摘 要

随着Internet的迅猛发展，Web上可获得的Web服务急剧增多，如何从庞大的服务群中有效地获得所需功能的Web服务和如何在功能相似的Web服务中找到最佳服务成为了值得关心的问题，这些正是Web服务发现的任务。然而传统的服务发现技术是在语法描述的基础上，通过关键词匹配来实现的。这样的Web服务发现技术，由于缺乏对Web服务的语义信息的描述，智能化程度不高，无论在精确度还是在返回率上都无法满足人们日益增长的需求。要得到更好的结果，需要更高效、更完善的Web服务发现技术。把语义Web技术引入Web服务就可以解决Web服务的这个问题。语义Web的基本思想是为Web资源添加语义标注。它并不是另外一个独立的Web，而是对当前Web的扩展。在语义Web中，信息具有良好定义的语义，可以更好地促进人与机器间的协作。理想情况下，语义Web和Web服务技术应该可以很好地融合在一起，语义Web服务就是两种技术的结合。因此，基于语义的Web服务发现逐渐成为了当前的研究热点。研究结果表明语义描述和服务本体论的应用能够有效地提高服务发现结果。

自动化的 Web 服务发现是语义 Web 服务研究领域的基本问题，本质上它是一个自动定位满足用户需求的 Web 服务的过程。本文提出一个能够实现自动化的 Web 服务发现的语义 Web 服务发现模型，它主要具有以下三个特点：

首先，提出了 Web 服务语义描述模型，综合使用 OWL-S 和信誉度本体来描述 Web 服务。OWL-S 作为一种语言，被广泛地用于描述 Web 服务的语义信息以实现自动化的语义 Web 服务发现。但是，OWL-S 并不成熟，还存在一些缺点，尤其是 OWL-S 不能描述 Web 服务的信誉度。为了解决这个问题，本文定义了 Web 服务的信誉度本体 ROWS（Reputation Ontology for Web Services）并用它来描述 Web 服务的可信任性和服务质量等信息。

其次，服务匹配算法采用基于语义相似度的匹配。本文使用相似度用来度量发布的服务和请求的服务之间的“相似程度”。发布的服务和请求的服务之间的相似度由 OWL-S Profile 的功能相似度和信誉度本体的相似度联合决定。

再次，在匹配器中引入了 Web 服务的信誉度管理器。匹配器是实现自动化的 Web 服务发现的软件设施。信誉度管理器负责 Web 服务的信誉度管理。在信誉度管理器的理想模型中，可以使用第三方的权威机构来管理 Web 服务的信誉度，但

目前还不存在这样的机构，只能使用反馈机制来实现信誉度管理。信誉度管理器收集客户的反馈信息并把它们保存到持久的数据中

关键字：语义Web服务；服务发现；信誉度本体；语义相似度

ABTRACT

With the fast development of internet, the growing number of web services available on the Web raises new and challenging search problems: efficiently locating functionality-desired web services among numbers of web services and selecting a best one among large numbers of functionality-similar web services. While these just are the task of web service discovering. However, traditional web service discovering technology is done by keyword match based on the syntactic description of web service. Such web service discovering technology can not capture the semantic information of web service and is lack of intelligence, so it can not meet the growing demand of people. In order to get a better result of discovering, a more efficient and perfect web service discovering technology is needed. This is where the Semantic Web comes to play. The Semantic Web is an effort to facilitate the web resources with semantic descriptions. It is not a separate Web but an extension of current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation. Ideally, Web Services and the Semantic Web should support each other but there are still some gaps. The Semantic Web Service is an attempt to bridge the gap between Web Services and the Semantic Web. So Semantic-based Web Service Discovery is becoming a hot research topic. The researches show that the use of semantic description and ontology of web service can improve the result of service discovering

Automatic Web Service Discovery is an automated process to locate the Web Services that match the client' s requirements appropriately. It is the basic issue of Semantic Web Services' research. This paper describes a model for flexible discovery of Semantic Web Services. The proposed approach has three main features compared with other approaches.

Firstly, OWL-S and Reputation Ontology are used to describe the Web Services. As both a language and ontology, OWL-S is widely used to add

semantic annotations to Web Service and enable automatic discovery of Web Service. However, OWL-S is still evolving and has some shortcomings, especially that it can not describe the reputation of Web Services. In order to overcome this, the Reputation Ontology for Web Services is defined by the author in this paper. The proposed Reputation Ontology can describe the reputation of Web Services. In order to overcome this, the Reputation Ontology for Web Services is defined by the author in this paper. The proposed Reputation Ontology can describe whether a Web Service is trusty, as well as the Quality of Service and other things.

Secondly, the matching algorithm is based on semantic similarity between the candidate services and client' s request. The concept "Similarity" is used to describe how much the candidate services match the client' s request constrains. Both OWL-S Profile Functional Similarity and Reputation Ontology Similartiy are calculated to measure the similarity between the candidate services and the client' s request.

Thirdly, the Reputation Manager of Web Services is introduced in the Matchmaker which is a software infrastructure that enables automatic discovery of Web Services. It is responsible for reputation management of Web Services. In the ideal concept model of Reputation Manager, there is a third-part Reputation Authority which is in charge of the measurement of Reputation for Web Services. But currently the third-part Reputation Authority does not exist, so a feedback mechanism is used by the Reputation Manager which collects the feedback from the clients and save them in a persistent database.

Keywords: Semantic Web Services, Web Services Discovery, Reputation

原 创 性 声 明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的科研成果。对本文的研究作出重要贡献的个人和集体，均已在文中以明确方式标明。本声明的法律责任由本人承担。

论文作者签名：王艳　　日　期：2006.5.18

关于学位论文使用授权的声明

本人完全了解山东大学有关保留、使用学位论文的规定，同意学校保留或向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅；本人授权山东大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或其他复制手段保存论文和汇编本学位论文。

(保密论文在解密后应遵守此规定)

论文作者签名：王艳　导师签名：　日　期：2006.5.18

第1章 序论

1.1 引言

Web服务是Internet上松散耦合并可重用的分布式软件组件。它对离散的功能进行语义上的封装，通过标准的Web协议向外部程序提供服务，包括提供信息(如天气预报)或完成一些动作从而对现实世界产生影响(如在线预订机票)。由于Web服务具有完好的封装性、松散耦合性、使用协议规范性等特点，在很多领域得到广泛的应用。

随着互联网络的广泛应用和高速发展，各种基于网络的Web服务如雨后春笋般出现了。Web已从静态的页面的存储库发展到了今天的交互式的，自动的，智能的Web服务的存储库[3] 多个Web服务的协作能满足用户动态地、即时地提出的任务执行，信息提供、商业交易的需要。Web服务为应用开发者和终端用户带来了前所未有的优势。由于Web服务采用被广泛接受的标准如SOAP等，Web服务应用模式简化了商业应用的开发和交互，实现了代码重用和松散耦合。此外，它还提供给终端用户直观的浏览界面，使得他们来选择，设定和组装自己的Web服务。但是，与此同时，用户面对表现形式和复杂性都可能不相同的服务，如何正确、高效地从如此庞大的Web服务群中找到自己所需的服务则变得越来越具有挑战性。特别地，Web数量的急剧增多，用户将面临大量功能相似的服务选择，如何从众多的功能相似的Web服务中发现最佳服务，无歧义地规范化Web服务的质量同样也成为当今Web服务的热点研究之一。而这些正是Web服务发现的任务。因此，进行提高Web服务发现的能力的研究具有重要意义。高效的Web服务发现技术成为用户有效利用Web服务的关键。为此，研究者们针对提高Web服务发现高效性和准确性展开了研究。目前，基于Web服务发现的研究主要体现在如下几个方面:(1)基于UDDI扩充服务语义信息，以提供对服务发现的支持；(2)如何描述服务使之有语义；(3)在服务描述的基础上研究支持智能化的语义匹配，以提供对服务准确定位；(4)指导性地指出了支持Web服务发现的相关研究热点[4,5]，如服务质量评价等问题。

1.2 web服务发现

所谓 Web 服务发现，就是客户以某种方式在不同类型的 Web 服务中找到其想要的服务，以执行 Web服务请求[6]. Web 服务发现是 Web 服务系统架构中的一个重

要部分。Web 服务发现整个过程由如下四个步骤组成[7]（如图 1-1）：

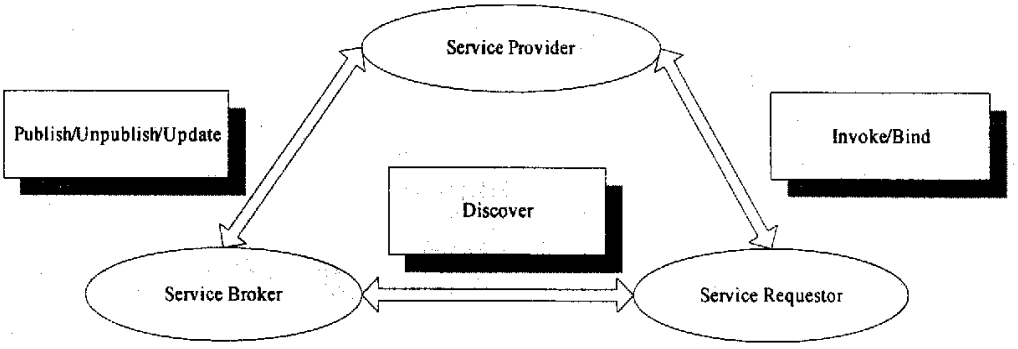

图 1-1: web 服务发现模式

Fig1-1: web Service Discover Model

(1)服务提供者描述其提供的服务(Service Description)

(2)服务代理把服务描述进行分类和发布(Service Publishing)

(3)服务请求者向服务代理提出请求是否有那些能提供所需功能的服务

(4)服务代理把服务请求与已存储的服务进行匹配，然后把匹配结果返回给服务请求者(Service Matchmaking)，然后，服务请求者就能够根据已发现的服务来调用该服务。为了达到高效性，整个发现过程必须达到一些基本的需求。接下来，我们来分析一下上面提到的服务发现过程中四个步骤的各自的基本需求。

* 服务描述需求

对服务的分类，发现和使用而言，Web 服务的功能的描述是很重要的。服务描述应该包括服务的功能性的属性描述(如说明服务是干什么的，what is service does)和非功能性的属性描述(服务的分类，信息交换相关的等等)，文献[8]中给出了一系列的非功能性属性以及他们的使用。服务描述需要能被人工和机器理解。这意味着服务属性需要在语法层和语义层上进行描述。语法信息是关于服务实现方面的，专门针对编程人员的需要。语义信息是关于服务的概念方面的，旨在为终端用户避开底层的技术细节，同时帮助开发者找到最合适的服务。假设有一个股票报价服务，输入为一个表示股票符号的字符串，返回为一个表示股票报价结果的数字。语法信息代表输入参数是一个字符申输出是一个数字，而语义信息则传达了股票报价市场环境下字符串和数字现实意义。因此，服务描述语言应该能用语义和语法的表示方式来描述 Web 服务功能性的，非功能信息。描述语言应该支持基于描述的推理。例如:我们想要找一个 Booking Service，我们希望我们的

请求与任何提供旅店预定的服务进行匹配。一种可行的方法就是使用本体。本体包含了对领域中基本概念以及它们之间关系的计算机可使用的定义。本体用于人之间，数据库之间和应用之间的领域信息的共享。

* 服务发布需求

服务发布是一个使得服务能够通过 Web 广而告知，因而能被广泛使用的基本步骤。它提供了服务共享的平台。因而需要建立一个全球化的、与平台无关的、开放式的架构以共享信息，共享应用。UDDI 是一种解决方案。它是一套基于 Web 的、分布式的、为 Web 服务提供信息注册和查询的实现标准规范，UDDI 注册中心为 Web 服务提供了一个良好的服务发布、维护和管理环境，受到业界的深厚的支持。

* 服务请求描述的需求

服务请求描述是服务发现过程重要的一部分，必须能描述服务的功能。为了能使服务请求与广告服务能在共享语义基础上进行匹配，遵循与服务发布时的相同的服务描述规范。

* 服务匹配需求

服务匹配过程是在服务描述的基础上，对服务请求与已发布的服务描述进行匹配。因此，为了提供服务匹配能力，需要利用服务请求描述和广告服务描述中的属性的语义和语法信息基础上，以准确和高效性为目标，进行智能化的匹配。

1.2.1 Web 服务发现所面临的问题

目前因特网在 Web 服务的表达和检索方面，仍然存在着许多技术缺陷，制约着 Web 服务的准确、高效的发现。主要体现在以下一些方面：

* 以语法性语言表达的 Web 服务，主要是面向用户直接阅读的，不利于计算机直接阅读和处理；

* 不同团体对同一领域事物的认识和表示往往不同，使得来自服务提供者与服务请求者关于同一 Web 服务的描述存在着冲突，这种认识上的差异所产生的描述差异可被称作语义异构，具体表现在：

(1)不同的服务描述使用多种术语(词汇)表示同一概念；

(2)同一概念在不同的服务描述中表达不同的含义；

(3)各服务描述使用不同的结构来表示相同(或相似)的信息；

* 以关键字匹配的方式为主的检索，根据广告服务描述中是否包含请求查询中的关键词来返回结果，由于许多不相关的服务也会在它的描述中包含查询关键词，检索的结果往往会出现很多不相关的Web服务，随着服务数量的增大，检索的准确率就越低。同时这种关键字匹配的方法，查询关键词与广告服务描述中的关键词可能是语义相同但是非语法相同的，遗漏了大量与检索概念同义或相关的内容信息，因此检索在查全率方面不高，难以达到期望效果；

* 服务的检索只是对服务功能描述的关键词匹配，无法充分反映服务所提供服务的功能信息，造成服务检索结果不理想；

* 针对大量相似服务的发现，缺乏Web服务质量支持以实现最佳服务的发现。

1.2.2 研究现状

针对Web服务描述信息的丰富程度的不同，目前Web服务发现方法的主要研究可分为两类：

（1）语法级

在描述语言上，其着重描述Web服务接口的语法，对行为约束缺乏有效支持；在匹配算法上，大多是基于关键字匹配。典型系统有IBM，Microsoft,SUN等公司各类UDDI系统（基于预定义分类和WSDL描述语言）。语法级服务发现方法实现相对简单，但查准率低，很难保证服务组合的相容性，对Web服务复用、验证和管理的支持远远不够。

（2）语义级

在描述语言上，采用本体论来解决传统语法级Web服务描述的异构性,增强对Web服务的功能、行为的语义描述；在匹配算法上，依赖于逻辑演绎和推理，具有查准率高、匹配效率不佳、实用性差等特点。典型研究有卡内基+梅隆大学的augment UDDI Registry系统（基于本体DAML的非轻量级语言DAML-s），乔治亚大学的SpeedR项目（基于本体OWL的WEB服务描述语言OWL-s，即DAML-S1后续版本）。总体来说，现有语义级方法采用的Web服务描述语言过于复杂而不全面（普遍缺乏服务信誉度描述），且缺乏灵活、有效的服务匹配算法，不利于组合过程的系统监控、性能分析和动态调整。

1.3 Web服务发现语义化的提出

1.3.1 自动、智能化Web服务发现

如何有效地将web服务的功能及行为特征表示出来，进一步的，这些表示一定是计算机可以理解的形式，使其能自动智能的根据服务功能性进行推理，并有效的集成相关服务的功能，完成用户任务。Web服务不同于以往的web应用程序的地方是它强调了功能的分割及其模块化，而不是传统的面向过程的思想。

这样，迫切需要web基础设施能理解web服务的功能性及行为，使web服务能在智能web上自动的被发现和集成，甚至扩展已有的功能性。这样，web服务的自动和智能化越来越成为其不断发展的迫切要求。这样，就产生了两种不同的策略实现:对现有的web改造，使其本身具有支持自动和智能性，语义web的研究就是这个思路的代表，从简化实现和强调过渡出发，语义web倡议对web上的资源以结构半结构化的语义描述，然后建立一个能消费和发布语义增强的web资源的语义web，包括web服务在内的所有web资源以相同的方式为计算机解释，使用，最终实现web的智能性和自动化；另一种策略试图从技术层面上提高web服务描述(主要是技术性的语法信息)的标准性，在规格化所有web服务相关技术的基础上，提高web服务的自动化水平，并具有可控的智能性。

Web服务和语义Web都属于Web上的本体论研究，其共同目标都是通过利用Web上人和机器都能够存取的内容，创建智能自动服务及商务处理基础设施。考虑以上二者的结合，实现功能互补是一种自然的选择。首先，用于描述Web服务的WSDL并不能很好地表达Web服务的语义信息；不包含服务执行过程的信息，从而不支持动态性和自动化。其次，从语义Web的角度看，一系列本体论能够通过Agent使Web服务自动地被发现、激活及执行，并且本体论为服务及服务间关系的描述提供了强有力的手段。语义Web服务是语义Web和Web服务的结合，可为Web服务的发现、执行、解释和组合的自动化提供有效的支持。

1.3.2 Web语义化—语义Web

随着 Web应用的迅速发展，Web应用面临着自动化及语义保持的问题。对于用户提交的应用请求，如何根据语义信息执行分布在Web上的相关服务，并自动地进行这些服务间的切换，是语义Web被提出的动机所在。使机器可存取Web数据并实现处理的自动化是语义Web的目标。就语义Web本身而言，它是目前Web

应用的扩展。它能够描述一定的语义，使计算机及人类能够更好地协调合作。从语义 Web 的表示方式来看，可以认为它是 RDF 和元数据(metadata)对 Web 上数据的抽象表示。是本体论领域模型的具体表示和应用实例。而本体论是一组概念及这些概念间关联描述的集合，它描述了包括客观事物及它们之间联系的领域知识。基于语义 Web 的服务描述语言如 DAML-S, OWL，建立在 XML 和 RDF 的基础上，为机器提供了读取数据以及对数据进行解释和推理的能力，使得面向人的 Web 转换到了语义 Web。

Web 服务发现的研究目标是服务发现的高效率和自动化，在服务发现技术中，利用语义描述和服务本体论是达到该目标的有效途径:而如何用 OWL 等语义标记语言描述服务，如何将语义 Web、本体论和语义推理应用到 Web 服务的查找和匹配中极具研究价值。从这个意义上说，Web 服务发现是语义 Web 在 Web 服务中的一种应用。

对 Web 服务发现技术来说，语义 Web 无疑是一种根本的解决方案，它力图将人类的智慧转化为机器的智能，使其能代表人类工作，智能 Agent 技术代表了这类研究的成果。但是，实现语义 Web 的目标相当困难，在语义 Web 还未实现的情况下，考虑如何利用语义 Web 研究领域对 Web 资源语义化的成果，使 Web 服务的语义信息能够有效地被描述和发现、推理，是当前提高 Web 服务发现水平、解决 Web 服务发现和集成中的问题的捷径。

1.4 本文主要研究内容

1.4.1 论文背景

本课题来源于实验室的科研项目-旅游服务集成平台——IPVita (the Intelligent Platform of Virtual Travel Agency)。将各个企业提供的旅游服务根据类别按照流程组织起来，为游客提供满意的优质的旅游服务。该平台在语义的支持下，首先将用户的需求形式化，然后根据形式化的需求自动生成组合流程，之后自动为组合流程匹配满意的服务从而形成旅游流程，最后实施旅游流程，为用户提供满意的旅游服务。整个过程最大的体现了平台的智能化。

整个平台分为四个模块，语义管理模块、组合流程生成模块、服务注册中心和流程执行模块，其中，注册中心中的服务发现模块在整个平台中占据重要地位，是实现服务共享、复用的重要前提。Web 服务发现的效果直接关系服务复用的质量，

影响到服务组合的相容性（compatibility）和可替换性（sub-stitutability），关系到能否真正实现服务的“即插即用”。

本论文主要论述Web Service中面向服务发现的语义描述，以及语义Web相关技术，旨在实现基于语义Web的Web服务发现机制。主要讨论的关键技术和提出的创新研究包括：

（1）研究OWL-S的应用及扩展。OWL-S规范虽然得到广泛的应用，但还存在很多不足，尤其是不能很好地支持语义Web服务的信誉度[44]，如QoS、可信任性等。本文的一个重要研究内容就是分析OWL-S的应用和不足，研究并设计一个可以描述Web服务的信誉度的本体，并在此基础上提出了Web服务的语义描述模型。

（2）研究基于语义的Web服务自动发现。在现有算法的基础上提出一个更优的基于语义推理的服务匹配算法，这个匹配算法应该支持信誉度的匹配。

（3）研究并设计基于UDDI的语义Web服务匹配的框架系统。该模型应该能够有效地利用Web服务的信誉度描述。

1.4.2 论文的结构

本论文首先论述语义 Web 服务的背景技术，然后分析自动化的 Web 服务发现所要满足的条件以及当前的 Web 服务发现技术，接着论述作者的几项主要工作成果：

（1）Web 服务的语义描述模型，采用了 OWL-S 和信誉度本体 ROWS 相结合的方法；

（2）服务匹配算法，采用基于语义相似度的匹配算法并引入了信誉度相似度的匹配；

（3）IPVita 的总体结构以及组成模块尤其是语义 Web 服务发现模块的设计与实现。

本论文共分五章，各章的主要内容介绍如下：

第一章 绪论

本章全面介绍Web服务发现技术，分析自动智能化的Web发现Web服务发现。论述论文选题的背景和意义，并简单介绍了作者的研究工作。

第二章 基于语义的Web服务描述模型

本章在语义扩展传统Web服务描述模型的基础上，提出一种Web服务的语义描述方法，该方法综合使用了OWL-S和信誉度本体ROWS。ROWS是本文在文献[44]

的基础上引入的概念，它的定义是本章的重点。

第三章 语义Web服务的匹配模型

本章首先引入相似度的概念，使用相似度来度量服务请求和待选的服务之间的相似程度，然后在文献[28,44]的基础上提出了一个基于语义信息的相似度匹配的服务匹配算法，该算法综合考虑了服务的功能匹配和信誉度匹配。

第四章 语义Web服务发现在IPVita中的实现

本章给出了旅游服务集成平台——IPVita（the Intelligent Platform of Virtual Travel Agency）的总体结构，并分别介绍了各个模块的功能和实现，尤其是服务发现模块的实现。

第五章 总结与展望

本章对本文的研究工作进行总结并提出进一步的研究方向。

第2章 Web服务的语义描述

当前的Web服务描述技术的最大问题是没有包含语义信息，不能支持自动化的Web服务发现，因此为Web服务提供语义描述信息是实现自动化的Web服务发现的首要条件。本章首先分析当前的Web服务的UDDI表示及其不足，然后研究设计了Web服务信誉度的本体ROWS，综合使用OWL-S和ROWS提出一种Web服务的语义描述模型对Web服务进行语义描述，最后定义了语义描述到UDDI的映射。

2.1 Web服务的UDDI表示

UDDI定义了5个主要的数据结构，这些结构用于表示一个机构、机构的服务、实现技术以及与其它商务实体之间的关系。这5个数据机构及其功能如下所示：

a) BusinessEntity：表示提供Web服务的商业或者机构；

b) BusinessService：表示一个Web服务或其它某些电子服务；

c) BindingTemplate：表示Web服务到其访问点（它的URL）以及到tModel的技术绑定；

d) TModel：表示一个特殊类型的技术（如SOAP、WSDL），或者表示一种分类系统，BindingTemplate引用的tModel说明了Web服务使用的技术类型；

e) PublishAssertion：表示两个商务实体之间的关系。

BusinessEntity结构的XML Schema定义如图2-1所示。其中，businessKey，operator，和authorizedName这三个businessEntity的直接属性分别表示businessEntity的主键、实施注册的UDDI操作入口站点以及对该businessEntity拥有所有权的用户ID，businessKey是在注册后由UDDI注册中心自动赋予，并在businessEntity整个生命周期中有效；discoveryURLs是一个充分体现UDDI的发现能力的属性，这个结构包含了多个discoveryURL，访问其中的每个discoveryURL都应当可以获得这个businessEntity的完整XML文本(这个XML文本的顶级元素一定是businessEntity)；name、description和contacts分别表示该商业实体的名、描述和联系方法等；businessServices是一个businessService的容器，它表示了这个businessEntity所能提供的所有Web服务，在businessServices中的每个businessService条目都描述了一个Web服务。

```
<element name = "businessEntity">
```

```
<type content = "elementOnly">
<group order = "seq">
<element ref = "discoveryURLs" minOccurs = "0" maxOccurs = "1"/>
<element ref = "name"/>
<element ref = "description" minOccurs = "0" maxOccurs = "*"/>
<element ref = "contacts" minOccurs = "0" maxOccurs = "1"/>
<element ref = "businessServices" minOccurs = "0" maxOccurs = "1"/>
<element ref = "identifierBag" minOccurs = "0" maxOccurs = "1"/>
<element ref = "categoryBag" minOccurs = "0" maxOccurs = "1"/>
</group>
<attribute name = "businessKey" minOccurs = "1" type = "string"/>
<attribute name = "operator" type = "string"/>
<attribute name = "authorizedName" type = "string"/>
</type>
</element>
```

图 2-1 businessEntity 的结构

Fig2-1 Structure of businessEntity

BusinessService 结构的 XML Schema 定义如图 2-2 所示：

```
<element name = "businessService">
<type content = "elementOnly">
<group order = "seq">
<element ref = "name"/>
<element ref = "description" minOccurs = "0" maxOccurs = "*"/>
<element ref = "bindingTemplates"/>
<element ref = "categoryBag" minOccurs = "0" maxOccurs = "1"/>
</group>
<attribute name = "serviceKey" minOccurs = "1" type = "string"/>
<attribute name = "businessKey" type = "string"/>
</type>
</element>
```

图 2-2 businessService 的结构

Fig2-2 Structure of businessService

其中，serviceKey 和 businessKey 两个直接属性分别表示 businessService 的主键和 businessService 的父类容器 businessEntity 的主键标识,serviceKey 是在注册后由 UDDI 注册中心自动赋予，并在 businessService 整个生命周期中有效，而 businessKey 的值仅当 businessService 的父类容器发生变化时才会被修改；name、description 分别表示该服务的名、描述等信息；categoryBag 的作用与 businessEntity 中是类似的；bindingTemplates 是一个 bindingTemplate 的容器，它表示了这个 businessService 所包含的所有技术绑定信息。

UDDI缺乏语义描述，只能针对一些属性进行基于分类及关键字的搜索，查全率和查准率并不理想，而且需要人的手工参。在UDDI之上添加一个语义层，实现基于语义的服务发现，则可以解决这些问题。

2.2 Web服务的信誉度

语义Web和Web服务相结合称为语义Web服务。带有语义信息的Web服务描述有利于实现自动化的Web服务发现，使得Web服务发现技术不再拘泥于传统的分类和基于关键字查询的技术，而是可以进行基于语义的模糊匹配，使Web服务发现技术更有效和准确。但是满足发现条件的Web服务可能同时有多个，服务质量孰优孰劣如何判定；所得到的Web服务是否可用，即其描述语义的真实性如何判定等都是现实应用中人们所关心的问题，这就是Web服务的信誉度(Reputation)问题。

在网格计算和电子商务中同样有信誉度的问题。但与前两者不同的是，语义Web服务的信誉度除了要关注资源(即Web服务)的质量之外，还要关注其语义信息的信誉问题。因此语义Web服务信誉度应该从两个层次上来解释。第一层次是Web服务的语义真实性。语义真实性是指一个Web服务描述所体现的语义是否真实，有没有不确切的甚至是虚假的部分，譬如某个Web服务宣称自己可以处理任何事情，这当然是虚假的。在确认了Web服务的语义真实性的基础上需要考虑Web服务的质量。Web服务质量的优劣影响客户选择，例如同样是汽车租赁的Web服务，但是服务A提供的车要比服务B提供更为物美价廉，这样用户当然会去选择服务A。

因此，Web服务信誉度的作用主要体现在：1)影响到服务的选择，用户在既能满足预算和时间约束，同时等价格的前提下，优先选择信誉度好的服务；2)对服务质量要求较高的用户，在满足预算和时间约束的前提下，宁肯高价选择信誉度好的服务，而可能不选择信誉度低的服务。显然，这会给信誉度好的服务带来更多的利益；3)信誉度也是影响Web服务QoS的因素之一。

本文的Web服务的语义描述采用OWL-S和信誉度本体。OWL-S作为一种语言，被广泛地用于描述Web服务的语义信息以实现自动化的Web服务发现。但是，OWL-S并不成熟，还存在一些缺点。尤其是OWL-S不能描述Web服务的信誉度。为了解决这个问题，本文定义了Web服务的信誉度本体ROWS（Reputation Ontology for Web Services）并用它来描述Web服务的可信任性和服务质量等信息。

2.2.1 影响 Web 服务的信誉度的因素

影响语义 Web 服务的信誉度的首要因素是服务描述的真假性，Web 服务的描述的真假性用符号 D 表示（各个因素的符号表示见表 2-1），它的值要么为 0 要么为 1。有些服务提供商不够诚实，为了提高自己的 Web 服务被查找到的机率，夸大 Web 服务的能力，把自己的服务说成无所不能。为了杜绝这种情况的发生，只要发现某个服务提供者造假，就把它的信誉度设为 0。

Web 服务的 QoS 是影响 Web 服务信誉度的另外一个重要的因素。Web 服务的 QoS 属性很多，我们主要考虑响应时间、可用性、可维护性和满意度四个属性，分别用符号 T、A、M 和 SA 来表示。后续的研究中，将会增加对其它属性的支持。

响应时间（T）：客户从提交服务请求到完成服务请求所花费的时间，包括服务执行时间（$T_{service}$）、网络上请求的传输时间（$T_{request}$）和响应的传输时间（$T_{response}$）。即

$$T = T_{service} + T_{request} + T_{response} \quad \text{（公式 2-1）}$$

可用性（A）：Web 服务正常运行的概率。由于 Web 服务是按次进行调用的，可以采用统计的方法来计算 Web 服务的可用性。公式如下：

A＝（成功调用的次数）/（调用的总次数） （公式 2-2）

或者

A＝ 1－（不成功调用的次数）/（调用的总次数） （公式 2-3）

可维护性（M）：Web 服务在出现意外的情况下能正确维护的概率。公式如下：

M＝(成功维护意外情况的次数)/(出现意外情况的总次数) （公式 2-4）

满意度（SA）：用户对 Web 服务的满意程度。一般用[0, 1]区间上的一个小数或区间表示。Web 服务的满意度由用户决定，即用户得到服务的结果后对服务所做的评价。公式如下：

$$SA=(SA_1+SA_2+\ldots+SA_n)/n \quad \text{（公式 2-5）}$$

其中 SA_i 表示用户对第 i 次服务的满意度。

还有其它的一些附加属性会影响 Web 服务的信誉度，如机密性和价格。机密性和价格分别用符号 S 和 P 表示。

机密性（S）：用来描述 Web 服务及其提供者是否会泄漏消费者的隐私（现在很多不法的网络服务提供商靠出卖客户的隐私来牟取非法利润），它的取值和 D 一

样，要么为0要么为1。0表示Web服务及其提供者会泄漏用户隐私信息，是不可靠的，这时相应的信誉度应该是0；1表示Web服务及其提供者不会泄漏用户隐私信息，是可靠的，Web服务的信誉度应该由其它因素联合判定。

价格（P）：每次调用Web服务所需的费用。有些Web服务不是免费的，使用它要收取一定的费用。价格作为信誉度的参考因素，不是必须的，只有用户有价格限定或者其它因素相当的情况下才考虑该要素。

除了上述这些因素外，各个领域的Web服务也有特定的影响Web服务的信誉度的因素，即特定领域的信誉度因素。

由这些因素计算Web服务信誉度（用符号R表示）的公式为：

$$\mathbf{R}=\begin{cases}0, D=0\\ 0, S=0\\ f(T,A,M,SA,P), D=1\text{并且}S=1\end{cases} \qquad \text{（公式 2-6）}$$

由公式2-4可以看出，当服务描述不真实或者会泄漏用户机密时，Web服务的信誉度为0；当服务描述真实并且不会泄漏用户机密时，Web服务的信誉度由响应时间、可用性和价格来决定，暂不考虑特定领域的信誉度因素。

因素名字	符号	取值范围
服务描述的真假性	D	0或1
响应时间	T	{t| $t\geq 0$，t是小数}
可用性	A	{a| $0\leq t\leq 1$}
机密性	S	0或1
可维护性	M	{m| $0\leq t\leq 1$}
满意度	SA	{sa|$t\geq 0$}
价格	P	{p| $t\geq 0$}

表2-1 影响Web服务信誉度的因素

Table2-1 The Elements of Web Service Reputation

2.2.2 Web服务的信誉度本体ROWS

Web服务的信誉度本体ROWS（Reputation Ontology for Web Services）的整个结构大致如图2-3所示。

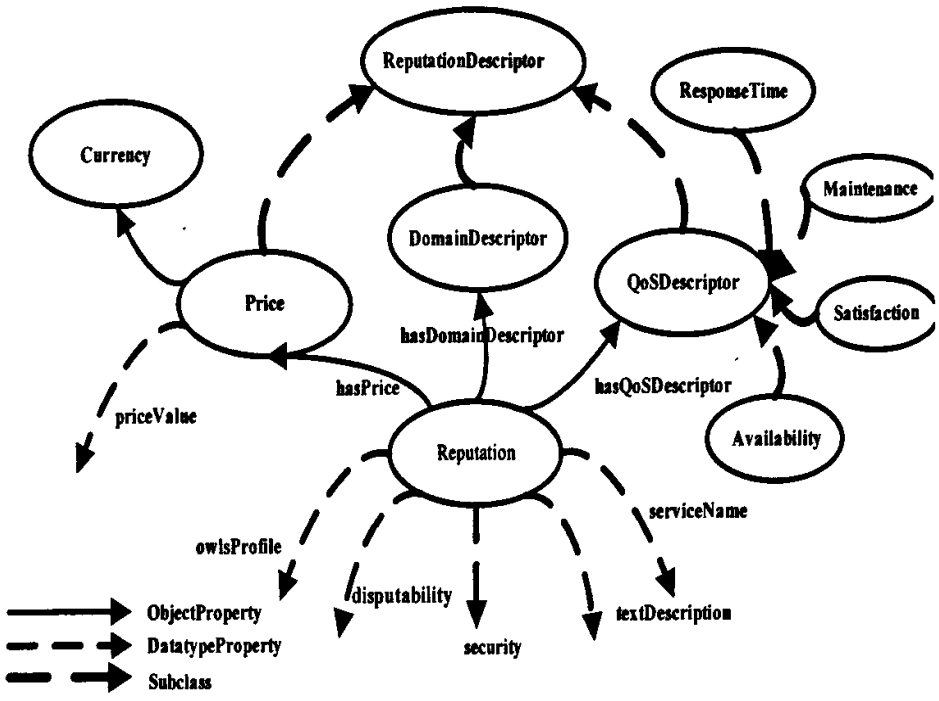

图 2-3 ROWS 的结构

Fig2-3 Architecture of ROWS

ROWS 使用 OWL 创建，主要用来描述 2.2.1 里所述的因素，以后还要进行扩充。类 Reputation 表示 Web 服务的信誉度，它有五个数据属性和三个对象属性。

（一）数据属性

serviceName 表示信誉度所描述的 Web 服务的名字，每个信誉度本体只能有一个 serviceName 属性；textDescription 是针对信誉度本体的一个简单的文字描述，每个信誉度本体只能有一个 textDescription 属性；owlsProfile 是和本体所描述的 Web 服务的 OWL-S Profile 的 URL，每个信誉度本体只能有一个 owlsProfile 属性；disputability 用来描述 Web 服务是否可信，判定 Web 服务提供者是否提供了虚假的描述，每个信誉度本体只能有一个 disputability 属性；security 用来描述 Web 服务是否泄漏消费者的隐私，保护客户的机密信息，每个信誉度本体只能有一个 security 属性。

（二）对象属性

所有的对象属性的基类都是 ReputationDescriptor。hasPrice 用来描述 Web 服务的价格，priceValue 描述币值，而货币种类用对象属性 hasCurrency 描述，hasCurrency 的值域采用 http://www.daml.ecs.soton.ac.uk/ont/currency.owl 定义的 Currency 本体；

hasDomainDescriptor 用来描述 Web 服务的领域属性，目前还没有定义它的子类，有待于进一步进行研究；hasQoSDescriptor 用来描述 Web 服务的 QoS 信息，responseTime 描述 Web 服务的响应时间，availability 描述 Web 服务的可用性，maintenance 描述 Web 服务的可维护性，satisfaction 描述 Web 服务的满意度。

2.2.3 Web 服务信誉度的操作

公式 2-4 的意义不在于计算 Web 服务的信誉度的具体值，而在于比较两个 Web 服务的信誉度。如果两个 Web 服务的信誉度分别为 r_1 和 r_2，各个因素分别为 R_1（d_1，t_1，a_1，s_1，m_1，sa_1 和 p_1）以及 R_2（d_2，t_2，a_2，s_2，m_2，sa_2 和 p_2）。

下面定义两个操作分别用来判断 Web 服务的信誉度是否为 0 和比较两个 Web 服务的信誉度的大小。

* isReputationZero：判断 Web 服务的信誉度是否为 0，如算法 1 所示；

* compareReputation：判断两个 Web 服务的信誉度的大小，如算法 2 所示。

算法 1 的描述如下所示。

算法 1：判断 Web 服务的信誉度是否为 0

输出：为 0 返回 true，否则为 false

```
1   bool isReputation(R(d, t, a, s, m, sa, p)){
2       return (d == 0 || s == 0)? true: false;
    }
```

算法 2 的描述如下所示。

算法 2：判断两个 Web 服务的信誉度 R_1 和 R_2 的大小

输出：$\begin{cases} 0, r_1 = r_2 \\ -1, r_1 \ \pi \ r_2 \\ 1, r_1 \ \phi \ r_2 \end{cases}$

```
1   int compareReputation(R1, R2){
2     if (isReputationZero(R1) && isReputationZero(R2)) {
3       return 0;
4     }else if (isReputationZero(R1) && ! isReputationZero(R2)){
5       return -1;
```

```
6     }else if (!isReputationZero(R1) && isReputationZero(R2)){
7       return 1;
8     }else{
9       if (t1==t2 && a1==a2 && m1==m2 && sa1==sa2 && p1==p2) {
10        return 0;
11      else
12        return compare(t1, a1, m1, sa1, p1, t2, a2, m2, sa2, p2);
      }
      }
13  int compare(t1, a1, m1, sa1, p1, t2, a2, m2, sa2, p2){
14    if((a1 * sa1 * sqrt(t1^2+t2^2)/t1 * sqrt(m1^2+m2^2)/m1 * sqrt(p1^2+p2^2)/p1)
          ϕ (a2 * sa2 * sqrt(t1^2+t2^2)/t2 * sqrt(m1^2+m2^2)/m2 * sqrt(p1^2+p2^2)/p2))
15      return 1;
16    else
17      return -1;
}
```

2.3 Web 服务的语义描述模型

2.3.1 Web 服务的语义描述模型

在因特网环境下，为了实现 Web 服务准确高效的发现，采用标准的方式来描述 Web 服务的各种信息，以减少服务提供者和服务请求者之间所需的共识的程度量，是进行 Web 服务发现的基础。结合 OWL-S 语言和以上介绍，本文构造了语义 Web 服务描述模型，目的就是为服务提供者和服务请求者提供一种标准的方式来描述服务的各种信息。如图 2-4 所示：

如果 Web 服务是一个原子服务，那么如图所示也就是对此 Web 服务的表示，如果 Web 服务是包含多个服务的组合服务，那么对 Web 服务的描述就由对服务组件的描述组合而成。

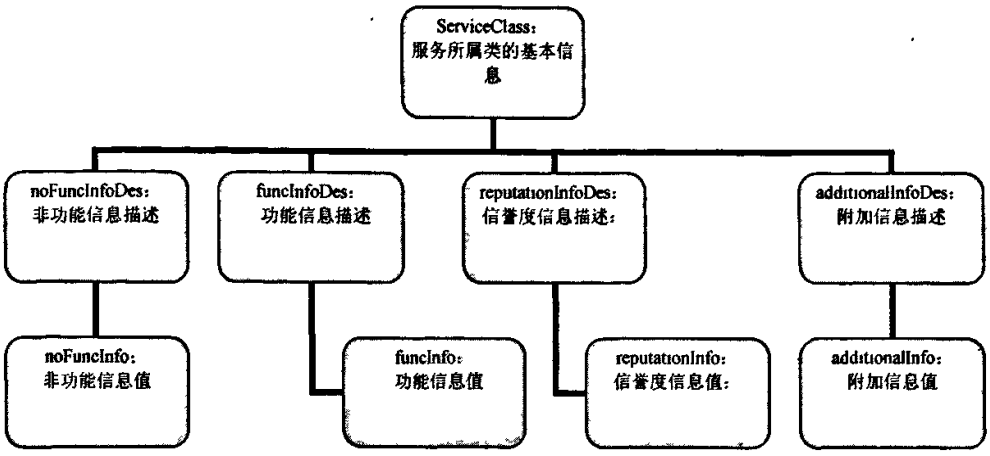

图 2-4: Web 服务语义描述模型

Fig2-4: Web Service Description Model Based on Semantic

此模型提供服务发现所需要的信息，主要由四类信息构成，我们可以对Web服务的语义描述定义为：

$$S=(noFunc(s),Func(s),Rep(s),Add(s))$$

(1) noFunc(s)—服务的非功能信息描述

服务的非功能信息描述包括服务基本信息和服务提供者的信息

服务基本信息提供可供人工阅读的信息，包括 serviceName, textDescription 两个元素。serviceName 指定了所提供服务的名称，可以作为该服务的标识。textDescription 提供了服务的简要描述，一般总结服务的功能，描述服务所需要的输入以及编写者希望服务请求者知道的其他信息。服务提供者的信息包含了指向提供服务的实体的联系信息，由 contactInformation 元素来描述。contactInformation 指定了服务提供者给出的个人或者其他实体的联系方法，对应于 Actor 类的实例。类 Actor 提供的信息包括:name:名称，可以是人名，也可以是公司名称;title:联系人的头衔;phone:联系电话;fax:传真号码;email:电子邮件地址;physicalAddress:物理地址(如邮政地址); WebURL:产品或公司的网址。

(2) Func(s)—服务的功能信息描述

在模型中，服务的功能信息描述表达了服务功能性的两个方面:信息的转变和服务执行引起的状态变化。服务产生的信息转变通过输入和输出来表示。输入属性指定了服务进行计算需要的信息，对应于元素 input。例如购书服务可以要求信

用卡号和用户所购书的书目信息作为输入。输出指定了服务操作的结果，对应于元素 output。例如可以是购书的收据以确认交易。input 元素、output 元素的值都通过类 ParameterDescription 来描述。input (output)包括三个属性:parameterName 提供参数的名称;restrictedTo 提供对参数值的约束，weight 提供参数的权重。

(3)Rep(s)--服务的信誉度信息描述

可以通过 Web 服务的信誉度本体 ROWS 直接提供对服务质量的描述，如 2.2 节所述。

(4)Add(s)--服务的附加信息描述

服务的附加信息由 2 个元素来描述:serviceCategory 和 serviceParameter serviceCategory 引用某本体论或者服务分类法中的项来指定服务所属的分类，其值是类 ServiceCategory 的实例，包含四个属性:分类法的名称 categoryName、对分类法模式的引用 taxonomy、服务在该分类法中所对应的值 value。及相应的代码值 code。serviceParameter 为一可扩展的属性，其值是 ServiceParameter 类的实例，包含属性名 serviceParameterName 和属性值 sParameter 两个属性。serviceParameter 可用来包含任何信息。这些信息可能包括服务的位置等属性。

通过如上所述的 Web 服务语义描述模型，服务可以提供自己的能力说明，服务匹配就可以通过服务提供的上述模版的信息判断服务是否适合用户的要求。

2.3.2 语义信息到 UDDI 的映射

语义信息到 UDDI 的映射就是把 Web 服务语义描述模型中的信息描述转换成 UDDI 描述。语义信息到 UDDI 的映射如图 2-5 所示。下面分别介绍上述 Web 服务语义描述的信息到 UDDI 的映射。

对于 OWL-S 描述中的非功能信息，直接转换成 UDDI 中对应的描述就可以。从图 2-5 可以看出 OWL-S Profile 中的 serviceName 和 textDescription 直接映射到 UDDI BusinessService 数据结构中的 name 和 description 元素；OWL-S Profile 中的 contactInformation 直接映射到 UDDI BusinessEntity 中的 contacts 元素；WebURL 映射到 UDDI BusinessEntity 中的 discoveryURL 元素。但 OWL-S Profile 中也有些元素没有对应的元素，如 contactInformation 中的 title 和 fax 在 UDDI 没有对应的元素。

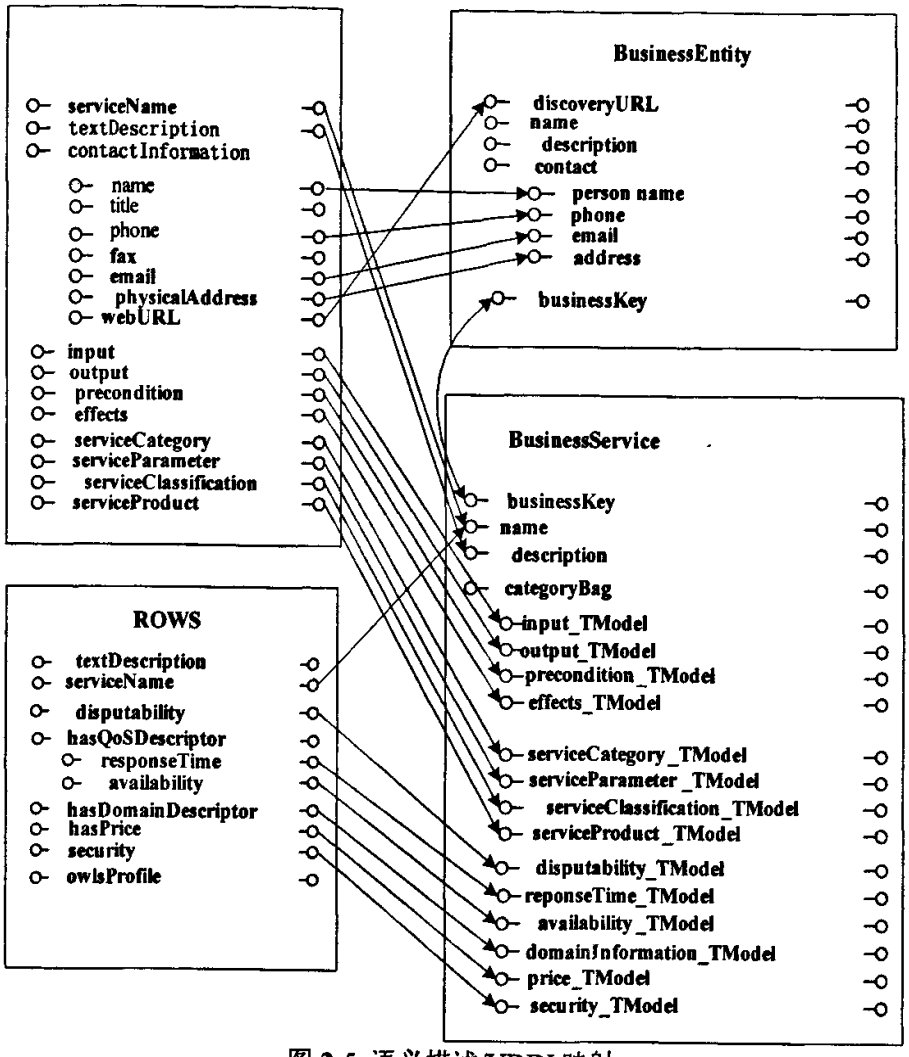

图 2-5 语义描述/UDDI 映射

Fig2-5 The mapping between Semantic Description and UDDI

OWL-S Profile 中的功能描述在 UDDI 中没有对应的描述元素，要映射到 UDDI 中必须采用 tModel 结构。tModel 结构从本质上讲是一个带元数据的命名空间，它表示一个规范、分类法、模型、分类、标识系统或其它的技术概念。Web 服务的 OWL-S 的功能描述信息 input、output、precondition 和 effect 分别使用 tModel 数据结构封装嵌入到 UDDI 中。

同功能描述信息的映射相似，serviceParameter、serviceCategory、serviceClassification 和 serviceProduct 属性也以 TModel 结构嵌入到 UDDI 中。信誉度本体的几个属性 security、disputability、responseTime、availability 和 price 也是分别使用 tModel 数据结果封装嵌入到 UDDI 中，当前虽然没有定义特定的

domainInformation，为了完整性，也把它映射到一个 tModel 数据结构中。也有属性没有映射到 UDDI 中，如 textDecription，但是这并不影响服务的信誉度，信誉度信息并没有丢失。

2.4 小结

本章重点研究和论述 Web 服务的语义描述。传统的 Web 服务的 UDDI 描述缺乏语义信息，不能实现自动化的 Web 服务发现。使用 OWL-S 可以为 Web 服务添加语义信息，但 OWL-S 不能描述 Web 服务的信誉度，所以本章使用 OWL 定义了描述 Web 服务的信誉度本体 ROWS，使用 ROWS 可以有效地描述 Web 服务的信誉度。在此基础之上，我们建立了 Web 服务的语义描述模型，为 Web 服务提供完整的语义描述，并且分析了语义信息到 UDDI 的映射。

第3章 基于语义相似度的Web服务匹配

匹配是在请求者和提供者之间查找可能的匹配资源的过程。匹配不同于简单的发现（完全匹配），它应该能接受和存储请求者和发布者的信息，并能随着新加入的发布和请求发现最满意的匹配，并返给他们。

当一个待选的服务和服务请求"足够相似"的时候，我们认为该服务匹配这个请求。问题是如何解释"足够相似"。当请求的服务和待选的服务正好是同一个服务的时候，当然"足够相似"，但这个条件过于苛刻，服务匹配应该具有很好的灵活性，我们可以使用相似度的概念来量化"足够相似"，这样可以满足服务匹配灵活性的要求。

本章首先分析了当前匹配算法的研究，以及在文献[28]提出的基于语义相似度的服务匹配算法的优点及其不足，并在此基础上提出了一个新的服务匹配算法，引入了信誉度的匹配。

3.1 服务匹配研究现状

语义Web服务发现框架和服务发现模型为服务发现提供了基础设施，对所有的与服务发现推理相关的环节（服务描述、服务发布、服务匹配、服务调用和服务组合）提供了支持。其中，服务匹配环节中的服务匹配算法是整个基于语义的Web服务发现技术的关键问题和难点。服务匹配算法的基本作用就是根据输入的服务请求，从已发布的所有候选服务中选取匹配请求的Web服务，并将匹配的服务列表排序输出。

目前，学术界已经对服务匹配算法做了广泛而深入的研究，研究方法和侧重点各不相同。

文献[2,3,5,6,7]使用OWL-S（以前的版本叫做DAML-S）进行Web服务的语义匹配。它根据请求与服务的输入和输出的匹配情况将服务匹配程度分为四种：精确匹配、插入匹配、包含匹配和不匹配。服务请求和待选的服务之间的匹配取决于它们所有的输出和输入之间的匹配。每个输出或者输入之间的匹配又取决于它们的概念之间的包含关系。

文献[20]将Web服务匹配问题转换为获取请求（概念的析取）的最佳覆盖（Best Cover）问题，即给定一个请求和知识库，要找到作为请求的最佳覆盖的服务集合，这个集合中的每个服务描述都包含与请求尽可能多的公有信息和尽可能少的多余

信息。他们首先形式化定义了描述逻辑中的概念差异，并以此为基础定义了概念之间的相对的多余（Rest）和缺失（Miss）。接着又指出概念的覆盖就是指知识库中任何与该概念共享一些公有信息的任意概念的析取的集合。从而概念的最佳覆盖就是相对于此概念具有最小多余和最小缺失的一个覆盖。最后提出了一个基于超图的算法来有效地计算最佳覆盖，并且通过计算超图横截的代价来量化匹配度。该算法能有效地在服务描述和请求之间进行灵活匹配，但是由于最佳覆盖大多数情况下是一些服务的集合，因此很多时候得到的匹配服务是多个，而不是精确到某个服务。这些服务可能提供对应请求的不同部分或者相互冗余，从而导致服务选择不能准确的完成，甚至导致服务匹配结果是无效的。

文献[21,22]提出了一种高精度的服务匹配算法，它使用过程模型来捕捉服务的语义，使用模式匹配算法来查找满足请求的服务。该方法认为提高服务匹配的精度的关键是能够捕捉服务和请求的足够的语义信息，而过程模型则能很好地提供了这种信息的描述，因此匹配的第一步是服务提供者必须把服务建模成过程模型，然后使用过程查询语言来进行检索。这种方法大大提高了服务匹配的精度，但是它严重地依赖于所建的过程模型，如果建模方式不同或者所建的过程模型不够准确，那么该方法的缺点是显而易见的。

文献[23]提出一种基于逻辑推理和内容检索的混合服务匹配算法。文献[23]认为纯粹的基于对加注的语义信息进行逻辑推理是不够的，因为它不能有效利用隐藏的语义信息。

文献[24,25,26]把服务匹配分为两个阶段。第一阶段，先根据请求的服务的功能性需求（服务做什么、它的输入输出、前提及结果）。这个阶段确保了返回的服务能够满足请求的基本需求。第二阶段，为当前请求任务确定最适合的服务，这基于非功能性需求，比如第一阶段返回的服务的 QoS。第二阶段在过程联合和服务发现领域是一个重要的研究方向，因为动态绑定和调用是一种更加优越的方法。为了实现第二阶段，他们定义了 QoS 本体。使用 OWL-S 描述和 QoS 本体描述进行服务匹配。

文献[27]的服务匹配算法比较灵活，提供了多种匹配策略，并考虑了服务质量与服务效率的平衡。采用了语义距离的机制来实现近似服务匹配。尽管使用 WordNet 等工具可以更方便地计算概念之间的语义距离（WordNet 是一种基于认知

语言学的英语词典），但因为语义距离仍然需要人工建立，工作量大、主观因素和不确定性因素很重，所以服务匹配算法的实用性和可靠性都有一定的局限性。

文献[28]提出了一种基于语义相似度的服务匹配方法。两个 OWL 对象的相似度使用它们共有的信息量来度量，这样，通过计算 OWL-S Profile、Process Model 和 Grounding 的相似度就可以匹配服务请求。这种方法的优点是匹配的精度高，但算法描述过于理想化，不易于实现，计算服务请求和候选服务的相似度时使用 Process Model 和 Grounding 也没有太大的意义，并且也没有包含服务的信誉度描述信息。

3.2 请求服务和发布服务的相似度

文献[28]提出了一种基于语义相似度的服务匹配方法，两个 OWL 对象的相似度使用它们的共有信息量来度量。对于请求服务 REQ 和公布的服务 ADV，通过计算这两个 OWL 对象的 OWL-S Profile、Process Model 和 Grounding 的联合相似度就可以匹配服务请求。该算法可以精确的计算服务请求和待选的服务之间的相似度。但是，它存在以下几点不足：

* 计算服务请求和待选的服务的相似度时没有考虑服务的信誉度信息，这样，提供了虚假描述的服务、速度极慢可用性不高的服务以及价格很高的服务就会顺利地通过了服务匹配，但是这些服务实际上不可以用的。

* 计算服务请求和待选的服务的相似度时还要考虑 Process Model 和 Grounding，这样增加了计算的复杂性，而且所带来的实际效益并不大。实际上，只需要计算 Profile 和信誉度本体的联合相似度就足够了。

* 相似度的计算方法比较复杂，不利于实现。

本章在文献[28]的基础上提出了一个新的计算服务请求和待选服务的相似度的方法，弥补了文献[28]的算法的以上几点不足。

由 2.3 可知，服务的描述模型由四大部分组成：非功能信息、功能信息、信誉度信息和附加信息。服务匹配主要指服务功能匹配以及服务的信誉度匹配，ProcessModel、Grounding 以及 Profile 中非功能描述和附加描述可以忽略。假设请求服务和发布的服务分别用 REQ 和 ADV 来表示，则它们之间的相似度 sim(REQ, ADV)可以用公式 3-1 表示：

$$sim(REQ, ADV) = w_1 sim_{functional}(REQ, ADV) + w_2 sim_{reputation}(REQ, ADV) \quad \text{（公式 3-1）}$$
$$\sum w_1 + w_2 = 1$$

其中，$sim_{functional}(REQ, ADV)$ 表示请求和公布的服务之间的功能相似度，$sim_{reputation}(REQ, ADV)$ 表示请求和公布的服务之间的信誉度相似度。

3.3 请求服务和发布服务的功能相似度

功能相似度的定义基本上基于文献[28]描述的相似度计算方法。OWL-S Profile 分别通过 hasInput、hasOutput、hasPrecondition 和 hasResult 来描述输入、输出、前置条件和效果。公告和请求的功能相似度的定义如公式 3-2 所示：

$$sim_{functional}(REQ, ADV) = w_1 sim_{input}(REQ, ADV) + w_2 sim_{output}(REQ, ADV) + w_3 sim_{precondition}(REQ, ADV) + w_4 sim_{effect}(REQ, ADV) \quad \text{（公式 3-2）}$$
$$\sum w_1 + w_2 + w_3 + w_4 = 1$$

3.3.1 OWL 对象的相似度

两个 OWL 对象 a 与 b 之间的相似度由它们的公共信息量来描述，如公式 3-3 所示：

$$sim(a,b) = \frac{f_{common}(a,b)}{f_{desc}(a,b)} \quad \text{（公式 3-3）}$$

$f_{common}(a,b)$ 表示 a 与 b 之间的公共信息，$f_{desc}(a,b)$ 表示 a 和 b 的并集信息。

由公式可知函数 sim 具有如下属性

属性 1：$0 \le sim \le 1$

属性 2：$\forall a : sim(a,a) = 1$

属性 3：$\forall a,b : sim(a,b) = sim(b,a)$

属性 1 说明了函数 sim 的取值范围，对于相等的两个对象 a 和 b，$f_{common}(a,b) = f_{desc}(a,b)$，它们的相似度值为 1，对于没有公共值的两个对象，$f_{common}(a,b) = 0$，它们的相似度为 0；

属性 2 说明了相似度函数具有自反性，任何对象都与它自己完全相同；

属性 3 说明了相似度函数的对称性，对于任意两个对象 a 和 b，a 到 b 的相似

度与 b 到 a 的相似度相等，同样可知 $f_{common}(a,b)=f_{common}(b,a)$

任意 OWL 对象的 f_{common} 和 f_{desc} 值都是从它的描述中派生的，接下来，本文定义了 OWL 对象描述集合的概念，这是 OWL 对象描述并集和公共描述集的基础。

3.3.2 描述集合

公式 3-3 中的公共信息表示是基于描述集合的，OWL 对象 a 的描述集合 *desc(a)* 用公式 3-4 定义：

$$desc(a)=\{(a,p,o)\in O\} \quad \text{（公式 3-4）}$$

其中，*O* 表示 OWL 本体，*o* 是本体中的对象，*p* 是谓词，*(a,p,o)* 是一个 RDF 三元组。

公式 3-4 只是定义了直接描述对象的三元组，即为在 RDF 图表中，与对象一边相联的结点（边代表三元组中的谓词）。但是，1-edge-separation 不能用来表达 OWL 对象的所有描述。当描述一个约束类时，需要两条边到达约束语句（如图 3-1 所示）。为了进一步表示 OWL 对象的信息量，可以使用 n 阶描述集合。

如公式 3-5 所定义：

$$\begin{aligned} desc_n(a)&=\{(a,p,o)\in O\}\cup\bigcup_{x\in Obj(a)}\{desc_{n-1}(x)\} \\ Obj(a)&=\{o\mid(a,p,o)\in desc(a)\} \end{aligned} \quad \text{（公式 3-5）}$$

Obj(a) 包括了 *des(a)* 中的所有对象，因为高阶描述集和中的描述较多，所以比低阶描述集和所包含的信息要多，可以说，阶数代表着对象的边的数量。但是，高阶描述集和也并非都有用，这取决于上下文的联系，当距离对象阶数越多，描述所包含的信息值也就越少。

例 1：

对于图 3-1 中的类 A，$desc_1(A)$ 只包括(A,rdfs:subClassOf,:ex1),丢失了关于属性 P 的约束信息。如果增加描述的阶数，那么 $desc_2(A)$ 就可以包括(A,rdfs:subclassOf,:ex1),(:ex1,rdf:type,Class),(:ex1,onProperty,P), (:ex1,hasValue,X)。这样，就可以更加清晰的描述概念 A。

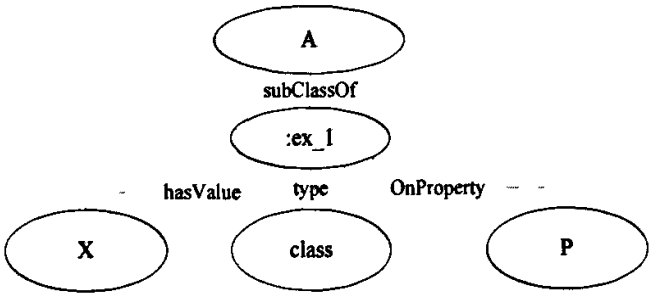

图 3-1：约束类的 RDF 图表描述

Fig3-1: RDF graph representation of a restriction class

3.3.3 并集描述和公共描述

a 与 b 的并集描述 $codesc_{m,n}(a,b)$ 和公共描述集 $com_{m,n}(a,b)$ 如公式 3-6 所示：

$$codesc_{m,n}(a,b) = desc_m(a) \cup desc_n(b)$$
$$com_{m,n}(a,b) = (desc_m(a) \cap desc_n(b)) \cup \{(a,p,o),(b,p,o) \mid (a,p,o) \in desc_m(a),(b,p,o) \in desc_n(b)\} \quad \text{（公式 3-6）}$$

两个 owl 对象 a 与 b 的公共信息量集合分为两个部分，一部分是 a 与 b 的多阶描述集合的交集，另外一部分是由 a 与 b 的多阶描述集合中谓词和对象相同的三元组组成的集合

例 2：

图 3-2 中的对象 A 和对象 B，公共描述集合中只有一个元素，((A,rdf:type,Class),(B,rdf:type,Class)),并集描述集合是 desc(a)加上附加元组(B,rdfs:subClassOf,A)。

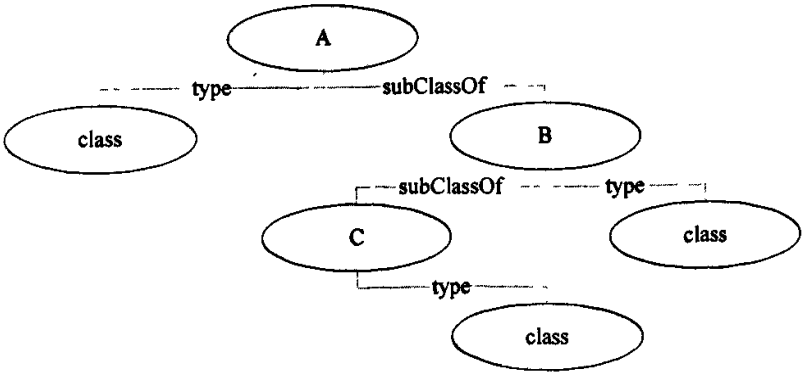

图 3-2：OWL 类

Fig3-2：An OWL SubClassOf hierarchy

3.3.4 基于推理的信息值

本体中一个元组所蕴涵的信息量可以由该元组的可推理性来度量，可推理性可以由该元组通过一些推理规则所生成的新的RDF元组数目来量化，如公式3-7所示：

$$R_r(t,O)=\begin{cases}\{b_1,b_2,...,b_i\}:\exists\{a_1,a_2,...,a_i\in O\}\\ \varnothing:\text{其它}\end{cases} \quad \text{（公式3-7）}$$

其中，r为推理规则，t为元组，O为本体，$\{b_1,b_2,...,b_i\}$为应用推理规则生成的新的元组集合。

在公式3-7的基础上可以定义元组针对某个OWL构造的可推理性：

$$inf_p((s,p,o),O)=\bigcup_{r\in rs}R_r((s,p,o),O) \quad \text{（公式3-8）}$$

其中，p为OWL构造，rs为针对构造p的推理集[28]。

元组t的n阶可推理性如公式3-9所定义：

$$inf_p^n(t,O)=\bigcup_{e\in inf_p^{n-1}(t,O)}inf_p(e,O) \quad \text{（公式3-9）}$$

元组t所蕴涵的信息量成为可推理的信息值（Inference-based Information Value，简称IBIV），它的值就是集合$inf_p^n(t,O)$元素的个数，即：

$$IBIV(t,O)=|inf_{rs}^n(t,O)| \text{ 或简写为 } IBIV(t)=|inf_p^n(t,O)| \quad \text{（公式3-10）}$$

t是本体O的三元组，$inf_p^n(t)$是对(s,p,o)n阶推理后得到的三元组集。

例3：

利用图3-2给出2阶推理集合：$inf_{rs}(B,rdfs:subClassOf,A)$包括type A的三元组，$inf_{rs}^2(B,rdfs:subClassOf,A)$包括所有以前的元素加上type C的三元组。通过综合(B,rdfs:subClassOf,A)和(C,rdfs:subClassOf,B)可以得出实例C属于type A。

例4：

本例示范推理集合的结构和通过本体Γ计算IBIV，如图3-3所示本体Γ具有简单的三层结构，包括类A，B和C，同样包括type B的实例b和type C的实例c。通过使用subclass推理集合rs，可以为RDF三元组(B,rdfs:subClassOf,A)构造基本推理集合

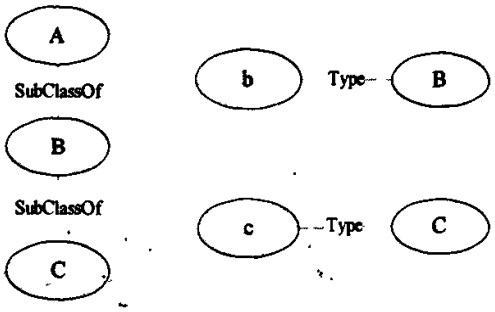

图 3-3：OWL 本体Γ

Fig3-3：An Example Of OWL Ontology，Γ

(1) 给出 rdfs:subClassOf 的 IBIV 计算

rdfs:subClassOf 结构的推理集合 rs 包括：

r1: $(X_1,\text{subClassOf},X_2),(X_2,\text{subClassOf},X_3)\rightarrow(X_1,\text{subClassOf},X_3)$

r2: $(X_1,\text{subClassOf},X_2),(x_1,\text{type},X_1)\rightarrow(x_1,\text{type},X_2)$

rs 的一阶推理集合是：

$\inf_{rs}((s,rdfs:subClassOf,o),O)=R_{r1}(\text{s,subClassOf,o})\cup R_{r2}(\text{s,subClassOf,O})$

O 是 OWL 本体，R_{r1} 包括从 subClassOf 结构的传递性派生出的三元组，R_{r2} 包括描述实例的三元组。

IBIV 等于推理集合中的元素的数量：

$\text{IBIV(s,rdfs:subClassOf,o)}=|\inf_{rs}^{n}(\text{s,rdfs:subClassOf,o})|$

(2) 为本体Γ的 RDF 三元组(B,rdfs:subClassOf,A)构造基本的推理集合如下：

$\inf_{rs}((B,rdfs:subClassOf,A),\Gamma)$

$=R_{r1}(\text{B,subClassOf,o})\cup R_{r2}(\text{B,subClassOf,A})$

$=\{(\text{C,rdfs:subClassOf,A})\}\cup\{(\text{b, rdf:type, A})\}$

使用 r_1 可以推理出(C,rdfs:subClassOf,A)，使用 r_2 可以派生出(b, rdf:type, A)。

(3) 假设 t=(A, rdfs:subClassOf, B)，则可得二阶推理集合：

$\inf_{rs}^{2}(\text{t})=\inf_{rs}(\text{t})\cup\inf_{rs}(\text{e})$

$=\inf_{rs}(\text{t})\cup\inf_{rs}(\text{C,rdfs:subClassOf,A}))\cup\inf_{rs}((\text{b,rdf:type,A}))$

$=\inf_{rs}(\text{t})\cup\{(\text{c, rdf:type, A})\}\cup\Phi$

={(C, rdfs:subClassOf, A), (b, rdf:type, A), (c, rdf:type, A)}

(4) IBIV(A, rdfs:subClassOf, B)

$=|\inf_{rs}^{2}(A, rdfs:subClassOf, B)|$.

=3

由公式 3-10 可以量化公式 3-3 中的 $f_{common}(a,b)$ 和 $f_{desc}(a,b)$：

$$f_{desc}(a,b)=\sum_{s\in codesc_{m,n}(a,b)} IBIV(s) \quad 及 \quad f_{common}(a,b)=\sum_{(x,y)\in com(a,b)}(IBIV(x)+IBIV(y))$$

把上述两式代入公式 3-3 可得：

$$sim(a,b)=\frac{\sum_{(x,y)\in com(a,b)}(IBIV(x)+IBIV(y))}{\sum_{s\in codesc_{m,n}(a,b)} IBIV(s)} \qquad (公式\ 3\text{-}11)$$

使用公式 3-2 和 3-11 就可以计算待选的服务和请求的功能相似度。

3.4 请求服务和发布服务的信誉度相似度

2.2.1 定义了 5 个影响 Web 服务信誉度的因素，但影响信誉度相似度的因素只有三个：响应时间、可用性和价格。信誉度相似度的定义如公式 3-12 所示：

$$sim_{reputation}=w_1 sim_T+w_2 sim_A+w_3 sim_M+w_4 sim_{SA}+w_5 sim_P$$
$$\sum w_1+w_2+w_3+w_4+w_5=1 \qquad (公式\ 3\text{-}12)$$

sim_T 表示响应时间的相似度，sim_A 表示可用性的相似度，sim_M 表示可维护性的相似度，sim_{SA} 表示满意度的相似度，sim_p 表示价格的相似度。

3.4.1 响应时间的相似度

历史记录中的响应时间的记录为 $t_1,t_2,...,t_n$，则平均响应时间 t 可以通过计算平均值，如公式 3-13 所示。

$$t=\frac{1}{n}\sum_{i=1}^{n} t_i \qquad (公式\ 3\text{-}13)$$

但是这些测量值可能存在误差，尤其是可能存在一些误差比较大的数据，所以必须剔除这些存在较大误差的数据。通过计算这些统计值的标准误差 σ，然后剔除误差大于 3σ 的数据，因为根据统计学的原理，测试误差落在-3σ 和 3σ 的概率是 99.7%。

$$\dot{O}=\left[\frac{1}{n-1}\sum_{i=1}^{n}(t_i-t)^2\right]^{\frac{1}{2}} \quad \text{（公式 3-14）}$$

所有应该剔除的数据集合如公式 3-15 所示：

$$\{a \mid -3\dot{O} < a-t < 3\dot{O}\} \quad \text{（公式 3-15）}$$

剔除误差大的数据后，重新使用公式 3-13 来计算平均值。

假设请求中设定的响应时间阈值为 $t_{request}$，则响应时间的相似度：

$$sim_T=\begin{cases}1: & t \le t_{request}\\ \dfrac{t_{request}}{t}: & t > t_{request}\end{cases} \quad \text{（公式 3-16）}$$

3.4.2 可用性的相似度

历史记录中的可用性的记录为 $a_1,a_2,...,a_n$，则可用性的平均值 a 可以通过计算平均值，如公式 3-17 所示：

$$a = \frac{1}{n}\sum_{i=1}^{n} a_i \quad \text{（公式 3-17）}$$

可以同样采用公式 3-14 和 3-15 所示的方法来消除较大的误差。

假设请求中设定的可用性的阈值为 $a_{request}$，则可用性的相似度：

$$sim_A=\begin{cases}1: & a \ge a_{request}\\ \dfrac{a}{a_{request}}: & a\ \pi\ a_{request}\end{cases} \quad \text{（公式 3-18）}$$

3.4.3 可维护性的相似度

历史记录中的可维护性的记录为 $m_1,m_2,...,m_n$，则可维护性的平均值 m 可以通过计算平均值，如公式 3-19 所示：

$$m = \frac{1}{n}\sum_{i=1}^{n} m_i \quad \text{（公式 3-19）}$$

可以同样采用公式 3-14 和 3-15 所示的方法来消除较大的误差。

假设请求中设定的可维护性为 $M_{request}$，实际的可维护性是 m，则可维护性的相似度：

$$sim_m = \begin{cases} 1: & m \le m_{request} \\ \dfrac{m_{request}}{m}: & m \phi\ m_{request} \end{cases} \qquad \text{（公式 3-20）}$$

3.4.4 满意度的相似度

历史记录中满意度的记录为 $sa_1, sa_2, ..., sa_n$，则可维护性的平均值 sa 可以通过计算平均值，如公式 3-21 所示：

$$m = \frac{1}{n}\sum_{i=1}^{n} m_i \qquad \text{（公式 3-21）}$$

可以同样采用公式 3-14 和 3-15 所示的方法来消除较大的误差。

假设请求中设定的满意度为 $SA_{request}$，实际的满意度是 sa，则满意度的相似度：

$$sim_{sa} = \begin{cases} 1: & sa \le sa_{request} \\ \dfrac{sa_{request}}{sa}: & sa \phi\ sa_{request} \end{cases} \qquad \text{（公式 3-22）}$$

3.4.5 价格的相似度

假设请求中设定的价格为 $p_{request}$，实际的服务价格 p，则价格的相似度：

$$sim_P = \begin{cases} 1: & p \le p_{request} \\ \dfrac{p_{request}}{p}: & p \phi\ p_{request} \end{cases} \qquad \text{（公式 3-23）}$$

把公式 3-16、3-18、3-20 和 3-22 代入公式 3-12 就可以得到信誉度相似度的完整的计算公式。

3.5 服务匹配算法

匹配算法的基本思想是：遍历所有的待选服务，过滤掉信誉度为 0 的服务，计算信誉度不为 0 的待选服务与服务请求的相似度，如果相似度满足给定的阈值，则记录到匹配结果链表中；根据相似度的大小对匹配结果进行排序并返回结果。匹配的执行过程如图 3-3 所示。

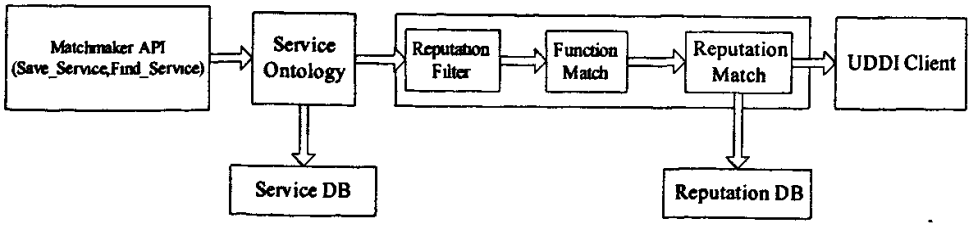

图 3-3：匹配执行过程

具体如算法 3 所示。

算法 3：服务匹配算法

输出：满足匹配要求的服务的描述列表

```
1   MatchedList match(ServiceRequest request){
2     MatchedResultList resultList = new MatchedResultList();
3     for each adv in advertisement database{   //遍历所有的待选服务
4       if (adv.getReputation()==0)             //过滤信誉度为 0 的服务
5         continue;
      //用公式 3-1 计算请求和待选服务的相似度
6     float similarity = sim(request, adv);
7       if (similarity >= request.getExpetedSim())
      //满足匹配请求则添加到匹配结果链表
8         resultList.add(adv);
9     }
10    resultList.sort();                        //按匹配度大小降序排序
11    return resultList;
  }
```

3.6 小结

本章重点介绍了基于相似度的服务匹配算法。相似度可以用来灵活、精确地度量服务请求和待选的服务的相似程度。本章提出的服务匹配算法的最大特点是引入了信誉度的匹配，在服务功能（IOPEs）匹配的基础上加上信誉度的匹配，可以避免为用户提供功能匹配但信誉度差（如提供虚假描述）的 Web 服务。

第4章 语义Web服务发现的应用

当前的旅游服务之间缺乏互相协调、协同工作的能力，因此按照流程将旅游服务组织起来，提供给用户真正满意的服务非常重要。若干跟旅游有关的企业迫切需要一个平台把他们提供的服务管理起来，根据用户的需求，结合平台掌握的信息，按照流程将服务组织起来，以更好的服务提供给用户，以最大的发挥他们的服务的功能。

本章中，建立一个旅游服务集成平台——IPVita（the Intelligent Platform of Virtual Travel Agency），阐述了平台的总体结构和各个模块的实现，并着重介绍了语义Web服务在IPVita中的应用及实现。

4.1 IPVita的总体结构

通过建立一个旅游服务集成平台——IPVita（the Intelligent Platform of Virtual Travel Agency），将各个企业提供的旅游服务根据类别按照流程组织起来，为游客提供满意的优质的旅游服务。该平台在语义的支持下，首先将用户的需求形式化，然后根据形式化的需求自动生成组合流程，之后自动为组合流程匹配满意的服务从而形成旅游流程，最后实施旅游流程，为用户提供满意的旅游服务。整个过程最大的体现了平台的智能化。如图4-1所示：

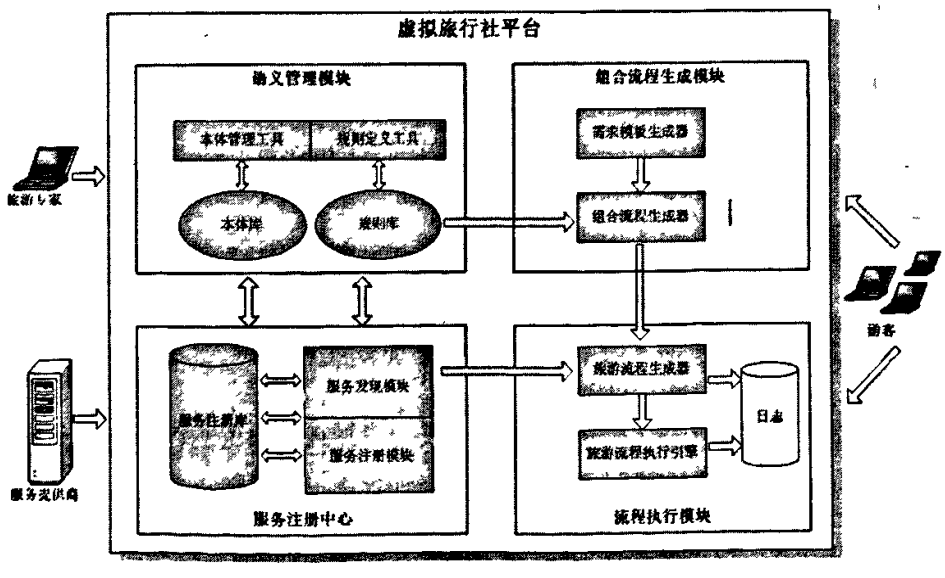

图4-1：IPVita的总体结构

Fig4-1: The Structure of IPVita

4.2 IPVita 的组成模块

由图 4-1 可知整个平台分为四个模块，语义管理模块、组合流程生成模块、服务注册中心和流程执行模块。下面分别介绍平台的各个模块

4.2.1 语义管理模块

语义管理模块的作用是通过简洁的界面和方便的操作，对平台所需要的语义基础——本体和规则进行管理。平台其他三个模块都需要得到语义的支持，因此平台必须要有一个好的语义管理工具。该模块包括三个组成部分：本体管理工具和规则定义工具。

4.2.1.1 本体管理工具

本体管理工具的作用是提供方便的、人性化的界面，供旅游专家定义和修改本体。旅游本体是在旅游专家和软件专家的共同努力下建立起来的，它描述了旅游领域中的各个概念，以及概念之间的关系、概念之间存在的规则等。设计好的本体存放在本体库中。本体管理工具还可以根据对平台的监控结果，提示专家调整本体。

4.2.1.2 规则定义工具

规则是平台其他部分的推理基础。规则定义工具的作用就是为旅游专家提供方便的工具，定义和修改规则。规则定义工具还应该提供检查规则合法性和正确性的功能。

本体管理工具和规则定义工具借鉴了本体定义工具 protégé，使用 protégé 定义基础的旅游本体和规则，然后在此基础上使用平台的工具进行修改。

4.2.2 组合流程生成模块

组合流程生成模块的功能是将用户的需求形式化，然后根据形式化的需求自动生成组合流程。组合流程生成模块由两部分组成，需求模板生成器和组合流程生成器。

4.2.2.1 需求模板生成器

需求模板生成器的作用是根据旅游本体中关于服务的描述，生成需求模板，供用户填写。需求模板中既包括确定的要求，也包括扩展的要求。例如对旅游的要求既包括目的地、可以承受的花销、旅游时间的长短等确定要求，也包括温暖湿润、刺激度高、大快朵颐等附加的要求。填写完毕后的需求描述模板，最终成

为为一棵需求描述，树中的每个节点是一个描述类（Describe Class），描述了对旅游中某个概念的要求。概念可能是一类服务，也可能是服务的属性，甚至可能是整个旅游。

4.2.2.2 组合流程生成器

组合流程生成器的作用是根据需求描述树，结合事先定义好的组合操作[13]以及本体中定义的规则，自动生成一个虚拟的旅游流程。

组合流程生成器中的组合流程自动生成算法——AGCF4WS (an Auto Generation Algorithm for Composition Flow of Web Services)[13]是一个智能规划算法。该算法以形式化的需求描述为基础，在领域本体内定义的规则的支持下，生成组合流程。算法最终会生成一个虚拟旅游流程——组合流程，其指明了流程的控制流和数据流、流程的节点等，但是其中的每个节点并没有绑定到具体的服务，而只是对这个服务必须满足的条件的描述，所以说这是一个“虚拟”的旅游流程。组合流程生成器是 IPVita 的核心问题之一。

4.2.3 流程执行模块

流程执行模块的功能是为虚拟旅游流程绑定服务，变为真实的旅游流程，并交由工作流引擎执行。流程执行引擎由两部分组成：旅游流程生成器和旅游流程执行引擎。

4.2.3.1 旅游流程生成器

旅游流程生成器的功能是为虚拟旅游流程绑定服务，变为真实的旅游流程。旅游流程生成器从虚拟旅游流程的节点获取到对具体服务的需求信息，然后调用服务发现模块，在注册中心查找到跟需求信息匹配的服务，作为候选服务，最后根据候选服务的评估情况，选出得分最高的几个候选服务，供用户选择。

4.2.3.2 旅游流程执行引擎

旅游流程执行引擎负责执行匹配了具体的服务之后的旅游流程，其作用跟传统的工作流引擎类似，但是因为流程相对比较简单，出现异常时相应的处理也比较简单，所以整个执行机功能相对传统工作流引擎更为简单。

4.2.4 服务注册中心

服务注册中心提供了四个操作：发布操作、删除操作、查询操作和反馈操作。发布操作用来发布和更新语义 Web 服务；删除操作用来删除语义 Web 服务；查询

操作用来查询 Web 服务；反馈操作返回用户对 Web 服务的反馈信息（这些反馈信息保存到信誉度管理器的历史数据库中），当用户调用 Web 服务时，会监视 Web 服务的一些服务参数，调用结束后返回反馈信息；。

该平台的注册中心既不同于传统的 UDDI[29]，又不同于语义的 DAML_S[9]。服务注册中心由两部分组成：服务注册模块、服务发现模块。

4.2.4.1 服务注册模块

服务注册模块的作用是根据旅游本体，自动生成服务注册模板，由服务提供商填写，用于注册他们的服务。注册模板应该包括两部分，服务的描述模板和服务的实现模板。描述模板包括服务详细的描述，如 2.3 所述。实现模板描述平台如何调用服务，其作用类似于 UDDI 中的 WSDL[10]。服务注册信息存放在服务注册库中。服务注册模块还提供了服务注册信息的修改、删除等功能。在 IPVita 中，采用 UDDI 公共注册仓实现 Web 服务管理，通过在此基础上增加语义层的方式，实现 Web 服务的语义标注和匹配功能。这种通过封装 UDDI 实现 Web 服务高级功能的方式已经被证明是一种可行的方法。如图 4-2 所示：

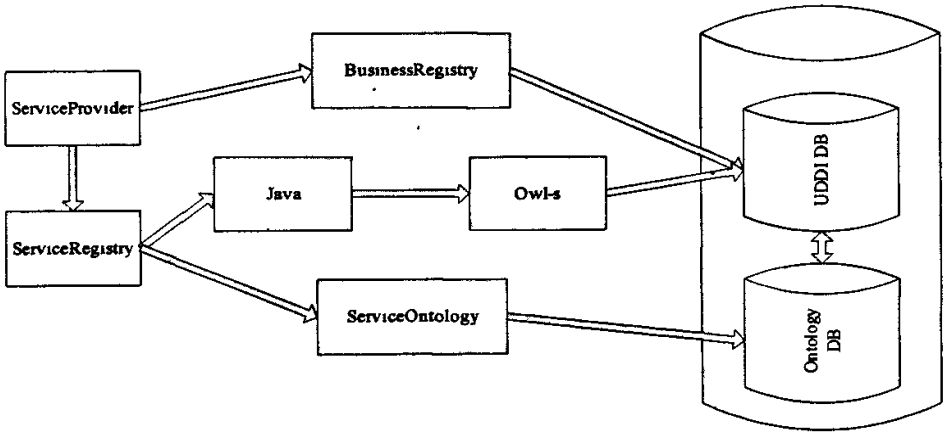

图 4-2：服务的注册过程

Fig4-2: Service Enroll Process

4.2.4.2 服务发现模块

服务发现模块的功能是为虚拟流程中的任务节点匹配真正的服务，其中虚拟流程是由组合流程生成器生成的。虚拟流程中每个任务节点只对该节点需要的服务进行了描述，服务发现模块的任务根据这些描述，在服务注册库中查找满足条件的服务，我们在第二章已经给出了 Web 服务的语义描述模型。

服务发现模块将需求描述树中的服务需求和服务注册库中的服务描述进行匹配，如果匹配成功，那么该服务即为候选服务。如何进行匹配是服务发现模块的难点，我们在第三章已经给出。服务匹配还要在旅游本体的支持下，进行必要的推理才能成功。例如虚拟流程中需要一个可以订到去北京的机票的订票服务，但是服务注册中心中注册的某个订票服务可能只表明它可以预定“所有国内城市”的机票，服务发现模块必须清楚“北京是国内城市”这一事实，才能不错过这个服务。针对目前旅游行业提供虚假服务的现象，本文提出了信誉度管理器对 Web 服务的信誉度进行管理，可以防止服务提供者提供虚假的描述，为服务请求者提供更好的服务，Web 服务信誉度的评估也可以由专门的第三方的信誉度权威评估机构（当然，这只是个理想模型，目前并不存在第三方的信誉度权威管理机构）进行，详细介绍见 4.3.1。

我们将在 4.3 节中详细介绍基于语义的服务发现模块的实现

4.2.4.3 服务注册中心的结构

IPVita 的服务注册中心是一个基于知识的信息发布，存储和检索过程，使用隐藏在 Web 搜索后的用户兴趣与语义知识来提高搜索的精度，其实质就是以有序的知识库对无序的 Internet。它能够对用户的检索内容定位的更快、更精确、更全面。本节给出完整的服务注册和服务发现的结构图，如图 4-3 所示：

4.3 基于语义的服务发现模块

将 Web 服务的 OWL-S 语义描述和 ROWS 语义描述与 UDDI 注册中心结合起来，可以进行语义 Web 服务的集中注册和发布。本文在服务发现模块中增加了信誉度管理器，管理 Web 服务的信誉度。IPVita 中的服务发现模块如图 4-4 所示。

通信模块对外提供发布、查询、删除语义 Web 服务和反馈服务信誉度的 Java 接口，用户可以在他们的应用中通过使用通信模块的接口调用匹配器以执行相应的操作

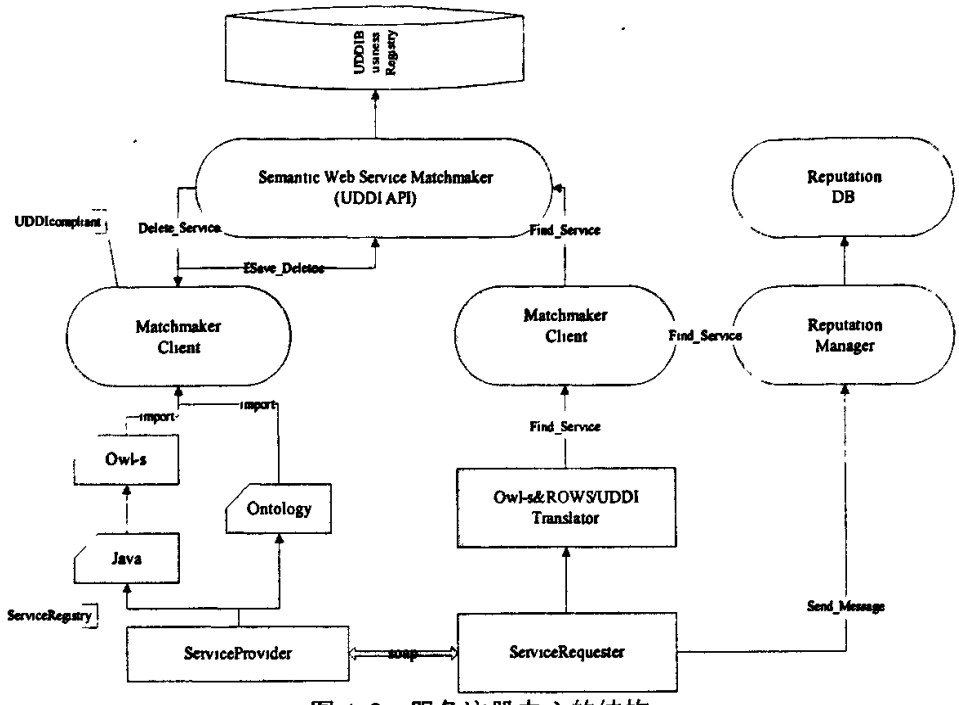

图 4-3：服务注册中心的结构

Fig4-3：Architecture of Service Enrollment Center

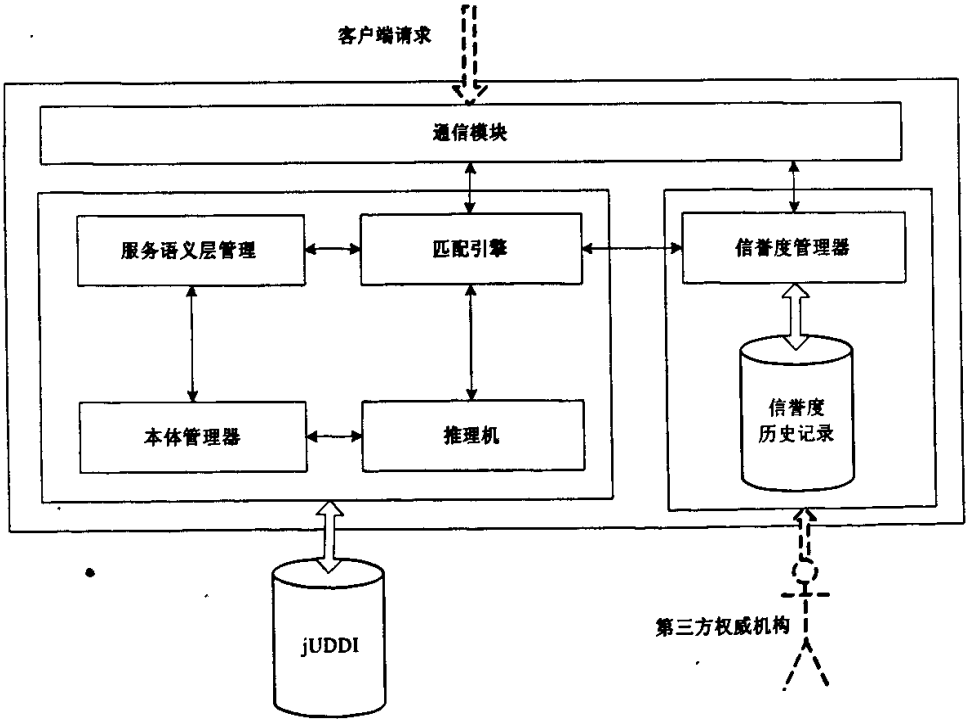

图 4-4：基于语义的服务发现模块

Fig 4-4：Semantic-based Service Find Model

4.3.1 信誉度管理器

信誉度管理器是服务发现模块的一个重要组成部分，管理 Web 服务的信誉度。信誉度的管理大致可以分为两种：收集用户反馈和自动化的测试。收集用户反馈是把用户的反馈信息保存到持久存储介质中，然后通过统计学的方法来计算服务的信誉度；自动化的测试方法则是通过自动化的测试设施来周期性地检查 Web 服务的各个信誉度因素。前者可能会存在一定的误差，但采用合理的统计学方法可以减少误差；后者是比较理想一种方法，是实现起来比较有难度，目前的技术还不能实现。所以，本文采用的是收集用户反馈的方法。

如图 4-4 所示，信誉度管理器主要和通信模块以及匹配引擎交互，从中可以看出信誉度管理器的几个主要功能。

1. 收集客户反馈：通信模块拦截到客户的反馈操作，然后调用信誉度管理器的接口来保存客户反馈。
2. 为 Web 服务创建一个唯一的 ReputationID：当新发布一个 Web 服务时，首先调用信誉度管理器的接口来创建一个唯一的 ReputationID，然后把 ReputationID 使用 TModel 技术封装到 Web 服务的 UDDI 记录中。
3. 判断某个 Web 服务的信誉度是否为 0：匹配引擎有一个过滤器，用来过滤掉信誉度为 0 的 Web 服务，匹配引擎会调用信誉度管理器的接口，使用算法 2-1 来判断指定的 Web 服务的信誉度是否为 0。
4. 比较两个 Web 服务的信誉度大小。使用算法 2-2 可以比较两个 Web 服务的大小。

4.3.2 服务发现算法

将 Web 服务的语义描述与 UDDI 注册中心结合起来，可以进行语义 Web 服务的集中注册和发布。现在我们解决的主要问题是如何根据用户输入的描述信息自动定位服务，而不仅仅是依赖关键词进行搜索，而是找到最能满足服务请求的服务匹配，并对所发现的服务进行评估排序。为了提高查找速度，指定了相似度阈值。只有不低于该相似度阈值的服务才会作为查询结果显示，其余的服务在经过匹配过滤器时将被淘汰。服务发现算法(FindingAlgorithom) 具体描述如下：

步骤 1 请求者（用户或程序）按照 2.3 的 Web 服务语义描述模型对 Web 服务进行语义描述并且提交给 Web 服务发现模块，通信模块通过本体管理(OntoBroker)获得服务本体对请求进行分析，并翻译成针对目标领域的形式化表示，查找语义注册中心，获得满足服务操作规范的服务集；

步骤 2 针对请求的目标和可能的约束，结合相关的领域本体进行推理。在服

务集中进行输入、输出概念的匹配，获得满足服务一般属性的服务集，即满足服务的功能需求。包括查询条件标准化和泛化的过程；

步骤3 结合基于语义相似度的Web服务匹配算法计算给出每个备选服务的匹配度，获得有效的、按服务性能排序的服务集。该服务集是为服务消费者按需推荐的服务，最终获得最佳的服务；

步骤4 根据匹配过程返回的服务的标识从UDDI中检索，以获取服务的WSDL文件，以便对目标服务进行调用。

4.4 本章小结

本章提出了旅游服务集成平台——IPVita (the Intelligent Platform of Virtual Travel Agency)，阐述了平台的总体结构和构成平台的四个模块，并着重介绍了语义Web服务发现在IPVita中的应用及实现，通过这种方式，使得IPVita能根据用户需求自动地匹配、发现和调用Web服务，动态生成和执行业务流程。

第5章 总结与展望

Web服务的自动化发现是语义Web服务的基本研究课题，也是实现自动化的Web服务调用和组合等技术的基础。本文分析了国内外针对该课题的研究成果，在此基础上提出了一个新的语义Web服务发现模型，这个模型具有如下几个特点：

（一）服务描述采用OWL-S和ROWS相结合的方式

OWL-S是一个比较成熟的语义Web服务描述语言，使用它可以在当前的Web服务基础设施之上添加一层语义设施，实现Web服务的自动化发现、调用、组合和监视。但是，OWL-S存在一个严重的不足，它不能很好地描述Web服务的信誉度。这里的信誉度是一个广义的概念，包括服务的可信任性、QoS等属性。本文分析了影响Web服务信誉度的一些因素，并在此基础上定义了信誉度本体ROWS，弥补了OWL-S的不足。使用OWL-S和ROWS相结合的方法描述Web服务，可以实现服务描述的完整性以及灵活性。

（二）服务匹配算法基于语义相似度的计算

当一个待选的服务和服务请求“足够相似”的时候，我们认为该待选的服务匹配这个请求。本文使用相似度的概念来刻划“足够相似”，这样可以满足服务匹配灵活性的要求。待选的服务和服务请求的相似度由它们的OWL-S Profile的功能描述（IOPEs）和信誉度本体联合决定。IOPEs的相似度由OWL对象的公共信息量来度量，信誉度本体的相似度由各个属性的相似度联合决定。

（三）服务发现模块中引入信誉度管理器部件

服务发现模块是旅游服务集成平台——IPVita（the Intelligent Platform of Virtual Travel Agency）的重要模块，本文中，通过为UDDI添加了一个语义层来实现基于语义推理的服务匹配，并对外提供服务的发布、查询、删除和反馈接口。信誉度管理器是服务发现模块的一个重要部件，它实现Web服务的信誉度管理。服务匹配时需要调用它来计算信誉度相似度，用户的反馈信息也是通过信誉度管理器保存在持久存储介质中。另外，在信誉度管理器的概念模型中，有一个第三方的权威机构来管理维护Web服务的信誉度。

由于研究时间有限，也有一些技术点需要进行进一步更深入的研究。主要有以下几点：

1. 信誉度本体的进一步细化。本文只定义了几个基本的影响Web服务的信誉度的

因素，还有许多其它的因素需要研究，尤其是 QoS 属性和面向特定领域的属性。

2. 信誉度的管理。本文采用收集用户反馈的方式来记录影响 Web 服务信誉度的信息，然后通过统计学的方式来计算服务的相似度。如果采用自动化的测试框架，可以周期性地检测 Web 服务的信誉度，则可以大大提供相似度的精确程度。

3. 本体的存储与管理。本文没有把本体的管理列为研究内容，但是它可以影响服务匹配的效率，因此后续的研究工作应该把本体的存储与管理作为一个主要的研究课题。

4. 匹配算法的扩展。第五章计算功能相似度的公式目前更适用于 OWL Lite，需要进一步扩展以支持 OWL DL。

参考文献

[1]. Tim Berners-Lee, James Hendler and Ora Lassila. "The Semantic Web," Scientific American, Vol. 284, No.5, May 2001, pp.34-43

[2]. Massimo Paolucci and Katia Sycara. "Autonomous Semantic Web Services," IEEE Internet Computing, Vol.7, October 2003, pp.34-41

[3]. D.Martin, M.Paolucci and S.McIlraith. "Bring Semantics to Web Services: The OWL-S Approach," In Proc of the International Workshop on Semantic Web Services and Web Process Composition (SWSWPC 2004), SanDiego, California, USA, July 6-9, 2004

[4]. Mark Burstein, Christoph Bussler, Michal Zaremba, et al. "A Semantic Web Services Architecture," IEEE Internet Computing, September/October 2005, pp.72-81

[5]. Massimo Paolucci, Takahiro Kawamura, Terry R. Payne, et al. "Semantic Matching of Web Services Capabilities," ISWC2002, 2002

[6]. Sheila A. McIlraith and David L. Martin. "Bringing Semantics to Web Services," IEEE Intelligent Systems, January/February 2003

[7]. Sivashanmugam K., Verma K. ,Sheth A., et al. "Adding Semantics to Web Services Standards,"The International Conference on Web Services, Las Vegas, USA, 2003.

[8]. Massimo Paolucci, Takahiro Kawamura, Terry R. Payne, et al. "Importing the Semantic Web in UDDI," In Proceedings of Web Services, E-business and Semantic Web Workshop, 2002

[9]. Anupriya et al. "DAML-S: Web Services Description for the Semantic Web," In Proceedings of the First International Semantic Web Conference (ISWC), 2002

[10]. Anupriya Ankolenkar et al. "DAML-S: A Semantic Markup Language for Web Services," In Proceedings of 1st Semantic Web Working Symposium (SWWS' 01), Stanford, USA, August 2001, pp.411-430

[11]. OWL-S Specification. http://www.daml.org/services/OWL-S/1.1/

[12]. WSMO Primer, http://www.wsmo.org/TR/d3/d3.1/v0.1/#S1

[13].D. Fensel and C. Bussler, "The Web Services Modeling Framework WSMF," http://www.swsi.org/resources/wsmf-paper.pdf

[14]. SWRL Specification. http://www.daml.org/2003/11/swrl/#contents

[15]. Kunal Verma, Kaarthik Sivashanmugam, Amit Sheth, et al. "METEOR-S WSDI: A Scalable Infrastructure of Registries for Semantic Publication and Discovery of Web

Services," Journal of Information Technology and Management, Special Issue on Universal Global Integration, Vol. 6, No. 1, 2005, pp. 17-39

[16]. Abhijit Patil, Swapna Oundhakar, Amit Sheth, et al. "METEOR-S Web service Annotation Framework," The Proceedings of the Thirteenth International World Wide Web Conference, May, 2004 (WWW2004), pp. 553-562

[17]. METEOR-S Project. http://lsdis.cs.uga.edu/projects/meteor-s

[18]. A. Sheth, K. Verma, J. Miller and P. Rajasekaran (University of Georgia), "Enhancing Web Service Descriptions using WSDL-S," Research-Industry Technology Exchange at EclipseCon 2005, Burlingame, CA, Feb. 28 - Mar. 3, 2005.

[19]. M. Schlosser, M. Sintek, S. Decker, et al. "A Scalable and Ontology-based P2P Infrastructure for Semantic Web Services," In Proceedings of the Second International Conference on Peer-to-Peer Computing, 2002, pp.104—111

[20]. Boualem Benatallah, Mohand-Said Hacid, Alain Leger, et al. "On Automating Web Service Discovery," The VLDB Journal, Vol.14, Issue 1 (March 2005), pp. 84 – 96

[21]. Mark Klein and Abraham Bernstern. "Toward High-Precision Service Retrieval," In Proceedings of the First International Semantic Web Conference on The Semantic Web, 2002.

[22]. Mark Klein and Abraham Bernstern. "Searching for Services on the Semantic Web Using Process Ontologies,"
www.ifi.unizh.ch/ddis/staff/goehring/btw/files/SWWSFinal.pdf

[23]. Matthias Klusch, Benedikt Fries, Mahboob Khalid, et al. "OWLS-MX: Hybrid OWL-S Service Matchmaking," http://www.dfki.de/Eklusch/owls-mx/owlsmx-aaai.pdf

[24]. Chen Zhou, Liang-Tien Chia and Bu-Sung Lee. "Semantics in Service Discovery and QoS Measurement," IT Professional, Vol. 7, No. 2. (2005), pp. 29-34

[25]. Chen Zhou, Liang-Tien Chia and Bu-Sung Lee. "Service Discovery and Measurement based on DAML-QoS Ontology," http://citeseer.ist.psu.edu/730652.html

[26]. Chen Zhou, Liang-Tien Chia and Bu-Sung Lee. "DAML-QoS Ontology for Web Services,"
http://www.ntu.edu.sg/home5/pg04878518/Articles/icws04_235_Chen_Z.pdf

[27]. Katia Sycara, Widoff S, Klusch M, et al. "LARKS: Dynamic Matchmaking among Hetero-geneous Software Agents in Cyberspace," Autonomous Agents and Multi-Agent Systems, Vol. 5 (2002), pp. 173-203.

[28]. Jeffrey Hau, William Lee and John Darlington. http://www.ai.sri.com/WSS2005/

final-versions/WSS2005-Hau-Final.pdf

[29].UDDI4J Project. http://sourceforge.net/projects/uddi4j/

[30].Doug Tidwell. "UDDI4J: Matchmaking for Web Services,"
http://www.ibm.com/developerworks/library/ws-uddi4j.html?dwzone=xml

[31]. OWL-S API. http://projects.semWebcentral.org/projects/OWL-S-api/

[32]. Web Service Architecture. http://www.w3.org/TR/ws-arch/#id2260892

[33]. jUDDI Project. http://www.juddi.org/

[34]. Apache Tomcat. http://tomcat.apache.org/

[35]. MySQL. http://www.mysql.com/

[36]. Apache Axis. http://ws.apache.org/axis/

[37].Jena 框架. http://jena.sourceforge.net/

[38].Racer 推理机. http://www.sts.tu-harburg.de/~r.f.moeller/racer/

[39].沈坚，隋鑫，罗引等，Semantic Web Service，2004 年 Web Service 课程讨论报告

[40].胡建强，邹鹏等，Web 服务描述语言 QWSDL 和服务匹配模型研究，计算机学报，2005 年第 28 卷第 4 期

[41].陈丁剑，吴健等，语义 Web 服务信誉度模型及其实现，计算机应用，2005 年第 25 卷第 8 期

[42].岳昆，王晓玲，周敖英，Web 服务核心支撑技术：研究综述，软件学报，2004 年第 15 卷第 3 期

[43].梁邦勇，徐剑军，李涓子等，面向 Web 服务的分布式本体系统，计算机工程与应用，2003 年 11 月：159~161

[44].任波，基于语义的 Web 服务发现研究，浙江工业大学硕士学位论文，2004 年 12 月

[45]. 隋琪，王海洋. IPVita:一个基于语义的智能虚拟旅行社平台. 计算机科学 2005, Vol.32, No.9A:176 – 179, 180

致谢

光阴荏苒，三年紧张而又充实的研究生生活即将结束，在硕士论文完成之际，我要向所有支持、关心和帮助过我的人们表示最诚挚的谢意！

感谢导师王海洋教授三年来的谆谆教诲。从论文的酝酿到写作，王老师都给予了精心指导和热情关怀，为我提供一切可能的机会进行学习和锻炼，而且在生活中也给予我无微不至的关心。王教授敏锐的科学洞察力、渊博的学识、坦荡的胸怀和平易近人的作风使我受益终身。

在多年学习和生活中感谢李庆忠教授、洪晓光教授、张世栋老师、董国庆老师、李晖老师、王新军老师、闫中敏老师在学习及科研实践中对我的精心指导，在此致以最诚挚的谢意！

在本文的写作过程中，隋琪博士、崔立真博士给予了很大的支持和帮助，同时我要感谢数据库教研室的各位同学，他们忘我工作、精诚团结的精神激励着我，最后，特别感谢所有曾给予我鼓励、支持和关心的亲人、朋友和同学。

发表的学术论文

1. 王艳，基于从 OWL-S 到 BPEL 映射的 UDDI，全国第二届 WEB 信息系统及其应用学术会议论文集，2005 年 9 月
2. 王艳，基于工作流视图的跨组织工作流模型，计算机工程与应用（2005 增刊）