TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 0?/?? pp???-???
Volume 22, Number 3, June 2017

Quality Model and Metrics of Ontology for Semantic Descriptions of Web Services

Hong Zhu*, Dongmei Liu, Ian Bayley, Arantza Aldea, Yunfei Yang, and Ying Chen

Abstract: An ontology is a conceptualisation of domain knowledge. It is employed in semantic web services technologies to describe the meanings of services so that they can be dynamically searched for and composed according to their meanings. It is essential for dynamic service discovery, composition, and invocation. Whether an ontology is well constructed has a tremendous impact on the accuracy of the semantic description of a web service, the complexity of the semantic definitions, the efficiency of processing messages passed between services, and the precision and recall rates of service retrieval from service registrations. However, measuring the quality of an ontology remains an open problem. Works on the evaluation of ontologies do exist, but they are not in the context of semantic web services. This paper addresses this problem by proposing a quality model of ontology and defining a set of metrics that enables the quality of an ontology to be measured objectively and quantitatively in the context of semantic descriptions of web services. These metrics cover the contents, presentation, and usage aspects of ontologies. The paper also presents a tool that implements these metrics and reports a case study on five real-life examples of web services.

Key words: semantic web services; ontology; quality model; ontology evaluation; ontology metrics

1 Introduction

An ontology is an explicit specification of a conceptualisation[2]. It thereby represents knowledge of a specific application domain. Most ontologies in computer science and information technology contain a vocabulary that represents the concepts of a specific domain as classes, individuals as instances of concepts, attributes as properties of the objects and classes, and relationships between the classes as relations. Ontologies are employed in semantic web services technologies to describe the meanings of services. This is true both for the so-called Big Web Services[3], based on WSDL, SOAP, UDDI, and for the RESTful Web Services[4], built directly on the HTTP protocol. In both approaches, the vocabulary defined in an ontology of the application domain is used to annotate on the service operations and their input and output parameters. When services are registered and published with such machine readable semantic descriptions, they can be dynamically searched for and composed according to their meanings. The messages passed between services, such as service requests and service responses, can also be correctly interpreted by machines at run time. Thus, ontologies play a fundamental role in semantic web services.

Languages for defining ontologies and their uses in

• Hong Zhu, Ian Bayley, and Arantza Aldea are with the Department of Computing and Communication Technologies, Oxford Brookes University, Oxford OX33 1HX, UK. E-mails: hzhu@brookes.ac.uk; ibayley@brookes.ac.uk; and aaldea@brookes.ac.uk.

• Dongmei Liu, Yunfei Yang, and Ying Chen are with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China. E-mail: dmliukz@njust.edu.cn.

∗ To whom correspondence should be addressed.

Manuscript received: 2016-07-07; revised: 2016-10-30; accepted: 2016-12-20

Note. This paper is an extended, corrected, and revised version of the conference paper[1].

the semantic descriptions of web services have been much researched in the past decade. Among the most well-known are OWL-S[5], MicroWSMO/hRESTS[6], WADL[7], SA-REST[8], and WSML[9], etc. However, far less research has gone into evaluating the quality of such ontologies[10] even though this is of great importance to service discovery, composition and interaction. In particular, whether an ontology is well constructed has tremendous impact on the accuracy of the semantic description of a web service, the complexity of the semantic definitions, the efficiency of processing messages passed between services, and the precision and recall rates of service retrieval from service registrations. The evaluation of the quality of ontologies has been studied in the literature, see Ref. [10] for recent surveys. However, little research has been conducted in the context of web services. This paper addresses the open problem of ontology quality evaluation with a metrics-based framework.

The rest of this paper is organised as follows. Section 2 presents a quality model of ontology, Section 3 presents a set of metrics based on that, and Section 4 introduces the tool that implements these metrics. Section 5 uses the tool to apply the metrics to a case study. Section 6 compares the paper with related work, summarises the main contributions, and concludes.

2 Quality Model of Ontologies

The concept of quality in software engineering is notably elusive[11]. Since 1970s, numerous quality models have been proposed in the literature to define the relevant properties of software quality, the relationships between various factors and attributes of software quality, and metrics for measuring them either qualitatively or quantitatively. This section proposes a quality model of ontologies, while the metrics on various quality attributes will be defined in the next section.

2.1 Overview of the quality model

Existing software quality models focus on the executable source code and on the processes surrounding its development, evolution, and maintenance. They are not applicable directly to ontologies because the structure and functionality of ontologies are fundamentally different from programs, and the development, evolution, and maintenance processes for ontologies are also fundamentally different. It is highly desirable to develop quality models for ontologies.

Existing software quality models fall into two types: *hierarchical* and *relational*[12].

A *hierarchical model* defines several aspects of software quality, then decomposes each aspect into a number of factors, each of which is in turn decomposed into a number of attributes, which can sometimes be measured by several metrics. One typical example is the quality model for object-oriented design[13]. Another example is the quality model in the ISO/IEC 25010 standard[14]. In general, in a hierarchical quality model, more abstract qualities are higher up the tree, whereas more concrete qualities, some even directly measurable, are lower down. Each quality has a positive contribution to its parent in the sense that an improvement in the child implies an improvement in the parent.

A *relational model*, in contrast, also defines negative and neutral relations between quality properties. If two quality attributes are related to each other *negatively* then that means an improvement in one attribute implies a deterioration in the other. Or, if the relationship is *neutral*, an improvement in one does not necessarily imply any change to the other. Such models have more relationship information and are therefore more accurate. Examples include Perry's quality model of information systems[15] and Zhang's quality model of web-based information systems[16].

The model proposed in this paper is hierarchical. In particular, the quality attributes are divided into three aspects: *contents*, *presentation*, and *usage*. For each aspect, the attributes and factors that influence it are identified and defined. Not all of quality attributes and factors are measurable, but for those that are, metrics are defined. Many of them have been proposed, studied, and used in the literature, as discussed in Section 6.1, but quite a few of them are new ones for evaluating ontology in the context of semantic web services. The quality model and the metrics proposed in this paper are based on knowledge gleaned from the literature and on our empirical understanding too, but validation of them is beyond the scope of the paper.

The following discusses each of the aspects in turn, their attributes and the factors influencing those attributes. The relationships between these attributes and factors are discussed at the end of the section.

2.2 Contents

The contents aspect has quality attributes that focus on the contents of the ontology and, in particular, on how

well it represents the subject domain. There are two such quality attributes:

- *Correctness*: whether the ontology accurately represents the knowledge of a subject domain.
- *Completeness*: whether the ontology represents all of the knowledge of a subject domain.

These attributes are both relative to human knowledge. To make evaluations and comparisons possible when examining ontology contents, a standard is needed. In existing work on evaluating ontologies, such a standard is often presented in the form of an ontology of the same application domain, and called *gold standard*[10,17]. It acts as the ideal representation of the domain knowledge.

To evaluate an ontology's correctness, one can check whether its contents, such as concepts, relations, attributes, and instances, are compatible with a given gold standard, so compatibility is one factor.

- *Compatibility*: whether the ontology contains junk, i.e., contents not in the subject domain.

Another factor is consistency, of which there are two types.

- *Internal consistency*: whether there is no self-contradiction within the ontology.
- *External consistency*: whether the ontology is consistent with the subject domain knowledge.

The internal consistency of an ontology cannot be measured because it is non-numerical and plays the role of a pre-condition for all future analysis. External consistency can only be judged relative to a given standard. In that case, all concepts, relationships between concepts, attributes of concepts, and instances of concepts must be defined to be the same as in the standard. We will define metrics for each of these in the next section.

Compatibility is also relative to a standard and is judged by calculating the number of elements that cannot be derived from the ontology to which it is being compared.

Completeness is assessed by determining the number of elements in the standard that are covered by the ontology. Two types are recognised:

- *Syntactic completeness*: how much the vocabulary of the ontology matches exactly that of the standard;
- *Semantic completeness*: how much the vocabulary of the standard can be derived from the ontology.

Both of these can be measured by metrics, as will be seen in the next section.

2.3 Presentation

The presentation aspect has quality attributes related to the way in which the ontology presents the domain knowledge. The main ones are:

- *Well formedness*: syntactic correctness with respect to the rules of the language in which it is written. Like internal consistency, it is a precondition to all further analysis.
- *Conciseness*. The key attribute for this is the lack of *redundancy* within the ontology, where an element is redundant if it can be derived from other elements of the ontology. This can also be measured by metrics.
- *Structural complexity*. This is particularly important when the ontology has redundancy. This is because redundancy often helps to improve usability but since each redundancy increases the complexity in different ways, it is useful to measure complexity and choose the least complex ontology. The two key factors are:
  - *Size*: in terms of the number of elements;
  - *Relation*: in terms of the number of relationships between elements
- *Modularity*: how well the ontology is decomposed into smaller parts, to make it easier to understand, use, and maintain. These smaller parts, or modules, form ontologies of subdomains and may themselves refer to other modules. The two factors are:
  - *Cohesion*: The degree of interaction within one module or ontology,
  - *Coupling*: The degree of cross-referencing between different modules or ontologies.

There are metrics for both of these and the ideal is high cohesion and low coupling.

2.4 Usage

The usage aspect has quality attributes that manifest themselves when the ontology is used. They measure how easy it is to use the ontology. This is the most elusive of the three aspects, in part because ontologies can be used for many different purposes, such as annotating meta-data on web pages and informing intelligent search engines. Here, our focus is on the use of ontology to describe web service semantics.

The most important quality attribute is *applicability*. This relates to whether the ontology is easy to apply for a specific task, which in this case is the description of web service semantics. A service consists of a

number of operations that inspect and/or modify the system state. A semantic service description describes the meanings of those states, service requests, service responses, and operations in the vocabulary of the ontology. In this context, two important factors are identified, both of which are measurable.

- *Definability*: whether the states and functions of the services can be defined;
- *Description complexity*: how complex are those descriptions if they are definable.

The following three other attributes can be affected by factors not actually related to the ontology, as will be seen later. This is another reason why the usability aspect is elusive.

- *Adaptability*: how easily the ontology can be changed to meet the specific purposes of developing a particular web service. There are several different factors due to the different ways in which an ontology can be changed to fulfil a specific task.
  - *Tailorability*: how easily a subset of the contents can be removed if it is irrelevant to the specific application;
  - *Composibility*: how easily it can be combined with another ontology;
  - *Extendability*: how easily extra concepts can be added;
  - *Transformability*: how easily it can be changed to a different form.
- *Efficiency*: how easily semantic information can be processed for various purposes. Services must be searched for in a registry, composed either statically or dynamically and then invoked using the service requests and responses encoded in the ontology. The different factors, each a type of efficiency, are therefore:
  - *Search and discovery efficiency;*
  - *Composition efficiency;*
  - *Invocation efficiency.*
- *Comprehensibility*: whether human readers can easily understand the semantic description.

Adaptability and comprehensibility are influenced by human factors, which are independent of the ontology itself. Adaptability is affected by the user's familiarity with the ontology. Comprehensibility is equally dependent on the person and cannot be measured except indirectly through definability and complexity metrics. As for processing efficiency, this depends both on implementation details and on the choice of hardware and software platforms.

To summarise, the quality model for ontology is depicted in Fig. 1. The metrics associated to the quality attributes and factors are defined in Section 3.

2.5 Relationship between quality attributes and factors

It is apparent that both content attributes and presentation attributes have affect on usage attributes.

Taking applicability in particular, the definability factor can be affected by completeness and the description complexity factor can be affected by conciseness, structural complexity, and modularity. These may also affect all the factors of the adaptability attribute. In the case of conciseness (a:k:a:, non-redundancy) this would, on the other hand, reduce the complexity of service description.

Taking efficiency, on the other hand, it seems possible that this would be reduced if the structural complexity was higher. It is not clear at all whether redundancy (i.e., low conciseness) would be good or bad for processing efficiency. Further empirical study is needed to answer this and further questions, but such a study is beyond the scope of this paper.

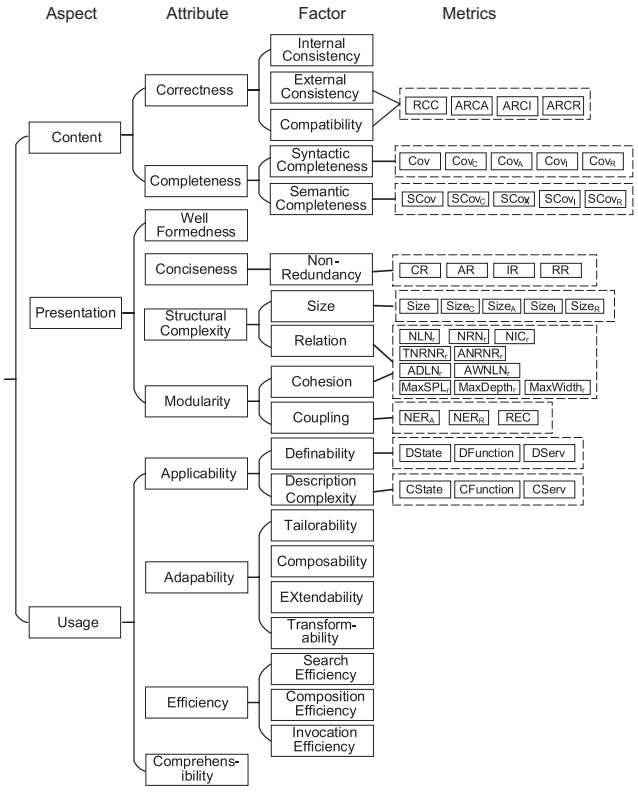


Fig. 1 The quality model of ontology.

3 Metrics of Ontology

This section defines a collection of metrics of ontologies. In order to make the definitions of metrics independent of the ontology specification language, and thus generally applicable, an abstract model of ontologies is formally defined first based on the structural similarities shared by all ontologies.

3.1 Abstract model of ontologies

Definition 1 (Domain ontology)

A domain ontology **O** is a tuple (C, I, A, R), where

(1) C is a finite set of classes. Each element $c \in C$ represents a concept of the domain.

(2) $I = \{I^c | c \in C\}$ is a collection of finite sets indexed by C. Each $\alpha \in I^c$ is an instance of the concept c.

(3) $A = \{A^c | c \in C\}$ is a collection of finite sets of attributes. For each $c \in C$, $\varphi \in A^c$ is an attribute of concept c. The value of an attribute φ for an instance $\alpha \in I^c$ of concept c, denoted by $\varphi(\alpha)$, is either a data value of type T or an instance of a class c'. In the former, φ is a *data property* and in the latter, it is an *object property*. In both cases, we say that c and T (or c') are the domain and codomain of attribute φ, and write $\varphi : c \to T$ (or $\varphi : c \to c'$).

(4) $R = \{r_1, \cdots, r_k\}$ is a finite set of binary relations on the set of concepts. For each $r \in R$ and $r \subseteq C \times C$, we have that $(c, c') \in r$ means that concept c is related to c' by r. □

Two examples of relations that are particularly widely used are the *is-a* and *has-a* relations, defined as follows:

- $(c, c') \in$ *is-a* means that each instance α of concept c is also an instance of concept c', i.e. $\forall \alpha \in I^c \Rightarrow \alpha \in I^{c'}$. The *is-a* relation is also called the *sub-super* relation, or *inheritance* relation between concepts.
- $(c, c') \in$ *has-a* means that for each instance α of concept c, there is an instance α' of class c' such that α' is a part of α. The *has-a* relation is also called the *whole-part* relation between concepts.

It is worth noting that there are other types of relations between concepts. For example, in the English lexical database WordNet, the *member-of* and *substance-of* relations are commonly used between nouns, in addition to the *is-a* and *has-a* relations.

Figure 2 shows an example of an ontology of people and family relationships. It will be used as the running example throughout the paper. Here, classes are depicted in solid line boxes, data types in solid line boxes with rounded corners, and instances in dotted boxes. If a concept referred to is external to the ontology then a grey box is used. The *is-a* and *has-a* relations are represented with unfilled and filled arrows respectively. Arrows representing attributes are labelled with the name of the attribute and depicted in red for data properties and green for object properties.

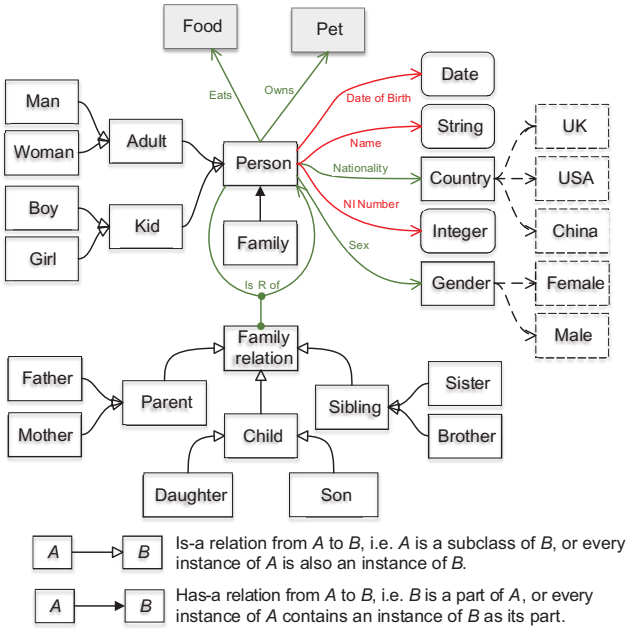


Fig. 2 Example of ontology: *Family*.

We now define a set of metrics on ontologies. They are classified according to the aspects of the ontology that they measure. In the sequel, we use $O = (C, I, A, R)$ to denote a given ontology to be measured.

3.2 Vocabulary

The vocabulary of an ontology is the set of names defined in it, comprising the names given to the concepts, instances, attributes, and relations. The following defines a set of basic metrics for the sizes of an ontology on various aspects.

Definition 2 The sizes of an ontology are defined as follows.

$$\text{Size}_C(O) = ||C||, \qquad \text{Size}_I(O) = \sum_{c \in C} ||I^c||,$$

$$\text{Size}_A(O) = \sum_{c \in C} ||A^c||, \quad \text{Size}_R(O) = \sum_{r \in R} ||r||,$$

$$\text{Size}(O) = \text{Size}_C(O) + \text{Size}_I(O) + \text{Size}_A(O) + \text{Size}_R(O).$$ □

An ontology's coverage of an application domain

is often evaluated against another ontology, which is called *gold standard* in Ref. [17]. The following metrics measure the domain coverage of an ontology O with respect to a second ontology $\Omega = (C', I', A', R')$.

Definition 3 (Vocabulary Coverage)

Ontology O's vocabulary coverage with respect to Ω on O's constituent parts is defined as follows.

$$\begin{aligned}
&\mathrm{Cov}_C^{\Omega}(O) = S_C/\mathrm{Size}_C(\Omega),\\
&\mathrm{Cov}_I^{\Omega}(O) = S_I/\mathrm{Size}_I(\Omega),\\
&\mathrm{Cov}_A^{\Omega}(O) = S_A/\mathrm{Size}_A(\Omega),\\
&Cov_R^{\Omega}(O) = S_R/\mathrm{Size}_R(\Omega),\\
&\mathrm{Cov}^{\Omega}(O) = \frac{(S_C + S_I + S_A + S_R)}{\mathrm{Size}(\Omega)},
\end{aligned}$$

where

$$S_C = ||C \cap C'||, \qquad S_I = \sum_{c \in C} ||I^c \cap I'^c||,$$

$$S_A = \sum_{c \in C} ||A^c \cap A'^c||, \quad S_R = \sum_{r \in R} ||r \cap r'||,$$

where r' is the relation corresponding to r in R'. □

Example 1 Let Ω be the ontology *Family*, and O be *Family* with the class *Pet* removed. Then, O's vocabulary coverages with respect to Ω are as follows.

$$\begin{aligned}
&S_C = 21,\ \mathrm{Size}_C(\Omega) = 22,\ \mathrm{Cov}_C^{\Omega}(O) = 21/22,\\
&S_I = 5, \quad \mathrm{Size}_I(\Omega) = 5, \quad \mathrm{Cov}_I^{\Omega}(O) = 1,\\
&S_A = 7, \quad \mathrm{Size}_A(\Omega) = 8, \quad \mathrm{Cov}_A^{\Omega}(O) = 7/8,\\
&S_R = 16,\ \mathrm{Size}_R(\Omega) = 16,\ \mathrm{Cov}_R^{\Omega}(O) = 1,\\
&Cov^{\Omega}(O) = 49/51.
\end{aligned}$$

□

These coverage metrics measure the proportion of classes, instances, attributes, and relations that are defined directly. However, an ontology may define only the key components and leave out the other components definable from the basic ones. Such coverage will be investigated in Section 3.4.

3.3 Structure

Structural metrics are among the most widely explored metrics and of these, in particular, cohesion metrics measure the degree of relatedness between concepts in an ontology and coupling metrics measure the interactions across different ontologies.

3.3.1 Cohesion metrics

We construct a directed graph for each relation separately and measure the relatedness and complexity of the ontology relative to that, using the notions of graph theory.

Definition 4 (Graph on Relation)

Let $r \in R$ be any given relation of ontology O. The graph of O on relation r, denoted by $\mathrm{Graph}_r(O)$, is a directed graph $G = (N, E)$, in which the nodes $n \in N$ are concepts of the ontology, and the edges $e \in E$ are the r-relationships between the concepts, i.e., there is an edge $e \in E$ from node c to c', if and only if $(c, c') \in r$. □

The graph for the *is-a* relation of Ontology *Family* is shown in Fig. 3.

Before defining metrics on the ontology, we will first recap a few graph theory notions that will be needed.

A node a in graph G is a *root* node, if there is no edge e that enters node a. A node a is a *leaf* node, if there is no edge e that leaves a. A node a is an *isolated* node if it is both a root node and a leaf node, i.e., it is not linked to any other node in the graph. Formally, the notions of root and leaf nodes are defined as follows.

$$\begin{aligned}
&\mathrm{Root}_r(O) = \{c \in C | \neg\exists x \in C \cdot (x, c) \in r\},\\
&\mathrm{Leaf}_r(O) = \{c \in C | \neg\exists x \in C \cdot (c, x) \in r\}.
\end{aligned}$$

A *path* p in $\mathrm{Graph}_r(O)$ is a sequence $(n_1, n_2, ...n_K)$ of $K > 0$ nodes in the graph such that for all $i = 1, \cdots, K-1$, there is an edge e in the graph from node n_i to node n_{i+1}. A path is a *simple path* if all the nodes on the path are different. The *length* of path p, written $\mathrm{Length}(p)$, is the number of nodes on the path. A node b is *reachable* from node a in the graph, and written $a \rightsquigarrow b$, if there is a path from node a to node b in the graph. $p : a \rightsquigarrow b$ denotes that p is a path from node a to b. For $c \in \mathrm{Root}_r(O)$, $\mathrm{Reachable}_r^O(c)$ is the set of nodes reachable from c. Formally,

$$\mathrm{Reachable}_r^O(c) = \{x \in C | c \rightsquigarrow x\}.$$

Definition 5 (Relation-Based Structural Complexity

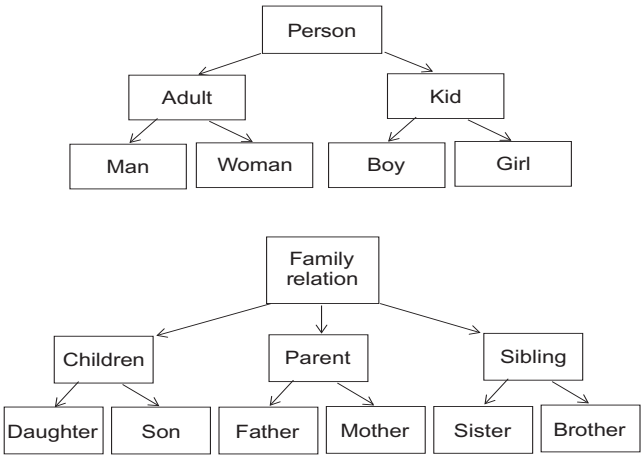


Fig. 3 Graph for *is-a* relation of *Family* Ontology

Metrics)

Let $r \in R$ be any given relation of the ontology. The following are a few structural metrics.

(1) Number of Root Nodes:
$$\mathrm{NRN}_r(O) = ||\mathrm{Root}_r(O)||.$$
(2) Number of Leaf Nodes:
$$\mathrm{NLN}_r(O) = ||\mathrm{Leaf}_r(O)||.$$
(3) Maximal Length of Simple Paths:
$$\mathrm{MaxSPL}_r(O) = \max_{p \in \mathrm{Path}_r(O)} (\mathrm{Length}(p)).$$
(4) Number of Isolated Nodes:
$$\mathrm{NIC}_r(O) = ||\mathrm{Root}_r(O) \cap \mathrm{Leaf}_r(O)||.$$
(5) Total Number of Reachable Nodes from Roots:
$$\mathrm{TNRNR}_r(O) = \sum_{x \in \mathrm{Root}_r(O)} ||\mathrm{Reachable}_r^O(x)||.$$
(6) Average Number of Reachable Nodes from Roots:
$$\mathrm{ANRNR}_r(O) = \mathrm{TNRNR}_r(O)/||\mathrm{NRN}_r(O)||. \quad \square$$

Example 2 For the *Family* ontology in Fig. 2, the *is-a* relation-based structure metrics are:

$$\mathrm{NRN} = 2,\ \mathrm{NLN} = 10,\ \mathrm{MaxSPL} = 3,$$
$$\mathrm{NIC} = 0,\ \mathrm{TNRNR} = 15,\ \mathrm{ANRNR} = 7.5.$$

The *has-a* relation-based structure metrics are:

$$\mathrm{NRN} = 1,\ \mathrm{NLN} = 1,\ \mathrm{MaxSPL} = 2,$$
$$\mathrm{NIC} = 0,\ \mathrm{TNRNR} = 1,\ \mathrm{ANRNR} = 1. \quad \square$$

For any well-structured ontology, the graph of the *is-a* and *has-a* relations must be acyclic. For these and other acyclic graphs, the depth of a node c is the length of the longest path from a root node to c. More formally,

$$\mathrm{Depth}_r^O(c) = \max_{x \in \mathrm{Root}_r(O)} \mathrm{Length}(p : x \rightsquigarrow c).$$

The width of a node is the number of nodes it is related to. More formally,

$$\mathrm{Width}_r^O(c) = ||\{x \in C | (c, x) \in r\}||.$$

Further metrics for acyclic relations can be defined as follows.

Definition 6 (Metrics for Acyclic Relations)

The following are structural complexity metrics for acyclic graphs.

(1) Average Depth of all Leaf Nodes:
$$\mathrm{ADLN}_r(O) = \frac{\sum_{c \in \mathrm{Leaf}_r(O)} \mathrm{Depth}_r^O(c)}{\mathrm{NLN}_r(O)}.$$
(2) Average Width of all Non-Leaf Nodes:
$$\mathrm{AWNLN}_r(O) = \frac{\sum_{c \notin \mathrm{Leaf}_r(O)} \mathrm{Width}_r^O(c)}{\mathrm{NAN}_r(O) - \mathrm{NLN}_r(O)},$$
where $\mathrm{NAN}_r(O)$ is the number of all nodes.
(3) Maximal Depth of all Leaf Nodes:
$$\mathrm{MaxDepth}_r(O) = \max_{c \in \mathrm{Leaf}_r(O)} \left(\mathrm{Depth}_r^O(c)\right).$$
(4) Maximal Width of Non-Leaf Nodes:
$$\mathrm{MaxWidth}_r(O) = \max_{c \notin \mathrm{Leaf}_r(O)} \left(\mathrm{Width}_r^O(c)\right). \quad \square$$

Example 3 For the *Family* ontology in Fig. 2, the structural metrics for the acyclic relations are as follows.

For the *is-a* relation:

$$\mathrm{ADLN} = 3,\ \mathrm{AWNLN} = 15/7,$$
$$\mathrm{MaxDepth} = 3,\ \mathrm{MaxWidth} = 3.$$

For the *has-a* relation:

$$\mathrm{ADLN} = 2,\ \mathrm{AWNLN} = 1,$$
$$\mathrm{MaxDepth} = 2,\ \mathrm{MaxWidth} = 1. \quad \square$$

3.3.2 Coupling metrics

An ontology may refer to concepts defined in other ontologies through attributes and relations. This means that the ontologies are coupled together.

Definition 7 (Coupling Metrics)

The following are coupling metrics.

(1) Number of External References through Attributes:
$$\mathrm{NERA}(O) = \sum_{c \in C} ||\{\varphi \in A^c | \varphi : c \rightarrow c', c' \text{ is external}\}||.$$
(2) Number of External References through Relations:
$$\mathrm{NERR}(O) = \sum_{r \in R} ||\{(c, c') \in r | c \text{ or } c' \text{ isexternal}\}||.$$
(3) Ratio of External Concepts:
$$\mathrm{REC}(O) = \frac{||\{c \in C | c \text{ is external}\}||}{\mathrm{Size}_C(O)}. \quad \square$$

Example 4 Suppose that the classes *Food, Pet* and their subclasses are defined in another ontology and referenced in the *Family* ontology, then the coupling metrics are:

$$\mathrm{NERA}(O) = 2,\ \mathrm{NERR}(O) = 0,$$
$$\mathrm{REC}(O) = 1/11. \quad \square$$

3.4 Semantics

Now, we define a set of metrics that measure the extent to which two ontologies of the same domain are semantically compatible with each other.

Ontologies not only introduce terminology but also represent knowledge about the world by specifying a conceptualisation through axioms to constrain the possible interpretation of the defined terms. It is

worth noting that, based on the concepts, relations, and attributes defined directly in an ontology, further elements can be defined by employing a formal logic system. This can also be used for reasoning about the knowledge. For the sake of generality, instead of specifying such a logic system, it is simply assumed that for an ontology O and associated axioms, a certain set of components can be defined and further statements can be inferred. In the sequel, we write $O \vdash x$ to denote that x is definable in ontology O, where x can be a concept, an attribute, an instance, or a relation.

The semantical completeness of an ontology can therefore also be measured by the following metrics of derivable spaces. Similar to Section 3.2, we will assume the existence of a *gold standard* ontology Ω of the domain that contains all concepts, relations, attributes, and instances of the domain.

Definition 8 (Semantic Coverage)

The semantic coverage of ontology O with respect to Ω on various components is defined as follows.

$$\begin{aligned}
&\mathrm{SCov}_C^{\Omega}(O) = D_C/\mathrm{Size}_C(\Omega),\\
&\mathrm{SCov}_I^{\Omega}(O) = D_I/\mathrm{Size}_I(\Omega),\\
&\mathrm{SCov}_A^{\Omega}(O) = D_A/\mathrm{Size}_A(\Omega),\\
&\mathrm{SCov}_R^{\Omega}(O) = D_R/\mathrm{Size}_R(\Omega),\\
&\mathrm{SCov}^{\Omega}(O) = (D_C + D_I + D_A + D_R)/\mathrm{Size}(\Omega),
\end{aligned}$$

where

$$\begin{aligned}
D_C &= ||\{c' \in C' | O \vdash c'\}||,\\
D_I &= \sum_{c' \in C'} ||\{\alpha \in I'^{c'} | O \vdash c', O \vdash \alpha\}||,\\
D_A &= \sum_{c' \in C'} ||\{\varphi \in A'^{c'} | O \vdash c', O \vdash \varphi\}||,\\
D_R &= \sum_{r' \in R'} ||\{(a,b) \in r' | O \vdash (a,b) \in r'\}||.
\end{aligned}$$

□

Example 5 Let Ω be ontology *Family* again, and let O be *Family* with *Pet* and its subclasses removed. The semantic coverage of O with respect to Ω is as follows:

$$\begin{aligned}
&\mathrm{SCov}_C^{\Omega}(O) = 21/22,\ \mathrm{SCov}_I^{\Omega}(O) = 1,\\
&\mathrm{SCov}_A^{\Omega}(O) = 7/8,\ \mathrm{SCov}_R^{\Omega}(O) = 1,\\
&\mathrm{SCov}^{\Omega}(O) = 49/51.
\end{aligned}$$

□

A domain ontology is semantically correct only if its contents are consistent with the ideal ontology of the gold standard. So, we have the following metrics.

Definition 9 (Semantic Compatibility)

Ontology O's semantic compatibility with regard to Ω on various components is defined as follows.

(1) Ratio of Correct Concepts:

$$\mathrm{RCC}^{\Omega}(O) = \frac{||\{c \in C \mid \Omega \vdash c\}||}{\mathrm{Size}_C(O)}.$$

(2) Average Ratio of Correct Instances:

$$\mathrm{ARCI}^{\Omega}(O) = \frac{\sum\limits_{c \in C} \mathrm{RCI}^{\Omega}(c)}{\mathrm{Size}_C(O)}.$$

(3) Average Ratio of Correct Attributes:

$$\mathrm{ARCA}^{\Omega}(O) = \frac{\sum\limits_{c \in C} \mathrm{RCA}^{\Omega}(c)}{\mathrm{Size}_C(O)}.$$

(4) Average Ratio of Correct Relations:

$$\mathrm{ARCR}^{\Omega}(O) = \frac{\sum\limits_{r \in R} ||\{(x,y) \in r \mid \Omega \vdash (x,y) \in r\}||}{||R||}.$$

where for each $c \in C$,

$$\mathrm{RCI}^{\Omega}(c) = \begin{cases} 1, & \text{if } I^c = \varnothing;\\ \dfrac{||\{\alpha \in I^c \mid \Omega \vdash \alpha \in I^c\}||}{||I^c||}, & \text{if } I^c \neq \varnothing. \end{cases}$$

$$\mathrm{RCA}^{\Omega}(c) = \begin{cases} 1, & \text{if } A^c = \varnothing;\\ \dfrac{||\{\varphi \in A^c \mid \Omega \vdash \varphi \in A^c\}||}{||A^c||}, & \text{if } A^c \neq \varnothing. \end{cases}$$

□

Example 6 Let Ω be the *Family* ontology, O is obtained from *Family* by adding a class *Profession*, an attribute *occupation* to class *Person* with *Profession* as its codomain. Then, ontology O's semantic compatibility w.r.t. Ω are:

$$\begin{aligned}
&\mathrm{RCC}^{\Omega}(O) = 22/23,\ \mathrm{ARCI}^{\Omega}(O) = 1,\\
&\mathrm{ARCA}^{\Omega}(O) = 206/207,\ \mathrm{ARCR}^{\Omega}(O) = 1.
\end{aligned}$$

□

An ontology may contain concepts, instances, attributes, and relations that are redundant. Here, an element in an ontology is redundant if it can be defined or derived from other elements of the ontology. More formally, redundancy can be defined as follows.

Definition 10 (Redundant Elements in Ontology)

(1) A subset $C' \subseteq C$ of concepts is redundant, if

$$\forall c \in C'.\left((C - C', I, A, R) \vdash c\right).$$

(2) A subset $I' = \{I'^c \subseteq I^c | c \in C\}$ of instances is redundant, if

$$\forall \alpha \in I'^c, c \in C.\left((C, I - I', A, R) \vdash \alpha \in I^c\right).$$

(3) A subset $A' = \{A'^c \subseteq A^c | c \in C\}$ of attributes is redundant, if

$$\forall \varphi \in A'^c, c \in C.\left((C, I, A - A', R) \vdash \varphi \in A^c\right).$$

(4) Let $R' = \{r'_1, \cdots, r'_k\}$, where for each $r'_i \in R'$, $r'_i \subseteq r_i$. The collection of relations R' is redundant, if

$$\forall (c, c') \in r'_i, i \in \{1, \cdots, k\}. \left(O' \vdash (c, c') \in r_i\right).$$

where $O' = (C, I, A, R - R')$. □

The metrics of semantic redundancy of an ontology can be defined as follows. Note that, for a set of concepts there may be many different subsets of redundant concepts. The same is true for instances, attributes, and relations.

Definition 11 (Redundancy Metrics)

(1) Concept Redundancy:

$$\mathrm{CR}(O) = ||C_R||/\mathrm{Size}_C(O),$$

where C_R is the largest set of redundant concepts of O.

(2) Instance Redundancy:

$$\mathrm{IR}(O) = \sum_{c \in C} ||I_R^c||/\mathrm{Size}_I(O),$$

where $I_R = \{I_R^c | c \in C\}$ is the largest collection of redundant instances of O.

(3) Attribute Redundancy:

$$\mathrm{AR}(O) = \sum_{c \in C} ||A_R^c||/\mathrm{Size}_A(O),$$

where $A_R = \{A_R^c | c \in C\}$ is the largest collection of redundant attributes of O.

(4) Relation Redundancy:

$$\mathrm{RR}(O) = \sum_{r' \in R'} ||r'||/\mathrm{Size}_R(O),$$

where R' is the largest collection of redundant relations of O. □

Example 7 Suppose that an attribute *Age* for the class *Person* is added in the ontology *Family*. Then the redundancy metrics are as follows:

$$\mathrm{CR} = 0,\ \mathrm{IR} = 0,\ \mathrm{AR}(O) = 1/9,\ \mathrm{RR}(O) = 0.$$

This is because *Age* can be derived from the attribute *Date of Birth*. □

3.5 Context

In computer science and information technology, ontologies are used to support certain computation and information processing tasks. A key question to be answered when evaluating an ontology is whether it is easy to use for these tasks. This paper focuses on service-oriented computing and is concerned with whether an ontology is a good basis for describing the semantics of services.

Assume that a web service S provides m operations $\mathrm{Op}_1, \cdots, \mathrm{Op}_m$ and stores $s_1, \cdots, s_l$ types of internal data. We write $\mathrm{Op}_i : (x_{i,1}, \cdots, x_{i,n_i}) \rightarrow (y_{i,1}, \cdots, y_{i,k_i})$ to denote that operation Op_i takes a service request containing parameters $(x_{i,1}, \cdots, x_{i,n_i})$ and responds with a message containing parameters $(y_{i,1}, \cdots, y_{i,k_i})$. An ontological description of the semantics of such a web service consists of the following expressions:

- $\mathrm{Exp}_{\mathrm{Op}_i}$, which describes the functionality of the service operation Op_i, where $i = 1, \cdots, m$;
- $\mathrm{Exp}_{x_{i,j}}$, which defines the meaning of the parameter $x_{i,j}$ in the service request, where $j = 1, \cdots, n_i$;
- $\mathrm{Exp}_{y_{i,j}}$, which describes the meaning of the parameter $y_{i,j}$ in service response, where $j = 1, \cdots, k_i$;
- Exp_{s_i}, which describes the meaning of the internal state of the service, where $i = 1, \cdots, l$.

Ideally, these expressions should be statements in the application domain ontology. A good domain ontology will enable these expressions to be simple, but in a badly-designed domain ontology they might be complicated, or even impossible to formulate. The following metrics measure the definability of a service semantics.

Definition 12 (Definability of Service Semantics)

The definability of the state of a web service S, denoted by $\mathrm{DState}^S(O)$, is defined as follows:

$$\mathrm{DState}^S(O) = l_\mathrm{d}/l,$$

where l_d is the number of state components that are definable in O.

For each operation $\mathrm{Op}_i : (x_{i,1}, \cdots, x_{i,n_i}) \rightarrow (y_{i,1}, \cdots, y_{i,k})$ provided by the service, the definability of the operation in O, denoted by $\mathrm{DFun}^{\mathrm{Op}_i}(O)$, is defined as follows:

$$\mathrm{DFun}^{\mathrm{Op}_i}(O) = \mathrm{ND}_\mathrm{Op}/(n_i + k_i + 1),$$

where $\mathrm{ND}_\mathrm{Op} = n_\mathrm{d} + k_\mathrm{d} + \mathrm{Op}_\mathrm{d}$, n_d and k_d are the numbers of request parameters and response parameters definable in O, respectively. $\mathrm{Op}_\mathrm{d} = 1$, if the functionality of the operation is definable in O; otherwise, $\mathrm{Op}_\mathrm{d} = 0$.

The definability of the service system S in ontology O, denoted by $\mathrm{DServ}^S(O)$, is defined as follows:

$$\mathrm{DServ}^S(O) = \mathrm{DState}^S(O) \times \sum_{i=1}^{m} \mathrm{DFun}^{\mathrm{Op}_i}(O)/m.$$

□

Better metrics of definability of a service in an ontology should take into consideration the complexity of the semantic descriptions. Here, the following simple metrics of the complexities of expressions is used.

Let Exp be an expression that uses the vocabulary defined in O and also logic operators, qualifiers, and data operators. The complexity of the expression Exp,

denoted by Cmplx(Exp), is defined as follows:

$$\text{Cmplx(Exp)} = \text{NOP} + \text{NV},$$

where NOP is the number of occurrences of logic and data operators in the expression, and NV is the number of occurrences of vocabulary defined in ontology O in the expression.

Definition 13 (Complexity of Semantic Description)

The Complexity of Semantic Description of the State of Service S in O is defined as follows:

$$\text{CState}^S(O) = \sum_{i=1}^{l} \text{Cmplx}(\text{Exp}_{s_i}).$$

The Complexity of Semantic Description of the functionality of operation Op_i of Service S in O, denoted $\text{CFun}^{\text{Op}_i}(O)$, is a metric defined as follows:

$$\text{CFun}^{\text{Op}_i}(O) = \text{Cmplx}(\text{Exp}_{\text{Op}_i}) + \sum_{j=1}^{n_i} \text{Cmplx}(\text{Exp}_{x_{i,j}}) + \sum_{j=1}^{k_i} \text{Cmplx}(\text{Exp}_{y_{i,j}}).$$

The Complexity of Semantic Description of Service S in O, denoted by $\text{CServ}^S(O)$, is a metric defined as follows.

$$\text{CServ}^S(O) = \text{CState}^S(O) + \sum_{i=1}^{m} \text{CFun}^{\text{Op}_i}(O). \quad \square$$

Example 8 Consider a web service that provides services for registering personal information and answers queries about family relationships between persons, etc. It contains a database of personal information, which is the internal state of the system. The service operations provided are:

$$\text{Register} : (n : \text{String}, f, m : \text{Person}, d : \text{Date}) \rightarrow (\text{id} : \text{Nat});$$

$$\text{QueryParents} : (\text{id} : \text{Nat}) \rightarrow (f, m : \text{Person}).$$

Table 1 gives the semantic descriptions of the service and the complexity measures. $\square$

3.6 Association between metrics and quality attributes

The metrics defined above are measurements of various aspects of an ontology. They are associated with various quality attributes in the quality model, as shown in Fig. 1. Completeness can be measured by vocabulary coverage and semantic coverage. Correctness can be measured by semantic compatibility. Conciseness can be measured by redundancy. Modularity can be measured by cohesion and coupling. Finally, applicability can be measured by definability and description complexity.

Table 1 Example: Semantics description.

Op/Param	Semantics description	Comp
State	List of person	2
n	Name of the Person	2
d	Date of Birth of the Person	2
f	Father of the Person	2
m	Mother of the Person	2
id	NI Number of the Person	2
Register	Add the Person into the List of Person f, m are in List of Person.	3
QueryParents	f and m are the Father and Mother of the Person. The Person's NI number is id.	2
$\text{CFun}^{\text{Register}}$		13
$\text{CFun}^{\text{QueryParents}}$		8
CServ		22

The dependence between the metrics are depicted in Fig. 4.

4 Tool Support of Ontology Assessment

The metrics defined in this paper have been implemented as a part of our formal engineering environment for services, which is called *ASWebService*. As shown in Fig. 5, the environment

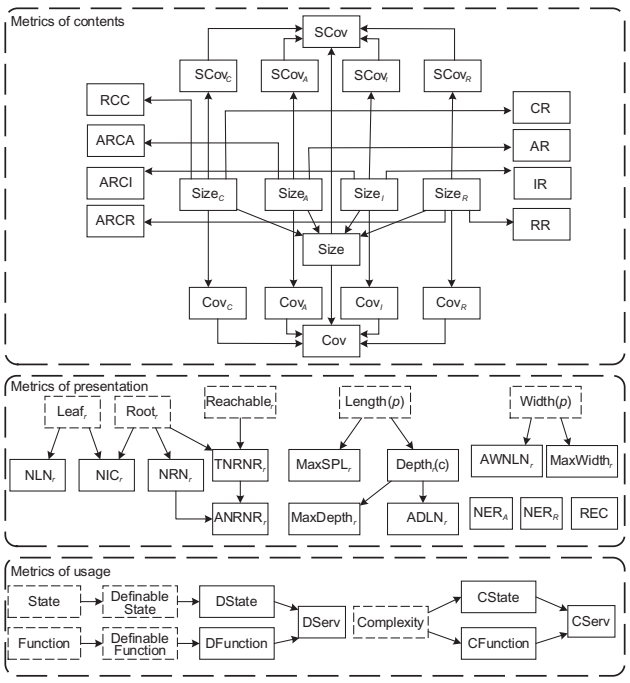


Fig. 4 Dependence between the metrics.

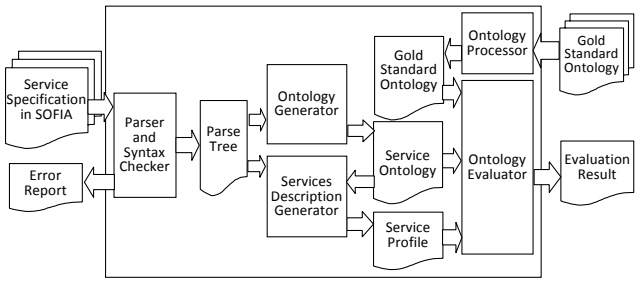


Fig. 5 Overall structure of the tool ASWebService.

contains the following five main components.

(1) *Specification Parser*, which parses a collection of algebraic specification units written in a language called SOFIA, generates a parse tree for each specification unit, checks whether each specification unit is syntactically well-formed, and finally checks that the axioms in the collection of specification units are type correct and consistent with each other.

(2) *Ontology Generator*, which takes the parse trees of a collection of specification units as input, and generates domain ontologies and ontological descriptions of web services according to the rules given in Ref. [18]. Figure 6 shows the interface for this component.

(3) *Service Description Generator*, which takes a domain ontology and a collection of specifications of web services as input and generates a semantic description of the web service in the form of an OWL-WS profile. See Fig. 7.

(4) *Ontology Processor*, which takes a file that contains an ontology written in OWL-WS/XML format, checks its syntactic well-formedness, and translates it into internal representation for use in the assessment and evaluation. See Fig. 8.

(5) *Ontology Evaluator*, which takes two ontologies as inputs, one generated from a web service and the other a gold standard for it to be assessed against. Typically the latter has been generated from the Ontology Processor component. The tool applies the metrics and reports the results. See Fig. 9.

5 Case Study

This section reports a case study on real world examples of web services. The purpose of the case study is to demonstrate that the quality model and metrics presented in the paper are applicable to an assessment of the quality of ontology.

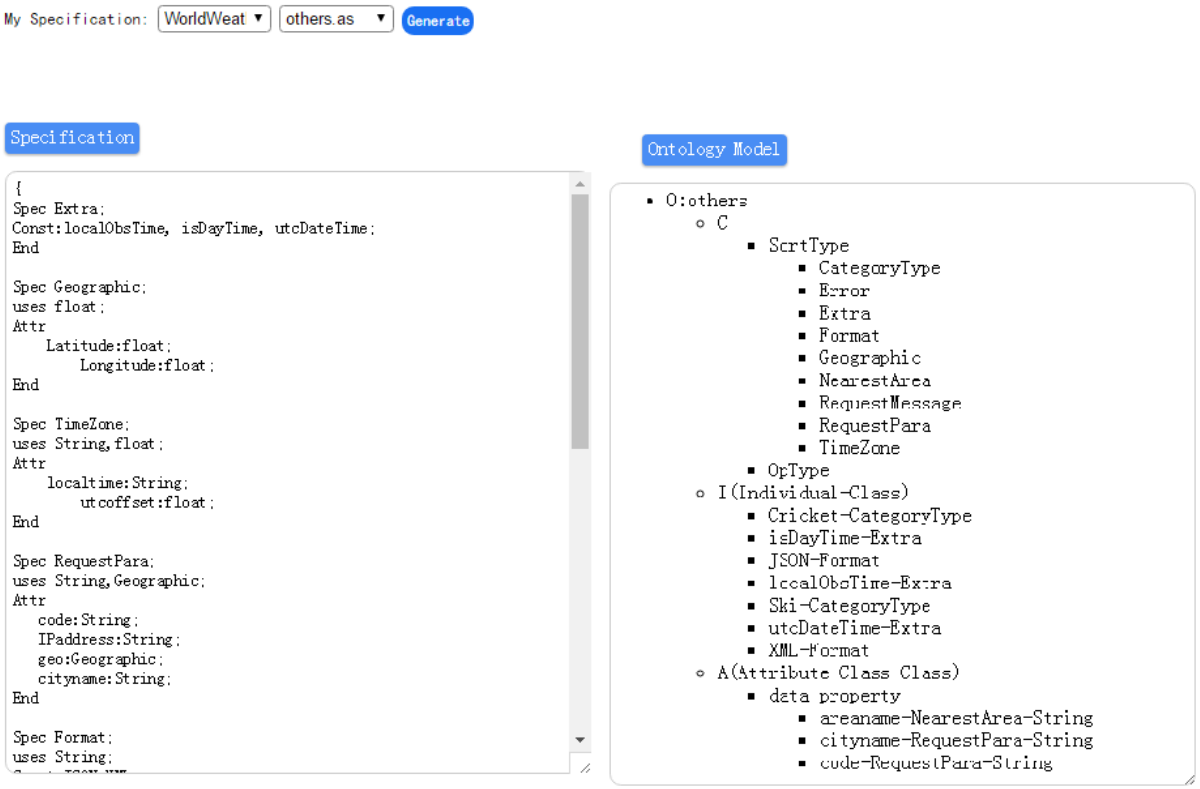


Fig. 6 Extraction of ontology from specification.

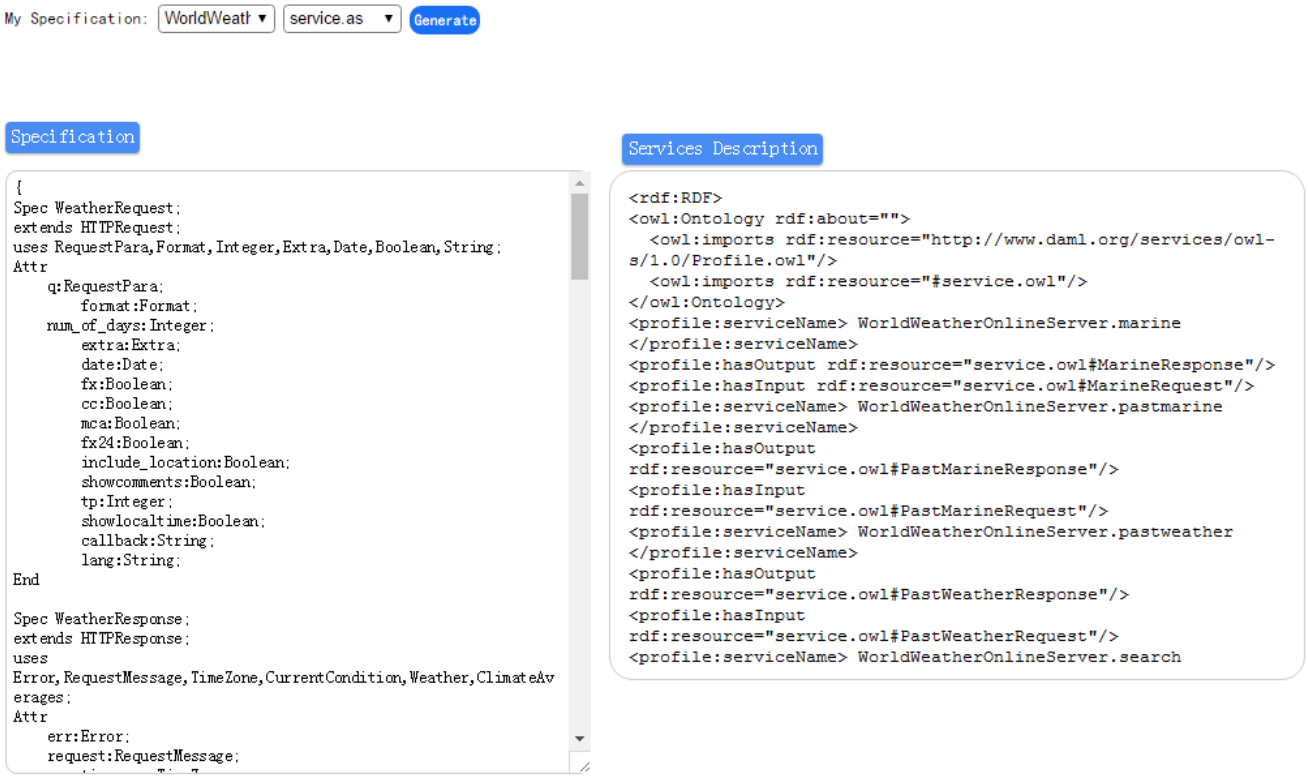


Fig. 7 Generation of semantics descriptions of web services.

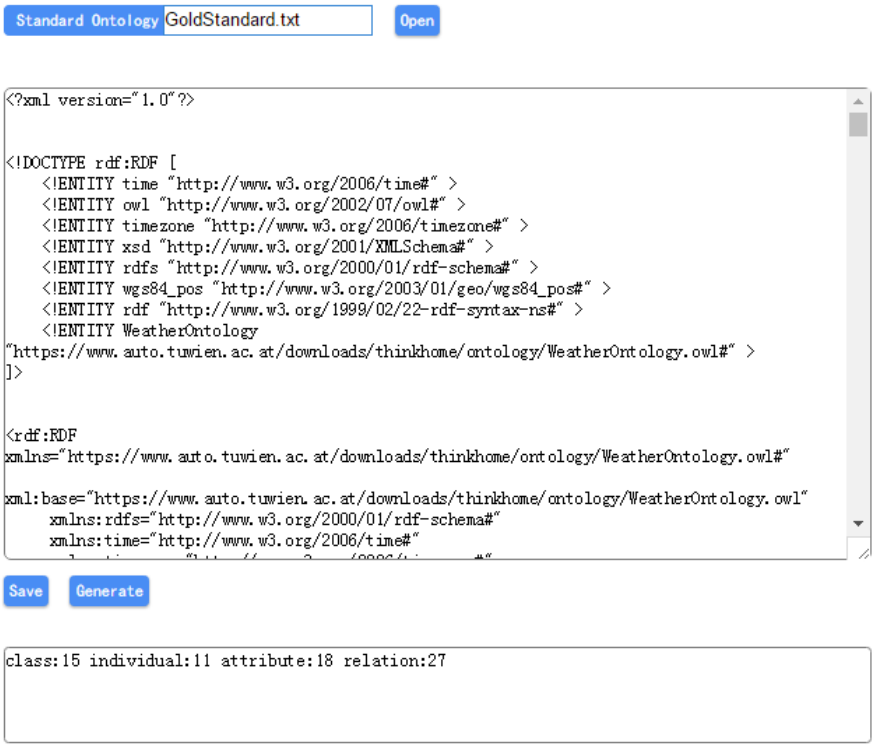


Fig. 8 Loading and checking user provided ontology.

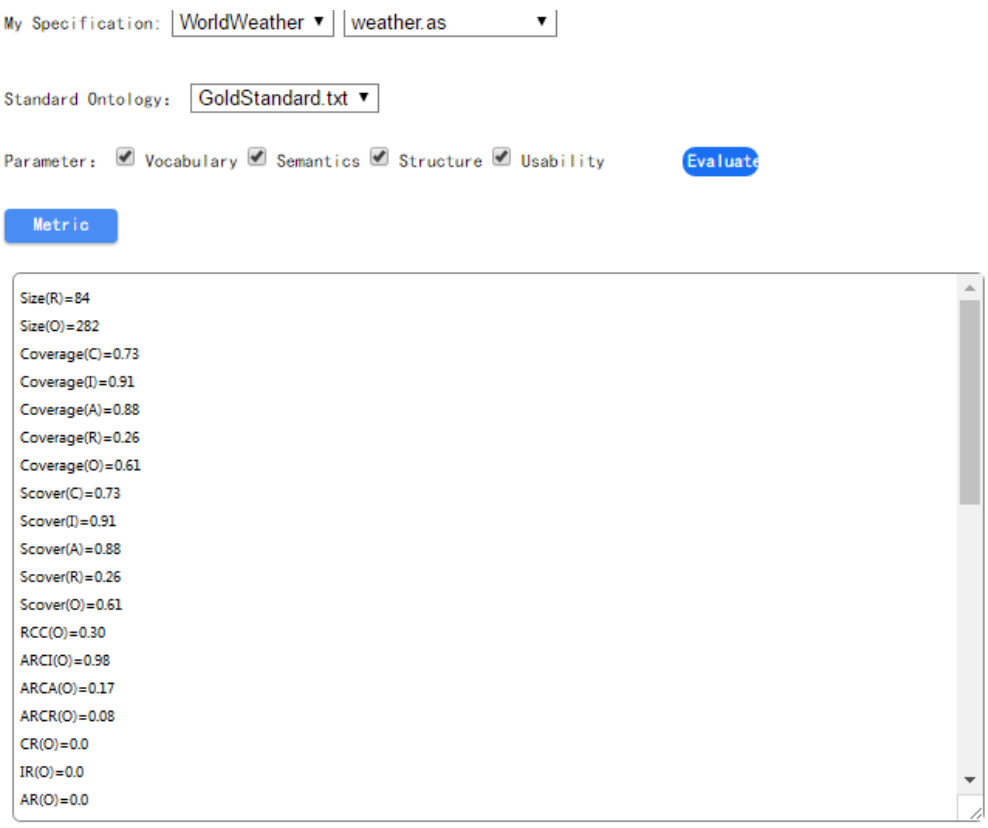


Fig. 9 Results of applying metrics to an ontology.

5.1 Design of the case study

A set of five real web services for the same application domain of reporting weather forecasts are used in the case study. The ontologies underlying each of these web services are semantic descriptions generated from their formal specifications. Table 2 lists the web services used in the case study, together with the IDs that will be used to refer to them in the sequel. The gold standard (GS) of the domain knowledge is the weather ontology developed by the Automation Systems Group of the Institute of Computer Aided Automation, Technical University of Wein, Austria. See Table 2 for its URL.

Table 2 Subject web services.

Ref ID	URL
WS1	http://www.webservicex.net/globalweather.asmx
WS2	http://openweathermap.org
WS3	http://www.webxml.com.cn/WebServices/WeatherWS.asmx
WS4	https://developer.yahoo.com/weather/documentation.html
WS5	http://developer.worldweatheronline.com/api/marine-weather-api.aspx
GS	https://www.auto.tuwien.ac.at/downloads/thinkhome/ontology/WeatherOntology.owl

These five web services are formally specified in an algebraic formal specification language called SOFIA[19]. A specification in SOFIA consists of a collection of specification units. Each unit specifies one concept from the real world or the type of a software entity. In this way, the structure of the specification can reflect that of the software systems and also that of the concepts that underlie the software. The specification units are grouped into closely associated packages.

In the case of the web services, each ontology is divided into three packages. The first describes concepts related to weather, since that is the application domain. The second describes external concepts such as *location*, *city*, etc. The third specifies the operations, including valid requests and responses, as well as the semantics of the operations. The numbers of specification units for each of these packages are shown in Table 3.

The ontology underlying each web service and the ontological semantic descriptions of the services are extracted from the formal specifications by using the TrS2O tool[18,20]. Each package transforms into an ontology module. Table 4 gives the sizes of these

Table 3 Sizes of the formal specifications.

Package	WS1	WS2	WS3	WS4	WS5
Definition of weather entities	10	13	19	19	35
Definition of external entities	2	6	2	3	9
Definition of operations	3	3	3	3	15

Table 4 Sizes of the ontologies.

Ontology		#Cls	#Insts	#Attrs	#Rels
Domain	GS	15	11	18	27
	WS1	12	2	26	13
	WS2	13	72	31	11
	WS3	12	9	28	11
	WS4	19	11	52	37
	WS5	35	49	114	84
Op	WS1	8	–	6	18
	WS2	19	–	28	35
	WS3	5	–	6	9
	WS4	4	–	11	14
	WS5	22	–	83	108

ontologies, where #Cls, #Insts, #Attrs, and #Rels are the numbers of classes, instances, attributes and relations, respectively.

The ontologies generated from specifications are then analysed for syntactic matching and semantic definability before metrics are applied. Table 5 gives the comparison of classes between GS and WS1, where *Point* and *Spatial Thing* are external classes that are referred to in the weather domain ontology.

5.2 Results of evaluation

The metrics defined in Section 3 are applied to the ontologies by using our implementation of the metrics. The results are given in Table 6.

In Table 6, the second column has the metrics used to evaluate the quality factors given in the first column.

Table 5 Comparison of classes between GS and WS1.

Class types	Classes	Nunmber
Defined both in GS and WS1	Wind, dew point temperature, cloud cover, weather state, atmospheric pressure, temperature, point	7
Defined in GS and derivable from WS1	Weather phenomenon	1
Defined in GS but not in WS1	Air pollution, precipitation, solar irradiance, weather condition, humidity, spatial thing, thing	7
Defined in WS1 but not in GS	Visibility, speed, direction, TEMUnit	4

Table 6 Evaluation of the *Weather* ontologies.

Factor	Metric	WS1	WS2	WS3	WS4	WS5
Syntactic completeness	Cov_C^Ω	0.47	0.57	0.43	0.4	0.73
	Cov_I^Ω	0	0.64	0	0	0.91
	Cov_A^Ω	0.67	0.56	0.28	0.44	0.88
	Cov_R^Ω	0.37	0.52	0.19	0.19	0.26
	Cov^Ω	0.41	0.55	0.23	0.24	0.61
Semantic completeness	$SCov_C^\Omega$	0.53	0.6	0.4	0.4	0.73
	$SCov_I^\Omega$	0	0.64	0	0	0.91
	$SCov_A^\Omega$	0.67	0.56	0.28	0.44	0.88
	$SCov_R^\Omega$	0.37	0.52	0.19	0.19	0.26
	$SCov^\Omega$	0.42	0.56	0.23	0.24	0.61
External consistency, compatibility	RCC^Ω	0.75	0.64	0.32	0.3	0.3
	$ARCI^\Omega$	1.00	0.86	0.95	0.6	0.98
	$ARCA^\Omega$	0.53	0.47	0.12	0.25	0.17
	$ARCR^\Omega$	0.34	0.47	0.13	0.14	0.08
Redundancy	CR	0	0	0	0	0
	IR	0	0	0	0	0
	AR	0	0	0	0	0
	RR	1	1	1	1	0.93
Cohesion (Relation)	NRN	1	2	2	1	9
	NLN	9	9	9	20	55
	MaxSPL	3	3	3	6	5
	NIC	0	0	0	0	0
	TNRNR	13	11	14	35	94
	ANRNR	13.0	5.5	7.0	35	10.44
Cohesion (Acyclic)	ADLN	2.56	2.33	2.89	4.00	3.27
	AWNLN	2.6	2.75	2.0	6.5	1.81
	MaxDepth	3	3	3	6	5
	MaxWidth	8	7	4	7	10
Coupling	NERA	0	0	0	2	1
	NERR	1	0	0	2	1
	REC	0.08	0	0	0.11	0.03
Definability	DState	1	1	1	1	1
	DFun	1	1	1	1	1
	DServ	1	1	1	1	1
Description complexity	CState	2	4	1	1	7
	CFun	12	24	6	6	42
	CServ	14	28	7	7	49

The data in the columns of WS1, WS2, WS3, WS4, and WS5 are the values of the ontologies on the metrics given in the second column. Note that ontology generated from WS5 has two relations has-a and *is-a*, while ontologies generated from the other four web services only have one relation *has-a*. In order to make a fair comparison among these ontologies, the *is-a* relation of WS5 is not included.

5.3 Analysis of the results and observations

5.3.1 Completeness

When measuring the completeness of an ontology with

respect to a given gold standard, one can use either purely syntactic vocabulary coverage metric, or use a semantic coverage metric instead. The experimental data shows that these are highly correlated, with correlation coefficients ranging from 0.9584 (WS2) to 1.000 (WS4 and WS5), see Fig. 10f, where the scores of ontologies WS1 to WS5 on syntactic and semantic coverage metrics are given in Figs. 10a–10e. A lower coefficient would be seen only where the domain has a large number of synonyms or varying non-standard terminologies. For a scientific subject area such as weather forecasting, both of these are unlikely, so inspection of the vocabulary coverage alone should be sufficient to judge the completeness of the ontology. This is practically useful because semantic coverage involves logical inference and possibly even human intervention, which can be time-consuming.

5.3.2 Compatibility with the gold standard

The compatibility scores of the extracted ontologies are quite low, as Fig 11 shows. This is of concern because compatibility with the gold standard is normally an indication of correctness. However, the extracted ontologies contain many elements, shown in Table 3, that are not about domain knowledge but rather the data structures, infrastructure, and operations that are specific to the particular web service. As these are implementation-specific details, they do not have equivalents in the gold standard. It can be seen in Table 4 that compatibilities with regard to instances are high because the instances in the extracted ontologies are mostly in the application domain.

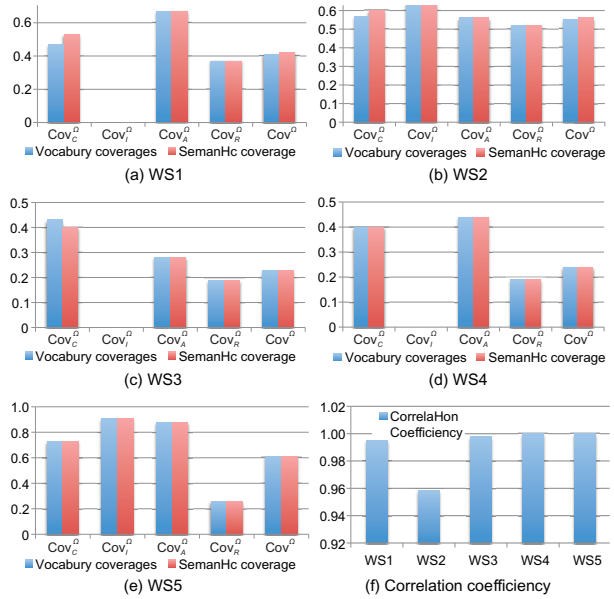


Fig. 10 Correlations between vocabulary and semantic coverages.

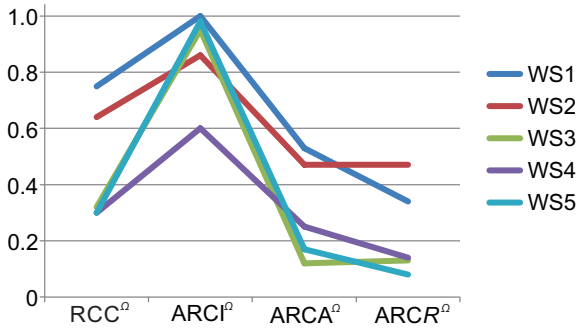


Fig. 11 Compatibility of the extracted ontologies.

5.3.3 Redundancy

The redundancy scores for all extracted ontologies are very low, especially for concepts, attributes, and instances, because the web service designs themselves have few redundancies.

5.3.4 Cohesion and structural complexity

Figure 12 shows that the structural complexity does not vary much even when the ontology size does.

WS5 is more than twice the size of WS4 which is itself more than twice the size of WS1, WS2, and WS3. The biggest differences between WS4/WS5 and WS1/WS2/WS3 are on the TNRNR and NLN. However, these are more indicative of size than structure. Moreover, all five ontologies have low scores for the coupling metrics so we can conclude that they are well-structured.

5.3.5 Usability

The ontologies are extracted from formal specifications so it is unsurprising that they score highly on definability metrics. They do score differently on description complexity, however, as can be seen in Fig. 13. An interesting observation is that the complexities

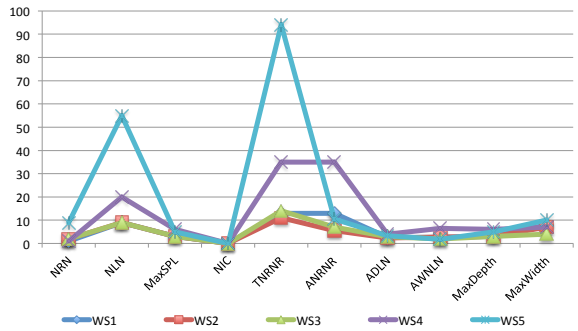


Fig. 12 Cohesion metrics of the extracted ontologies.

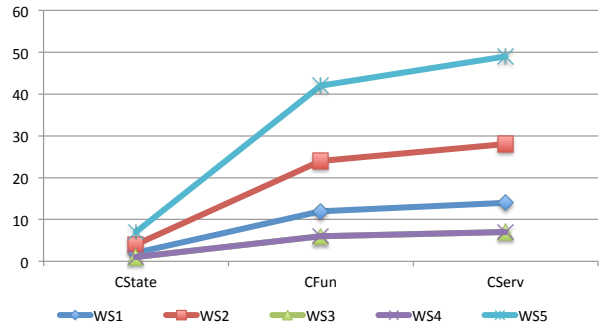


Fig. 13 Description complexity of the extracted ontologies.

of the descriptions of a web service on various aspects are highly correlated with each other. If an ontology scores high in one of these three metrics, it will score high on the other metrics. A practical implication of this means that there is no need for a trade-off when selecting an ontology.

5.3.6 Overall comparison of quality

The metrics need to be normalised for comparison between different aspects of quality as some metrics are relative values (e.g., completeness and conciseness) and some are absolute values (e.g., structural complexity and usability). We adopt the widely-used subrange technique in which subrange 1 means the poorest quality of a specific attribute or factor, and subrange 5 is the highest quality. Table 7 gives details of how the metrics are mapped to the subranges and Fig. 14 shows the ontology scores after normalisation.

Table 7 Mapping from metrics values to subranges.

Metric	Subranges				
	1	2	3	4	5
Cov	[0,0.2]	(0.2,0.4]	(0.4,0.6]	(0.6,0.8]	(0.8,1]
SCov	[0,0.2]	(0.2,0.4]	(0.4,0.6]	(0.6,0.8]	(0.8,1]
RCC	[0,0.2]	(0.2,0.4]	(0.4,0.6]	(0.6,0.8]	(0.8,1]
ARCI	[0,0.2]	(0.2,0.4]	(0.4,0.6]	(0.6,0.8]	(0.8,1]
ARCA	[0,0.2]	(0.2,0.4]	(0.4,0.6]	(0.6,0.8]	(0.8,1]
ARCR	[0,0.2]	(0.2,0.4]	(0.4,0.6]	(0.6,0.8]	(0.8,1]
CR	(0.8,1]	(0.6,0.8]	(0.4,0.6]	(0.2,0.4]	[0,0.2]
IR	(0.8,1]	(0.6,0.8]	(0.4,0.6]	(0.2,0.4]	[0,0.2]
AR	(0.8,1]	(0.6,0.8]	(0.4,0.6]	(0.2,0.4]	[0,0.2]
RR	(0.8,1]	(0.6,0.8]	(0.4,0.6]	(0.2,0.4]	[0,0.2]
MaxSPL	>8	(6,8]	(4,6]	(2,4]	[1,2]
NIC	>3	3	2	1	0
ANRNR	[1,3]	(3,5]	(5,8]	(8,12]	>12
ADLN	>8	(6,8]	(4,6]	(2,4]	[1,2]
AWNLN	>8	(6,8]	(4,6]	(2,4]	[1,2]
REC	(0.8,1]	(0.6,0.8]	(0.4,0.6]	(0.2,0.4]	[0,0.2]
DServ	[0,0.2]	(0.2,0.4]	(0.4,0.6]	(0.6,0.8]	(0.8,1]
CServ	>80	(60,80]	(40,60]	(20,40]	[2,20]

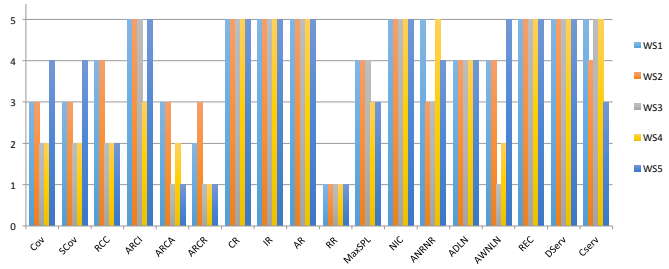


Fig. 14 Normalised metric scores of the ontologies.

Once the data has been normalised, the various completeness, structuredness, and usability metrics can be aggregated into a single value for each just by taking the average. This is shown in Fig. 15. A weighted average can be used, however, in situations where particular attributes are known to have greater or lesser importance.

Figure 15 shows the scores are high for usability and well-structuredness. This indicates that ontologies extracted from formal specifications are high quality and the extraction method is both feasible and useful in practice. The scores are low for completeness though. This is because the ontologies use only part of the terminology of weather forecasting. WS5 has the highest completeness score simply because it offers the most weather forecasting functions and thereby covers the gold standard most completely.

6 Conclusion

6.1 Related work

Ontology evaluation is a technical judgment of the quality of a given ontology with respect to certain criteria for a particular purpose. In one of the earliest works on ontology evaluation, Gomez-Perez[21] emphasised the following five aspects of ontology quality: Consistency, Completeness, Conciseness, Expandability, and Sensitiveness. This work forms a preliminary form of a quality model of ontologies. Vrandecic[22] further developed a

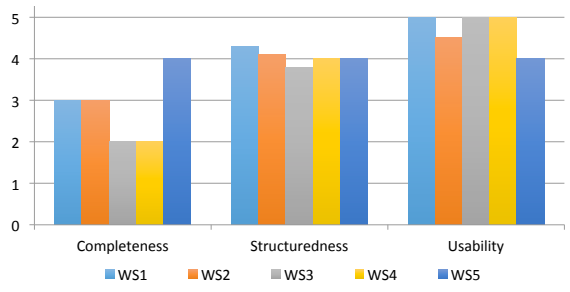


Fig. 15 Comparison of ontologies on quality attributes.

quality model of ontologies. It aimed at establishing a systematic approach to the evaluation of ontologies and proposed eight criteria to cover a wider range of quality attributes: Accuracy, Adaptability, Clarity, Completeness, Computational efficiency, Conciseness, Consistency, and Organizational fitness. Gangemi et al.[23] considered ontology evaluation as a diagnostic task, and classified criteria into three categories: Structural, Functional, and Usability, and defined nine quality attributes. Unlike the previous works, which use a flat structure, this quality model has a two-level hierarchy.

Recent work has adapted and applied software quality standards in an attempt to propose standardised methods for evaluating the quality of ontologies. Duque-Ramos et al.[24] proposed a general framework called OQuaRE to define the quality of ontologies. The aspects of ontology quality that they measured are: Structural, Functional adequacy, Reliability, Performance efficiency, Operability, Maintainability, Compatibility, Transferability, and Quality in use. However, most of the criteria are subjective and very hard to measure directly.

As Vrandecic pointed out in Ref. [22], an ontology is a complex, multi-layered information resource. He identified a number of aspects that vary between ontologies and therefore need to be evaluated. These include *Vocabulary*, *Syntax*, *Structure*, *Semantics*, *Representation*, and *Context*. Our quality model covers three aspects: *content*, *presentation*, and *usage* to give a clearer classification of the quality attributes.

Most existing ontology metrics are structural, see Ref. [25] for a survey. Tao et al.[26] used the number of root classes, the number of leaf classes, and the average depth of inheritance tree as cohesion metrics to measure modular relatedness of OWL ontologies. Later, Yang et al.[27] measured the complexity of ontology in the context of ontology evolution with a number of metrics. These were the number of concepts, the total number of relations, the total number of paths, the average length of paths, the longest length of paths, the average number of relations per concept, the average number of paths per concept, and the ratio of max length of paths over average length of paths. In addition to cohesion metrics, Orme et al.[28] and Oh et al.[29] also proposed coupling metrics for the evaluation of ontologies. These metrics are insufficient for evaluating ontologies, especially against a gold standard. They have not taken semantics into consideration, and are not applicable in the context of using ontology in the semantic description of web services.

6.2 Main Contribution of the work

In this paper, we proposed a metrics-based approach to the evaluation of ontologies in the context of their uses in the semantic description of web services. The first main contribution is a quality model of ontology with associated metrics. The former consists of quality attributes for each of three aspects: the contents, the presentation, and the usage. Each quality attribute is decomposed into a number of factors and measured by a set of thirty-seven metrics. The quality model has three features that are distinctive in comparison with existing work:

(1) *Objectiveness*: The metrics used are objective. Many are novel, such as those that compare an ontology against a gold standard ontology and those concerned with the semantics of ontology. As far as we know, there is no other account of using a gold standard in the definition of ontology metrics[10,25].

(2) *Language Independence*: The metrics are defined based on an abstract general model of ontologies rather than on the concrete syntax of any specific ontology definition language. These metrics cover all variable aspects of ontology that are not related to the syntax and representation of the ontology definition language. They are therefore generally applicable.

(3) *Service Orientation*: We developed a set of new metrics for the evaluation of ontologies in its usage context of describing the semantics of web services. They have been successfully applied in the evaluation of the web services in our case study. The existing work on ontology evaluation and assessment has been in the context of semantic web and/or web search engines[10,25].

The second main contribution is that all the metrics are implemented as a part of our formal engineering environment for service oriented computing. We have also conducted a case study using five real-world examples. The case study demonstrated that the metrics are feasible and effective for measuring the quality of ontologies after they have been extracted from algebraic specifications. The results of the measurement provide good indications of ontology quality.

6.3 Future work

The work reported in this paper is a part of our

long-term research agenda on formal engineering of service-oriented systems. The evaluation framework and the implementation of the metrics is a part of our formal service engineering environment. An interesting research question is whether the metrics can be redefined to make the evaluation be automated. A possible avenue for future work is to check the consistency of specification using both ontological reasoning and equational logic inferences. This will further reduce the need for human input in the evaluation of ontologies.

As with other software quality models, the relationships between quality attributes and factors are derived from empirical knowledge and experiences. This makes it very difficult and labour-intensive to validate them. Moreover, our quality model is hierarchical. It will be interesting to develop it into a relational model. This will require more experiments and empirical studies.

Acknowledgment

The work reported in this paper was partially supported by the National Natural Science Foundation of China (No. 61502233) and Jiangsu Qinglan Project, and partially supported by EU FP7 project MONICA on Mobile Cloud Computing (No. PIRSES-GA-2011-295222).

References

[1] D. Liu, Y. Yang, Y. Chen, H. Zhu, I. Bayley, and A. Aldea, Evaluating the ontological semantic description of web services generated from algebraic specifications, in *Proc. of the 10th IEEE International Conference on Service Oriented System Engineering (SOSE 2016)*, Oxford, UK, 2016.

[2] T. Gruber, A translation approach to portable ontology specifications, *Knowledge Acquisition*, vol. 5, no. 2, pp. 199-220, 1993.

[3] S. A. Mallraith, T. C. Son, and H. Zeng, Semantic web services, *IEEE Intelligent Systems*, vol. 16, no. 2, pp. 46–53, 2001.

[4] L. Richardson and S. Ruby, *RESTful Web Services*. O'Reily, 2007.

[5] D. Martin, M. H. Burstein, J. Hobbs, O. Lassila, D. Mcdermott, S. A. Mcilraith, S. Narayanan, M. Paolucci, B. Parsia, T. R. Payne, et al., OWL-S: Semantic markup for web services, http://www.w3.org/Submission/OWL-S/, Nov. 2004. (last access: May 25, 2015)

[6] J. Kopecky, K. Gomadam, and T. Vitvar, hRESTS: An HTML microformat for describing RESTful web services, in *Proc. of WI-IAT'08*, 2008, pp. 619–625.

[7] M. J. Hadley, Web Application Description Language (WADL), Sun Microsystems Inc., CA, USA, Tech. Rep. SMLI TR-2006-153, 2006.

[8] J. Lathem, K. Gomadam, and A. P. Sheth, SA-REST and Smashups: Adding semantics to RESTful services, in *Proc. of ICSC'07*, pp. 469–476, 2007.

[9] J. Bruijn, H. Lausen, A. Polleres, and D. Fensel, The web service modelling language WSML: An overview, in *Proc. of the 3rd Europ. Semantic Web Conf.*, 2006, pp. 590–604.

[10] J. Raad and C. Cruz, A survey on ontology evaluation methods, in *Proc. of the 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2015)* , 2015, pp.179–186.

[11] B. Kitchenham and S. L. Pfleeger, Software quality: The elusive target, *IEEE Software*, vol. 13, no. 1, pp12–21, 1996.

[12] A. Gillies, *Software Quality: Theory and Management*. International Thomson Computer Press, 1997.

[13] J. Bansiya and C. G. Davis, A hierarchical model for object-oriented design quality assessment, *IEEE Transactions on Software Engineering*, vol. 28, no. 1, pp. 4–17, 2002.

[14] ISO and IEC, Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) - System and Software Quality Models, ISO/IEC 25010:2011, First Edition, March 1, 2011.

[15] W. E. Perry, *Quality Assurance for Information Systems: Methods, Tools and Techniques*. John Wiley and Sons, 1991.

[16] Y. Zhang, Quality modelling and metrics of Web Information Systems, PhD dissertation, Oxford Brookes University, Oxford, UK, 2005.

[17] E. Zavitsanos, G. Paliouras, and G. A. Vouros, Gold standard evaluation of ontology learning methods through ontology transformation and alignment, *IEEE Trans. On Knowledge and Data Engineering*, vol. 23, no. 1, pp. 1635–1648, 2011.

[18] D. Liu, H. Zhu, and I. Bayley, Transformation of algebraic specifications into ontological semantic descriptions of web services, *Int'l J. of Services Computing*, vol. 2, no. 1, pp. 58–71, 2014.

[19] D. Liu, H. Zhu, and I. Bayley, SOFIA: An algebraic specification language for developing services, in *Proc. of SOSE 2014*, 2014, pp. 70–75.

[20] D. Liu, H. Zhu, and I. Bayley, From algebraic formal specification to ontological description of service semantics, in *Proc. of ICWS 2013*, 2013, pp. 579–586.

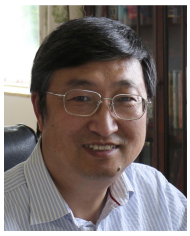
[21] A. Gomez-Perez, Towards a framework to verify knowledge sharing technology, *Expert Systems with Applications*, vol. 11, no. 96, pp. 519–529, 1996.

[22] D. Vrandecic, Ontology evaluation, in *Handbook on Ontologies*, S. Staab and R. Studer, eds. Springer, 2009, pp. 293–313.

[23] A. Gangemi, C. Catenacci, M. Ciaramita, and J. Lehmann, Modelling ontology evaluation and validation, in *The Semantic Web: Research and Applications*. Springer, 2006, pp. 140–154.

[24] A. Duque-Ramos, J. T. Fernndez-Breis, R. Stevens, and N. Aussenac-Gilles, OQuaRE: A SQuaRE-based approach for evaluating the quality of ontologies, *Journal of Research and Practice in Information Technology*, vol. 43, no. 2, pp. 159–176, 2011.

[25] J. Garca, F.J. Garca-Pealvo, and R. Thern, A survey on ontology metrics, in *Knowledge Management, Information Systems, E-Learning, and Sustainability Research*, 2010, pp. 22-27.
[26] H. Tao, A. M. Orme, and L. H. Etzkorn, Cohesion metrics for ontology design and application, *Journal of Computer Science*, vol. 1, no. 1, pp. 107–113, 2005.
[27] Z. Yang, D. Zhang, and Y. E. Chuan, Evaluation metrics for ontology complexity and evolution analysis, in *Proc. of ICEBE 2006*, 2006, pp. 162–170.
[28] A. M. Orme, H. Tao, and L. H. Etzkorn, Coupling metrics for ontology-based system, *IEEE Software*, vol. 23, no. 2, pp. 102–108, 2006.
[29] S. Oh, H. Y. Yeom, and J. Ahn, Cohesion and coupling metrics for ontology modules, *Information Technology and Management*, vol. 12, no. 2, pp. 81–96, 2011.



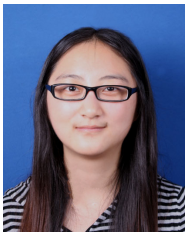
Hong Zhu obtained the BSc, MSc, and PhD degrees in computer science from Nanjing University, China, in 1982, 1984, and 1987, respectively. He worked at Nanjing University 1987 to November 1998. From October 1990 to December 1994 while on leave from Nanjing University, he was a research fellow at Brunel University and the Open University, UK. He joined Oxford Brookes University, UK, in November 1998 as a Senior Lecturer in Computing and became a Professor of Computer Science in October 2004. Prof. Zhu chairs the Applied Formal Methods Research Group of the Department of Computing and Communication Technologies. He is a senior member of IEEE Computer Society, a member of British Computer Society, ACM, and China Computer Federation. His research interests are in the area of software development methodologies, including formal methods, agent-orientation, automated software development, foundation of software engineering, software design, modelling and testing methods, Software-as-a-Service, etc. He has published 2 books and more than 180 research papers in journals and international conferences. He has been a conference program committee chair of SOSE 2012 and ICWS 2015, etc., a conference general chair of SOSE 2013, MobileCloud 2014, MS 2016, EDGE 2017, etc. He is a member of the editorial board of the journal of *Software Testing, Verification and Reliability, Software Quality Journal, International Journal of Big Data Intelligence*, and *the International Journal of Multi-Agent and Grid Systems*.

Arantza Aldea is a Senior Lecturer in the Department of Computing and Communication Technologies) at Oxford Brookes University (UK). She received the BSc and PhD degrees in 1989 and 1994, respectively. Previously she was a lecturer at Universitat Rovira I Virgili in Spain and a post-doctoral researcher at Herriot Watt University (UK). She has written more than 30 papers in international journals and conference proceedings. She has worked in international multi-disciplinary research projects and her research interests include knowledge representation, ontologies, and semantic web. She is also interested in the development of intelligent mobile tools to facilitate the self-management of chronic health conditions.

Dongmei Liu received the BS degree in computer applied technology, the MS degree in computer architecture, and the PhD degree in computer software and theory from Wuhan University, China, in 1999, 2001, and 2004, respectively. Currently, she is an associate professor at Nanjing University of Science and Technology, China. She is a member of IEEE and CCF. Her research interests include software testing, software reliability modeling, formal specification methods, and intelligent computation.

Ian Bayley achieved MEng Computing from Imperial College, London in 1997 and a DPhil Computing from Oxford University in 2002. He was a lecturer at Bournemouth University from 2002 and has been a lecturer at Oxford Brookes University since 2005. He has organised conferences in the UK such as QSIC 2008, CYBERPATTERNS 2012, and SOSE 2014. His work on design patterns achieved the award for best paper in 2008. His research interests include design patterns, formal specification, cybersecurity, and agent-oriented programming.

Yunfei Yang received the BS degree in computer science and technology and the MS degree in computer technology from Nanjing University of Science and Technology, China, in 2013 and 2016, respectively. Her research interests include quality model, ontology evaluation, semantic web services.



Ying Chen received the BS degree in computer science and technology from Nanjing Normal University, China, in 2015. Currently, she is now studying for a master degree in computer science at Nanjing University of Science and Technology, China. Her research interests include Web Services composition and formal specification methods.