一种类自然语言驱动的语义服务搜索方法

张桂刚
(武汉大学软件工程国家重点实验室 武汉 430072)

摘 要 传统的服务搜索方式需要通过编写固定格式的程序来搜索所需要的服务,事实上大部分非程序员很难掌握这种编程方式,这种传统的方式很难满足非程序员搜索语义服务的需求。针对这种缺陷,提出了一种类自然语言驱动的语义服务搜索框架。定义了一系列语义服务请求类自然语言的描述规则,提出了一种新的语义服务资源描述框架RDF4S(Resource Description Framework For Service),阐述了类自然语言驱动的语义服务的浅搜索和深搜索。

关键词 语义服务,本体,RDF4S,语义搜索
中图法分类号 TP301 **文献标识码** A

A Kind of Semantic Service Search Method Driven by Natural-like Language

ZHANG Gui-gang
(State Key Laboratory of Software Engineering, Wuhan University, Wuhan 430072, China)

Abstract The traditional method of searching service needs programmer program according to the fixed format. Actually, most of people especially the non-programmer have large difficulties to master it. The traditional search method can not satisfy non-programmer's requirements. To overcome the above limitations, the paper gave a kind of semantic service search framework driven by natural-like language. Defined a series of nature-like description rules to semantic service request and gave a kind of new semantic service description framework, which is RDF4S(Resource Description Framework For Service). This paper analysed the surface search and deep search about semantic service driven by natural-like language and analysed the core technology in deep search.

Keywords Semantic service, Ontology, RDF4S, Semantic search

1 引言

语义服务的应用越来越广,已经成为未来电子商务发展的一个重要基石。如何能够比较容易且准确地从世界各地找到自己所需要的语义服务已成为一个迫切需要解决的技术问题。目前研究搜索引擎和语义服务的文献很多。以Yahoo为代表的第一代搜索引擎主要侧重于将资源进行分类,通过目录分级来实现搜索。以Google为代表的第二代搜索引擎主要通过关键字匹配来实现数据的搜索。后来元搜索引擎又发展起来,最近有不少学者又提出了跨媒体搜索引擎等等。当然所有的这些搜索机制在技术上取得了很大的突破在商业上取得了成功。但是以上这些搜索引擎都用于搜索网页层面的资源,且主要用于静态页面的搜索,不能用于搜索网络服务。在搜索网络服务方面,目前尚无专业的搜索引擎来实现,只能通过程序员编写程序到UDDI中去提取已经注册了的网络服务。这种需要通过编写程序去搜索所需要的服务已经越来越难以满足普通人尤其是非程序员对网络服务的要求。如非程序员若想搜索"预订机票"这样一个网络服务,因为他不会编写这种程序,根本没有办法能够满足要求。本文试图设计一种可以面向非程序员的语义服务搜索方法,普通用户只要在搜索框输入满足要求的语义服务陈述句如"预订2008年8月10日北京饭店的标准间3间",语义搜索引擎就会去查找"预订房间"这种语义服务。语义服务的搜索涉及到的不仅仅是网页层面的资源搜索,而且涉及到隐藏在网页后面的数据的提取即深度搜索问题。当用户输入"预订2008年8月10日北京饭店的标准间3间"搜索请求时,语义搜索引擎首先要将"预订酒店"语义服务搜索到,然后要将隐藏在语义服务里的数据如"2008年8月10日"、"北京饭店"以及"标准间"、"3间"等这些隐藏在"预订酒店"这个语义服务中所隐藏的数据搜索出来。

近年来,国内外研究人员围绕基于语义的服务搜索展开了广泛的研究。IBM的Akkiraju[1]等人提出了采用领域无关和领域相关的本体来匹配服务。Georgia大学的Meteor-S[2]项目采用对UDDI进行语义扩展的方式来对服务进行语义的匹配。Paolucci[3]等人提出通过匹配OWL-S的Service Profile的基于语义的服务搜索算法。马应龙[4]等人提出了采用分布式描述逻辑扩展描述相互关联的异构分布式本体,并且用优先分布式知识库来描述分布式本体的进化和更新,从而适应于语义Web服务环境。这些服务匹配的方法都是在搜索的时候对服务库里面注册服务通过穷举的方式逐个进行匹配,并且对于语义匹配是在服务发现的时候才通过推理机访问本体库进行语义的推理。文献[5]基于WSDL-S规范基础

到稿日期:2008-08-22 返修日期:2009-02-04 本文受国家高技术研究发展计划(863)(2006AA01Z168)资助。
张桂刚(1978—),男,博士研究生,讲师,研究方向为语义计算、语义网络服务、语义软件工程,E-mail:zhangguigang@163.com。

提出了一种新颖的发现不同服务间语义关系的方法。通过服务的本体性描述规范来识别不同服务间前置条件和后置条件间的语义关系，并在此基础之上进行服务资源的搜索和发现。文献[6]提出能够在某种具体的服务描述规范基础之上进行基于语义的匹配，并基于 DAML-S 规范提出了 4 种服务资源匹配类型。提出了基于本体论和词汇语义相似度的服务发现方法。通过构建 Web 服务本体，给出一个明晰的 Web 服务发现对象，指出可对 Web 服务进行的几种相似度计算，为 Web 服务相似度计算、Web 服务发现提供了一种有效可行的方法。

假设用户将去参加 2008 年北京奥运会的开幕式，用户有如下语义服务的请求，即“预订 2008 年 8 月 7 日从武汉到北京的最便宜的飞机票 5 张”。这是一个自然语言的服务请求，翻译成英文就是：“Booking 5 air tickets from Wuhan to Beijing on August 7，2008”。如何找到上述用户的自然语言需求所描述的语义服务是要实现的目的。本文首先提出一个类自然语言驱动的语义服务搜索框架，然后详细分析了语义服务请求规则；接着介绍了语义服务搜索实现方法，包括语义服务浅搜索实现方法和语义服务深搜索实现方法；最后通过实验进行测试和验证。

2 类自然语言驱动的语义服务搜索框架

实现类自然语言驱动的语义服务搜索的关键技术需要解决如下几个关键核心问题。首先要将用户的自然语言请求（包括文字和语音）进行特殊的自然语言处理以便能够满足特殊语义服务的搜索要求。其次，要解决语义搜索引擎的设计问题，不仅要考虑语义服务的浅搜索要求，同时更要设计出满足要求的深搜索要求。另外由于要实现数据的提取还需要分析本体[14]技术对语义搜索的支持，本文采用本体查询语言查询语义服务中隐含的数据。类自然语言驱动的语义服务搜索框架如图 1 所示，其基本实现思想如下：

（1）首先用户提出语义网络服务的请求，这种需求表达方式包括文字或者语音方式，当然语音方式要通过语音识别技术最后转变成陈述句的自然语言表达方式。

（2）接着将用户提出的自然语言的需求，或者通过语音处理技术之后得到的自然语言的需求通过一个语义接口处理器进行类自然语言处理，得到本文所讲的“名词”、“动词”和“形容词”等。

（3）处理后得到的类自然语言以及它的“名词”、“动词”和“形容词”等，并通过“语义搜索引擎”去搜索所需要的服务。

（4）由于要实现语义服务搜索的“语义”功能，我们用到了本体技术。在本文中，本体具有两个主要功能。①语义功能，相当于本体功能。本体中存储了语义库，使得各种数据之间具有语义功能。比如“预订”和“订购”以及“购买”这些动词在处理时候，本体会将它们做同一语义处理；“火车票”和“列车票”也会将它们处理为同一语义。这样使得“订购火车票”与“购买列车票”这样的不同输入能够通过本体得到同样的结果。②存储功能。本体不仅具有上述的语义功能，而且能够将不同的数据源（数据库、文本、XML 文档、网络服务数据等）的数据存储起来。本文中的语义服务中的数据可以存储在本体中，从而可以实现深搜索（Deep Search），将语义服务中的数据取出来。如“搜索标间价格在 200 到 300 元之间的酒店”，本搜索不仅要将“酒店查询”该服务搜索出来，而且要将服务的“价格”的数据提取出来，即所要求的深搜索。

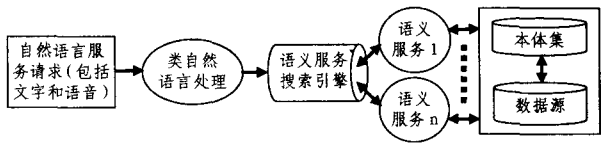

图 1 类自然语言驱动的语义服务搜索框架

3 语义服务请求规则及语义服务资源描述框架

3.1 类自然语言语义服务请求约定

需要说明的是：在这里所阐述的动词、名词和形容词和一般所理解的动词、名词和形容词有所不同。本文中所指的动词、名词和名词分别与本文的规定 1、规定 2 和规定 3 所指相对应。我们做如下规定。

规定 1 陈述句里的动词与本文中的动词意思表示一致。

规定 2 陈述句里的宾语里的名词是本文所指的名词。

规定 3 陈述句里的非宾语的名词或者其他任何修饰词，本文都将其认定为形容词。

语义服务表现形式是一个动宾结构如“订购机票”、“预订门票”等这种简单的动宾结构语义服务，也有如“订购一张从武汉到北京的飞机票”、“预订一张 2008 年 8 月 8 日的奥运会的开幕式门票”、“预订一张从武汉到上海的飞机票和一间靠近上海明珠广场的 5 星级酒店”等等这种复杂的动宾结构的语义服务。为了能够让上述自然语言所描述的语义服务能够被机器理解并通过语义服务搜索引擎查找到所需要的语义服务，我们做如下的变换：将一个语义服务请求的句子或者短语分解成 3 类词语。动词、名词和形容词。则上面的语义服务可以转换成“订购（动词）＋机票（名词）”、“预订（动词）＋门票（名词）”、“预订（动词）＋一张（形容词）＋2008 年 8 月 8 日（形容词）＋奥运会（形容词）＋开幕式（形容词）＋门票（名词）”、“预订（动词）＋一张（名词）＋武汉（形容词）＋上海（形容词）＋飞机票（名词）＋一间（形容词）＋上海明珠广场（形容词）＋5 星（形容词）＋酒店（名词）”。

变换规则如下：

规则 1 单个简单语义服务请求。

定义 对于只有一个动词和一个名词的短语则是一个最简单的单个语义服务请求。如“订购（动词）＋机票（名词）”和“预订（动词）＋门票（名词）”等都属于单个简单的语义服务请求。

规则 2 带修饰限制的单个语义服务请求。

定义 对于只有一个动词和一个名词的短语并且至少有一个形容词的陈述句短语则是一个带修饰限制的单个语义服务请求。如“预订（动词）＋一张（形容词）＋2008 年 8 月 8 日（形容词）＋奥运会（形容词）＋开幕式（形容词）＋门票（名词）”。在这样一个陈述句语义服务请求中只有一个动词和一个名词，但是包含了 4 个形容词。满足这种条件的都符合属于带修饰限制的单个语义服务请求。

规则 3 多个简单语义服务请求。

定义 对于只有一个动词和多个名词的陈述句或者有多个动词和多个名词构成的陈述句所形成的语义服务请求则是多个简单语义服务请求。如“订购机票和门票”“订购（动词）

+机票(名词)+门票(名词)"(一个动词和多个名词)或者"订购(动词)+机票(名词)并+预定(动词)+酒店(名词)"(多个动词和多个名词)等都属于多个简单语义服务请求。

规则 4 带修饰限制的多个语义服务请求。

定义 对于只有一个动词和多个名词并且至少有一个形容词或者对于多个动词和多个名词并且至少有一个形容词的陈述句则是属于带修饰限制的多个语义服务请求。如"预订(动词)+一张(名词)+武汉(形容词)+上海(形容词)+飞机票(名词)+一间(形容词)+上海明珠广场(形容词)+5 星(形容词)+酒店(名词)"。满足这种条件的都符合属于带修饰限制的多个语义服务请求。

规则 5 复杂的多陈述句语义服务请求。

定义 对于满足规则 1 到规则 4 但是由多个陈述句组成的语义服务请求块,我们将其称为复杂的多陈述句语义服务请求。如"预订一张从武汉到广州的 7 日的火车票。预订 8 号的广州饭店的标间房间一间。预订 9 号的广州工人体育馆羽毛球门票一张。"前面的这个用户的语义服务请求则是由多个陈述句构成的。满足这种条件的 H 都符合属于复杂的多陈述句语义服务请求。

3.2 语义服务资源描述框架 RDF4S

由于以陈述句自然语言驱动的语义服务请求和一般的语义服务请求有本质的区别,以前的以 OWL-S 方式描述的资源已经不能满足需要。提出一种特殊的语义服务资源描述框架 RDF4S(Resource Describe Framework for semantics),其基本模型如图 2 所示。

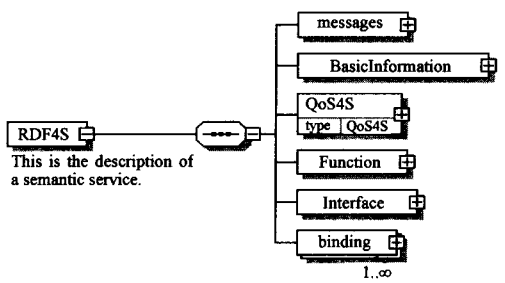

图 2 语义服务 RDF4S 框架

语义服务描述模型 RDF4S,是通过对 WSDL 进行语义扩展而得到的,保留了 WSDL 文档的基本描述信息和实现信息部分,另外添加了功能、接口、执行和 QoS 4 层语义信息,这 4 层语义信息如下。

(1)服务的功能语义

服务的基本属性(Profile)包括 4 个部分:服务的简单描述信息(BasicInformation)、联系人信息(Contact)、分类信息(Category)和服务提供的能力(Capability)。

(2)服务的接口语义

语义服务的接口语义包括经本体标注的操作名、操作的输入消息 input、输出消息 output 和错误消息 fault,前置条件 precondition 和后置条件 postcondition。

(3)服务的执行语义

语义服务的执行语义主要用于描述操作的执行,采用操作的状态 *State* 来描述。操作的执行一般会经历 4 种外部可观察的状态:准备状态 *Ready*,开始状态 *Start*,激活状态 *Active* 和结束状态 *End*。状态之间的依赖关系用→表示,如 *S1*→*S2* 表示状态 *S1* 一定在状态 *S2* 之前发生。操作的状态变化一般按如下次序进行:*Ready*→*Start*→*Active*→*End*。如果未对操作进行请求,操作将一直置于 *Ready* 状态,开始状态 *Start* 意味着操作已被初始化,激活状态 *Active* 表示操作已被初始化并正处于处理请求的过程中,请求被处理之后,操作返回相应的结果并到达结束状态 *End*。调用某一操作的时候可能会涉及对其他的操作的调用,为此定义了两类操作之间的关系:前置操作 *PreOperations* 和后置操作 *PostOperations*。

(4)服务的 QoS 语义

语义服务的 QoS 信息决定了服务的可用性和实用性等方面的内容,可为用户提供更符合需求的语义服务,提升用户的满意度。

例 1 旅行预订资源 RDF4S 描述文本

```
〈? xml version="1.0" encoding="UTF-8"?〉
〈WSDL4S xmlns="www.sklse.org/rdf4s" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="WSDL4S_Schema.xsd"〉
  〈BasicInformation xmlns="" BusinessName="BookTravel" Description="Text"〉
    〈Category〉
      〈CategoryName ontoreference="string" name="string" /〉
      〈TaxonomyURI〉String〈/TaxonomyURI〉
      〈TaxonomyValue〉String〈/TaxonomyValue〉
      〈TaxonomyCode〉0〈/TaxonomyCode〉
    〈/Category〉
    〈Capability〉
      〈ontoreference〉String〈/ontoreference〉
      〈value〉String〈/value〉
    〈/Capability〉
    〈ProviderInformation contact="String" location="String" domain="String" name="String" /〉
  〈/BasicInformation〉
  〈BusinessProfile xmlns="" name="Text"〉
    〈Service ontoreference="Onto4T#BookTravel" name="BookTravel"〉
      〈Function ontoreference="Onto4T#BookTravel" name="BookTravel"〉
        〈Preoperation name="String" /〉
        〈input ontoreference="Onto4T#BookTravelRequestMessage" name="BookTravelRequestMessage" /〉
        〈pattern ontoreference="String" name="String" /〉
        〈output ontoreference="Onto4T#BookTravelResponseMessage" name="BookTravelResponseMessage" /〉
        〈error ontoreference="String" name="String" /〉
        〈precondition ontoreference="String" name="String" /〉
        〈effect ontoreference="String" name="String" /〉
        〈Postoperation name="String" /〉
      〈/Function〉
    〈/Service〉
  〈/BusinessProfile〉
  〈TechnologyProfile xmlns="" name="Text"〉
    〈QoS〉
      〈Reliabiligy value="0" /〉
      〈Cost value="10RMB" /〉
      〈Reputaion value="0.9" /〉
```

```
〈ResponseTime value="15s" /〉
〈Accessbility value="0" /〉
〈Availibility value="0" /〉
〈Security value="0" /〉
〈/QoS〉
〈TechnologyContext〉
〈ServiceContext name="Text"〉
〈sproperty ontoreference="String" value="String" name="String" /〉
〈/ServiceContext〉
〈OperationContext name="BookTravel" TransProtocol="Text" PortURL="Text"〉
〈oproperty ontoreference="TechOnto#http" value="HTTP" name="transportProtocol" /〉
〈/OperationContext〉
〈MessageContext MessageStyle="Text" MessageProtocol="Text" MessageCoded="Text" name="BookTravelRequestMessage"〉
〈mproperty ontoreference="TechOnto#soap2.0" value="SOAP2.0" name="messageProtocol" /〉
〈mproperty ontoreference="TechOnto#RSA" value="RSA" name="Encryption" /〉
〈/MessageContext〉
〈MessageContext MessageStyle="Text" MessageProtocol="Text" MessageCoded="Text" name="BookTravelResponseMessage"〉
〈mproperty ontoreference="TechOnto#soap2.0" value="SOAP2.0" name="messageProtocol" /〉
〈mproperty ontoreference="TechOnto#RSA" value="RSA" name="Encryption" /〉
〈/MessageContext〉
〈/TechnologyContext〉
〈/TechnologyProfile〉
〈Mapping xmlns=""〉
〈businessprofilemapping〉
〈servicemapping service="String" interface="String"〉
〈functionmapping operation="String" function="String"〉
〈inputmapping functioninput="String" operationinput="String" /〉
〈outputmapping operationoutput="String" functionoutput="String" /〉
〈errormapping error="String" fault="String" /〉
〈/functionmapping〉
〈/servicemapping〉
〈/businessprofilemapping〉
〈technologyprofilemapping〉
〈technologycontextmapping〉
〈operationcontextmapping〉
〈PortURLmapping location="String" PortURL="String" /〉
〈TransportProtocolmapping transport="String" TransportProtocol="String" /〉
〈/operationcontextmapping〉
〈messagecontextmapping〉
〈MessageProtocolmapping SOAP="SOAP" MessageProtocol="String" /〉
〈MessageStylemapping MessageStyle="String" style="String" /〉
〈MessageCodedmapping use="String" MessageCoded="String" /〉
〈/messagecontextmapping〉
〈/technologycontextmapping〉
〈/technologyprofilemapping〉
〈/Mapping〉
〈/WSDL4S〉
```

4 语义服务搜索实现

语义服务搜索与一般的搜索不同，主要体现在如下几个方面：①搜索的对象不同。传统的搜索引擎主要用于搜索网页，而语义服务搜索的对象是以 RDF4S 格式描述的语义服务。②搜索程度不同。传统的搜索引擎一般只需要搜索到页面所能展示的数据，而语义搜索不仅要将 RDF4S 描述文档搜索到，也要将隐含在 RDF4S 里的“深”数据搜索出来。③搜索请求不同。传统的搜索引擎输入的一般是一个名词作为关键词，而语义服务搜索引擎输入的是以类自然语言表达的一个至少包含“动＋宾”语义服务表达语言。如“预订酒店”。④语义体现不同。一般搜索引擎没有体现语义，而语义服务搜索引擎针对的就是语义服务。

下面从语义服务浅搜索和语义服务深搜索两个方面来详细分析语义服务的搜索实现方法。

(1)语义服务浅搜索方法

服务浅搜索的搜索原理和目前各种商业搜索引擎的搜索机制和原理基本一样，唯一的区别是：一般的搜索引擎搜索的是网页。而这里所指的服务浅搜索所要搜索的内容是以 RDF4S 格式所描述的资源文件。至于前面浅搜索中的索引库的建立，爬虫的工作原理和其他商业搜索引擎一样。在这里就不再讲述。本文仅仅从陈述句自然语言语义服务请求出发来看是如何搜索 RDF4S 语义服务资源的，图 3 是语义服务浅搜索的方法示意图。

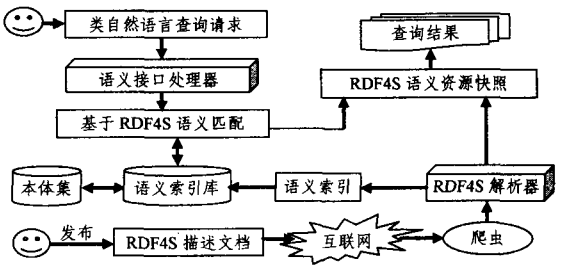

图 3 语义服务浅搜索方法

语义服务浅搜索方法的基本实现原理如下：①发布部分。各种商家将各自制作的语义服务的 RDF4S 描述文档发布在互联网上。②语义爬虫部分。语义爬虫主要就是用来专门爬取各种以 RDF4S 格式发布在互联网上的语义服务描述文件。③建立语义索引部分。由于通过语义爬虫爬出来的资源 RDF4S 具有语义信息，故通过 RDF4S 解析器将 RDF4S 描述文件中的语义信息解析出来，生成语义索引并将它们存储到语义索引库中。④语义匹配部分。语义索引库中的每条语义索引存储了各种特定领域的语义数据，但是语义的匹配需要一个语义库来解析，传统的语义解析一般都使用本体库来实现，本文中使用本体来进行语义解析，从而实现基于 RDF4S

的语义匹配功能。⑤语义服务查询请求和解析部分。各个语义服务资源的用户在搜索输入文本框中输入“动宾”结构的陈述句，通过 SemanticTalk 解析得到符合本文规定的“动词”、“名词”和“形容词”这些本体的语义小粒度对象，将这些对象用于和以 RDF4S 格式描述的语义服务进行匹配。⑥语义服务结果展示部分。当用户搜索到所需要的语义服务时，系统会通过语义服务资源快照将结果反映出来。

假设有这样一条陈述句自然语言驱动的服务请求语句“query the airline price from Wuhan to Beijing and query the start time and arrive time.”

自然语言 Parser 的结果如下：

Query(verb)＋ airline(noun)＋ price(adjective) ＋ Wuhan(adjective) ＋Beijing(adjective) ＋ start time(adjective) ＋ arrive time(adjective)

本语句在查询时候要经过如下两个过程：

①对请求语句"query the airline's price from Wuhan to Beijing and query the airline's start time and arrive time."进行简化找出其相应的动词(query)和名词(airline)。(注意这里的名词不是"price"＋"time"，请参看前面的规定)。

②很明显通过自然语言 Parser 的上述句子符合前面的规则 2：带修饰限制的单个语义服务请求。在第 2 步就是要将其中的形容词也就是限制修饰词查找出来。

经过上述两个步骤，在用户搜索界面接口中输入的陈述句自然语言的服务请求所需要的服务就可以通过图 4 中的语义服务浅搜索方法搜索出来，从而将用户所需要的语义服务结果返回给用户，至此，页面浅搜索也就完成了。

如图 4 所示，索引存储方法是搜索引擎最常用的存储方法。索引表由若干索引项组成。在语义服务资源搜索引擎中，采用稠密索引和稀疏索引结合的方式，以达到快速查找的目的。

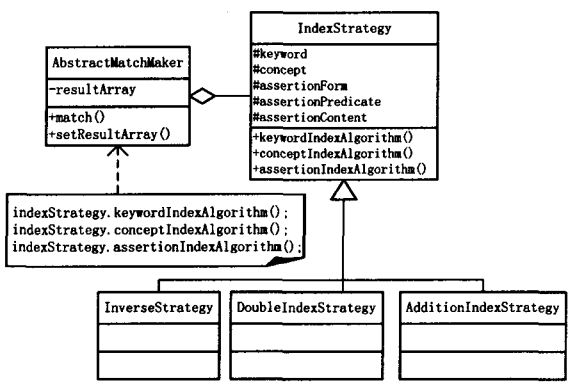

图 4　索引存储结构图

(2)语义服务深搜索方法

语义服务搜索引擎应能处理深层(DEEP Semantic Service)语义服务里所包含的数据。所谓深层搜索是指搜索那些放在数据库中的信息。语义服务深搜索与语义服务浅搜索不一样，页面深度搜索不是要搜索出要找的服务是什么，而是要找出服务所提供的内容是什么。对请求语句"query the airline price from Wuhan to Beijing and query the airline's start time and arrive time."页面浅搜索只需要找到满足条件的服务，至于服务中隐含的数据不要求搜索出来。而对于深搜索需要找出隐含在服务描述文件后面的在数据库、本体或者其他存储方式里面的数据。比如价格的具体数值，航班的具体起飞时间和到达时间等等。图 5 描述了语义服务深搜索的基本方法。

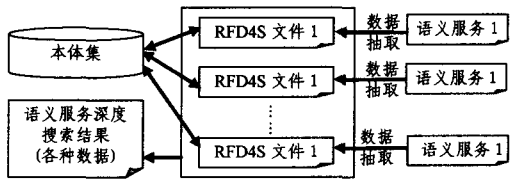

图 5　语义服务深搜索方法

语义服务深搜索的实现方法是：①通过前面的搜索得到以 RDF4S 格式描述的所需要的语义服务。②通过语义本体查询语言将各种语义服务中所隐含的“数据”查找出来。

一般来说语义服务的真正数据都是存储在数据库或者本体库里。本文所定义的语义服务的隐含数据的存储方式是通过 SQL Server2000 数据库变换后存放在本体里面。本体有两个重要的功能，首先它可以起到一个数据库的作用。本体的另外一个重要功能是它具有语义描述功能，相当于一个本体，可以提供语义服务。

5　实验测试

本文实验在国家高技术研究发展计划(863)项目“面向语义 Web 服务资源的软件设计语言的实现及其应用”(项目资助号：2006AA01Z168)资助下，在分析和研究已有旅游系统的基础上，设计并定义了满足电子商务旅游系统典型应用场景开发的 31 类语义服务。这 31 类语义服务包含基本的原子服务和用于服务组合的组合服务。这 31 个服务囊括了几乎旅游领域的全部内容。这些旅游服务严格按照 RDF4S 的规范并按其规范中的基本本体、业务本体、技术本体详细描述了各个语义部分所需要的资源属性等。这些 RDF4S 格式的资源集的设计成功可以为语义服务资源制作、语义服务组合设计案例提供了基本的资源范本和支撑。具体的 31 类语义服务如表 1 所列。

表 1　31 类语义服务

服务编号	服务名称	服务编号	服务名称
SWS001	国内航班服务	SWS002	国际航班服务
SWS003	国内酒店服务	SWS004	国际酒店服务
SWS005	出租车服务	SWS006	火车服务
SWS007	长途汽车服务	SWS008	轮船服务
SWS009	城市链旅游计划服务	SWS010	城市酒店服务
SWS011	城市间交通服务	SWS012	城市酒店出行服务
SWS013	餐厅服务	SWS014	市内交通服务
SWS015	奥运赛事服务	SWS016	体育馆查询服务
SWS017	景点观光服务	SWS018	游客支付服务
SWS019	航空公司收取款项语义服务	SWS020	出租车收取款项语义服务
SWS021	长途汽车公司收取款项语义服务	SWS022	轮船轮渡公司收取款项语义服务
SWS023	商家收取款项语义服务	SWS024	银行预存款项语义服务
SWS025	银行撤销预存语义服务	SWS026	银行转帐服务
SWS027	用户折扣服务	SWS028	导游预定服务
SWS029	投诉服务	SWS030	翻译服务
SWS031	休闲娱乐服务		

根据这些 RDF4S 格式的资源集，建立了一个语义电子商务旅游服务平台。在此平台上，我们部署了上述 18 类各种电

子商务语义服务资源在不同的机器上进行实验验证。下面是搜索语义服务测试示例：在搜索输入框输入“query the airline price from Wuhan to Beijing and query the start time and arrive time”，点击搜索后得到实验结果。

通过本文的方法建立的电子商务旅游语义服务搜索平台，在搜索的文本输入框输入动宾结构的陈述句，在部署好的以 RDF4S 为描述格式的语义服务中能够得到比较准确的实验结果。图 5 和图 6 是本文的一个实验过程。

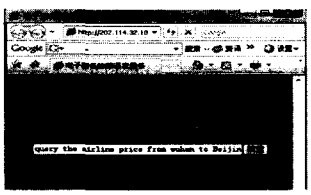

图 5 输入语义服务搜索条件

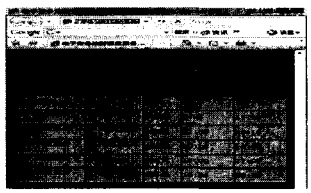

图 6 语义服务搜索结果

结束语 传统的搜索引擎都是按照关键字搜索网页，且大部分只是静态网页，因此对于语义服务，只能通过编写程序到 UDDI 中去提取已经注册了的网络服务。这种需要通过编写程序去搜索所需要的服务已经越来越难以满足普通人尤其是非程序员对网络服务的要求。为了解决这些面临的困难，本文试图设计一种面向非程序员的语义服务搜索方法。基于此，本文详细分析了一种基于类自然语言驱动、以 RDF4S 格式发布的语义服务的搜索方式。

基于类自然语言驱动的语义服务搜索在目前仍然是一个新的领域，还有很多的关键问题需要去解决。尤其这种面向非程序员的语义服务搜索方法，它面临一个“人”和“软件(这里是指语义服务)”进行交互的一个巨大鸿沟。为了逐步解决人机交互的每一步，哪怕是取得一点点进展都需要进行大量的研究，这里不仅涉及到语义计算的研究、自然语言理解、自然语言处理，还涉及到形式化验证以及各种优化问题的研究。为了不断地完善这种基于类自然语言处理的语义服务搜索方法，今后的研究主要从以下几个方面进行：①分布式环境下的语义服务搜索优化；②并行环境下语义服务搜索优化；③问答式的语义搜索服务方式实现等等。

参 考 文 献

[1] Verma K, Sivashanmugam K, Sheth A, et al. METEOR-S WSDI: A Scalable P2P Infrastructure of Registries for Semantic Publication and Discovery of Web Services[J]. Journal of Information Technology and Management, 2005, 6(1): 17-39

[2] Paolucci M, Kawamura T, Payne T R, et al. Semantic matching of Web services capabilities[C]// Horrocks, ed. Proc. of the Int'l Semantic Web Conf. Sardinia: Springer-Verlag, 2002: 333-347

[3] 马应龙，金蓓弘，冯玉琳. 基于进化分布式本体的语义 Web 服务动态发现[J]. 计算机学报，2005，28(4)：603-615

[4] Lin Lin , Arpinar I B. Discovery of Semantic Relations between Web Services[C]// Web Services, 2006. ICWS '06. International Conference. on Sept. 2006: 357-364

[5] Paolucci M, Mawamura T, Payne T R, et al. Semantic Matching of Web Service Capabilities[C]// First International Semantic Web Conference. 2002: 333-347

[6] 吴健，吴朝晖，李莹，等. 基于本体论和词汇语义相似度的 Web 服务发现[J]. 计算机学报，2005，28(4)：595-600

(上接第 87 页)

值的发生及其发生的时间，是我们下一步工作的目标。

参 考 文 献

[1] Paxson V. End-to-end Internet packet dynamics[J]. IEEE ACM, Trans Networking, 1999, 7(6): 277-292

[2] Allman M , Paxson V . On estimating end - to - end network path properties [C]// Proc. ACM SIGCOMM. 1999: 263-274

[3] Johari R, Tan D K H. End-to-end congestion control for the Internet: Delays and stability[J]. IEEE/ACM Trans on Networking, 2000, 9(6): 818-832

[4] Wong J W. Queuing network modeling of computer communication networks[J]. Computing Surveys, 1978, 10(3): 343-351

[5] Jiao Liangbao, Zhang De. Differential AR algorithm for packet delay prediction[J]. Progress in Natural Science, 2006, 16(4): 437-440

[6] Li Q, Millis D L. Jitter-basd delay-boundary prediction of wide area networks[J]. IEEE Trans. on Networking, 2001, 9(5): 578-590,

[7] Yang M, Li X R. Predicting end-to-end delay of the internet using times series analysis[R]. University of New Orleans, 2003

[8] Parlos A G. Identification of the Internet end-to-end delay dynamics using multi-step neuro-predictors[C]// Proc. of the 2002 Int. Joint Conf. on Neural Networks. 2002

[9] Srikar D. Empirical Modeling of End-to-End Delay Dynamics in Best-effort Networks[D]. Texas A&M Univ, 2004

[10] Gelenbe E. Stability of the random neural network model[J]. Neural Computation, 1990, 2(2): 239-247

[11] Gelenbe E. Learning in the recurrent random neural network [J]. Neural Computation, 1993, 5(1): 154-164

[12] Mohamed S, Rubino G. A study of real-time packet video quality using random neural networks[J]. IEEE Trans on Circuits and Systems for Video Technology, 2002(12): 1071-1083

[13] Gelenbe E, Liu P, Laine J. Genetic Algorithms for Route Discovery[J]. IEEE Trans on Systems, Man, and Cybernetics, 2006, 36(6): 1247-1254

[14] Kocak T, Seeber J, Terzioglu H. Design and implementation of a random neural network routing engine[J]. IEEE Trans. on Neural Networks, 2003, 14(5): 1128-1143

[15] Fujimoto K, Ata S. Statistical analysis of packet delays in the Internet and its application to playout control for streaming applications[J]. IEICE Transactions on Communications, 2001, E84-B(6): 1504 -1512

[16] Mallik K. Predictor development for controlling real-time applications over the Internet[D]. Texas A&M Univ, 2005

[17] www. caida. org