Evaluating the Ontological Semantic Description of Web Services Generated from Algebraic Specifications

Dongmei Liu, Yunfei Yang, Ying Chen
School of Computer Science and Engineering
Nanjing University of Science and Technology
Nanjing, 210094, P.R. China
dmliukz@njust.edu.cn, 870766592@qq.com
1967394547@qq.com

Hong Zhu, Ian Bayley, Arantza Aldea
Dept of Comp. and Comm. Technologies
Oxford Brookes University
Oxford OX33 1HX, UK
hzhu@brookes.ac.uk, ibayley@brookes.ac.uk
aaldea@brookes.ac.uk

***Abstract*—The semantics of web services can be described using ontology or formally specified in mathematical notations. The former is comprehensible and searchable, while the latter is testable and verifiable. To take advantage of both, we proposed, in our previous work, a transformation that takes an algebraic specification of a web service to generate a domain ontology and a semantic description of the service on that ontology.**

This paper investigates the quality of these two outputs by proposing a general framework of ontology evaluation that assesses them on 4 aspects of quality, which are decomposed into 8 factors and then measured by a set of 37 metrics. It reports a case study on 3 real-life examples of web services. The results show that the ontologies and semantic descriptions generated from formal specifications are of satisfactory quality.

***Keywords*-Service Semantics; Algebraic Specifications; Ontology Evaluation; Ontology Metrics.**

I. Introduction

The accurate description of semantics plays a crucial role in service discovery, composition and interaction. Existing techniques fall into two categories: *ontology-based* approach and *formal methods based* approach. The former uses a vocabulary defined in application domain ontologies to annotate services, while the latter uses mathematical notations to formally define the functions of the software system.

Semantic Web Services have been advanced in the context of Big Web Services (i.e. those based on WSDL, SOAP and UDDI, etc.) as well as RESTful Web Services [1]. They describe services using domain ontologies [2], which are defined in ontology definition languages, such as OWL-S [3], MicroWSMO/hRESTS [4], WADL [5] and SA-REST [6], WSML [7], etc. In this approach, the domain ontologies give metadata-like labels that can be applied to the operations as well as to their input and output parameters. Such labels are easy for human developers to understand and efficient for computers to process. However, they cannot provide a verifiable and testable definition of a service's function, because ontology is limited to stereotypes formed from the relationship between the concepts and their instances.

Formal methods, which we consider as an alternative to the ontological approach, have been developed over the past 40 years to define the semantics of software systems in mathematical notations. One such formal method, algebraic specification, was first proposed in the 1970s for specifying the semantics of abstract data types. Since then, it has been extended to concurrent systems, state-based systems, software components and service-oriented systems [8], [9].

Algebraic specifications are at a very high level of abstraction, independent of any implementation details, and can be used directly in automated software testing [10]–[12]. However, algebraic specifications do not directly support efficient searching of services, and neither do other formal methods. To bridge the gap between algebraic specifications and ontological descriptions, we proposed a transformational approach in [13]. In this approach, algebraic specifications of services are transformed automatically into an ontology-based semantics description. This confers on formal specifications the machine-readability and human-understandability benefits of ontologies. The tool is called TrS2O (Translator from Specification to Ontology) and has been implemented in Java [14]. Its input formal specifications are written in SOFIA and its output ontological descriptions of services in OWL-S.

It remains an open question, however, whether the generated ontological semantic descriptions are of good quality. This paper addresses this problem by developing a metrics-based ontology evaluation framework and conducting a case study on three real-world examples. The remainder of the paper is organised as follows. Section II briefly reviews the related work. Section III gives the framework of our evaluation of domain ontology. Section IV formally defines a set of metrics of ontologies. They are applied to real web services in a case study in Section V. Section VI concludes the paper with a summary of our main contributions and a discussion of future work.

II. Related Work

Ontology evaluation is a technical judgment of the quality of a given ontology with respect to certain criteria for a particular purpose. Among the early work on ontology

evaluation, Gomez-Perez [15] emphasised on the following five aspects of ontology quality: Consistency, Completeness, Conciseness, Expandability and Sensitiveness.

Aiming at establishing a systematic approach to the evaluation of ontologies, Vrandecic [16] proposed eight quality criteria: Accuracy, Adaptability, Clarity, Completeness, Computational efficiency, Conciseness, Consistency and Organizational fitness. Gangemi et al. [17] considered ontology evaluation as a diagnostic task, and classified criteria into three categories: Structural, Functional and Usability, and defined nine quality attributes.

Recent work adapted and applied the software quality standards, trying to propose standardised methods for evaluating the quality of ontologies. Duque-Ramos [18] 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. In this paper, we only employ objectively measurable criteria for ontology evaluation.

As Vrandecic pointed out [16], an ontology is a complex, multi-layered information resource. He identified a number of aspects that are variable for an ontology, thus they are the subject to be evaluated. These include *Vocabulary*, *Syntax*, *Structure*, *Semantics*, *Representation*, and *Context*. In this paper, we are only concerned with four of these six aspects identified by Vrandecic. *Syntax* and *Representation* are irrelevant because the ontologies that we evaluate are generated from algebraic specifications automatically. With minor changes to the transformation tool, we can generate the same ontology in different ontology definition languages and with different representations. Therefore, we define metrics at a higher level of abstraction so that they are independent of the ontology definition language. Consequently, our metrics can be used to evaluate and compare ontologies defined in different languages.

Most existing ontology metrics are structural; see [19] for a survey. Yao et al. [20] 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. [21] measured the complexity of ontology in the context of ontology evolution using 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, the ratio of max length of paths over average length of paths. In addition to cohesion metrics, Orme et al. [22] and Oh, et al. [23] also proposed coupling metrics for the evaluation of ontologies.

In this paper, we propose a metrics-based approach to evaluate ontology in the context of their uses in semantic description of web services and apply it to the evaluation of ontologies generated from algebraic specifications.

III. The Evaluation Framework

An ontology is an explicit specification of a conceptualisation [24]. It represents the knowledge of a specific application domain by means of conceptualisation. Most ontologies used 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. They share many structural similarities regardless of the language in which they are expressed. Ontologies in many different languages can be obtained from a single algebraic specification by changing the transformation rules slightly. For this reason, our framework must be language-independent. Therefore, we define an abstract model of domain ontologies based on their structural features common to all ontology description languages. The metrics of ontologies will be defined based on this model.

A. Abstract Model of Ontologies

Definition 1: (Domain Ontology)
A domain ontology $\mathbf{O}$ 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 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.

Figure 1 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.

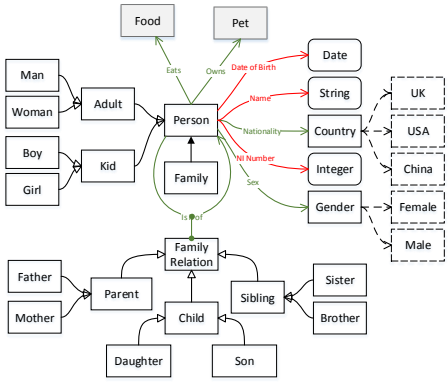

Figure 1. Model of *Family* Ontology

B. Quality Attributes and Factors

We identify four quality attributes of ontologies, which are decomposed into a few factors as shown in Fig. 2. Each quality factor is then measured by a number of metrics.

1) Completeness: how well the ontology represents the knowledge of a subject domain.

We employ an "gold standard" ontology as a representative of the domain knowledge. Whether an ontology completely represents the knowledge of a domain can be measured by the coverage of the gold standard. Two levels of coverage are recognised: *syntactic coverage* refers to the vocabulary that exactly matches the corresponding vocabulary of the gold standard; *semantic coverage* refers to the subset of vocabulary of the gold standard that can be derived semantically from the ontology. Another factor of an ontology's completeness with respect to a domain is the *compatibility* of the ontology to the gold standard. Metrics are defined to measure both coverage and compatibility on various aspects of ontology, including concepts, instances, attributes and relations.

2) Conciseness: whether the ontology is informative.

The key factor of conciseness is the *redundancy* within the ontology. Metrics are defined on the redundancy of various types of ontology components.

3) Well Structuredness: whether it can be decomposed into smaller modules, to make the ontology easier to understand, use and maintain. These smaller modules often form ontologies of sub-domains that can be used separately with references to other modules/ontologies.

Whether the decomposition is well structured can be judged on two factors: the *cohesion* and *coupling* between the modules. Here, cohesion means the interaction within one module/ontology, while coupling means the cross references or relations between different modules/ontologies. A well-structured decomposition should have high cohesion and low coupling. A number of coupling and cohesion metrics are defined.

4) Usability: whether an ontology is easy to use for a specific task, which, in this paper, is to describe the semantics of services.

A service consists of a number of operations on the state of the system. A semantic service description describes the meanings of the states of the system, the service requests, responses and the operation in the vocabulary of the ontology. Two factors of usability in this context are recognised: *definability* refers to whether the states and functions of the services can be defined within the ontology; *description complexity* refers to how complex the description of the state and functions of a service, if they are definable. Metrics are defined for both of them.

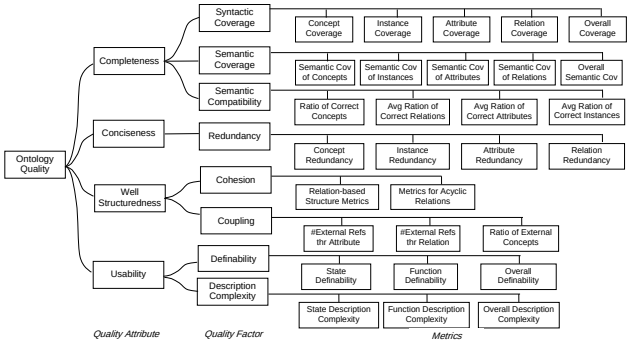

Figure 2. Factors of Quality Attributes

IV. Metrics of Ontology

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.

A. 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. We first define a set of basic metrics for the sizes of an ontology on various aspects.

Definition 2: We define sizes of an ontology as follows.

$$Size_C(O) = ||C||, \qquad Size_I(O) = \sum_{c \in C} ||I^c||,$$

$$Size_A(O) = \sum_{c \in C} ||A^c||, \qquad Size_R(O) = \sum_{r \in R} ||r||,$$

$$Size(O) = Size_C(O) + Size_I(O) + Size_A(O) + Size_R(O). \quad \square$$

In the evaluation of an ontology's coverage of an application domain, a *gold standard* is often used [25]. This acts as the ideal representation of domain knowledge. The following metrics measures the domain coverage of an ontology O with respect to a second ontology $\Omega = (C', I', A', R')$ as the gold standard.

Definition 3: (Vocabulary Coverage)

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

$$Cov_C^\Omega(O) = S_C/Size_C(\Omega); \qquad Cov_I^\Omega(O) = S_I/Size_I(\Omega);$$
$$Cov_A^\Omega(O) = S_A/Size_A(\Omega); \qquad Cov_R^\Omega(O) = S_R/Size_R(\Omega);$$
$$Cov^\Omega(O) = \frac{(S_C + S_I + S_A + S_R)}{Size(\Omega)}$$

where

$$S_C = ||C \cap C'||, \qquad S_I = \sum_{c \in C} ||I^c \cap I'^c||,$$
$$S_A = \sum_{c \in C} ||A^c \cap A'^c||, \qquad S_R = \sum_{r \in R} ||r \cap r'||,$$

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.

$$S_C = 21, \quad Size_C(\Omega) = 22, \quad S_I = 5, \quad Size_I(\Omega) = 5,$$
$$S_A = 7, \quad Size_A(\Omega) = 8, \quad S_R = 16, \quad Size_R(\Omega) = 16,$$
$$Cov_C^\Omega(O) = 21/22, \quad Cov_I^\Omega(O) = 1, \quad Cov_A^\Omega(O) = 7/8,$$
$$Cov_R^\Omega(O) = 1, \qquad Cov^\Omega(O) = 49/51. \qquad \square$$

These coverage metrics measure the proportion of classes, instances, attributes and relations that are defined directly in the reduced ontology. However, an ontology may define only the key components and leave out the other components definable from the basic ones. We will investigate this in Section IV-C.

B. 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.

- *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 $Graph_r(O)$, 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 Figure 3.

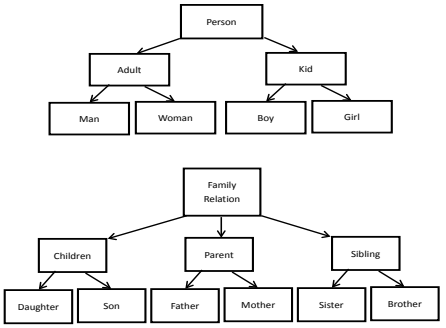

Figure 3. Graph for *is-a* realtion of *Family* Ontology

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, we define root and leaf nodes as follows.

$$Root_r(O) = \{c \in C | \neg \exists x \in C \cdot (x, c) \in r\}$$
$$Leaf_r(O) = \{c \in C | \neg \exists x \in C \cdot (c, x) \in r\}$$

A *path* p in $Graph_r(O)$ 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 $Length(p)$, is the number of nodes on the path. If there is a path from node a to node b in the graph, we say that node b is *reachable* from node a in the graph, and write $a \rightsquigarrow b$. We also write $p : a \rightsquigarrow b$ to denote that p is a path from node a to b. For $c \in Root_r(O)$, we define $Reachable_r^O(c)$ as the set of nodes reachable from c. Formally,

$$Reachable_r^O(c) = \{x \in C | c \rightsquigarrow x\}$$

Definition 5: (Relation-Based Structure Metrics) Let $r \in R$ be any given relation of the ontology. We define the following structural metrics.

1) Number of Root Nodes: $NRN_r(O) = ||Root_r(O)||$.
2) Number of Leaf Nodes: $NLN_r(O) = ||Leaf_r(O)||$.
3) Maximal Length of Simple Paths:
$$MaxSPL_r(O) = \max_{p \in Path_r(O)} (Length(p)).$$
4) Number of Isolated Nodes:
$$NIC_r(O) = ||Root_r(O) \cap Leaf_r(O)||.$$
5) Total Number of Reachable Nodes from Roots:
$$TNRNR_r(O) = \sum_{x \in Root_r(O)} ||Reachable_r^O(x)||.$$
6) Average Number of Reachable Nodes from Roots:
$$ANRNR_r(O) = TNRNR_r(O)/||NRN_r(O)||. \qquad \square$$

Example 2: For the *Family* ontology in Figure 1, the *is-a* relation-based structure metrics are:

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

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

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

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

$$Depth_r^O(c) = \max_{x \in Root_r(O)} Length(p : x \rightsquigarrow c)$$

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

$$Width_r^O(c) = ||\{x \in C | (c, x) \in r\}||$$

We can now define the following further metrics for acyclic relations.

Definition 6: (Metrics for Acyclic Relations)

For acyclic $Graph_r(O)$, we define the following metrics.

1) Average Depth of all Leaf Nodes:
$$ADLN_r(O) = \frac{\sum_{c \in Leaf_r(O)} Depth_r^O(c)}{NLN_r(O)}.$$
2) Average Width of all Non-Leaf Nodes:
$$AWNLN_r(O) = \frac{\sum_{c \notin Leaf_r(O)} Width_r^O(c)}{NAN_r(O) - NLN_r(O)},$$
where $NAN_r(O)$ is the number of all nodes.
3) Maximal Depth of all Leaf Nodes:
$$MaxDepth_r(O) = \max_{c \in Leaf_r(O)} \left(Depth_r^O(c) \right).$$
4) Maximal Width of Non-Leaf Nodes:
$$MaxWidth_r(O) = \max_{c \notin Leaf_r(O)} \left(Width_r^O(c) \right). \qquad \square$$

Example 3: For the *Family* ontology in Figure 1, the structural metrics for the acyclic relations are as follows.

For the *is-a* relation:

$$ADLN = 3, AWNLN = 15/7, MaxDepth = 3, MaxWidth = 3.$$

For the *has-a* relation:

$$ADLN = 2, AWNLN = 1, MaxDepth = 2, MaxWidth = 1. \qquad \square$$

- *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)

We define the following coupling metrics.

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

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:

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

C. 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, which can also be used for reasoning about the knowledge. For the sake of generality we will not specify such a logic system but simply assume 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 use $O \vdash x$ 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 IV-A, we will assume the existence of a *gold standard* ideal ontology Ω of the domain that contains all concepts, relations, attributes and instances of the domain.

Definition 8: (Semantic Coverage)

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

$$SCov_C^\Omega(O) = D_C / Size_C(\Omega);$$
$$SCov_I^\Omega(O) = D_I / Size_I(\Omega);$$
$$SCov_A^\Omega(O) = D_A / Size_A(\Omega);$$
$$SCov_R^\Omega(O) = D_R / Size_R(\Omega);$$
$$SCov^\Omega(O) = (D_C + D_I + D_A + D_R) / Size(\Omega);$$

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

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:

$$SCov_C^\Omega(O) = 21/22, \quad SCov_I^\Omega(O) = 1, \quad SCov_A^\Omega(O) = 7/8,$$
$$SCov_R^\Omega(O) = 1, \quad SCov^\Omega(O) = 49/51. \quad \square$$

A domain ontology is only semantically correct if its relations are, so we need to check whether they are consistent with the ideal ontology.

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:
$$RCC^\Omega(O) = \frac{||\{c \in C \mid \Omega \vdash c\}||}{Size_C(O)}.$$
2) Average Ratio of Correct Instances:
$$ARCI^\Omega(O) = \frac{\sum\limits_{c \in C} RCI^\Omega(c)}{Size_C(O)}$$
3) Average Ratio of Correct Attributes:
$$ARCA^\Omega(O) = \frac{\sum\limits_{c \in C} RCA^\Omega(c)}{Size_C(O)}.$$
4) Average Ratio of Correct Relations:
$$ARCR^\Omega(O) = \frac{\sum\limits_{r \in R} ||\{(x,y) \in r \mid \Omega \vdash (x,y) \in r\}||}{||R||}.$$

where for each $c \in C$,

$$RCI^\Omega(c) = \begin{cases} 1, & if\ I^c = \emptyset \\ \frac{||\{\alpha \in I^c \mid \Omega \vdash \alpha \in I^c\}||}{||I^c||}, & if\ I^c \neq \emptyset \end{cases}$$

$$RCA^\Omega(c) = \begin{cases} 1, & if\ A^c = \emptyset \\ \frac{||\{\varphi \in A^c \mid \Omega \vdash \varphi \in A^c\}||}{||A^c||}, & if\ A^c \neq \emptyset \end{cases}$$

$\square$

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:

$$RCC^\Omega(O) = 22/23, \quad ARCI^\Omega(O) = 1,$$
$$ARCA^\Omega(O) = 206/207, \quad ARCR^\Omega(O) = 1. \quad \square$$

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 $O' = (C, I, A, R - R')$. $\square$

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:
$$CR(O) = ||C_R||/Size_C(O),$$
where C_R is the largest set of redundant concepts of O.
2) Instance Redundancy:
$$IR(O) = \sum_{c \in C} ||I_R^c||/Size_I(O),$$
where $I_R = \{I_R^c | c \in C\}$ is the largest collection of redundant instances of O.
3) Attribute Redundancy:
$$AR(O) = \sum_{c \in C} ||A_R^c||/Size_A(O),$$
where $A_R = \{A_R^c | c \in C\}$ is the largest collection of redundant attributes of O.
4) Relation Redundancy:
$$RR(O) = \sum_{r' \in R'} ||r'||/Size_R(O),$$
where R' is the largest collection of redundant relations of O. $\square$

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

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

This is because Age can be derived from the attribute *Date of Birth*. $\square$

D. 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. In this paper, we focus on service-oriented computing and ask whether an ontology is a good basis for describing the semantics of services.

Assume that a web service S provides m operations $Op_1, \cdots, Op_m$ and stores $s_1, \cdots, s_l$ types of internal data. We write $Op_i : (x_{i,1}, \cdots, x_{i,n_i}) \rightarrow (y_{i,1}, \cdots, y_{i,k_i})$ to denote that operation Op_i takes a service request containing

parameters $(x_{i,1}, \cdots, x_{i,n_i})$ and responses 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:

- Exp_{Op_i} that describes the functionality of the service operation Op_i, where $i = 1, \cdots, m$;
- $Exp_{x_{i,j}}$ that defines the meaning of the parameter $x_{i,j}$ in the service request, where $j = 1, \cdots, n_i$;
- $Exp_{y_{i,j}}$ that describes the meaning of the parameter $y_{i,j}$ in service response, where $j = 1, \cdots, k_i$;
- Exp_{s_i} that 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. We start with metrics to 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 $DState^S(O)$, is defined as follows.

$$DState^S(O) = l_d/l,$$

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

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

$$DFun^{Op_i}(O) = ND_{Op}/(n_i + k_i + 1),$$

where $ND_{Op} = n_d + k_d + Op_d$, n_d and k_d are the numbers of request parameters and response parameters definable in O, respectively. Op_d = 1, if the functionality of the operation is definable in O; otherwise, $Op_d = 0$.

The definability of the service system S in ontology O, denoted by $DServ^S(O)$, is defined as follows.

$$DServ^S(O) = DState^S(O) \times \sum_{i=1}^{m} DFun^{Op_i}(O)/m. \qquad \square$$

Better metrics of definability of a service in an ontology should take into consideration the complexity of the semantic descriptions. Here we use the following simple complexity metrics of expressions.

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.

$$Cmplx(Exp) = NOP + NV,$$

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.

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

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

$$CFun^{Op_i}(O) = Cmplx(Exp_{Op_i}) + \sum_{j=1}^{n_i} Cmplx(Exp_{x_{i,j}}) + \sum_{j=1}^{k_i} Cmplx(Exp_{y_{i,j}})$$

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

$$CServ^S(O) = CState^S(O) + \sum_{i=1}^{m} CFun^{Op_i}(O). \qquad \square$$

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:

$$Register : (n : String, f, m : Person, d : Date) \rightarrow (id : Nat)$$
$$QueryParents : (id : Nat) \rightarrow (f, m : Person)$$

Table I gives the semantic descriptions of the service and the complexity measures. □

Table I
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	2
QueryParents	f, m are in List of Person. f and m are the Father and Mother of the Person. The Person's NI number is id.	2
$CFun^{Register}$		12
$CFun^{QueryParents}$		8
$CServ$		22

We have implemented the metrics defined in this paper as a part of our formal engineering environment of services.

V. CASE STUDY

This section reports a case study on real world examples of web services.

The purpose of the case study is to assess the quality of the ontology generated from formal specification of web services. The subjects of the case study are 3 different web services of weather information. They are referred to as *WS1* [26] , *WS2* [27] and *WS3* [28] respectively in the sequel. One weather ontology [29] is used as the *gold standard (GS)* of the domain knowledge.

Table II
NUMBER OF UNITS IN SPECIFICATIONS OF *Weather Services*

Package	WS1	WS2	WS3
Definition of Weather Entities	12	14	12
Definition of External Entities	3	6	2
Definition of Operations	5	10	3

Table III
SIZES OF THE ONTOLOGY MODULES

Ontology Module		#Cls	#Insts	#Attrs	#Rels
Domain	GS	14	11	17	10
	WS1	12	2	26	13
	WS2	13	72	31	11
	WS3	12	9	28	11
Op	WS1	8	0	6	18
	WS2	19	0	28	35
	WS3	5	0	6	9

A. Specifications and Ontologies of Web Services

These three weather information web services are formally specified in SOFIA. A SOFIA specification consists of a collection of specification units. Each unit specifies one concept of the real world or a type of software entity. This enables the specification structure to reflect the structure of software systems and also the conceptual structure of real world. Units closely related to each other are encapsulated into a package. The specification of each weather web service consists of three packages. The first contains units that specify the structure and semantics of various application domain related entities and concepts, i.e. concepts related to weather. The second is for external entities and concepts, such as *location*, *city* etc. The third is for specifying the operations of the services, including the valid requests and responses of service operations as well as the semantics of the operations. The numbers of specification units in the specifications are shown in Table II.

Using the TrS2O tool [14], we transformed specifications of the web services into ontologies. Each package transforms into an ontology module. We focus on the weather and service operation modules of ontologies in our case study. Table III gives the sizes of these ontology modules, 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 IV 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.

B. Results of Evaluation

The metrics defined in Section IV are applied to the ontologies through using our implementation of the metrics. The results are given in Table V.

In Table V, the second column is the metrics used to evaluate the quality factors given in the first column. The data in columns of *WS1*, *WS2* and *WS3* are the values of the ontology on the metrics given in the second column.

It is worth noting that the metrics produce quantitative values in different ranges. For example, the metrics for measuring completeness and conciseness produce relative values, but some metrics for structuredness and usability produce absolute values. To make the results of measurement on various metrics clearly indicate the quality of the ontology on the specific quality attributes and factors, we employ the subrange technique of data normalisation, which is widely used in software quality evaluation. In particular, for each metric, we divide the range into 5 subranges so that a value in the subrange 1 means poorest quality on the specific attribute and factor as measured by the metric, while a value in the subrange 5 indicates a best possible quality. Table VI gives the details of the mapping from metrics' values to the subranges.

Figure 4 shows the result of evaluation of three ontologies after the metrics are normalised. Figure 5 compares three ontologies using the average normalised scores of the metrics on each quality attributes. As shown in Figure 5, *WS2* gets the highest completeness and *WS1* gets the highest score for well structuredness. It is apparent that all of these three ontologies scored very well on conciseness, structuredness and usability. The weakest aspect of quality for all three ontologies is on the completeness. This is because the web services in the case study do not use all knowledge about weather contained in the gold standard ontology. An overall conclusion that can be drawn from the evaluation is that the quality of these ontologies generated from the algebraic specifications is high. This demonstrates that the transformation of formal specification into ontological semantic descriptions of web services is both feasible and practically useable.

VI. CONCLUSION

The main contributions of the paper are as follows.

First, we proposed a framework for evaluation of the quality of ontologies on four quality attributes, which is

Table IV
COMPARISON OF CLASSES BETWEEN GS AND WS1

Class Types	Classes	No.
Defined both in GS and WS1	Wind, Dew Point Temperature, Cloud Cover, Weather State, Humidity, Atmospheric Pressure, Temperature, *Point*	8
Defined in GS and derivable from WS1	Weather Phenomenon	1
Defined in GS but not in WS1	Air Pollution, Precipitation, Solar Irradiance, Weather Condition, *SpatialThing*	5
Defined in WS1 but not in GS	Visibility, Speed, Direction, TEMUnit	4

Table V
EVALUATION OF THE *Weather* ONTOLOGIES

Quality Factor	Metric	WS1	WS2	WS3
Vocabulary Coverage	Cov_C^Ω	0.57	0.57	0.43
	Cov_I^Ω	0	0.64	0
	Cov_A^Ω	0.65	0.59	0.29
	Cov_R^Ω	0	0	0
	Cov^Ω	0.37	0.48	0.21
Semantic Coverage	$SCov_C^\Omega$	0.64	0.64	0.43
	$SCov_I^\Omega$	0	0.64	0
	$SCov_A^\Omega$	0.65	0.59	0.29
	$SCov_R^\Omega$	0.5	0.5	0
	$SCov^\Omega$	0.48	0.6	0.21
Semantic Compatibility	RCC^Ω	0.69	0.64	0.5
	$ARCI^\Omega$	0.92	0.86	0.83
	$ARCA^\Omega$	0.53	0.47	0.23
	$ARCR^\Omega$	0.26	0.29	0
Redundancy	CR	0	0	0
	IR	0	0	0
	AR	0	0	0
	RR	1	1	1
Relation-Based Cohesion	NRN	1	2	2
	NLN	9	9	9
	MaxSPL	3	3	3
	NIC	0	0	0
	TNRNR	13	11	14
	ANRNR	13.0	5.5	7.0
Acyclic Cohesion	ADLN	2.56	2.33	2.89
	AWNLN	2.6	2.75	2.0
	MaxDepth	3	3	3
	MaxWidth	8	7	4
Coupling	NERA	0	0	0
	NERR	1	0	0
	REC	0.08	0	0
Definability	DState	1	1	1
	DFun	1	1	1
	DServ	1	1	1
Description Complexity	CState	2	4	1
	CFun	12	24	6
	CServ	14	28	7

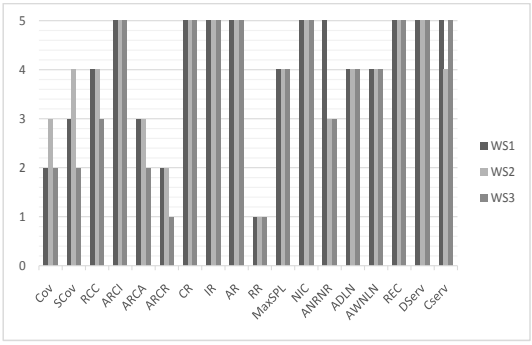

Figure 4. Normalised Metric Scores of the Ontologies

decomposed into eight factors and measured by a set of thirty seven metrics. All the metrics are implemented as a part of our formal engineering environment for service oriented computing. The framework has the following three distinctive features.

1) *Objectiveness*: The evaluation is completely based on objective metrics. Many of the metrics are novel, such as those that compare an ontology against a gold standard ontology and those about the semantics of ontology.
2) *Language Independence*: Our metrics are defined based on a 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. They are therefore generally applicable.
3) *Usage Orientation*: We developed a set of new metrics for the evaluation of ontologies in its usage context of describing the semantics of web service. They have been successfully applied in the evaluation of the web services in our case study.

Table VI
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]

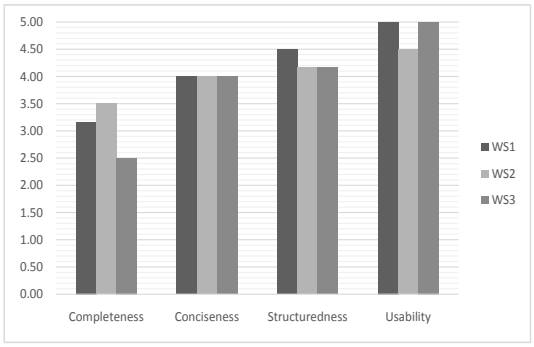

Figure 5. Comparison of Ontologies on Quality Attributes

Secondly, we have conducted a case study with the proposed framework on 3 real-world examples. The case study demonstrated that ontologies obtained from the transformation of algebraic specifications are of satisfactory quality.

Thus, for the first time we have proved the feasibility and effectiveness of the transformation from algebraic specifications to ontological descriptions of services. Therefore, the gap between formal methods and ontological approaches to semantic description of web services can be bridged.

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 reported in this paper is a part of our formal service engineering environment. It serves as a means of automated evaluation of the quality of the specification and design of web services. We are conducting more empirical studies for the identification of more quality attributes and factors, and metrics as well. Another 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 interaction in the evaluation of ontologies.

ACKNOWLEDGEMENT

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

REFERENCES

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

[2] Mallraith, S. A., Son, T. C. and Zeng, H, Semantic web services, *IEEE Intelligent Systems* (March/April), pp. 46–53, 2001.

[3] Martin, D., et al., *OWL-S: Semantic Markup for Web Services*, http://www.w3.org/Submission/OWL-S/, Nov. 2004. (last access:May 25, 2015)

[4] Kopecky, J., Gomadam, K. and Vitvar, T., hRESTS: An HTML microformat for describing RESTful web services, in *Proc. of WI-IAT'08*. Dec. 2008, pp619–625.

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

[6] Lathem, J., Gomadam, K. and Sheth, A. P., SA-REST and Smashups: Adding semantics to RESTful services, in *Proc. of ICSC'07*, pp469–476, 2007.

[7] Bruijn, J., Lausen, H., Polleres, A. and Fensel, D., The web service modelling language WSML: An overview, in *Proc. of the 3rd Europ. Semantic Web Conf*. pp590–604, 2006.

[8] Zhu, H. and Yu, B., Algebraic specification of web services, in *Proc. of QSIC 2010*, pp457–464, 2010.

[9] Liu, D., Zhu, H. and Bayley, I., SOFIA: An algebraic specification language for developing services, in *Proc. of SOSE 2014*, pp70–75, 2014.

[10] Gaudel, M. C. and Gall, P. L., Testing data types implementations from algebraic specifications, *CoRR*, vol. abs/0804.0970, 2008.

[11] Yu, B., Kong, L., Zhang, Y. and Zhu, H., Testing java components based on algebraic specifications, in *Proc. of ICST 2008*, pp190–199, 2008.

[12] Liu, D., Liu, X., Zhang, X., Zhu, H. and Bayley, I., Automated testing of web services based on algebraic specifications, in *Proc. of SOSE 2015*, pp143–152, 2015.

[13] Liu, D., Zhu, H. and Bayley, I., From algebraic formal specification to ontological description of service semantics, in *Proc. of ICWS 2013*, pp579–586 ,2013.

[14] ——, Transformation of algebraic specifications into ontological semantic descriptions of web services, *Int'l J. of Services Computing* 2(1), pp58–71, 2014.

[15] Gomez-Perez, A., Towards a framework to verify knowledge sharing technology, *Expert Systems with Applications* 11(96), pp519–529, 1996.

[16] Vrandecic, D., Ontology evaluation, in *Handbook on Ontologies*, S. Staab and R. Studer (eds.), Springer, pp293–313, 2009.

[17] Gangemi, A., Catenacci, C., Ciaramita, M., et al., Modelling ontology evaluation and validation, *The Semantic Web: Research and Applications*. Springer, pp140–154, 2006.

[18] Duque-Ramos, A., Fernndez-Breis, J. T., Stevens, R., et al., OQuaRE: A SQuaRE-based approach for evaluating the quality of ontologies, *Journal of Research and Practice in Information Technology* 43(2), pp159–176, 2011.

[19] Garca, J., Garca-Pealvo, F, J. and Thern, R., A survey on ontology metrics, *Knowledge Management, Information Systems, E-Learning, and Sustainability Research*, pp22–27, 2010.

[20] Tao, H., Orme, A. M. and Etzkorn, L. H., Cohesion metrics for ontology design and application, *J. of Comp. Sci.* 1(1), pp107–113,2005.

[21] Yang, Z., Zhang, D. and Chuan, Y.E., Evaluation metrics for ontology complexity and evolution analysis, In *Proc. of ICEBE 2006*, pp162 - 170, 2006.

[22] Orme, A. M., Tao, H. and Etzkorn, L. H., Coupling metrics for ontology-based system, *IEEE Software* 23(2), pp102–108, 2006.

[23] Oh, S., Yeom, H. Y. and Ahn. J., Cohesion and coupling metrics for ontology modules, *Inf. Technol. and Management* 12(2), pp81-96, June 2011.

[24] Gruber, T., A Translation Approach to Portable Ontology Specifications. *Knowledge Acquisition* 5(2), pp199-220, 1993.

[25] Zavitsanos, E., Paliouras, G. and Vouros, G. A., Gold standard evaluation of ontology learning methods through ontology transformation and alignment, *IEEE Trans. on Knowledge and Data Engineering* 23(1), pp1635 –1648, Nov. 2011.

[26] Webservicex.net, http://www.webservicex.net/globalweather.asmx. (last access: Nov.10, 2015)

[27] OpenWeatherMap.org, http://openweathermap.org/. (last access: Nov.10, 2015)

[28] Webxml.com.cn, http://www.webxml.com.cn/WebServices/WeatherWS.asmx. (last access: Nov.10, 2015)

[29] Automation System Group, https://www.auto.tuwien.ac.at/downloads/thinkhome/ontology/WeatherOntology.owl. (last access: Nov.10, 2015)