evaluation of ontologies in a engineering activity.

Further research should be carried out in a series of directions. First, a series of quality metrics and measurement primitives have been defined and implemented by reusing and adapting state-of-the-art metrics. A series of thresholds and mappings have been proposed for those metrics with the aim of obtaining values in the range 1 and 5 suggested by SQuaRE. For this purpose, the recommendations and best practices of the Software Engineering and Ontology Engineering

communities were followed. We think that such thresholds and the value mappings to be used in a engineering environment. Second, we should work on the optimization of the mappings between the metrics scores and the SQuaRE evaluation scale. Third, we should evaluate whether all the subcharacteristics and metrics should be equally weighted. We believe that such issues should be the result of a discussion in the ontology evaluation community, and were out of the scope of this work. Finally, we should extend OQuaRE with the requirements and evaluation processes in order to complete the evaluation framework.

ACKNOWLEDGEMENTS

This research was funded by the Spanish Ministry of Science and Innovation through grant TIN2010-21388-C02-02.

REFERENCES

BECK, K. and CUNNINGHAM, W. (1987): Using pattern languages for object-oriented programs. *Workshop on the Specification and Design for Object-Oriented Programming, OOPSLA*. Orlando, Florida, United States: ACM.

BLANCO, C., LASHERAS, J., FERNÁNDEZ-MEDINA, E., VALENCIA-GARCÍA, R. and TOVAL, A. (2011): Basis for an integrated Security Ontology according to a systematic review of existing proposals. *Computer Standards & Interfaces* 33(4): 372–388.

BRANK, J., GROBELNIK, M. and MLADENIC, M. (2005): A survey of ontology evaluation techniques. *Conference on Data Mining and Data Warehouses (SiKDD 2005)*. Ljubljana, Slovenia.

CHIDAMBER, S.R. and KEMERER, C.F. (1994): A metric suite for object oriented design. *IEEE Transactions on Software Engineering*, 467–493.

CHIKOFSKY, E.J. and CROSS, J.H. (1990): Reverse engineering and design recovery – A Taxonomy. *IEEE Software*, 7:13–17.

CORCHO, O., GÓMEZ-PÉREZ, A., GONZÁLEZ-CABERO, R., SUÁREZ-FIGUEROA, M.C. (2004): Odeval: A tool for evaluating RDF(S), DAML+OIL and OWL concept taxonomies. *1st IFIP Conference on Artificial Intelligence Applications and Innovations*. Toulouse, France. 369–382.

EGAÑA, M., ANTEZANA, E., KUIPER, M. and STEVENS, R. (2008): Ontology design patterns for bio-ontologies: a case study on the cell cycle ontology. *Bmc Bioinformatics*, 9.

FERNÁNDEZ-BREIS, J., EGAÑA, M. and STEVENS, R. (2009): A quality evaluation framework for bio-ontologies. In: PRECEDINGS, N., ed. *ICBO International Conference on Biomedical Ontology*, University at Buffalo, NY. 136–139.

GANGEMI, A., CATENACCI, C., CIARAMITA, M. and LEHMANN, J. (2006): Modelling ontology evaluation and validation. *Semantic Web: Research and Applications, Proceedings*, 4011: 140–154.

GANGEMI, A. and PRESUTTI, V. (2007): Ontology design for interaction in a reasonable enterprise. In: RITTGEN, P. (ed.) *Handbook of Ontologies for Business Interaction*. IGI.

GASEVIC, D., DJURIC, D., DEVEDZIC, V. and SELIC, B. (2006): *Model Driven Architecture and Ontology Development*, New YorK, Springer-Verlag.

GASEVIC, D., KAVIANI, N. and MILANOVIC, M. (2008): Ontologies and software engineering. In: STAAB, S.S.R. (ed.) *Handbook on Ontologies*. Springer.

GÓMEZ-PEREZ, A. and ROJAS-AMAYA, M.D. (1999): Ontological reengineering for reuse. In: SPRINGER-VERLAG (ed.) *Lecture Notes in Computer Science* 1621.

GRUBER, T.R. and OLSEN, G.R. (1994): An ontology for engineering mathematics. In: DOYLE, J., SANDEWALL, E. and TORASSO, P. (eds.) *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourth International Conference*.

GUAN, S. and ZHU F. (2004): Ontology acquisition and exchange of evolutionary product-brokering agents. *Journal of Research and Practice in Information Technology*, 36: 35–46.

GUARINO, T.R. and OLSEN, G.R. (1994): An ontology for engineering mathematics. In: DOYLE, J., SANDEWALL, E. and TORASSO, P. (eds.) *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourth International Conference*.

IEEE (1997): *IEEE Standard for Developing Software Life Cycle Processes*, Software Engineering Standards Committee of the IEEE Computer Society.

ISO9126 (2001): ISO/IEC9126 Software engineering – Product quality. Geneva, Switzerland.: International Organization for Standardization.

ISO14598 (1999): ISO/IEC 14598 – Information technology – Software Product Evaluation. International Organization for Standardization.

ISO25000 (2005): ISO/IEC 25000 2005, Software engineering – Software product quality requirements and evaluation (SQuaRE) – guide to square (ISO/IEC 25000). Geneva, Switzerland: International Organization for Standardization.

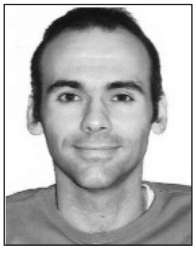
LASHERAS, J., VALENCIA-GARCÍA, R., FERNÁNDEZ-BREIS, J.T. and TOVAL, A. (2009): Modelling reusable security requirements based on an ontology framework. *Journal of Research and Practice in Information Technology*, 41: 119–133.
LI, W. and HENRY, S. (1993): Object-oriented metrics that predict maintainability. *Journal of Systems and Software*, 23: 111–122.
LINCKE, R. and LOWE, W. (2007): Compendium of software quality standards and metrics – Version 1.0. Available: http://www.arisa.se/compendium/.
LÓPEZ, M.F., GÓMEZ-PÉREZ, A., SIERRA, J.P. and SIERRA, A.P. (1999): Building a chemical ontology using methontology and the ontology design environment. *IEEE Intelligent Systems and Their Applications*, 14: 37–46.
LOZANO-TELLO, A. and GÓMEZ-PÉREZ, A. (2004): Ontometric: A method to choose the appropriate ontology. *Journal of Database Management*, 15: 1–18.
MA, Y.L., MA, X.Y., LIU, S.H. and JIN, B.H. (2009): *A Proposal for Stable Semantic Metrics Based on Evolving Ontologies*.
NETZER, Y., GABAY, D., ADLER, M., GOLDBERG, Y. and ELHADAD, M. (2009): Ontology evaluation through text classification. In: CHEN, L. and LIU, C. (eds.) *Advances in Web and Network Technologies, and Information Management*. Berlin: Springer-Verlag Berlin.
PARK, J., OH, S. and AHN, J. (2011): Ontology selection ranking model for knowledge reuse. *Expert Systems with Applications,* 38: 5133–5144.
ROGERS, J.E. (2006): Quality assurance of medical ontologies. *Methods of Information in Medicine*, 45: 267–274.
STEARNS, M.Q., PRICE, C., SPACKMAN, K.A. and WANG, A.Y. (2000): SNOMED clinical terms. *British Journal of Healthcare Computing and Information Management,* 17: 27–31.
STEVENS, R. and LORD, P. (2009): Application of ontologies in bioinformatics. In: BERNUS, P., BŁAŻEWICS, J., SCHMIDT, G., SHAW, M., STAAB, S. and STUDER, R. (eds.) *Handbook on Ontologies*. Springer Berlin Heidelberg.
TARTIR, S. and ARPINAR, I.B. (2007): Ontology evaluation and ranking using OntoQA. *ICSC 2007: International Conference on Semantic Computing, Proceedings*, 185–192.
VOROBIEV, A. and BEKMAMEDOVA, N. (2010): An ontology-driven approach applied to information security. *Journal of Research and Practice in Information Technology*, 42: 61–76.
VÖLKER, J., VRANDECIC, D. and SURE, Y. (2005): Automatic evaluation of ontologies (AEON). *Semantic Web – Iswc 2005, Proceedings*, 3729: 716–731.
VRANDECIC, D. (2010): *Ontology Evaluation*. Ph.D thesis. University of Karlsruhe.
WANG, T.D. (2006): Gauging ontologies and schemas by number. *EON Workshop on Evaluation of Ontologies for the Web, Proceedings* (eds.) ACM Press. USA.
WANG, T.D. and PARSIA, B. (2007) Ontology performance profiling and model examination: First steps. In: ABERER, K., CHOI, K.S., NOY, N., ALLEMANG, D., LEE, K.I., NIXON, L., GOLBECK, J., MIKA, P., MAYNARD, D., MIZOGUCHI, R., SCHREIBER, G. and CUDREMAUROUX, P. (eds.) *Semantic Web, Proceedings*.
YAO, H., ORME, A. and ETZKORN, L. (2005): Cohesion metrics for ontology design and application. *Journal of Computer Science*, 1.

BIOGRAPHICAL NOTES

Astrid Duque-Ramos is a PhD student at the University of Murcia in Spain. She received a degree in computer science from the University of Antioquia, Colombia. She is a member of the Knowledge Modelling, Processing and Management Technologies research group of the University of Murcia. Her current research interests include software quality, ontology quality and requirements engineering.

Astrid Duque-Ramos

Dr Jesualdo Tomás Fernández-Breis received his BA, MSc and PhD degrees in computer science from the University of Murcia. He is an associate professor at the Department of Informatics and Systems, University of Murcia. His research interests include ontology engineering and the development and application of semantic web technologies in biomedical domains. He is currently a member of the International Association of Ontologies and its Applications (IAOA).



Jesualdo Tomás Fernández-Breis

Dr Robert Stevens is a reader in BioHealth Informatics in the Bio and Health Informatics Group at the University of Manchester. He has a BSc in biochemistry, an MSc in biological computation, and a DPhil in computer science. His main research areas are bioinformatics and e-Science, with special interest in how ontologies based on description logics can improve the use of data in bioinformatics analysis.

Robert Stevens

Nathalie Aussenac-Gilles's main research topics focus on bottom-up approaches for knowledge acquisition and modeling since she obtained her PhD in 1989. She became a CNRS researcher in 1991. Since then she has been a member of the IRIT laboratory in computer science at University "Paul Sabatier" of Toulouse, where she currently heads the IC3 team Knowledge Engineering, Cooperation and Cognition. Her research interests include knowledge engineering, natural language processing and terminology based approaches for ontology engineering from text. Her work is influenced by long term cross-disciplinary collaborations with linguists and researchers in human factors. She has chaired the Terminology and Artificial Intelligence group from 2000 to 2008 and the GDR I3 SIG on knowledge engineering since 2002.

Nathalie Aussenac-Gilles