Journal of Southeast University (English Edition) Vol. 22, No. 3, pp. 365 – 369 Sept. 2006 ISSN 1003—7985

Semantic analysis approach for construction of OWL ontology

Liu Hongxing Yang Qing

(School of Computer Science and Technology, Wuhan University of Technology, Wuhan 430063, China)

Abstract: A semantic analysis approach is proposed, by which semantic relationships between concepts are identified and defined, and then mapped or transformed to OWL (web ontology language) ontology. The most common abstractions (namely, inclusion, aggregation and association) and their implication in ontology are discussed; then the OWL implementation for three abstractions are analyzed and illustrated. Taxonomies, constraints on properties for each class, and the relations between taxonomies in OWL ontology are established after all the semantic relationships are identified and described. This research is the basis for the development of the ontology conceptual model (OCM) and the mapping from OCM to OWL ontology.

Key words: ontology; semantic web; OWL

The purpose of ontology is to establish the shared and common understanding of some domain that can be communicated across people and computers[1]. Because of its potential for supporting the interoperability and integration among heterogeneous resources on semantic layer, ontology is now being widely thought valuable for many areas.

Semantic web is a vision for the future of the web, on which the information is given explicit meaning. Such a commitment makes it easier for machines to automatically manipulate and integrate information on the web. In order to achieve this mission, it is necessary to establish an additional layer which captures and represents the semantics of information on the web. OWL (web ontology language) was thus promoted as a standard for web ontology language by W3C. It is foreseeable that, in the future, a considerable number of ontologies will be created in OWL and available on the web. Many reasoning tools or software agents based on OWL ontology will be put into use.

Although some development tools have provided support for OWL ontology[2-3], the construction of OWL ontology is still time-consuming and easy to be out of control. The reason is that the methodologies for building ontologies are relatively immature. There is not a widely accepted methodology so far. Ontology development still largely depends on the experience of developers.

Ontology is a formal, explicit specification of a shared conceptualization[1], and it may take a variety of forms. Generally, ontologies explicitly represent the concepts of some domain, relationships between these concepts, and relevant constraints on concepts. The analysis and identification of basic semantic relationships between concepts should be central activities in the process of ontology development. In this paper, we propose a semantic analysis approach, by which semantic relationships between concepts are identified and expressed. The skeletal OWL ontology is established after the semantic relationships are mapped or transformed to OWL descriptions.

1 Architecture for Construction of OWL Ontology

Ontology is a kind of software or a part of software for knowledge representation. The knowledge is captured and represented in specific ways and finally transformed to computable expression in the process of ontology development. The same as with the development of other kinds of software, the development of ontology should also be a modeling driven process. The most important activities involved are construction and transformation of models. Fig. 1 describes the architecture for ontology representation and transformation, in which knowledge in the real world is identified and expressed in three layers.

The real world is composed of things having properties that are inherent and exist objectively whether or not they are observed or recorded. A property can be characterized as descriptive or associative according to its semantic role: a descriptive property expresses some characteristic of a thing; an associative property relates two or more things[4]. What we are

Received 2006-04-25.

Foundation item: The Natural Science Foundation of Hubei Province (No. 2004ABA040).

Biography: Liu Hongxing(1963—), male, associate professor, liuhx@public. wh. hb. cn.

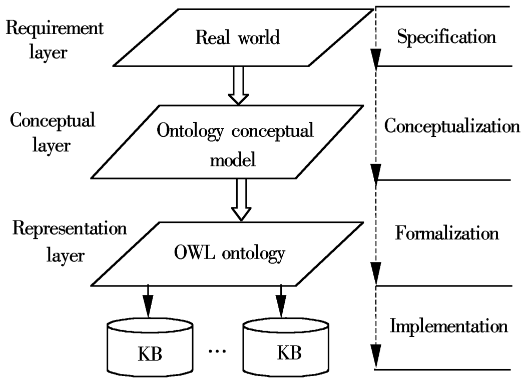

Fig. 1 Architecture for ontology representation and transformation

concerned with are those highly relevant things and properties in a domain. This is called domain knowledge in this paper.

The ontology conceptual model (OCM) is the abstraction of domain knowledge in the real world; the things and properties of the real world are represented by concepts, attributes, semantic relationships, and constraints in abstract and visual ways. The OCM is independent of any specific knowledge language, and is the common understanding or agreement shared by domain experts and ontology developers. So the conceptual layer is the bridge for the transformation from domain knowledge to computable expression.

In the representation layer, ontology is the model which expresses domain knowledge in computable or semi-computable knowledge language. The knowledge language can be roughly classified as description-logic based, and frame and first-order-logic based[5]. OWL is description-logic based ontology language; the ontology described in OWL is called OWL ontology. Knowledge base (KB) is the instantiation of OWL ontology, and is composed of substantial and meaningful individuals of some domain.

Development-oriented activities include specification, conceptualization, formalization and implementation[6]. In the context of this research, conceptualization refers to the activity which analyzes and abstracts the domain knowledge, and models the domain knowledge in OCM; the domain knowledge expressed in OCM is transformed to OWL ontology in the process of formalization; implementation is the activity of building related KB. In this paper, the focus is on the formalization, and the construction of OWL ontology is discussed in detail.

2 Semantic Relationships in Ontology Conceptual Model

In OCM, concept and attribute are respectively the abstract representation of "thing" and "property" in the real world; the abstract representation of descriptive property and associative property are respectively called simple-attribute and associative-attribute.

Semantic relationships are the abstraction of the inherent relations among things and properties. The database community has made much effort to identify and represent semantic relationships. Storey[7] summarized and presented a taxonomy of semantic relationships. Goldstein et al. [4] further analyzed the three most common abstractions (namely, inclusion, aggregation and association). The semantic relationships existing in the real world are universal. Ontology should be able to reveal the most common abstractions. Three abstractions and their implication in ontology are defined and discussed as follows.

2.1 Inclusion

The inclusion relationship represents the subtype/supertype relationship. Two kinds of inclusion should be distinguished in ontology.

1) Is-a

Ca and Cb are concepts. Ca is-a Cb means that Ca is a specialization of Cb and Cb is a generalization of Ca; Ca is relatively a generic concept and Cb is a specific concept. For example, "Graduate" is a "Student", "Graduate" is a specialization of "Student", and "Student" is a generalization of "Graduate".

The most important feature of "is-a" is inheritance: Ca is-a Cb, Ca thus inherits all the attributes of Cb. Multiple inheritances are permitted in OCM: Ca is-a Cb, Ca is-a Cc, Ca thus inherits the attributes from both Cb and Cc. Another important feature of "is-a" is transitivity: Ca is-a Cb, Cb is-a Cc, then, Ca is-a Cc.

2) Subset hierarchy

When a generic concept is defined as the union of non-overlapping specific concepts, the specific concepts form a partition of the generic concept. For example, "Student" is a union of "Graduate" and "Undergraduate", so "Graduate" and "Undergraduate" comprise a partition of "Student".

All the partitions relating to a generic concept form a subset hierarchy. For example, "Student" is the generalization of "Undergraduate", "Graduate", "Part-time", "Full-time", "Overseas" and "Home". Three groups, "Undergraduate" and "Graduate", "Part-time" and "Full-time", "Overseas" and "Home", form three partitions of "Student". Clearly, overlapping exists among three partitions.

The taxonomy among a group of concepts is established after all the inclusion relationships are identified and expressed.

2.2 Aggregation

Aggregation is an abstraction so that the relationship among concepts is considered as a higher level concept. For example, the aggregation of ("name", "affiliatedSubject", "degree", "duration", "description" and "steeringCommittee") represents a concept "Program"; conversely, "name", "affiliatedSubject", "degree", "duration", "description", "steering-Committee" are respectively the component-of concept "Program". In some literature, component-of is called part-of.

If Ca is component-of Cb, Ca will be regarded as an attribute of Cb. A concept could be an aggregation of both single-attributes and associative-attributes. A concept and its attributes are thus grouped together by aggregation.

2.3 Association

Association is a triple relationship which represents the link of two concepts by an associative-attribute. There are two kinds of association existing in ontology.

1) Instance-of

Cb is a concept, and Ax is an associative-attribute of some concept. Ax is instance-of Cb means that the type of Ax is Cb and the occurrence of Ax should be an instance of Cb.

For example, the concept "Program" has attributes ("name", "degree", "duration", "affiliatedSubject", "steeringCommit-tee"). "affiliatedSubject" is an associative-attribute. Its type is concept "Subject", and each occurrence of "affiliated-Subject" is an instance of the concept "Subject". So "affiliatedSubject" establishes a link between "Program" and "Subject".

2) Member-of

Cb is a concept, and Ax is an associative-attribute of some concept. Ax is member-of Cb means that the type of Ax is Cb and the occurrence of Ax should be a set of instances of Cb. In other words, where the occurrence of Ax is an aggregation of instances of Cb, Ax could be considered as another (higher) concept.

For example, the concept "Program" has attributes ("name", "degree", "duration", "affiliatedSubject", "steeringCommit-tee"). "steeringCommittee" is also an associative-attribute. Each occurrence of "steeringCommittee" is a set of instance of concept "Academic". So "steeringCommittee" establishes another kind of link between "Subject" and "Academic", and can be considered as a concept because it is the aggregation of instances of concept "Academic".

The cardinality of Ax is equal to the cardinality of the set attached to the occurrences of Ax. Obviously, the cardinality of Ax with relationship member-of is not less than one (that is, the set which is attached to Ax should not be empty). For example, the cardinality of "steeringCommittee" is five, which means that each occurrence of "steeringCommittee" should consist of five instances of "Academic".

The cardinality of Ax with relationship instance-of can be regarded as exactly one, for the occurrence of Ax can be regarded as a set which consists of exactly one instance of Cb. To some extent, the relationship instance-of is the specialization of relationship member-of; but the difference is that an associative-attribute with a relationship instance-of cannot be considered as a higher concept.

3 Semantic Analysis Based Construction of OWL Ontology

Generally, ontology includes taxonomies, relationships between taxonomies, and constraints on attributes for each concept. As discussed above, the concepts with attributes are grouped into taxonomy after inclusion (i.e., is-a, and subset hierarchies) and aggregation (conversely, component-of or part-of) are identified and expressed. The relationships between taxonomies are established after association (i.e., instance-of, and member-of) is identified and expressed.

We analyze and illustrate the OWL implementation for three abstractions. Concept, simple-attribute and associative-attribute, and instance are respectively mapped to class, datatype property and object property, and individual in OWL[8-9].

3.1 Implementation of inclusion

Class axiom defines necessary and sufficient characteristics of class by class descriptions. The basic class descriptions are listed below in the left column, and the right column lists more advanced descriptions based on description logic.

● owl: Class	● owl: unionOf
● rdfs: subClassOf	● owl: intersectionOf
● owl: equivalentClass	● owl: complementOf
● owl: disjointWith	

The class definition blocks below give the definitions for class "Student", "Undergraduate" and "Graduate"; ("Under-graduate", "Graduate") forms a partition of "Student". The definition for the partition is incorporated into the class definition blocks. Other partitions of "Student", ("Part-time", "Full-time") and ("Oversea", "Home"), can be defined in the same way. The taxonomy "Student" is established after all the "Inclusion" relations among "Student" and its subclasses are well defined.

```
〈owl: Class rdf: about = "#Student"〉
  〈owl: equivalentClass〉
    〈owl: Class〉
      〈owl: unionOf rdf: parseType = "Collection"〉
        〈owl: Class rdf: about = "#Undergraduate"/〉
        〈owl: Class rdf: about = "#Graduate"/〉
      〈/owl: unionOf〉
    〈/owl: Class〉
  〈/owl: equivalentClass〉
〈/owl: Class〉
〈owl: Class rdf: about = "#Undergraduate"〉
  〈rdfs: subClassOf rdf: resource = "#Student"/〉
〈/owl: Class〉
〈owl: Class rdf: about = "#Graduate"〉
  〈rdfs: subClassOf rdf: resource = "#Student"/〉
  〈owl: disjointWith rdf: resource = "#Undergraduate"/〉
〈/owl: Class〉
```

3.2 Implementation of aggregation and association

1) Property axiom

The property in OWL is the independent existence (i. e., the resource in the term of RDF). A property axiom defines characteristics of a property; the basic descriptions for property definitions are listed as follows:

- owl: ObjectProperty
- owl: DatatypeProperty
- rdfs: domain
- rdfs: range

The property definition blocks below give a definition for the properties "registeredTime" and "registeredProgram". The first definition block clearly indicates that class "Student" has an object property "registeredTime", and the type of "registeredTime" is class "Program". The second definition block indicates that class "Student" has a datatype property "registeredTime" with data type xsd: dateTime.

```
〈owl: ObjectProperty rdf: ID = "registeredProgram"〉
〈rdfs: domain rdf: resource = "#Student"/〉
〈rdfs: range rdf: resource = "#Program"/〉
〈/owl: ObjectProperty〉
〈owl: DatatypeProperty rdf: ID = "registeredTime"〉
〈rdfs: domain rdf: resource = "#Student"/〉
  〈rdfs: range rdf: resource = "&xsd; dateTime"/〉
〈/owl: DatatypeProperty〉
```

2) Property restriction

Based on the property axiom, a property with constraints can be incorporated into a class axiom by a property restriction. A property restriction is a special kind of class description and should be the most important part in a class axiom.

OWL provides two kinds of property restriction: value constraint and cardinality constraint. A value constraint puts constraints on the range of the property when this property is incorporated into a class axiom (domain of the property). A cardinality constraint puts constraints on the number of values that a property can take in the context of the particular class description. The descriptions for property restrictions are as follows: the left column is for value constraints and the right column is for cardinality constraints.

● owl: allValueFrom	● owl: maxCardinality
● owl: someValueFrom	● owl: minCardinality
● owl: hasValue	● owl: cardinality

The "aggregation" relationships among a class and its properties are well defined after all the property restrictions are incorporated into the class axiom. The "association" relation can be implemented by the cardinality constraints as illustrated below.

As discussed in section 2, the class "Program" has properties ("name", "degree", "duration", "affiliatedSubject", "steeringCommittee"). "affiliatedSubject" is instance-of "Subject". This relation can be interpreted so that the occurrence of "affiliatedSubject" for each individual of "Program" is a set which consists of exactly one instance of "Subject". "steeringCommittee" is member-of "Staff" and the cardinality of "steeringCommittee" is not less than five. This relation can be interpreted so that the occurrence of "steeringCommittee" for each individual of "Program" is a set that consists of at least five instances of "Staff". The implementation of properties "affiliatedSubject" and "steeringCommittee", and part of implementation of class "Program" are as follows.

Obviously, "instance-of" can be defined by description owl: cardinality; "member-of" can be defined by descriptions owl: maxCardinality and owl: minCardinality.

```
〈owl: Class rdf: about = "#Program"〉
〈owl: Restriction〉
〈owl: onProperty rdf: resource = "#affiliatedSubject "/〉
〈owl: cardinality rdf: datatype =
"&xsd; nonNegativeInteger"〉1〈/owl: minCardinality〉
〈/owl: Restriction〉
〈owl: Restriction〉
〈owl: onProperty rdf: resource = "#steeringCommittee"/〉
〈owl: minCardinality rdf: datatype =
"&xsd; nonNegativeInteger"〉5〈/owl: minCardinality〉
〈/owl: Restriction〉
〈/owl: Class〉
```

```
〈owl: ObjectProperty rdf: ID = " affiliatedSubject "〉
〈rdfs: domain rdf: resource = "#Program"/〉
〈rdfs: range rdf: resource = "#Subject"/〉
〈/owl: ObjectProperty〉

〈owl: ObjectProperty rdf: ID = " steeringCommittee "〉
〈rdfs: domain rdf: resource = "#Program"/〉
〈rdfs: range rdf: resource = "#Staff"/〉
〈/owl: ObjectProperty〉
```

The skeletal OWL ontology (i. e., taxonomies, the relationships between taxonomies, some constraints on property) has been well established. Further work remains to thoroughly complete the constrain definitions in terms of both property axioms and property restrictions.

4 Conclusion

The construction of OWL ontology is the process driven by analysis and transformation of semantic relationships. The taxonomy for a class is established after all the "inclusion" relationships among this class and its subclasses are defined; the relationship between taxonomies is established after the "association" relationship between two classes is described; some constraints on property for a class are defined when the property restrictions are aggregated into a class axiom. The skeletal OWL ontology is thus well established.

Based on the research in this paper, we first develop OCM, which is the extension of UML and provides support for the expression of the most common semantic relationships. OCM facilitates developers to model the domain knowledge in visual and natural ways. Secondly, we further define the mapping rules from OCM to OWL descriptions, so as to support the automatically or semi-automatically transformation from OCM to OWL ontology.

References

[1] Studer Rudi, Benjamims V Richard, Fensel Dieter. Knowledge engineering: principles and methods [J]. *Data & Knowledge Engineering*, 1998, **25**(102): 161 - 197.

[2] Knublauch Holger, Fergerson Ray W, Fergerson W, et al. The protégé OWL plugin: an open development environment for semantic web applications[A]. In: *Proc of ISWC* 2004, *Lecture Notes in Computer Science*[C]. Springer, 2004, **3298**: 229 - 243.

[3] Horridge Matthew, Knublauch Holger, Rector Alan, et al. A practical guide to building OWL ontology using the protégé-OWL plugin and COODE tools [EB/OL]. (2004-08-27) [2006-04-10]. http://www.co-ode.org/resources/tutorials/ProtegeOWLTutorial.pdf.

[4] Goldstein R C, Storey V C. Data abstraction: why and how? [J]. *Data & Knowledge Engineering*, 1999, **29**(3): 293 - 311.

[5] Corcho Oscar, Fernández-López M, Gómez-Pérez A. Methodologies, tool, and languages for building ontologies, where is their meeting point? [J]. *Data & Knowledge Engineering*, 2003, **46**(1): 41 - 64.

[6] Fernández-López M, Gómez-Pérez A, Pazos-Sierra A, et al. Building a chemical ontology using methontology and the ontology design environment [J]. *IEEE Intelligent Systems & Their Applications*, 1999, **4**(1): 37 - 46.

[7] Storey V C. Understanding semantic relationship [J]. *Vary Large Data Base Journal*, 1993, **2**(4): 455 - 488.

[8] W3C. OWL web ontology language guide [EB/OL]. (2004-02-10) [2006-04-10]. http://www.w3.org/TR.

[9] W3C. OWL web ontology language reference [EB/OL]. (2004-02-10) [2006-04-10]. http://www.w3.org/TR.

构造 OWL 本体的语义分析方法

刘洪星　　杨　青

(武汉理工大学计算机科学与技术学院,武汉 430063)

摘要:提出了一种支持本体构造的语义分析方法,该方法识别和定义概念间的语义联系,并将概念间的语义联系映射或转换成 OWL 原语. 首先讨论了 3 种最常见的抽象(即包含、聚集和联系)以及它们在本体中含义,然后分析并用实例演示了 3 种抽象的 OWL 实现方法. 当所有的语义联系都被识别和描述后,就产生了 OWL 本体的主干部分(即分类、对属性的约束、以及分类间的关系). 该研究是发展本体概念模型(OCM)以及实现模型间映射或转换的基础.

关键词:本体;语义网;OWL

中图分类号:TP182