WUJNS
Wuhan University Journal of Natural Sciences

Vol.11 No.1 2006 057-062

Article ID:1007-1202(2006)01-0057-06

A Framework of Semantic Information Representation in Distributed Environments

ZHANG Lin[1], CHEN He-ping[2]

1. Department of Computer and Information Engineering, Wuhan Polytechnic University, Wuhan 430023, Hubei, China;

2. Department of Information, Wuhan University of Science and Technology, Wuhan 430081, Hubei, China

Abstract: An information representation framework is designed to overcome the problem of semantic heterogeneity in distributed environments in this paper. Emphasis is placed on establishing an XML-oriented semantic data model and the mapping between XML data based on a global ontology semantic view. The framework is implemented in Web Service, which enhances information process efficiency, accuracy and the semantic interoperability as well.

Key words: semantic; XML; ontology; Web service

CLC number: TP 391

Received date: 2005-05-06

Foundation item: Supported by the Natural Science Foundation of Hubei Province of China (2003ABA049) and Natural Science Foundation of Hubei Education Agency of China (Z200511005, 2003A012).

Biography: ZHANG Lin (1979-), female, Master, research direction: semantic Web and XML data management.
E-mail:zhl_wh@sina.com

0 Introduction

On the global information infrastructure, the great expansion of the communication networks has made available to users a large number of autonomous data repositories, however, these repositories present different structures and semantics result from that distinct data sources may use different modeling methods, making very difficult to share and exchange information. Recently, there have been some works, such as OntoBroker in Ref. [1] and system in Ref. [2], that focus on the umform representation for these heterogeneous data source, to facilitate semantic information access and integration.

This paper builds on previous works, solves the problems of semantic mapping between XML-based semi-structured data, and provides an information representation framework to improve the semantic interoperability of information in distributed environment.

1 Ontology: A Key to Information Semantic

Communication among computer systems in distributed environments is difficult due to lack of a standard in syntax or semantic, which becomes much worsen along with a dramatic growth in the number of information. Recently, a widely accepted solution to lighten this problem consists in taking XML as a common syntax for exchanging heterogeneous information, especially those complex semi-structured information in Web. In spite of many positive features, it is wrong to assume that XML also eliminate semantic heterogeneity. Firstly, XML Schema, a modeling tool of XML data, cannot define standard domain terminology. The semantics specified by XML Sche-

ma are only for human consuming, however, would not be understood and deployed by computers without particular explanation. Secondly, it does not ensure consistent use of terminology in different XML documents that have the same set of labels. The problem of semantic heterogeneity will still exist in a world where all data is exchanged using XML structured according to XML schema.

In this paper we make use of ontology to translate between semantically heterogeneous XML data. Ontology provides a formal, shared specification of concepts, their relationships, and other realities of some domain, which can reduce or eliminate confusion of terminologies, enable computers to process domain knowledge more precisely and conveniently. Since 1990s, ontology has developed from AI field to computer field, and becomes a popular research topic in various communities such as knowledge engineering, natural language processing, intelligent information integration and knowledge management, etc. The existing famous ontologies are CYC, WordNet, CIA World FactBook, $(KA)^2$ [3] and Tourism.

Various kind of formal languages are used for representing ontologies, among other things Description Logics, Frame-Logic[4], and special languages for Semantic Web[5] such as OXL, OWL and OIL, etc. Table 1 shows an ontology represented in Frame-Logic, which is a subset of $(KA)^2$ ontology that provides a conceptual model of researchers and publications domain.

In Table 1, the formula $c_1 :: c_2$ means that c_1 is a subclass of c_2. $c[a \Rightarrow\!\!> r]$ means that an attribute a is of domain c and range r. $o : c[a \Rightarrow\!\!> v]$ means that o is instance of c and has the value v for a. <— means logical implication and <—> logical equivalence.

Table 1 ontology in Frame-Logic

Concept Hierarchy	Attribute Definitions	Rules

Object[].
Person :: Object.
Employee :: Person.
AcademicStaff :: Employee.
Researcher :: AcademicStaff.
PhDStudent :: Researcher.
Student :: Person.
PhDStudent :: Student.
Publication :: Object.
Book :: Publication.
Article :: Publication.
	Person [name =>> STRING;
	mail =>> STRING;
	address =>> STRING;
	publication =>> Publication].
	Employee [employeeNo =>> STRING].
	AcademicStaff [supervises =>> PhDStudent].
	Researcher [cooperatesWith =>> Researcher].
	Student [studentID =>> NUM].
	PhDStudent [supervisor =>> AcademicStaff].
	Publication [author =>> Person;
	title =>> STRING;
	year =>> NUM;
	abstract =>> STRING].
		FORALL Pers1, Pers2
		Pers1: Researcher[cooperatesWith ->> Pers2] <—>
		Pers2: Researcher[cooperatesWith —>> Pers1].
		FORALL Pers1, Publ1
		Publ1: Publication[author —>>Pers1] <—>
		Pers1: Person[publication —>> Publ1].
		FORALL Pers1, Pers2
		Pers1: PhDStudent[supervisor —>> Pers2] <—>
		Pers2: AcademicStaff[supervises —>> Pers1].

To solve the semantic heterogeneity, we could firstly make use of ontology to provide a machine-processable semantics that can be communicated between different data sources, and then translates between XML data referring to this global semantic view, the complexity here is $O(n)$, which is better than that of translation without ontology as a intermediary-$O(n^2)$.

In the next section, we will present a semantic data model via comparing ontology with XML Schema.

2 Semantic Data Model

2.1 From Conceptual Model to Data Model

Although ontology and XML schema[6] serve very different purposes, the former is a means to specify domain theories[7, 8] and the latter is a means to provide integrity constraints for information sources, however, they have one main goal in common: both provide model and vocabulary for describing information sources that are aimed at interoperation, what's the relationship between them? With regard to this question, we refer to the multi-level models theory of database field.

In database theory, a conceptual model, which usually presented in E-R diagram, is a specification of domain concepts (include entities, attributes, etc.) and their relations. A data model is the implementation of corresponding conceptual model in machine world. We might as well consider ontology as conceptual model, and XML Schema as data model, therefore, an XML-orient semantic data model can be derived from an ontology conceptual model by designing a mapping algorithm originating from ontology to XML Schema.

2.2 OTX Algorithm

First of all, we make a formalization definition of ontology conceptual model so that the OTX algorithm could be suitable for ontology in different representation

languages:

We shall define a ontology conceptual model as a structure $S=\langle C(S), R(S)\rangle$. $C(S)=\{c_1, c_2, \cdots, c_n\}$ is a set of concepts/classes; $R(S)=\{r_1, r_2, \cdots, r_m\}$ is a set of relations; $C(S)\cap R(S)=\varnothing$.

The element $c_i \in C(S)$ corresponds to an entity/object of the real world, which is composed of a concept name and a set of attributes, attributes are used to describe the static characteristics. Generally, $c_i=\langle n_{c_i}, CA(c_i)\rangle$, the attribute set $CA(c_i)=\{CA_1, CA_2, \cdots, CA_p\}$. CA_i, $i=1, 2, \cdots, p$, is determined by several attribute name—value pairs, i. e. $CA_i=\langle nCA_i, v_i\rangle$.

The element $r_j \in R(S)$ describes relationship between entities/objects of the real world, which is composed of a relation identifier and a subset of a Cartesian product of concepts that involved in the relation, i. e. $r_j=\langle n_{rj}, c_{1j}\times c_{2j}, \cdots, c_{qj}\rangle$, where $c_{pj}\in C(S)$, $p=1, 2, \cdots, q$. There are four basic relations: part-of, kind-of, instance-of and attribute-of, which denote respectively the relation between portion and whole, the relation of concept hierarchy, the relation between concept and instance, the relation of attributes.

The correspondences between ontology conceptual model and XML Schema can be listed as follows:

- Ontology concept c_i <—> XML Schema element
- Ontology attribute CA_j <—> XML Schema subelement of the corresponding element
- Ontology relation:

Kind-of <—> XML Schema element inheritance mechanism (XML Schema attribute base and derivedBy).

Attribute-of <—> XML Schema element content model(XML Schema attribute type).

OTX algorithm is described as follows:

Algorithm 1

Input: ontology in a certain representation language;

Output: XML Schema.

```
begin{
Parse ontology, acquire conceptual model S=⟨C(S), R(S)⟩;
for each c_i ∈ C(S) do
{
  if (kind-of(c_i, c_x) ∉ R(S))
    output ⟨complexType name=c_i + "Type"/⟩ ;
  else
    output ⟨complexType name=c_i + "Type" base=c_x + "Type"
derivedBy= "extension"⟩;
  for each CA_j ∈ CA(c_i) do
  {
    if (attibute-of(c_y, c_i) ∈ R(S))
      output ⟨element name= "nCA_j" type=c_y + "Type"/⟩;
    else
      output ⟨element name= "nCA_j" type= "v_j"/⟩;
  }
  output ⟨/complexType⟩;
}
  for each c_i ∈ C(S) do
      output ⟨element name= "c_i" type=c_i + "Type"/⟩;
  }
```

In algorithm 1, each ontology concept generates an element type in XML Schema; For each attribute of ontology, XML Schema defines a subelement; If the attribute represents a relation to another concept, the attribute element has as content the respective concept element, otherwise its content model is simply atomic. The result XML Schema keeps the original concept system of ontology naturally, we can used it to create XML instance documents, which has a markup that is well founded on an ontology and therefore can express explicit domain knowledge. This approach improves information from the syntactic or representational level to the more abstract level of concepts and relationships, realizes the consistent semantic representation of information.

We should point out that except for the first step-ontology parsing, which depends on different ontology representation languages, the algorithm 1 could be suitable for any ontology.

Actually, the representation language of ontology is unimportant, as long as concepts in a hierarchy and relationships between concepts can be defined. Table 2 shows the result conceptual model $S=\langle C(S), R(S)\rangle$ parsed from ontology in Frame-Logic.

Table 2 Parsing ontology conceptual model

Frame-Logic formula	$\langle C(S), R(S)\rangle$
$c[a_1 =\gg v_1; \cdots; a_n =\gg v_n;]$	$\langle c, \{\langle a_1, v_1\rangle, \cdots, \langle a_n, v_n\rangle\}\rangle$
$c_1 :: c_2$	kind-of(c_1, c_2)
$c_3[a =\gg c_4]$	attribute-of(c_4, c_3)

3 Framework of Information Representation

3.1 Architecture

Figure 1 is general architecture of the information representation framework based on Web Service[9].

Web Service provider accepts Simple Object Access Protocol (SOAP) message carried by HTTP from service agent of data sources in distributed environment, gets the

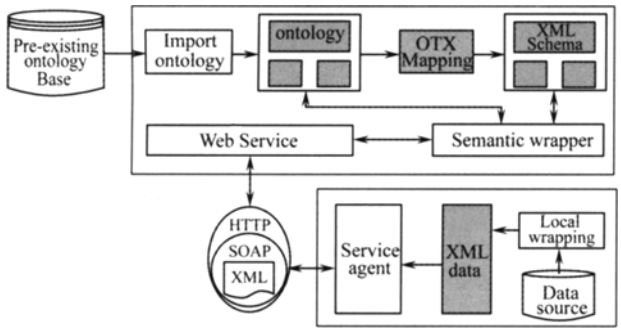

Fig. 1 The general architecture

service type and parameters, performs corresponding tasks, serializes the result into SOAP message and then sends it to Web Service Invoker.

3.2 Implementation

According to different demands of semantic representation, Web Service provider supplies the data sources with two aspects of services.

3.2.1 Schema only service

If a data source is capable of using XML tags to wrap information referring to the XML Schema generated by OTX algorithm, it may send a "Schema Only" request to service provider with the message for example:

```
POST someURL HTTP/1.1
Host:someURL
Content-Type:text/XML
  〈SOAP-ENV:Envelope
xmlns:SOAP-ENV = http://schemas.XMLsoap.org/soap/envelope/〉
  〈SOAP-ENV:Body〉
    〈request type="Schema Only"〉
      〈ontology〉(KA)²〈/ontology〉
    〈/request〉
  〈/SOAP-ENV:Body〉
〈/SOAP-ENV:Envelope〉
```

On receiving such a message, service provider will search for an XML Schema derived from ontology $(KA)^2$ in a local cache. If the Schema does not exist, "Import ontology" module will be launched to import the $(KA)^2$ ontology from a pre-existing ontology base, afterwards, the result XML Schema could be generated via OTX automatically and be sent to cache, at the same time, service provider return a SOAP message:

```
200 OK
Content-Type:text/XML
  〈SOAP-ENV:Envelpoe〉
    〈SOAP-ENV:Body〉
      〈Response type="Schema Only"〉
        〈result〉
          〈!--XML Schema goes here--〉
      〈complexType name="ObjectType"/〉
      〈complexType name="PersonType" base="ObjectType"
derivedBy="extension"〉
          〈elemen t name="name" type="string"/〉
          〈element name="email" type="string"/〉
          〈element name="address" type="string"/〉
          〈element name="publication" type="PublicationType"/〉
      〈/complexType〉
⋮
    〈complexType name="Researcher" base="AcademicStaffType" derivedBy="extension"〉
      〈element name="cooperatesWith" type="ResearchType"/〉
    〈/complexType〉
⋮
      〈element name="Person" type="PersonType"/〉
      〈element name="Employee" type="EmployeeType"/〉
      〈element name="Researcher" type="ResearcherType"/〉
⋮
        〈/result〉
      〈/Response〉
    〈/SOAP-ENV:Body〉
  〈/SOAP-ENV:Envelpoe〉
```

3.2.2 Semantic mapping service

Data sources have their own understand about domain knowledge and use different language or syntax to wrap information with XML, which forms distinct local semantic views. If a data source requests the "Semantic Mapping" Service, service provider will translate the source XML data serialized in the SOAP message from service agent into a semantic XML data compatible with ontology, which is the process of semantic representation for data source.

The following is a possible request message:

```
POST someURL HTTP/1.1
Host:someURL
Content-Type:text/XML
  〈SOAP-ENV:Envelope
xmlns:SOAP-ENV = http://schemas.XMLsoap.org/soap/envelope/〉
    〈SOAP-ENV:Body〉
     〈request type="Semantic Mapping"〉
      〈ontology〉(KA)²〈/ontology〉
       〈XML Data〉
        …〈!--source XML Data goes here--〉
       〈/XML Data〉
    〈/request〉
    〈/SOAP-ENV:Body〉
    〈/SOAP-ENV:Envelope〉
```

On receiving this message, service provider will take out the source XML data from it, and launch the "Semantic wrapper" module, which is illuminated in Fig. 2. There are following 3 steps of semantic wrapper.

1) Schema extraction

This step parses the source XML data, constructs a

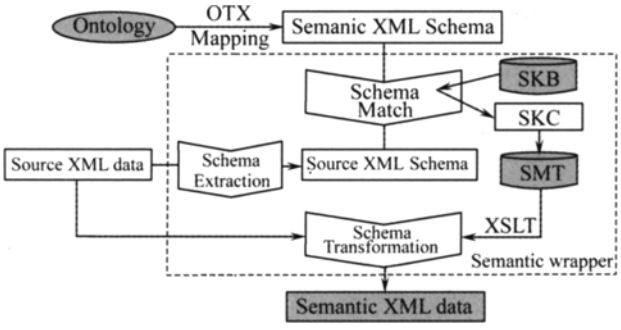

Fig. 2 Semantic wrapper

source node tree, and extracts the local semantic schema, i. e. makes out the element tags, attribute tags, and lists the tree-path for each node.

2) Schema match

Semantic Knowledge Constructor (SKC) loads corresponding XML Schema from the local cache according to ontology specified in SOAP message, parses it into a node tree (we call it a semantic node tree), and then performs schema match between the semantic node tree and the source node tree, schema match will produce the Schema Mapping Table (SMT), which facilitates translation between local and global semantic schema.

For the sake of accuracy of semantic mapping, schema match is divided into the following 3 subprocedures:

● Name match

Firstly, we should establish a Semantic Knowledge Base (SKB) using synonymous dictionary, with the help of which we may decompose and extend the node name. For example, the vocabulary EmailAddress can be decomposed into two tokens {Email, Address}; Abbreviation EMPNO can be extended to two tokens {Employee, No}. Secondly, we match the tokens with the node name of semantic node tree (i. e. the standard vocabulary provided by ontology). The match result is a series of triple-(ontology vocabulary, token, match degree), considering which we can determine the match degree[10] of the pair of nodes in source and semantic node tree.

● DataType match

On the basis of name match, we may compare the datatype of the pair of nodes by looking up the datatype table in SKB. If data type of the pair of nodes is identical, their match degree is considered as 1, or else takes value in the range [0,1]. For example, we specifies the match degree of an int type is 0. 8 with a float type, while that is 0. 6 with a string type, which denotes that float type is more close to int type semantically. Since XML Schema provides abundant datatypes, it can be assumed that datatype match should do a good job.

● SubTree match

Besides the two methods mentioned above, this paper provides another effective step-subtree match. It is to determine match degree of the pair of nodes by comparing their subtree respectively, in general, contents of an XML document are context-dependent, match degree between the descendent nodes could usually determine the match degree of the pair of nodes, moreover, leaf nodes in XML document are the entities to express contents, therefore the subtree match is actually leading the match process in the right direction.

It is worthy to be mentioned that in most cases not all of the descendent nodes can match exactly well, as a result, we define a threshold for subtree match aiming at different applications, for example, a threshold of 0. 6 means that if the match degree of their subtree (i. e. number of matching subtrees/ number of subtrees) is greater than 0. 6, the pair of nodes are considered to be matching. The threshold has the ability of learning from feedbacks. The schema match algorithm can be listed as follows:

Algorithm 2

```
Input: source node tree- * Source,
 semantic node tree- * Semantic;
Return value: 1 denotes that the pair of nodes is matching,0
denotes not.
int SchemaMatch( * Source, * Semantic)
{
if (Source! =Null &&. Semantic! =Null)
{if (NameMatch(Source, Semantic)>0. 6 &&.
  DatatypeMatch(Source, Semantic)>0. 6)
 {subtreeNum=max(number of subtrees of Source, number of
subtrees of
Semantic);
   match=0;
   for each Srcsubtree, in Source,
     each Smtsubtree, in Semantic do
match=match+SchemaMatch(Srcsubtree, , Smtsubtree, )
     if (match/subtreeNum)>0. 6 return 1;
     else return 0;
  }
else return 0;
}
elseif (Source==Null AND Semantic==Null) return 1;
else return 0;
}
```

The function SchemaMatch() of Algorithm 2 is to match between pair of nodes in source node tree and semantic node tree. If the result of name match and datatype match are both greater than 0. 6, subtrees of nodes will be

compared, actually, the subtree match is implemented *via* a recursive call of the function SchemaMatch() itself. When results of the three matching process reach the predefined threshold, the pair of nodes are considered to be matching, and SKC will store the relevant matching information in SMT, otherwise it can step into the matching of another pair of nodes.

When SKC fills in SMT, it will feed back information to SKB. With the increase of schema match, SKB will gradually gain new semantic knowledge.

3) Schema transformation

According to SMT (the result of schema match), this step will use XSLT[11] to transform source XML data into semantic XML data that coincides with ontology conceptual model. Firstly, establish some transformation templates based on SMT, once a node in source node tree is matching with a rule of the template, XSLT processor will then construct the content involved in this rule in the result tree, and finish the transformation step by step.

Finally, Web service provider will serialize the semantic XML data into a SOAP message and send it to the client:

```
200 OK
Content-Type:text/XML
  〈SOAP-ENV:Envelpoe〉
    〈SOAP-ENV:Body〉
      〈Response type="Semantic Mapping"〉
        〈result〉
        …〈! --Semantic XML Data goes here--〉
        〈/result〉
      〈/Response〉
    〈/SOAP-ENV:Body〉
  〈/SOAP-ENV:Envelpoe〉
```

4 Conclusion

Currently information is changing from single isolated data repositories to shared sources in a more opened environment. Therefore support in the exchange and integration of data, information, and knowledge is becoming the key issue in information process technology. The framework introduced in this paper provides a semantic wrapper to represent heterogeneous information in distributed environment, presents the strategy that matches and translates between ontology global semantic view and local data sources, which contributes to establish a uniform platform and lay the foundation for farther semantic information processing as well.

With a view to the ongoing data grid technology, it is possible to extend the Web Service architecture of this paper, so as to compatible with the advanced open standards, services and tools of grid. Our future work is to implement the Grid Service of semantic representation based on OGSA[12] making use of Globus Toolkit.

Acknowledgements Thanks to Dr. Gu Jin-guang for his assistance.

References

[1] Decker S, Erdmann M, Fensel D. ONTOBROKER: Ontology Based Access to Distributed and Semi-Structured Information. *Semantic Issues in Multimedia Systems*. Boston: Kluwer Academic Publiser, 1999. 351-369.

[2] Madhavan J, Bernstein P. A, Domingos P, *et al*. Representing and Reasoning about Mappings between Domain Models. *Proceedings of 18th National Conference on Artificial Zntelligence*. Edmonton, Canada: AAAI Press, 2002. 80-86.

[3] Benjamins R, Fensel D, Decker S, *et al*. (KA)^2: Building ontologies for the Internet. *International Journal of Human Computer Studies*, 1999, **51**(3): 687-712.

[4] Kifer M, Lausen G, Wu J. Logic Foundations of Object-Oriented and Frame-Based Languages. *Journal of the ACM*, 1995, **42**: 8-15.

[5] Farrugia J. Model-Theoretic Semantics for the Web. *Proceedings of the 12th International Conference on World Wide Web*. New York: ACM Press, 2003. 29-38.

[6] Fallside D. XML-Schema Part 0: Primer. *http://www.w3.org/TR/2000/WD-xmlschema-0/*, April 2000.

[7] Guarino N. Formal Ontology and Information System. *Proceedings of the 1 st International Conference on Formal Ontologies in Information Systems*. Trento, Italy: IOS Press, 1998. 3-15.

[8] Uschold M, Gruninger M. Ontologies: Principles, Methods and Applications. *Knowledge Engineering Review*, 1996, **11**(2): 93-155.

[9] W3C Workgroup. Web Service Activity. *http://www.w3.org/2002/ws/*, January 2002.

[10] Do Hong-hai, Rahm E. COMA——A System for Flexible Combination of Schema Matching Approaches. *Proceedings of the 28th VLDB Conference*. USA: Morgan Kaufmann Publisher, 2002. 610-621.

[11] Adler S, Deach S, Berglund A, *et al*. Extensible Stylesheet Language (XSL) Version 1.0. *http://www.w3.org/TR/xsl*, March 2000.

[12] Talia D. The Open Grid Services Architecture: Where the Grid Meets the Web. *IEEE Internet Computing*, 2002, **6**(6): 67-71.

□