SemQuaRE — An extension of the SQuaRE quality model for the evaluation of semantic technologies

Filip Radulovic *, Raúl García-Castro, Asunción Gómez-Pérez

Ontology Engineering Group, ETSI Informáticos, Universidad Politécnica de Madrid, Campus de Montegancedo, s/n, Boadilla del Monte 28660, Spain

A B S T R A C T

To correctly evaluate semantic technologies and to obtain results that can be easily integrated, we need to put evaluations under the scope of a unique software quality model. This paper presents SemQuaRE, a quality model for semantic technologies. SemQuaRE is based on the SQuaRE standard and describes a set of quality char-acteristics specific to semantic technologies and the quality measures that can be used for their measurement. It also provides detailed formulas for the calculation of such measures. The paper shows that SemQuaRE is com-plete with respect to current evaluation trends and that it has been successfully applied in practice.

1. Introduction

It is well known that software quality is a crucial need across domains (e.g., security and health) and technologies (e.g., operating systems and databases), and that to obtain high-quality software products, the specification and evaluation of quality are of pivotal importance [1].

Semantic technologies provide new ways to express knowledge and data in machine processable formats that can be exploited by software agents. There are different types of semantic technologies [2], which can be used for different tasks and which fulfill different requirements. For example, ontology editors are used for implementing ontologies, whereas ontology matching tools are used for mapping concepts of one ontology to concepts of another. However, some tools also share a number of requirements related to different technology types, as is the case of ontology engineering environments. An example of such tool is Protégé,[1] an ontology editor that provides additional functionalities such as reasoning (HermiT plug-in[2]) or querying (SWRL-IQ plug-in[3]). In recent years we have seen an exponential growth of semantic technologies, and in order to assess their quality, multiple evaluations of such technologies have been proposed, from general evaluation frameworks [3] to tool-specific evaluations [4,5] and even to characteristic-specific evaluations [6].

In order to benchmark different tools, which is an important aspect of evaluation [7], evaluation results from various sources need to be integrated and compared. In this way, through the observation of the results of quality evaluations, the quality of different products is compared in terms of meeting the users' needs. This is particularly significant for the integration of the evaluation results of different types of tools and for the selection of tools according to their quality. However, since no consistent terminology for describing semantic technology quality exists, the comparison of evaluation results sometimes requires additional effort. For example, the results of a specific reasoning task obtained by Baader et al. [8] and by Glimm et al. [9] can be compared, but the fact that such results are presented with different terminologies, "labeling time" and "classification time" respectively, requires a deeper analysis of the evaluation process. Furthermore, in some cases different characteristics are evaluated, and the analysis of these results or their reuse can be misleading or impossible. An example of this occurs in the analysis of reasoning times by Meditskos and Bassiliades [10] and by Urbani et al. [11]; in both cases the results are impossible to integrate because when they refer to reasoning times, they refer to different reasoning processes.

Software quality models provide the basis for software evaluation and give a better insight of the characteristics that influence software quality by specifying a consistent terminology for software quality and by providing guidance for its measurement. Evaluations from various sources which are driven by a quality model produce results that can be much more easily integrated and, hence, the comparison and benchmark of tools become easier. Generic software quality models exist (e.g., SQuaRE [12]) and in order to use a generic quality model in a

* Corresponding author. Tel.: +34 913363670; fax: +34 913524819.
E-mail addresses: fradulovic@fi.upm.es (F. Radulovic), rgarcia@fi.upm.es (R. García-Castro), asun@fi.upm.es (A. Gómez-Pérez).

[1] http://protege.stanford.edu/.
[2] http://protegewiki.stanford.edu/wiki/HermiT.
[3] http://protegewiki.stanford.edu/wiki/SWRL-IQ.

specific domain, the model usually has to be extended to include the particularities of such domain [13,14].

Various methods for extending quality models have been proposed in the literature [1,15]; they all follow a top-down approach, starting from general characteristics to concrete measures. However, as reported by some authors (e.g., Botella et al. [16]), identifying all the elements of a quality model can be a difficult task when using this approach. For some cases a bottom-up approach can be also applied, as those in which many evaluation results are available and from which the quality model can be extracted. An example of this case is the semantic technology field, in which various initiatives such as the SEALS project[4] [17] and the Ontology Alignment Evaluation Initiative[5] have performed evaluations and provided evaluation results.

Following the discussion above, it is important to have a quality model in the semantic technology domain that guides the evaluation of different semantic technology types. The goal of this paper is to define a quality model for semantic technologies that extends the SQuaRE quality model, starting from real semantic technology evaluations. The model here proposed has been evaluated in terms of completeness, flexibility, and applicability.

The quality model for semantic technologies is a hierarchical quality model developed from the evaluation results provided by the SEALS project. This model describes a set of quality characteristics and sub-characteristics for six different types of semantic technologies: ontology engineering tools, ontology matching tools, reasoning systems, semantic search tools, and semantic web services; ontology annotation tools have been included into the model during the process of evaluation. For each of these sub-characteristics, a set of quality measures is specified, and formulas for each measure are provided. These measures are organized into hierarchy of base measures, derived measures, and indicators.

The reminder of this paper is organized as follows. Section 2 gives an overview of existing generic software quality models and presents the elements of a software quality model with their definitions. Section 3 presents the current state of semantic technology quality specification. Section 4 describes the available top-down methods for extending software quality models. Section 5 describes how SemQuaRE, a quality model for semantic technologies, is defined; Section 6 presents the evaluation. And finally, Section 7 draws some conclusions and includes ideas for future work.

2. Software quality models

According to the ISO/IEC 9126 standard [18], one of the processes in the software development lifecycle is the evaluation of software products in order to satisfy software quality needs, where software quality is evaluated by measuring software attributes. Those attributes can be grouped into categories that are called quality characteristics, and these quality characteristics can be later decomposed into quality sub-characteristics [12]. Furthermore, the ISO/IEC 9126 standard states that software product quality should be evaluated using a quality model.

Various generic software quality models have been described in the literature: McCall's quality model [19]; Boehm's quality model [20]; the ISO 9126 quality model [18]; and the SQuaRE [12] quality model. Next, we give a brief description of the most recent software quality models.

The International Organization for Standardization (ISO) identified the need for a unique and complete software quality standard and, therefore, produced the ISO 9126 standard for software quality. The ISO 9126 standard offers a complete view of software quality with definitions for all the quality characteristics and sub-characteristics that the standard describes.

[4] http://www.seals-project.eu/.
[5] http://oaei.ontologymatching.org/.

The ISO 9126 is a hierarchical quality model that defines three types of qualities: internal quality, external quality, and quality in use. The standard describes six main quality characteristics for external and internal quality, which in turn are decomposed into sub-characteristics, and also provides the internal and external measures for sub-characteristics. For quality in use, the standard proposes four characteristics.

Although the ISO 9126 standard has been accepted and used successfully, some problems and issues related to its further use and improvement have been identified. These problems eventually arise mainly because of advances in technologies and changes in users' needs. As pointed out by Azuma [21], the main problem is due to issues on metrics (e.g., the existence of metrics that have no verified direct correlation with quality characteristics but that are generally recognized as related to product quality; and the distribution of metrics information in several parts of the standard series, among others).

With the object of harmonizing the content with the ISO 15939 standard [22] and taking into account the advancement in technology and environmental changes [23], the ISO 9126 standard has been redesigned and named SQuaRE (Systems and software Quality Requirements and Evaluation). The SQuaRE quality model [12] has the same hierarchical structure as the ISO 9126 quality model, with the difference that internal and external quality models are combined into a product quality model, which describes eight quality characteristics and thirty one quality sub-characteristics (Fig. 1), and is related to "static properties of software and dynamic properties of the computer system". As for quality in use, SQuaRE defines it as "the outcome of interaction when a product is used in a particular context of use", and describes five quality characteristics and nine quality in use sub-characteristics.

Taking into account the quality models described in the literature, we can define a quality model as a set of quality characteristics, sub-characteristics, and quality measures of a product and the relationships between them.

According to the ISO 15939 standard [22], software evaluation is performed with the goal of capturing some information about an attribute of a software product. In this process, a measurement method is applied, and the result is a base measure, which captures some information about an attribute. Two or more base measures can be combined in a measurement function in order to obtain *derived* measures. Finally, base and/or derived measures are later combined to obtain *indicators*, which are measures that provide evaluations of specific attributes. Therefore, the output of an evaluation process is a set of measures that, according to the SQuaRE standard, has to be specified in a quality model.

Finally, for any of those measures a measurement method and measurement scale, called metric, can be defined [18]. Additionally, a unit of measurement can also be defined for all measures.

As different quality models use different terminologies for their elements [24], we have based our terminology on the SQuaRE and ISO/IEC 15939 standards.

3. Specification of semantic technology quality

Different types of semantic technologies have been described in the literature [2,3]; however, to the best of our knowledge there is no consistent terminology for describing these technologies and the tasks they are designed to perform. The current scope of our research is limited to five types of semantic technologies (which do not exhaustively cover every type of semantic technology), for which we give the following definitions: *Ontology engineering tools*, which are software for managing ontologies and their components either through a user or a programming interface; *Ontology matching tools*, which are software programs for finding correspondences (i.e., alignments) between the components of two ontologies; *Reasoning systems*, which are software programs for inferring logical consequences from ontology instances; *Semantic search*

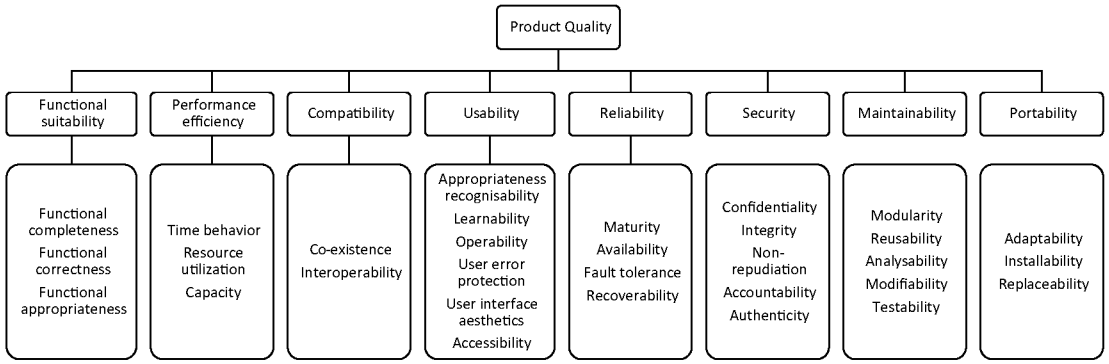

Fig. 1. SQuaRE quality model — product quality [12].

tools, which are software programs that provide answers to natural language queries by exploiting data semantics; and *Semantic web services*, which are software programs that support the use of semantic web services (i.e., web services enriched with semantic descriptions) for different tasks (e.g., discovery, composition and mediation, among others).

Multiple evaluations of semantic technologies have been performed to this date, both conducted by individuals and by the community. In order to obtain the current state of specification of semantic technology quality, we have performed a literature review. This section presents such a review on semantic technology evaluations, which was performed according to the procedure described by Kitchenham [25].

Next, we present the outcomes of each step described in the procedure:

- *Background*. The purpose of the research is to review previous evaluations of semantic technologies, including the evaluation results, with the object of having an overview of the current state of the quality of semantic technologies with respect to specification.
- *Research questions*. We stated two research questions. *RQ1: Is there a common approach for specifying the quality of semantic technologies?* This literature review will allow us to observe if there is a unique reference or guideline followed by researchers to specify the quality of semantic technologies. *RQ2: Which semantic technology characteristics are evaluated and which quality measures are used to evaluate them?* The goal here is to observe the quality characteristics and measures used in different evaluations as well as the terminology used.
- *Strategy for searching previous studies*. When looking for research relevant to our questions, in order to identify those publications that deal with semantic technology evaluation, we decided to analyze the proceedings of the two most relevant conferences and those of the three most important workshops in the semantic web area that are focused on the topic of evaluation. These proceedings are: International Semantic Web Conference (ISWC) — all eleven editions (2002–2012); European/Extended Semantic Web Conference (ESWC) — all nine editions (2004–2012); International workshop on Evaluation of Ontology-based Tools (EON) — all six editions (2002–2004;2006–2008); International Workshop on Evaluation of Semantic Technologies (IWEST) — all two editions (2010, 2012); and International workshop on Scalable Semantic Web Knowledge Base Systems (SSWS) — all eight editions (2005–2012).
  In the first phase of the search, the title and abstract of each paper were inspected in order to identify those publications that could answer our research questions. This phase resulted in 179 publications.
- *Study selection criteria and procedures*. All publications that are identified in the previous step were thoroughly revised with the object of selecting those that answered the research questions. The final selection was made with respect to several criteria.
  First, the selected studies included only publications dealing with the following types of semantic technologies: ontology engineering tools, reasoning systems, ontology matching tools, semantic search tools, and semantic web service tools. These are the types of semantic technologies most frequently evaluated in the literature, and are in the current scope of our work. Second, the selected studies included publications discussing or presenting measures for semantic technology evaluation, as well as publications reporting on evaluations of semantic technologies. Finally, publications suggesting new algorithms (e.g., for reasoning or semantic web service discovery) that were evaluated in the publication were also selected.
- *Data extraction strategy*. While inspecting the publications collected for this literature review, the data related to research questions were extracted. Extracted data include the type of semantic technology evaluated, as well as measures used in the evaluation. Measures were extracted in their original terminology, which means that we took into account the different terminologies that various authors use for the same measure.
  The data allowed us to analyze the publications and to answer our research questions.
- *Synthesis of the extracted data*. In order to answer the research questions, the data extracted was synthesized. First, the measures were extracted in their original terminology and were classified according to the measure evaluated. This helped us to observe the use of terminology and see whether different terms are used for the same measures. Then, each extracted measure was aligned to a quality characteristic from SQuaRE, and all the measures extracted in the review were classified according to those characteristics and to types of semantic technologies. This permitted us to observe which quality characteristics were evaluated. Finally, the frequencies of each measure and characteristic were also taken into account. This permitted us to observe which measures and characteristics are most common.

In total, we analyzed one hundred publications.[6] Table 1 shows the results of the analysis including, for each type of semantic technology, the number of papers in which each type is evaluated, and the quality characteristics used (shown in bold) that were extracted from SQuaRE standard. For every characteristic, a set of measures is presented and, in those cases where different terminologies were used, measures are grouped under a common name (shown in italics). For example, classification correctness for reasoner systems is represented as 'number of successes' and 'number of solved tasks'. In some cases, however, we were not able to determine the exact classification (e.g., in the case of time behavior of reasoning systems). The number of papers in which a measure was found, and the number of papers that evaluated a specific characteristic and a tool type are shown in brackets. Numbers are omitted when only one occurrence appears.

Next, we show the results of the literature review with respect to each research question.

RQ1. We have found that no common approach for specifying the quality of semantic technologies exists and that no quality model or

[6] The complete list of publications is available for download at http://goo.gl/LzWu0T.

Table 1
Measures used in conference publications.

Ontology engineering tools (8)
Functional correctness (4) – precision (2), recall (2), information added, information lost, benchmarks passed
Interoperability (3) – *information addition/loss* (3) [changes performed, import/export, knowledge preserved, knowledge lost]
Time behavior (2) – execution time (2)
Maturity (1) – execution;
Ontology matching tools (35)
Functional correctness (32) – precision (31); recall (30); *f-measure* (22) [f-measure (19), f-score (3)], weighted symmetric difference, found correspondences
Time behavior (4) – *matching time* (4) [execution time, time, mapping time, runtime]
Maturity (1) – error rate
Resource utilization (1) – memory consumption
Reasoning systems (31)
Time behavior (26) – *classification time* (11) [classification time (5), time (3), classification performance, total time, labeling time, lattice operation time, query time], *query execution time* (5) [query time (2), query response time, query answering time, time], *entailment time* (5) [time for entailment, time, justification time, reasoning time,kernel running time], *data loading time* (2) [loading time (2)], *ontology loading time* [loading time], *closure time* (2) [runtime, closure time], execution time, cpu time, response time, runtime, time cost, loading time
Functional correctness (11) – *classification* (4) [number of successes, number of correct classifications, number of wrong classifications, number of solved tasks, classifications], *entailment* (3) [number of solutions per second, soundness, logical entailment], *satisfiability* (2) [number of SAT instances solved, class satisfiability, ontology satisfiability], number of matches (2), number of refinements, effectiveness, gain, answer completeness
Resource utilization (2) – memory usage, network traffic
Maturity (2) – resources exceeded, errors, failure
Semantic search tools (15)
Functional correctness (11) – precision (9), recall (8), NDCG (2), *f-measure* (3) [f-measure (2), f-score], reciprocal rank, system accuracy, relevance, number of correct queries and answers
Time behavior (10) – *query execution time* (5) [time (2), query response time, running time, execution time], *question time* (2) [time, question time], input time (2), system speed, task completion time, execution time, response time
Operability (5) – SUS questionnaire (4), search experience
Satisfaction (3) – extended questionnaire (2), user satisfaction
Efficiency (2) – answer found rate (2), number of attempts (2), experiment time (2)
Learnability (1) – learning
Semantic web service tools (11)
Functional correctness (8) – precision (8), recall (8), f-measure, number of target sources returned, BPREF, reciprocal rank, success at cut-off point, cumulated gain
Time behavior (5) – *discovery time* (5) [processing time, query answering time, query response time, execution time, discovery time]

guideline has been developed in this domain. Furthermore, of all the publications analyzed, we have observed that only one publication provides instructions on the measures to be evaluated. In this publication, the authors propose a set of novel measures for ontology matching tools: comprehensive f-measure, STD score and ROC-AUR score, which have never been used before in evaluations [26]. Five publications provide some discussion on existing measures that have been used before, of which three are related to reasoning systems [7,27,28], and one to semantic web services [29] and ontology matching tools [30].

RQ2. Table 1 shows all the information related to the characteristics being evaluated and the quality measures used, and we can outline several conclusions.

When we see a specific type of semantic technology, we can observe that there is a variety of characteristics in the quality that are evaluated; i.e., in different publications we observe different characteristics evaluated. For example, while in some publications we find that time behavior of semantic web services is evaluated, in others only correctness is.

When we study specific quality characteristics, we can see that there is a variety of measures for measuring them. This means that, although some publications evaluate the same quality characteristic, they measure it with different measures. An example of this can be the classification and satisfiability evaluations of reasoning systems.

Different publications use different terminology for measures so their names are different even when they evaluate the same measures. This can be directly observed in Table 1 (e.g., in the case of classification correctness of reasoning systems).

As discussed before, the quality of software products is of high importance; therefore, the specification and evaluation of quality is of great concern. Software quality models provide the basis for quality specification and evaluation; however, since there is neither a unique specification nor a quality model for semantic technologies, the current state of semantic technology evaluations is characterized by a large variety of characteristics, measures, and terminology of the quality evaluated. A possible consequence of this variety is that the integration and analysis of different evaluation results (e.g., when performing benchmarks) might lead to misleading and wrong conclusions. Therefore, without a common ground to specify the quality of semantic technologies, it is very difficult to compare them and to assess their quality.

4. Extending software quality models

Software quality models (e.g., the ISO 9126 or SQuaRE ones) provide insight into characteristics that are generic [14]. However, different types of software products have characteristics specific to them and, therefore, the actual application of these software quality models usually requires refining the generic characteristics to conform to a specific software product or domain [13,14].

As pointed out by Al-Kilidar et al. [31], some practical problems with ISO 9126 arose, namely, ambiguity in metric definitions and usability interpretation. Furthermore, the authors argue that a number of attributes and measures are missing, that some characteristics are too abstract, and that the standard itself is open to interpretations, all of which calls its usefulness into question, according to the authors. On the other hand, some authors suggest that, taking into account the nature of the product itself, some new sub-characteristics can be added, the definitions of existing ones can be changed, or some sub-characteristics can be eliminated from the model [16]. By following these suggestions, the problems previously mentioned can be overcome.

For various types of applications, different authors have proposed software quality models in the domains of B2B [1], mail servers [32], web-based applications [33], e-learning systems [34], and ERP systems [16]. All those authors have used the ISO 9126 standard as the basis software quality model and have extended it to conform to the particular domain. Additionally, some authors have extended the SQuaRE quality model in the domains of web services [35] and software evolution [36].

Software quality model extensions can be performed following two main approaches [37]: a top-down approach that starts from the quality characteristics and continues towards the quality measures, and a bottom-up approach that starts from the quality measures and defines the quality sub-characteristics related to each specific measure.

Next, we describe some methods that we have found in the literature for extending software quality models.

Franch and Carvallo proposed a method for customizing the ISO 9126 quality model based on a top-down approach [15], which they latter applied in constructing the quality model for mail servers [32]. After defining and analyzing the domain of interest, their method proposes several steps.

In this first step, quality sub-characteristics are determined and, according to the domain, some new quality sub-characteristics are added while others are excluded or their definitions are changed. If needed, sub-characteristics are further decomposed according to some criteria, and a hierarchy of sub-characteristics is defined. Afterwards, sub-characteristics are decomposed into attributes, which are more concrete concepts that refer to some particular software attribute

(i.e., observable feature). Furthermore, attributes not directly measurable are further decomposed into basic ones.

After the quality entities have been defined (quality sub-characteristics and attributes), the relationships between these quality entities are explicitly defined. Three possible types of relationships are identified: *collaboration* means that increasing the value of one entity implies increasing the value of another entity; *damage* means that increasing the value of one entity implies decreasing the value of another entity; and *dependency* means that some values of one entity require that another entity fulfills some conditions. Finally, to be able to compare and evaluate quality, it is necessary to define metrics for all the attributes in the model.

In order to build their quality model for B2B applications, Behkamal et al. proposed a method that customizes the ISO 9126 quality model in five steps [1]. The main difference with the previous method is that in Behkamal's approach, the quality characteristics are ranked by experts; thus, experts should provide weights for all quality characteristics and sub-characteristics, and these weights are later used to establish their importance, which can be time consuming and resource demanding. Another important difference is that Behkamal's approach does not contemplate defining relationships between quality entities (quality characteristics and attributes).

At the time of writing this paper, we have not found any example of a bottom-up approach in the literature. In some scenarios, however, it can prove helpful to base the extension of the quality model on existing software evaluations and evaluation results when available. In some cases, both bottom-up and top-down approaches are important for building quality models [37]. Therefore, since a larger number of evaluation results are already available in the semantic technology domain, we have decided to follow a research methodology based on a bottom-up approach [38] to define a quality model for semantic technologies. Next, we present the steps of such methodology, including examples of each of them for the case of web browsers evaluation:

1. *To identify base measures.* The output of evaluating a software product with some input data allows identifying the base measures. For example, in web browser evaluation, base measures could be *page loading time*, *startup time*, *memory consumption*, or *number of open tabs*.
2. *To identify derived measures.* Base measures can be combined among them and/or with input data in order to obtain derived measures. Derived measures are defined in a way that they bring additional and meaningful information not provided by the base measures themselves; it is also possible to use one base measure in order to obtain more than one derived measure. In some cases it is possible to combine different derived measures to obtain other derived measures.
   A derived measure for web browsers could be *memory consumption per open tab*.
3. *To identify indicators.* Indicators are measures related to a whole evaluation and are obtained by the aggregation of base measures and/or derived measures. As in the previous case, a base or derived measure can be used in more than one indicator. For each indicator, a scale and a unit of measurement should be specified.
   In the case of web browsers, an indicator could be the *average startup time*. This indicator belongs to a ratio scale, and it is measured in seconds.
4. *To specify relationships between measures.* In this step, which can be performed in parallel with the previous ones, relationships between measures are expressed either in an informal way (e.g., the collaboration, damage and dependency categories proposed by Franch and Carvallo [15]), or more formally (e.g., with the formulas used for obtaining the measures, as proposed by Bombardieri and Fontana [36]). For every derived measure defined, it is recommendable to specify the formula (or set of formulas) that allows obtaining such derived measure from the base measures. Similarly, it is also advisable for any indicator to identify the formula that defines such an indicator based on other measures. Also, it is important to note that one indicator can be obtained by means of different formulas. Additionally, in order to have consistency in the quality model, all the lower level measures should be used for obtaining measures at the higher level. That means that the model cannot contain any base measure that is not used for obtaining any of the derived measures or indicators, or that an indicator cannot be obtained by means of a derived or base measure not defined in the model.
   In the web browser example, the formula for the memory consumption per open tab could be the ratio between the total memory consumption and the number of open tabs. The average startup time could be the average of startup times measured in the evaluation.
5. *To define domain-specific quality sub-characteristics.* Every software product from a particular domain has some sub-characteristics that are different from other software products and those sub-characteristics, together with more generic ones, should be identified and precisely defined. Every indicator provides some information about one or several software sub-characteristics; therefore, based on the indicators defined in the previous step, software quality sub-characteristics are specified. It is not necessary that every quality sub-characteristic has only one indicator but rather a set of indicators that determines it. Thus quality sub-characteristics can be examined through several different indicators which can be combined to measure certain sub-characteristics. Finally, if needed, some quality sub-characteristics can be combined into more general ones.
   Similarly, as in the case of the hierarchy of measures, all indicators should be used for quality sub-characteristics in the model, and no quality sub-characteristic should be measured with different indicators to those already specified, nor should they be measured without any of the indicators assigned.
   A quality sub-characteristic of a web browser measured using the average startup time could be the *browser time behavior*.
6. *To align quality sub-characteristics to a quality model.* In this step, the alignment with an existing quality model is established, i.e., the software quality sub-characteristics previously defined are related to others already specified in the existing model. Depending on the domain and nature of the software product, some new quality characteristics can be specified, or existing ones can be modified or excluded.
   The sub-characteristic defined for web browsers in the previous step can be aligned to the SQuaRE's *time behavior* sub-characteristic.

5. The SemQuaRE quality model for semantic technologies

This section describes each step of the definition of SemQuaRE, a software quality model in the domain of semantic technologies, by following the bottom-up approach and starting from evaluation results.

As mentioned in Section 3, the scope of our work does not cover all types of semantic technologies and is currently limited to only those types evaluated in the SEALS project; i.e., SemQuaRE is currently restricted to ontology engineering tools, ontology matching tools, reasoning systems, semantic search tools, and semantic web service tools.

5.1. Identifying base measures

The starting point to define software quality measures has been the set of evaluations performed in the international evaluation campaigns organized in the SEALS project. In these campaigns, 41 different tools from organizations in 13 countries have participated producing evaluation results for different types of semantic technologies.[7]

For each type of technology, different evaluation scenarios were defined, using different test data as input in each scenario. In this step we

[7] http://about.seals-project.eu/downloads/category/1-.

identified the base measures of each evaluation scenario (i.e., those outputs directly produced by the software during the evaluation).

Due to space reasons, we cannot present thorough details of all evaluation scenarios. The complete overview of the quality model can be found on the SemQuaRE wiki.[8]

In this section, we just present the outcomes of each step for one concrete scenario, that of evaluating the conformance of ontology engineering tools. Conformance evaluations assess the degree in which the knowledge representation model of a tool adheres to the knowledge representation model of an ontology language according to the different ontology components (e.g., classes, properties) of such knowledge representation models.

In this case, the evaluation process is based on the IEEE 829-2008 standard [39], and the evaluation data consists of different test suites, each containing a number of different test cases to be executed. These test suites are used for evaluating the conformance of ontology engineering tools, and each of their cases contains an *Origin ontology* (O_i), which is the ontology to be used as input.

An execution of each of these test cases in a test suite consists in importing the file that contains an origin ontology (O_i) into the tool and then exporting the imported origin ontology from a test case to another ontology file (O_i^{II}), as shown in Fig. 2. The ontology (O_i^{I}) represents the ontology inside the tool, for which the state is unknown. Therefore, the comparison of two ontologies (O_i and O_i^{II}) gives an insight of the extent to which the tool observed conforms to the ontology model.

The base measures obtained after a test case execution of conformance scenario of ontology engineering tools are:

- *Final ontology*. The ontology produced by the tool when importing and exporting the origin ontology, i.e., the resulting ontology after importing and exporting the origin ontology.
- *Execution problem*. Whether there were any execution problems in the tool when importing and exporting the origin ontology.

5.2. Identifying derived measures

In this step, the base measures identified in the previous step were combined with the test data to obtain derived measures.

In the conformance scenario of ontology engineering tools, based on the test data and the base measures of one test execution, the following derived measures were specified:

- *Information added*. The information added to the origin ontology after importing and exporting it.
- *Information lost*. The information lost from the origin ontology after importing and exporting it.
- *Structurally equivalent*. Whether the origin ontology and the final one are structurally equivalent.
- *Semantically equivalent*. Whether the origin ontology and the final one are semantically equivalent.
- *Conformance*. Whether the origin ontology has been imported and exported correctly with no addition or loss of information.

The derived measures previously defined give insight into additions or losses of information that are possible due to the tool's internal ontology model and to whether the semantics is preserved.

5.3. Identifying indicators

From the derived measures in the conformance scenario of ontology engineering tools, the following indicators were obtained:

- *Ontology language component support*. Whether the tool fully supports an ontology language component. This indicator has a nominal scale, with *yes* and *no* as possible values.

[8] http://semquare.oeg-upm.net.

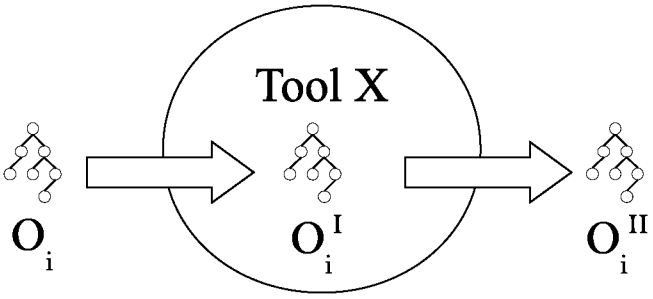

Fig. 2. Conformance test execution.

- *Ontology language component coverage*. The ratio of ontology components shared by a tool internal model and an ontology language model. This indicator has a ratio scale with values ranging from zero to one hundred, expressed in percentage.
- *Ontology information change*. The ratio of information additions or losses when importing and exporting ontologies. This indicator has a ratio scale with values ranging from zero to one hundred, expressed in percentage.

When using only the base measures, the following indicator is obtained.

- *Import/export errors*. The ratio of tool execution errors when importing and exporting ontologies. This indicator has a ratio scale with values ranging from zero to one hundred, expressed in percentage.

Due to space reasons, we have presented only the outcomes for one evaluation scenario. Similar to the example presented above, we have used the bottom-up approach to define measures for other types of tools. Table 2 summarizes the results obtained for each type of tool.[9]

5.4. Specifying relationships between measures

We have identified the relationships between measures in a formal way by defining the formulas used for obtaining all derived measures and indicators.

For example, the formulas for the *Information added* (1) and *Information lost* (2) derived measures calculate the structural differences between the origin and final ontologies in terms of triples, i.e., in terms of all the components of the two ontologies:

$$\text{final ontology} - \text{origin ontology} \tag{1}$$

$$\text{origin ontology} - \text{final ontology} \tag{2}$$

The formula for the *Structurally equivalent* (3) derived measure uses previously defined measures to determine whether there is a difference in the structure of the origin and final ontology:

$$(\text{information added} = \text{null}) \wedge (\text{information lost} = \text{null}). \tag{3}$$

The formula for the *Semantically equivalent* (4) derived measure calculates if the origin and final ontology carry the same amount of information:

$$\text{final ontology} \equiv \text{origin ontology}. \tag{4}$$

[9] As some measures are repeated across the tools, the total number of measures is different than the sum of all measures.

Table 2
Number of measures obtained for semantic technologies.

Tool type\measure	Base	Derived	Indicators
Ontology engineering tools	7	20	9
Ontology matching tools	2	5	6
Reasoning systems	11	0	16
Semantic search tools	12	8	13
Semantic web service tools	5	9	13
Total	34	41	55

Finally, the formula for the *Conformance* (5) derived measure observes the conformance of the process of importing and exporting an ontology:

$$\text{semantically equivalent} \wedge \neg \text{(execution problem)}. \tag{5}$$

Similarly, we have defined the formulas for the indicators obtained in the conformance scenario of ontology engineering tools. These formulas are the following: *Ontology language component support* (6), *Ontology language component coverage* (7), *Ontology information change* (8), *and Import/export errors* (9).

$$\frac{\#\text{ tests that contain the component where (conformance } = \text{ true)}}{\#\text{ tests that contain the component}} = 1 \tag{6}$$

$$\frac{\#\text{ components in the ontology language where (component support } = \text{ true)}}{\#\text{ components in the ontology language}} \times 100 \tag{7}$$

$$\frac{\#\text{ tests where (information added} \neq \text{null} \vee \text{information lost} \neq \text{null)}}{\#\text{ tests}} \times 100 \tag{8}$$

$$\frac{\#\text{ tests where (execution problem } = \text{ true)}}{\#\text{ tests}} \times 100 \tag{9}$$

In some cases, a measure can be obtained using more than one formula. For example, the *Ontology information change* measure could be also obtained with Formula 10.

$$\frac{\#\text{ tests where (structurally equivalent } = \text{ false)}}{\#\text{ tests}} \times 100 \tag{10}$$

Similarly, we have defined the formulas for all the derived measures and indicators identified in every evaluation scenario for the different types of semantic technologies. Furthermore, all the formulas defined are completely consistent, i.e., all the base and derived measures are used, and all the formulas contain measures already specified.

5.5. Defining domain-specific quality sub-characteristics

In this step, and starting from the indicators previously identified, we defined the set of quality sub-characteristics affected by those indicators. In some cases, we were able to reuse existing quality sub-characteristics but, in others, we had to define domain-specific ones.

In the conformance scenario of ontology engineering tools, based on the measures and analyses presented above, we have identified three quality sub-characteristics:

- *Ontology language model conformance*. The degree to which the knowledge representation model of the software product conforms to the knowledge representation model of an ontology language. It can be measured with
  - *Ontology language component coverage*
  - *Ontology language component support.*
- *Ontology processing accuracy*. The degree to which a product or system provides the correct results with the needed degree of precision when processing ontologies. It can be measured with
  - *Ontology information change.*
- *Ontology processing maturity*. The degree to which a system, product or component meets the needs for reliability while processing ontologies under normal operation. It can be measured with
  - *Import/export errors.*

Fig. 3 presents the base measures, derived measures, indicators, and quality sub-characteristics of the conformance evaluation for ontology engineering tools.

In total, we have identified fourteen semantic quality sub-characteristics. Three of them are those described for the conformance evaluation, and the others are the following:

- *Ontology language interoperability*. The degree to which the software product can interchange ontologies (importing and exporting an ontology using two different tools) and use the ontologies interchanged.
- *Ontology alignment accuracy*. The degree to which a product or system provides the correct results with the needed degree of precision when performing an ontology alignment task.
- *Reasoning accuracy*. The degree to which a product or system provides the correct results with the needed degree of precision when performing a reasoning task.
- *Semantic search accuracy*. The degree to which a product or system provides the correct results with the needed degree of precision when performing a semantic search task.
- *Semantic web service discovery accuracy*. The degree to which a product or system provides the correct results with the needed degree of precision when finding services that can be used to fulfill a given requirement from the service requester.
- *Ontology interchange accuracy*. The degree to which a product or system provides the correct results with the needed degree of precision when interchanging ontologies.
- *Ontology processing time behavior*. The degree to which the response and processing times and throughput rates of a product or system, when working with ontologies, meet requirements.
- *Reasoning time behavior*. The degree to which the response and processing times and throughput rates of a product or system, when performing a reasoning task, meet requirements.
- *Semantic search time behavior*. The degree to which the response and processing times and throughput rates of a product or system, when performing a semantic search task, meet requirements.
- *Ontology alignment time behavior*. The degree to which the response and processing times and throughput rates of a product or system, when performing an ontology alignment task, meet requirements.
- *Ontology alignment maturity*. The degree to which a system, product or component meets the needs for reliability while performing an ontology alignment task under normal operation

Besides these domain-specific quality sub-characteristics, we have identified that the following general ones (of which two come directly from SQuaRE) can also be defined for semantic technologies:

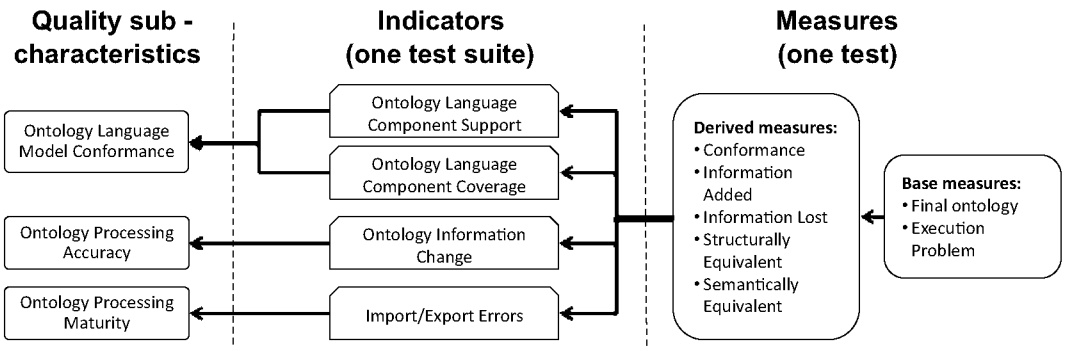

Fig. 3. Entities in the conformance scenario for ontology engineering tools.

- *Ease of use*. The degree to which a product or system is easy to operate and control by users.
- *Efficiency*. The degree of resources expended by a product or system in relation to the accuracy and completeness with which users achieve goals.
- *Satisfaction*. The degree to which users' needs are satisfied when a product or system is used in a specified context of use.

Ease of use is defined as a sub-characteristic of SQuaRE's *Operability* sub-characteristic and is related to product quality while, according to SQuaRE, *Efficiency* and *Satisfaction* are related to quality in use.

Finally, we have also identified those sub-characteristics contained in others (e.g., *Ontology alignment maturity* is a sub-characteristic of *Ontology processing maturity*).

All the quality sub-characteristics and indicators are completely consistent, i.e., all indicators are used to define quality sub-characteristics, and every quality sub-characteristic is measured using one or several indicators.

Table 3 shows all semantic quality characteristics and indicators defined for measuring them.

5.6. Aligning quality sub-characteristics with a quality model

Even though ISO 9126 is a widely adopted and used standard, it is now replaced by SQuaRE; that is why we have decided to adopt the latter (SQuaRE) for constructing the quality model for semantic technologies.

In the previous step we have identified the set of quality sub-characteristics specific to semantic technologies. In this step, all the sub-characteristics identified were properly assigned to those sub-characteristics that already exist in the SQuaRE quality model, which are as highly comprehensive as those in the ISO 9126 [1]. For example, *Reasoning time behavior* is aligned to the *Time behavior* sub-characteristic, which is a sub-characteristic from SQuaRE that, by its definition, is highly related to *Reasoning time behavior*. As suggested by Franch and Carvallo [15], we have introduced, where needed, new sub-characteristics to be aligned with existing ones. An example of this is *Functional compliance* sub-characteristic, which is introduced as a sub-characteristic of SQuaRE's *Functional suitability* characteristic.

Figs. 4 and 5 show the hierarchy of quality characteristics and sub-characteristics for semantic technologies for product quality and quality in use respectively.

6. Evaluation

Evaluation is an important phase in every engineering process [40], and this section presents the evaluation of the quality model here proposed. As there are no exact criteria to be referred to when evaluating quality models [1], the evaluation is based on similar evaluations found in the literature [1], and on criteria that we consider being the most important for our domain. Therefore, this section presents an initial evaluation of SemQuaRE with respect to three aspects: completeness, flexibility, and applicability; this evaluation is restricted to the

Table 3
Semantic quality sub-characteristics and indicators.

Semantic quality sub-characteristic	Indicators
Ontology language model conformance	Ontology language component support, ontology language component coverage
Ontology processing accuracy	Ontology information change
Ontology language interoperability	Ontology language component interoperability coverage, ontology language component interoperability support
Ontology alignment accuracy	Average alignment precision, average alignment recall, average alignment F-measure, average alignment harmonic measure
Reasoning accuracy	Class satisfiability correctness, ontology satisfiability correctness, classification correctness, entailment correctness, non-entailment correctness
Semantic search accuracy	Average search precision, average search recall, average search F-measure, number of completed queries, average number of results
Semantic web service discovery accuracy	Average number of retrieved documents, average number of relevant documents retrieved, average discovery precision, average normalized discounted cumulative gain, average normalized discounted cumulative gain at cutoff point, average binary preference, average reciprocal rank, average discovery precision at cutoff point, average discovery recall at cutoff point, average number of retrieved documents at cutoff point, average discovery precision at relevant retrieved cutoff point, average number of relevant documents retrieved at cutoff point
Ontology interchange accuracy	Interchange information change
Ontology processing time behavior	Ontology processing time, average loading time
Reasoning time behavior	Average class satisfiability time, average ontology satisfiability time, average classification time, average entailment time, average non-entailment time
Semantic search time behavior	Average query time, average time per search result, average execution time, average query input time, max query input time, average overall question time
Ontology alignment time behavior	Ontology alignment time
Ontology processing maturity	Import/export errors, ontology interchange errors, ontology satisfiability errors, class satisfiability errors, classification errors, entailment errors, non-entailment errors, average successful loads
Ontology alignment maturity	Ontology alignment errors

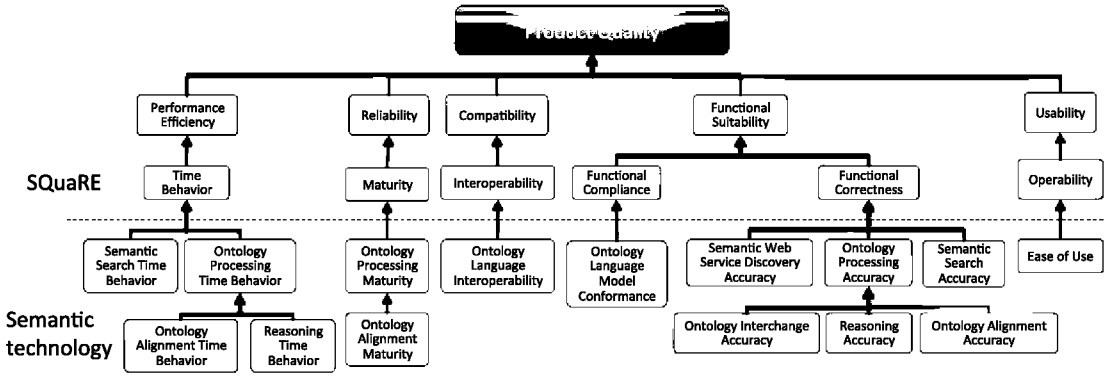

Fig. 4. Product quality characteristics and sub-characteristics of semantic technologies.

technology types currently covered by SemQuaRE. Obviously, the results of this evaluation are partial due to the current scope of SemQuaRE.

6.1. Completeness

The purpose of the completeness evaluation is to observe the inclusion of all quality characteristics in the area of study [1]. We have performed this evaluation because SemQuaRE was initially built by taking into account only the evaluation results obtained in the SEALS project. Therefore, the goal of this evaluation is twofold: first, to determine whether the quality model proposed describes those quality characteristics that are evaluated in practice when assessing the quality of semantic technologies; and second, to identify whether there are any quality characteristics evaluated that are not included in our model (which could be used to extend the quality model proposed).

In this evaluation, we have compared SemQuaRE to what we have found in the literature review that we presented in Section 3. From the literature review we can observe that, as regards the types of semantic technologies covered by SemQuaRE, the quality model that we have proposed is quite complete with respect to the current semantic technology evaluation and types of semantic technologies that it describes. Almost all the measures described in the publications conform to the quality characteristics that our model describes, and only three quality characteristics are not described in our model: time behavior of semantic web services (found in five publications), resource utilization of ontology matching tools (found in one publication) and resource utilization of reasoning systems (found in two publications).

One additional aspect of the completeness evaluation consists in comparing SemQuaRE to other semantic technology quality models. However, to the best of our knowledge, SemQuaRE is the only example of a quality model for semantic technologies.

6.2. Flexibility

As discussed in the introduction, due to the existence of tools that share functionalities of different types, it is important to have a unique quality model for all types of semantic technologies. As currently

Fig. 5. Quality in use characteristics of semantic technologies.

SemQuaRE does not include every type of technology or every quality characteristic for the current technology types it covers, the purpose of this evaluation is to observe how flexible the quality model proposed is in terms of its extension with new types of semantic technologies and with new characteristics.

As mentioned in the previous section, some measures found in the literature review were not observed by SemQuaRE; therefore we have defined new quality sub-characteristics for them. These are:

- *Semantic web service time behavior.* The degree to which the response and processing times and throughput rates of a product or system, when performing a semantic web service discovery task, meet the requirements.
- *Ontology alignment resource utilization.* The degree to which the amounts and types of resources used by a product or system, when performing ontology alignment task, meet the requirements.
- *Reasoning resource utilization.* The degree to which the amounts and types of resources used by a product or system, when performing reasoning task, meet the requirements.

These sub-characteristics have been completely included into SemQuaRE, together with their related set of quality measures and their formulas.

In order to further evaluate the flexibility of the SemQuaRE, we have analyzed the evaluations of a type of semantic technology, that of ontology annotation tools, that is not included in our model. The goal was to determine whether SemQuaRE is flexible enough so that evaluation results for other types of semantic technologies can be included.

We have analyzed the publications of the proceedings of the International Semantic Web Conference and European/Extended Semantic Web Conference, which in total were five publications related to ontology annotation tool evaluation. In four publications, the authors evaluate precision; three publications report results for recall; one publication provides results for normalized discounted cumulative gain and F-measure; and one publication provides results for correctness. All measures found are used to measure accuracy and, therefore, a new sub-characteristic, namely *Ontology annotation accuracy*, has been introduced into SemQuaRE. This sub-characteristic has been aligned with SQuaRE's *Functional correctness* quality sub-characteristic.

Table 4 shows quality sub-characteristics and quality measures added to SemQuaRE during the flexibility evaluation. After this evaluation, one new type of semantic technology, that of ontology annotation tools, has been added to the model, with one quality sub-characteristic, four indicators, four derived measures, and one base measure. Furthermore, with respect to semantic technologies that were already described in SemQuaRE, three new quality characteristics have been added, together with seven indicators and seven base measures.

Table 4
Semantic quality sub-characteristics added to SemQuaRE during the flexibility evaluation.

Quality sub-characteristic	Indicators	Derived measures	Base measures
Semantic web service time behavior	Average discovery time		Discovery time
Ontology alignment resource utilization	Ontology alignment memory consumption		Memory consumption
Reasoning resource utilization	Average classification memory consumption, average ontology satisfiability memory consumption, average class satisfiability memory consumption, average entailment memory consumption, average non-entailment memory consumption		Classification memory consumption, ontology satisfiability memory consumption, class satisfiability memory consumption, entailment memory consumption, non-entailment memory consumption
Ontology annotation accuracy	Average annotation precision, average annotation recall, average annotation F-measure, average NDCG	Annotation precision, annotation recall, annotation F-measure, normalized discounted cumulative gain	Output annotations

Fig. 6 shows the product quality part of SemQuaRE after flexibility evaluation. The quality in use part was not affected.

6.3. Applicability

The main purpose of software quality models is to provide consistent specification and guidelines for software evaluations; therefore, an important aspect of the quality model evaluation is its applicability.

Besides describing details about quality characteristics, SemQuaRE also describes a complete hierarchy of quality measures that can be used for assessing the quality of semantic technologies. Furthermore, a formula is specified for each measure in the hierarchy; such a formula can be used to obtain concrete results in the evaluations.

The SEALS European project has provided semantic technology evaluation results in two international evaluation campaigns. Those evaluation results are completely consistent with the quality model proposed, i.e., the evaluation results are obtained according to the guidance provided in the quality model. As an example, we can observe the evaluation results for ontology engineering tools [41], which show that the quality model proposed has been successfully applied in practice. SEALS also provides an independent platform for semantic technology evaluation to be used by any company or individual. The current evaluation services deployed in this platform produce evaluation results which are completely consistent with SemQuaRE, which can be used for comparing or benchmarking tools that are evaluated separately.

Furthermore, within its scope, the quality model proposed has been used as a basis of a semantic technology recommendation framework [42]. The recommendation framework has been implemented into a web application[10] which allows users to specify quality requirements in terms of the quality model (thus helping them to tailor the recommendations to their needs) and to get recommendations consisting of a tool or a set of different tools.

For example, a user might be interested in an ontology engineering tool that covers, at least, half of the ontology language features, and that does not make significant changes in information when importing/exporting ontologies. The user is able to specify those requirements in terms of an indicator and a desired threshold, which in this case are *Ontology language component coverage* with a threshold of 50%, and *Ontology information change* with a threshold of 20%. Based on these requirements, the recommendation framework can present the user a list of ontology engineering tools ordered according to how well each tool meets the requirements.

7. Conclusions and future work

This paper presents SemQuaRE, a quality model for the evaluation of semantic technologies, which extends the SQuaRE software quality model. The SemQuaRE quality model provides a framework for the evaluation and comparison of semantic technologies, and it is particularly significant for the community in the following aspects:

- It is a first step towards a consistent terminology for semantic technology quality, even if it is limited to its current scope.
- Within such scope, it gives comprehensive definitions of all its elements, i.e., quality characteristics and measures.
- Although some problems with the ISO quality models have been identified (as described by Al-Kilidar et al. [31]), SemQuaRE has introduced quality measures specific to semantic technologies; it has also specified formulas for all derived measures and indicators, which results in reducing ambiguities in the model and provides a detailed guidance of how to evaluate and measure the quality of semantic technologies.
- It serves developers of new semantic tools as a checklist for semantic technology quality requirements.

SemQuaRE has been built using the bottom-up methodology described in Section 4. Although we have not found any bottom-up approach for extending quality models in the literature, some authors suggest that a bottom-up approach is important when building quality models [37], and we share such opinion. The bottom-up approach that we have used is not validated, so in the future we plan to perform an evaluation and provide details about the validation and usefulness of such approach.

During the evaluation process, we have concentrated only on the literature review, i.e., limited venues that include the most relevant conferences and workshops in the semantic field and limited tool types. Although some venues were implicitly included in our study (some editions of the Ontology Alignment Evaluation Initiative[11] and the Semantic Web Service Challenge[12] have been addressed in the SEALS project and in the IWEST workshop series), we plan to extend SemQuaRE by analyzing other evaluation campaigns (e.g., the Semantic Web Challenge,[13] the Triplification Challenge,[14] or the Linked Data Cup[15]) and relevant journals, as well as by including other types of tools in order to get a more complete quality model.

SemQuaRE is a hierarchical quality model, as those of the ISO 9126 and the SQuaRE standards are, and this is regarded as an important factor for clear and unambiguous quality models [1]. Furthermore, Bertoa et al. [14] argue that the structure and organization of the model influence its understandability and, in this sense, we can assume that SemQuaRE provides a high degree of understandability to its users; however, we plan to evaluate the understandability of SemQuaRE in future work.

One line of future work includes building an ontology to describe SemQuaRE. Such an ontology could be used by developers as an

[10] The recommendation system is available at http://www.development.seals-project.eu/seals-recommendation/.

[11] http://oaei.ontologymatching.org/.
[12] http://sws-challenge.org/wiki/.
[13] http://challenge.semanticweb.org/.
[14] http://triplify.org/Challenge.
[15] http://i-semantics.tugraz.at/i-challenge/call-for-submissions.

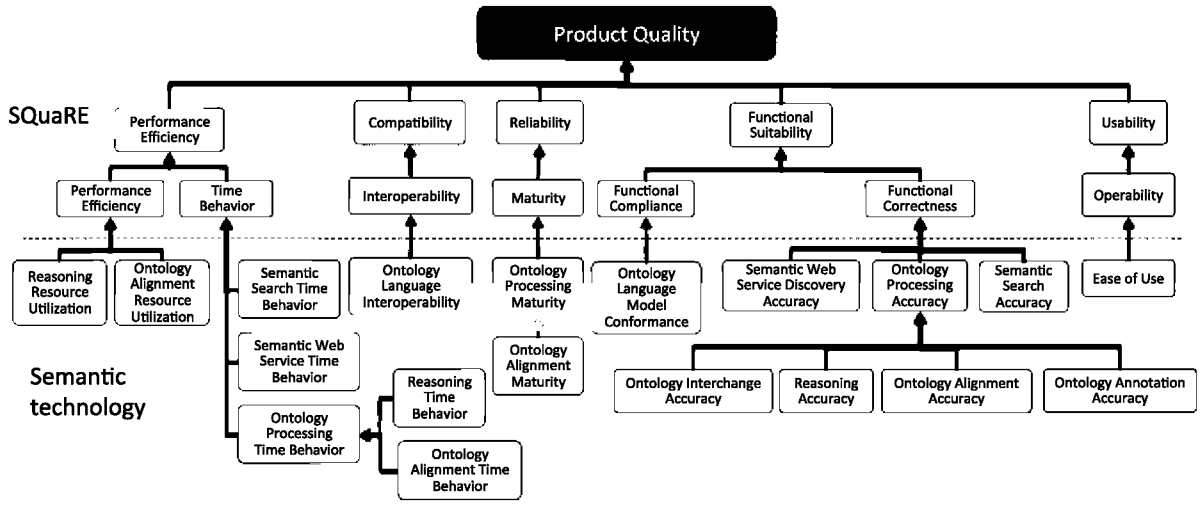

Fig. 6. Product quality of SemQuaRE upon completing the evaluation process.

information model and could take the form of a machine-readable artifact easily exploited by software in various applications.

Acknowledgments

This work has been supported by the SEALS European project (FP7-238975), by the EspOnt project (CCG10-UPM/TIC-5794) co-funded by the Universidad Politécnica de Madrid and the Comunidad de Madrid, and by an FPU grant (FPU2012/04084) of the Spanish Ministry of Education Culture and Sport.

Thanks to Rosario Plaza for reviewing the grammar of this paper.

References

[1] B. Behkamal, M. Kahani, M. Akbari, Customizing ISO 9126 quality model for evaluation of B2B applications, Inf. Softw. Technol. 51 (2009) 599–609.
[2] R. García-Castro, A. Gómez-Pérez, Óscar Muñoz-García, L. Nixon, Towards a component-based framework for developing Semantic Web applications, Proceedings of the 3rd Asian Semantic Web Conference, Springer, Bangkok, Thailand, 2008, pp. 197–211.
[3] OntoWeb, D1.3: a survey on ontology tools, Technical Report, IST OntoWeb Thematic Network, 2002.
[4] Y. Guo, Z. Pan, J. Heflin, LUBM: a benchmark for OWL knowledge base systems, Web Semant. Sci. Serv. Agents World Wide Web 3 (2005) 158–182.
[5] P. Lambrix, M. Habbouche, M. Perez, Evaluation of ontology development tools for bioinformatics, Bioinformatics 19 (2003) 1564–1571.
[6] R. García-Castro, A. Gómez-Pérez, Interoperability results for Semantic Web technologies using OWL as the interchange language, Web Semant. Sci. Serv. Agents World Wide Web 8 (2010) 278–291.
[7] C. Tempich, R. Volz, Towards a benchmark for Semantic Web reasoners – an analysis of the DAML ontology library, in: Proceedings of the 2nd International Workshop on Evaluation of Ontology-Based Tools, Sanibel Island, Florida, USA, 2003.
[8] F. Baader, M. Knechtel, R. Peñaloza, A generic approach for large-scale ontological reasoning in the presence of access restrictions to the ontology's axioms, Proceedings of the 8th International Semantic Web Conference, Chantilly, VA, USA, 2009, pp. 49–64.
[9] B. Glimm, I. Horrocks, B. Motik, G. Stoilos, Optimising ontology classification, Proceedings of the 9th International Semantic Web Conference, Shanghai, China, 2010, pp. 225–240.
[10] G. Meditskos, N. Bassiliades, Combining a DL reasoner and a rule engine for improving entailment-based OWL reasoning, Proceedings of the 7th International Semantic Web Conference, Karlsruhe, Germany, 2008, pp. 277–292.
[11] J. Urbani, S. Kotoulas, E. Oren, F. Harmelen, Scalable distributed reasoning using mapreduce, Proceedings of the 8th International Semantic Web Conference, Chantilly, VA, USA, 2009, pp. 634–649.
[12] ISO, ISO/IEC 25010:2011, Systems and software engineering – systems and software quality requirements and evaluation (square) – system and software quality models, Technical Report, International Organization for Standardization, 2011.
[13] P. Botella, X. Burgués, J. Carvallo, X. Franch, G. Grau, J. Marco, C. Quer, ISO/IEC 9126 in practice: what do we need to know? Proceedings of the 1st Software Measurement European Forum, 2004.
[14] M.F. Bertoa, J.M. Troya, A. Vallecillo, Measuring the usability of software components, J. Syst. Softw. 79 (2006) 427–439.
[15] X. Franch, J. Carvallo, Using quality models in software package selection, Software, IEEE 20 (2003) 34–41.
[16] P. Botella, X. Burgués, J. Carvallo, X. Franch, J. Pastor, C. Quer, Towards a quality model for the selection of ERP systems, Component-Based Software Quality 2693 (2003) 225–245.
[17] R. García-Castro, M. Esteban-Gutiérrez, A. Gómez-Pérez, Towards an infrastructure for the evaluation of semantic technologies, Proceedings of the eChallenges e-2010 Conference, Warsaw, Poland, IEEE, 2010, pp. 1–7.
[18] ISO, ISO/IEC 9126-1:2001, Software engineering – product quality – part 1: quality model, Technical Report, International Organization for Standardization, 2001.
[19] J. Cavano, J. McCall, A framework for the measurement of software quality, ACM SIGMETRICS Performance Evaluation Review 7 (1978) 133–139.
[20] B. Boehm, J. Brown, M. Lipow, Quantitative evaluation of software quality, Proceedings of the 2nd International Conference on Software Engineering, San Francisco, USA, 1976, pp. 592–605.
[21] M. Azuma, SQuaRE: the next generation of the ISO/IEC 9126 and 14598 international standards series on software product quality, European Software Control and Metrics Conference (ESCOM), London, UK, 2001, pp. 337–346.
[22] ISO, ISO/IEC 15939:2007, Systems and software engineering – measurement process, Technical Report, International Organization for Standardization, 2007.
[23] F. Domínguez-Mayo, M. Escalona, M. Mejías, M. Ross, G. Staples, Quality evaluation for model-driven web engineering methodologies, Inf. Softw. Technol. 54 (2012) 1265–1282.
[24] F. Garcia, M.F. Bertoa, C. Calero, A. Vallecillo, F. Ruiz, M. Piattini, M. Genero, Towards a consistent terminology for software measurement, Inf. Softw. Technol. 48 (2006) 631–644.
[25] B. Kitchenham, Procedures for performing systematic reviews, Joint Technical Report Software Engineering Group, Department of Computer Science Keele University (UK) and Empirical Software Engineering, National ICT Australia, 2004. 33.
[26] X. Niu, H. Wang, G. Wu, G. Qi, Y. Yu, Evaluating the stability and credibility of ontology matching methods, Proceedings of the 8th Extended Semantic Web Conference, Heraklion, Springer, Crete, Greece, 2011, pp. 275–289.
[27] A.N. Chitnis, A. Qasem, J. Heflin, Benchmarking reasoners for multi-ontology applications, in: Proceedings of the 5th International Workshop on Evaluation of Ontologies and Ontology-Based Tools, Busan, Korea, 2007.
[28] M. Yatskevich, I. Horrocks, F. Martin-Recuerda, G. Stoilos, Storage and reasoning systems evaluation campaign 2010, in: Proceedings of the International Workshop on Evaluation of Semantic Technologies, IWEST, Shanghai, China, 2010.
[29] U. Küster, B. König-Ries, Measures for benchmarking semantic web service matchmaking correctness, Proceedings of the 7th Extended Semantic Web Conference, Springer, Heraklion, Crete, Greece, 2010, pp. 45–59.
[30] C. Trojahn, C. Meilicke, J. Euzenat, H. Stuckenschmidt, Automating OAEI campaigns (first report), in: Proceedings of the International Workshop on Evaluation of Semantic Technologies, IWEST, Shanghai, China, 2010.
[31] H. Al-Kilidar, K. Cox, B. Kitchenham, The use and usefulness of the ISO/IEC 9126 quality standard, Proceedings of the 2005 International Symposium on Empirical Software Engineering, Noosa Heads, Australia, 2005, pp. 126–132.
[32] J. Carvallo, X. Franch, C. Quer, Defining a quality model for mail servers, Proceedings of the 2nd International Conference on COTS-based Software Systems, (ICCBSS 2003), Ottawa, Springer-Verlag New York Inc., Ottawa, 2003, p. 51.
[33] H. Zulzalil, A. Ghani, M. Selamat, R. Mahmod, A case study to identify quality attributes relationships for Web-based applications, Int. J. Comput. Sci. Network Secur. 8 (2008) 215.
[34] I. Padayachee, P. Kotze, A. van Der Merwe, ISO 9126 external systems quality characteristics, sub-characteristics and domain specific criteria for evaluating e-Learning systems, The Southern African Computer Lecturers' Association, University of Pretoria, South Africa, 2010.
[35] W. Abramowicz, R. Hofman, W. Suryn, D. Zyskowski, SQuaRE based web services quality model, Proceedings of The International Multi-Conference of Engineers and Computer Scientists, Hong Kong, 2008, pp. 827–835.

[36] M. Bombardieri, F. Fontana, A specialisation of the SQuaRE quality model for the evaluation of the software evolution and maintenance activity, Proceedings of the Automated Software Engineering Workshops, L'Aquila, Italy, IEEE, 2008, pp. 110–113.
[37] R. Dromey, Software Product Quality: Theory, Model, and Practice, Software Quality Institute, Brisbane, Australia, 1998.
[38] F. Radulovic, R. García-Castro, Extending software quality models — a sample in the domain of semantic technologies, Proceedings of the 23rd International Conference on Software Engineering and Knowledge Engineering (SEKE 2011), Miami, USA, 2011, pp. 25–30.
[39] IEEE, IEEE 829-2008, Standard for software and system test documentation, Technical Report, Institute of Electrical and Electronics Engineers, 2008.
[40] E. Dustin, J. Rashka, D. McDiarmid, Quality Web Systems: Performance, Security, and Usability, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2002.
[41] R. García-Castro, M. Yatskevich, C.T. dos Santos, S.N. Wrigley, L. Cabral, L. Nixon, O. Šváb-Zamazal, The State of Semantic Technology Today — Overview of the First SEALS Evaluation Campaigns, Technical Report, SEALS Consortium, 2011.
[42] F. Radulovic, R. García-Castro, Semantic technology recommendation based on the analytic network process, Proceedings of the 24th International Conference on Software Engineering and Knowledge Engineering (SEKE2012), Redwood City, California, USA, 2012, pp. 611–616.