第33卷 第21期 计 算 机 工 程 2007年11月
Vol.33 No.21 Computer Engineering November 2007
·软件技术与数据库· 文章编号：1000—3428(2007)21—0052—03 文献标识码：A 中图分类号：TP311

一种自动化的Web服务语义注释方法

王 超[1]，张宝山[2]，王春山[1]，牛晓霞[2]，刘 波[1]
(1. 河北农业大学信息科学与技术学院，保定 071001；2. 河北大学数学与计算机学院，保定 071002)

摘 要：分析了XML Schema和DAML文档，发掘二者在组成结构上的相似性，提出了一种联系WSDL文件和DAML本体描述文件的中间数据模型，通过将XML Schema格式的WSDL文件和DAML描述的本体文件映射到这种公共的数据模型上，使二者可以进行比较匹配，从而为自动化的语义注释提供支持。实验结果证明，该方法能为Web服务描述文件自动地添加语义信息。

关键词：Web服务；本体；语义；WSDL；DAML

Automatic Web Service Semantics Tagging Method

WANG Chao[1], ZHANG Bao-shan[2], WANG Chun-shan[1], NIU Xiao-xia[2], LIU Bo[1]
(1. College of Information Science & Technology, Aricultural University of Hebei, Baoding 071001;
2. College of Mathematics and Computer, Hebei University, Baoding 071002)

【Abstract】 This paper analyses the similarity between XML schema and DAML and develops a kind of common middle data model which can bridge both of them so that they can be matched by the way of mapping XML schema and DAML to the common data model. By doing so, this method can facilitate adding semantics to Web service automatically, which has been proved by experiments. The result shows that the method is worth being further studied and has a wide application foreground because it is necessary for adding semantics and efficiently to Web service precisely to store a mass of ontologies during the research work of Web service composition based on ontologies.

【Key words】 Web service; ontology; semantics; WSDL; DAML

Web服务合成[1]是一个分布式的开发架构，以现存的Web服务为基础去建构出一个新的Web服务，能为企业带来增值的商机。但由于参与服务合成的组件服务来源于不同的服务提供者，具有自治性和异构特性，导致服务合成远比传统的企业内的过程集成复杂。因此，必须在合成前解决Web服务间存在的异构问题[2]。采用本体论方法为Web服务添加语义信息[3]，从而解决异构的方法越来越被广泛认可，基于本体[3]的Web服务合成已经成为研究工作的热点问题。当前，对于Web服务语义信息的添加通常是采用人工手动添加的方式，然而随着本体库的不断完善，本体信息变得越来越庞大复杂，单靠人工的方式为Web服务添加语义信息难以满足实际需要，如何从海量的本体信息中为Web服务高效、准确、自动地添加语义信息已经成为基于本体的Web服务合成研究工作亟待解决的问题。

1 语义注释方案

WSDL[4]是XML Schema格式的Web服务描述语言，对于Web服务的数据交换格式，服务的调用方式等都作了详细的说明，用户根据WSDL文件便可以方便地调用远程服务。Web服务合成中的异构问题主要表现为服务接口不同上，即接口数据类型异构，导致各个Web服务之间难以直接进行合成。采用本体论，为WSDL文档的数据类型说明部分添加语义注释信息，能够较好地解决异构问题。然而，尽管结构上相似(都是基于XML格式)，本体的表达形式与WSDL使用XML schema的表达形式仍然有很大的不同。本体是用来获取真实世界的领域知识，因此，一般的本体描述语言主要针对现实世界的实体进行建模，通常是一种基于XML标准的网络资源标记语言，可对现实实体以及实体之间的关系、属性等进行显示的描述；而XML Schema的主要目的是定义文档语法和结构的约束，通过基本元素、简单数据类型和复杂数据类型的组合，为Web服务之间的数据交换提供基本的统一的数据结构。因此，很难对二者直接进行匹配比较。本文对目前比较流行的本体描述语言DAML[5]和XML Schema进行了分析，充分发掘和利用二者之间的共性，设计了一种基于文法的中间数据模型，作为联系本体与XML Schema的桥梁，使得二者可以进行比较、匹配，为Web服务自动化的语义注释提供支持。

文法是人工智能领域用来辅助计算机理解自然语句而定义的一种规则描述方法，通常是一个四元组$G=(V, T, P, S)$，其中：(1)V是非终结符（或称变元）的有限集；(2)T是终结符的有限集；(3)P是生成式的有限集，其中每个生成式都有形如$a\rightarrow b$，$a, b\in(V\cup T)^*$，且$a\neq b$；(4)$S\in V$，称为文法G的开始符号。

在普通文法中，生成式$a\rightarrow b$表示依据生成规则a可以转化为b，但没有描述出这种转化关系的具体意义，只是简单地用箭头标出，不能表示出元素之间的关系，因此，本文对普通文法的生成式进行了扩充，将生成式$a\rightarrow b$转化成aRb的形式，R表示a与b之间的具体的转化关系，即关系的取值。基于这种生成式的文法称为扩展文法G'。按照扩展文法G'的定义，可以将XML Schema和用DAML描述的本体映射成一个公共的中间数据模型，这里列举一段WSDL文档来说

基金项目：河北省科技攻关计划基金资助项目(20021124059)
作者简介：王 超(1978－)，男，硕士、助教，主研方向：信息系统与系统集成；张宝山，硕士研究生；王春山，硕士、助教；牛晓霞，硕士研究生；刘 波，硕士、助教
收稿日期：2006-11-10 **E-mail：**bdking78@126.com

明 XML schema 的映射规则。这部分 WSDL 文档如下所示：

```
- <complextype name="Pizza">
  - <all>
      <element name="size" type="xsd:string" />
      <element name="Topping" type="xsd:string" />
      <element name="quantity" type="xsd:int" />
      <element name="base" type="xsd:string" />
      <element name="customer" type="nso:Customer" />
    </all>
  </complextype>
+ <complextype name="Customer">
```

WSDL 文档到四元组的映射规则如下：

(1)顶级复杂数据类型(complexType)元素映射为开始元素，如 WSDL 文当中<complexType name=“Pizza”>映射为 S =“Pizza”。

(2)定义在顶级复杂数据类型下的简单数据类型元素映射为终结符集合中的元素，例如 WSDL 文当中“<element name=“quantity” type= “xsd:int” />”映射为“quantity”，其中，“quantity”∈T。

(3)定义在顶级复杂数据类型下的复杂数据类型元素映射为非终结有限集合中的元素，同时在生成式有限集中添加一个由开始元素到此元素的生成式元素，在内存中，将这个生成式定义为三元组。例如 WSDL 文当中的 <element name=“customer” type=“ns0:Customer” /> 映射为三元组(“Pizza”, “customer”, “Customer”)表示“Pizza”元素到“customer”元素之间存在着关系“Customer”。

根据映射规则，上述中的 WSDL 文档最终映射的中间数据模型为四元组：

({“customer”}, {“Pizza”}, {“quantity”, “size”, “topping”, “base”}, {({“Pizza”, “customer”, “Customer”})})

将用 DAML 描述的本体映射为文法四元组的映射规则同上很相似，这里同样列举 DAML 描述的本体进行说明，这段本体文件如下：

```
-<daml:Class rdf:ID="Food">
    <rdfs:comment>Super class for eatable things</rdfs:comment>
    <rdfs:lable>Food</rdfs:lable>
    <rdfs:subClassOf rdf:resourse="#Entitites"/>
  </daml:Class>
-<daml:Property rdf:ID="Topping">
    <rdfs:lable>food garmish</rdfs:lable>
    <rdfs:domain rdf:resourse="#Food"/>
    <rdfs:range
rdf:resourse="http://www.w3.org/2000/10/XMLShema#string"/>
  </daml:Property>
-<daml:Property rdf:ID="container">
    <rdfs:lable>Food container</rdfs:lable>
    <rdfs:domain rdf:resourse="#Food"/>
    <rdfs:range rdf:resourse="#Vessel"/>
 </daml:Property>
```

本体到四元组的映射规则如下：

(1)类(Class)映射为开始元素，如 DAML 本体实例中的<daml:Class rdf: ID= “Food”>映射为开始元素 S=“Food”。

(2)值域(range)为基本数据类型的类的属性映射为终结符集合中的元素。如 DAML 本体实例中的<daml:Property rdf:ID= “Topping”>映射为“Topping”，其中，“Topping”∈T。

(3)值域(range)为另一个本体概念或本体类的属性映射为非终结有限集合中的元素，同时在生成式有限集中添加一个由开始元素到此元素的生成式元素，在内存中，将这个生成式定义为三元组。例如 DAML 本体实例中的<daml:Property rdf: ID=“container”>映射为三元组(“Food”, “Vessel”, “container”)表示“Food”元素到“Vessel”元素之间存在着关系“container”。

根据生成规则，DAML 本体最终映射的中间数据模型为四元组：

({“Vessel”}, {“Food”}, {“Topping”}, {({“Food”, “Vessel”, “container”})})

2 匹配算法

依据四元组模型，从语法和结构两方面考虑，设计的匹配算法由两部分组成，如下所示：

$$MV=\frac{w1*elemSim(G1,G2)+w2*struSim(G1,G2)}{w1+w2}$$

其中，$0\le w1\le 1, 0\le w2\le 1$。这里 *elemSlim* 函数计算四元组中开始元素的语法匹配值，即 WSDL 文档中顶级复杂元素(ComplexTyp 元素)和 DAML 本体某一概念或类的语法匹配值；*struSlim* 函数表示四元组开始元素的结构匹配值；$w1$ 和 $w2$ 分别表示二者在最终的匹配度分值(MV)中所占的权值，权值由用户设定，一般设定 $w1=0.7$, $w2=0.3$，即认为语法匹配所占的比重要大一些。

2.1 语法匹配函数

语法匹配函数 *elemSim* 计算两个四元组中开始元素的语法匹配值，采用的是一种基于 WordNet 概念距离的匹配算法。WordNet 是普林斯顿大学的 Geo rgeMiller 等人开发的电子词典系。WordNet 的名词部分是按照相接近含义的概念组织的，提供了概念之间的 IS-A 关系，称为层次关系树(图 1)，它是用来衡量概念之间距离的重要特征。因此，可以通过 WordNet 的 IS-A 层次关系树计算两个四元组的开始元素的语义匹配值。定义如下：

$$elemSim(G1,G2)=\begin{cases}1: \text{if } s1=s2\\ 1/\text{edges: if edges} \le T\\ 0: \text{if edges} \ge T\end{cases}$$

其中，edges 是 $s1$ 和 $s2$ 在 IS-A 关系中的最短距离；$s1\in G1$, $s2\in G2$。

公式中的 T 是为正规化目的而预设的阈值，范围为[0,1]。例如，当 $T=5$ 时，由于“Pizza”和“Food”在 IS-A 层次关系树中的最短距离是 2，因此得出 $elemSlim(G1, G2)=1/2=0.5$。

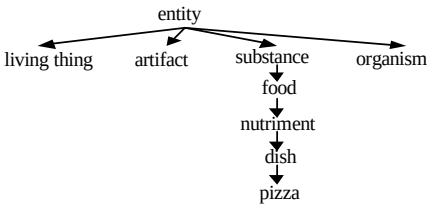

图 1 WordNet 层次关系树

2.2 结构匹配函数

结构匹配函数用来计算 2 个四元组开始元素的结构上的匹配值。所谓结构匹配是指 WSDL 一个复杂数据类型的各个子元素和一个本体概念各个属性之间的匹配值，定义为

$$struSim(G1,G2)=\frac{\sum_{j=1}^{n} MV(subElem_j)}{n}$$

其中，n 为 $G1$ 中所有终结符和非终结符元素的个数；*subElem* 表示四元组中各个终结符和非终结符元素，结构匹配由四元组的终结符元素和非终结符元素的匹配度值计算得出，因此，它是一个递归的过程，最终归结到终结符元素的语法匹配计算上。表 1 列举了一些结构匹配的实验结果。

表 1 结构匹配函数值

WSDL 子数据类型	DAML 本体属性	*MV*
Topping	Topping	1
Base	Container	0.667
Customer	无	0
struSim(*G*1,*G*2)= (1+0.667+0)/3=0.557		

3 语义注释

语义注释过程就是通过映射 WSDL 文件和 DAML 本体文件到文法四元组，根据匹配算法，找出与 WSDL 四元组最为匹配的 DAML 本体四元组，然后将其对应的本体概念或本体类添加到 WSD 文件中去的过程，下面为部分添加语义注释后的 WSDL 文件：

```
- <complextype name="Pizza" Ont-Class="Food">
    - <all>
          <element name="size" type="xsd:string" />
          <element name="Topping" Ont-Class="Food:topping" ype=
"xsd:string" />
          <element name="quantity" type="xsd:int" />
          <element name="base" Ont-Class="Food:Container" type=
"xsd:string" />
          <element name="customer" Ont-Class="human being" type=
"nso:Customer" />
       </all>
    </complextype>
+ <complextype name="Customer" Ont-Class="human being">
```

4 结束语

针对目前基于本体的 Web 服务合成的研究工作中存在的难以为 Web 服务描述文件添加语义注释信息的问题，提出了一种用于联系 WSDL 文件和 DAML 本体描述文件的文法四元组数据模型，使得二者可以进行匹配计算，依据匹配函数计算得出的匹配值，系统可以自动为 WSDL 文件添加语义注释信息。通过这种方法为 WSDL 文件添加语义注释，要比人工手动添加语义信息的方式，无论是在执行效率上，还是准确性上都有较大的提高，尤其是在本体库十分庞大的时候更是如此。这种基于文法四元组的语义注释方案在笔者开发的 Web 服务合成系统中得到了很好的应用，实践证明，能够较好地解决 Web 服务合成过程中存在的异构问题。

参考文献

1 Wang Hongbing, Huangc Joshua Zhexue, Qub Yuzhong, et al. Web Services: Problems and Future Directions[J]. Journal of Web Semantics, 2004, 1(3): 309-320.

2 申德荣, 于 戈, 张 蓉. Web 服务合成中的异构问题[J]. 东北大学学报, 2004, 25(3): 220-222.

3 宋 炜, 张 铭. 语义网简明教程[M]. 北京: 高等教育出版社, 2004.

4 Banerjee A, Corera A. C# WEB Services[M]. 北京: 清华大学出版社, 2002.

5 Sycara K, Paolucci M, Ankolekar A, et al. Automated Discovery, Interaction and Composition of Semantic Web Services[J]. Journal of Web Semantics, 2003, 1(1): 27-46.

~~~~~~

（上接第 51 页）

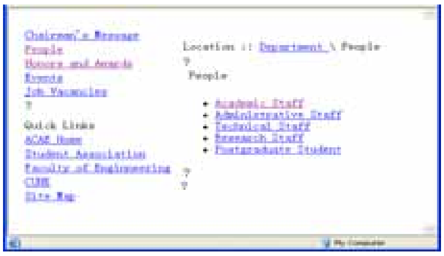

图 3 处理后的网页屏幕截图

3 算法测试

为了测试算法性能，本文开发了一个软件原型系统，开发平台是：编程语言为 Java，编译环境为 J2SE 5.0，网页 DOM 树解析软件包为 Xerces2 Java 解析器插件，下载的网页以及超链接存储在 MySQL 4.1 数据库中。本文在 4 个网站中对该算法进行了测试，并将算法执行的指标值列于表 1。表 1 中的编号 1～编号 4 分别是 4 个网址：http://www. cs.yale.edu(CS Dept.1)；http://www.acae.cuhk.edu.hk/en(ACAE Dept.2)；http://www.media.mit.eud(Media Lab3)；http://www.se.cuhk.edu.hk (SE Dept 4)，且算法对 4 个网页内容精化执行的平均精度均为 100%。

表 1 网页内容精化算法执行的效果

编号	网页总数	经处理的网页总数	剔除的字节总数/B	平均识别率/(%)
1	138	91	74 498(=745 473-670 975)	51.28
2	106	103	17 279(=584 470-567 191)	66.34
3	106	104	16 317(=582 162-565 845)	26.28
4	368	198	252 505(=1 932 806-1 680 301)	49.83

本文量化算法效果的指标如下：

(1)从网页中剔除的字节数。

(2)平均识别率。对于每个网页，假定其总共拥有 m 个无关信息分区，经过处理后剔除了 n 个无关信息分区，则该网页算法执行的识别率为 n/m。例如：对比图 1、图 3，可以计算出该网页的算法识别率为 66.67%(=2/3)。

(3)平均精度。对于每个网页，假定算法处理后提出了 y 个无关信息分区，其中，有 x 个分区确实为该网页的无关信息分区，则该网页算法执行的精度为 x/y。例如，图 3 所示网页的精度为 100%(=3/3)。

4 结束语

通过分析各大网站下网页内含“无关信息”的特点和模式，本文的网页内容精化算法以网站的拓扑为基础，过滤掉网页中出现在相同位置上的相同内容的信息，达到精化网页内容的效果。对算法的初步测试表明，该算法具有较高的正确率以及识别率。

参考文献

1 Cutler M, Shih Y, Meng W. Using the Structure of HTML Documents to Improve Retreival[C]//Proc. of USENIX Symposium on Internet Technologies and Systems. 1997: 241-251.

2 Brin S, Page L. The Anatomy of a Large Scale Hyper-textual Web Search Engine[J]. Computer Networks and ISDN Systems. 1998, 30(1/7): 107-117.

3 Davulcu H, Vadrevu S, Nagarajan S. OntoMiner: Bootstrapping and Populating Ontologies from Domain Specific Web Sites[J]. Intelligent Systems, 2003, 18(5): 24-33.

4 Buttler D, Liu L, Pu C. A Fully Automated Object Extraction System for the World Wide Web[C]//Proceedings of the 2001 International Conference on Distributed Computing Systems. 2001: 361-370.

5 傅 骞, 温晓辉. 开放式 Web 信息抽取系统研究与实现[J]. 北京师范大学学报, 2005, 41(6): 594-598.
~~~~~~