

基于 OWL-S 的 Web 服务发现系统的研究和实现

赵 军

(北京航空航天大学 计算机学院, 北京 100083)

摘 要:文中针对目前基于 WSDL, UDDI 等技术的 Web 服务发现存在的不足, 提出了一种使用 OWL-S 进行 Web 服务描述的新的服务发现机制。介绍了 Web 服务的语义描述语言 OWL-S, 设计并实现了一个 Web 服务发现系统。同时, 详细介绍了存储 Web 服务 OWL-S 描述信息的数据库结构的设计, 以及服务请求时进行服务匹配使用的匹配算法。

关键词:OWL-S; 服务发现; 本体; 服务匹配

中图分类号:TP393

文献标识码:A

文章编号:1673-629X(2006)10-0163-04

Research and Implementation of Web Service Discovery Based on OWL-S

ZHAO Jun

(School of Computer, Beijing University of Aeronautics & Astronautics, Beijing 100083, China)

Abstract: Proposed a new discovery mechanism of Web service with the description using OWL-S in order to solve the problem in the current discovery of Web service using WSDL and UDDI. This work introduced a semantic description language called OWL-S, designed and implemented a system of Web service discovery. At the same time, introduced in detail the design of database schema to store OWL-S files and the matching algorithm used in service matching.

Key words: OWL-S; service discovery; ontology; service matching

0 引 言

Web 服务作为一种新的 Web 应用模式, 具有自包容、自描述等特征, 可以通过 Web 来发布、查询和调用。用户可以将 Web 上发布的 Web 服务集成到自己的应用程序中完成任务而无需再重复开发。所谓 Web 服务发现就是客户以某种方式在这些不同类型的 Web 服务中找到其想要的服务, 以执行 Web 服务请求。服务发现包含了两个过程: 一个是服务的发布; 一个是从服务的描述数据库中查询服务。服务提供者将服务的描述发布在服务注册中心; 服务请求者则通过提交查询来得到所需服务的信息。通过 UDDI, WSDL 和 SOAP 等协议, 可以在 Internet 上发现 Web 服务, 得到其服务描述并动态绑定服务, 得到服务所提供的功能。目前 Web 服务发现的主要缺点是 WSDL 仅在语法层次上对服务进行描述, 缺乏针对应用的语义描述, 使得在服务匹配时只能采用简单的关键字搜索方法, 不能满足服务发现的需要。Tim Berners-Lee 在 2001 年的 Scientific American 中首次提出了语义 Web^[1]的概念。语义 Web 是国际互联网组织 W3C 制定的关于未来 Web 的一个蓝图, 主旨是在 XML 的基础上进一步提供

语义级的互操作。语义 Web 的应用遍及基于语义 Web 的互联网信息检索、语义网格技术及 Web 服务发现等。由于语义的引入, 作为集成了语义 Web 技术和 Web 服务技术而产生的语义 Web 服务使人们有望构造新的 Web 服务发现机制。

1 相关工作

语义 Web 的研究普遍采用本体论, 语义 Web 中的本体表示的是人们对特定领域中的概念统一的、本质的认识。对于网络上的应用, 重要的是需要定义一种具有统一语法的语言, 使得本体能够遵循统一的语法格式进行信息交换。OWL^[2]是 W3C 推荐的描述语言的标准, 它可以定义类、子类、等价类, 并且可以定义属性和子属性, 以及它们的约束, 如: 定义域(Domain)和值域(Range)等。

OWL-S^[3]是基于 OWL 语言的 Web 服务本体, 是 W3C 推荐的标准。OWL-S 定义了一组核心的语言构件, 用于对 Web 服务进行逻辑化描述, 所生成的描述文件支持机器理解, 从而支持代理程序基于逻辑语义实现对 Web 服务的自动发现、调用、组合与监控。OWL-S 由 3 部分组成, 如图 1 所示。

(1) Web 服务做什么, 例如服务实体、服务可以实现的功能, 以及服务的性能参数等。这方面的描述通过“ServiceProfile”来实现。基于这些描述, 服务请求者(人或

收稿日期: 2006-01-20

作者简介: 赵 军(1978-), 女, 河南驻马店人, 硕士研究生, 研究方向为语义 Web 服务; 导师: 马殿富, 教授, 研究方向为 Web 服务。

代理程序)可以发现满足特定功能需要的 Web 服务,可以确定需要满足哪些条件才能调用该服务。同时服务请求者也可以遵循“ServiceProfile”来描述自己的服务请求。

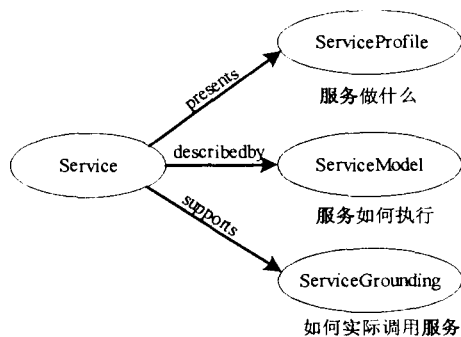


图 1 OWL-S 的顶级结构

(2) Web 服务如何执行,包括服务执行的先后顺序、过程流程等,这方面的描述通过“ServiceModel”来实现,服务请求者利用“ServiceModel”可组合多个服务以完成复杂任务,同时在服务执行过程中,可以利用“ServiceModel”来协调参与各方的动作。

(3) 如何实际调用 Web 服务,描述具体的绑定信息,例如服务地址、通信协议及消息格式等。这方面的描述通过“ServiceGrounding”来实现。

因此,文中将 ServiceProfile 作为广告发布在服务注册中心中,并利用 ServiceProfile 的信息进行服务匹配。同时,服务请求也可将 ServiceProfile 作为表达服务查询条件的语言,从而使得服务匹配能够更加方便。

目前已经有一些研究机构推出了一些语义 Web 服务的模型或者调用流程。其中由卡耐基-梅隆大学 Softagent 实验室开发的语义 Web 服务系统可以说是对实现语义 Web 服务的基本模型的一个设想。这个模型建立在 OWL-S 语言之上,也就是说提供 Web 服务的系统都采用 OWL-S 来描述自身的 Web 服务,各个系统之间也通过 OWL-S 来进行信息交换、服务合成。这个模型把 OWL-S 和 UDDI 结合,即采用文献[4]中描述的 OWL-S/UDDI 映射机制把 OWL-S 描述的信息转换成 UDDI 的数据结构,作为服务广告在 UDDI 中发布以及服务请求者在 UDDI 中查找服务,并设计了在 UDDI 中服务发现用到的查询匹配算法。但是,通过对此系统的深入研究发现,它为了利用已有的 UDDI 技术,把 Web 服务的 OWL-S 信息按 UDDI 的数据结构进行存储,这样做存在不妥之处。因此,文中对此进行了改进,专门设计了存储 OWL-S 本体信息的数据库结构,在发布服务时,使用 OWL-S API 和推理机把 OWL-S 本体信息映射到数据库中。

2 系统的设计

系统的体系结构如图 2 所示。

图 3 是服务提供者发布服务时的执行流程图:首先创建 Web 服务,服务开发包括编写 Java 代码;接着使用 Java2WSDL 转换器生成 WSDL 文档(由实现服务的 Java 接

口生成)。WSDL 文档接着通过 WSDL2OWLS 转换器转换为 OWL-S 文档。加载 OWL-S 文件到 OWL-S 编辑器,以补充不完整的信息,如服务组合信息、服务提供者的信息等。然后通过客户端提交,存储到数据库中,如果成功注册则返回服务的唯一标识符,否则返回出错信息。

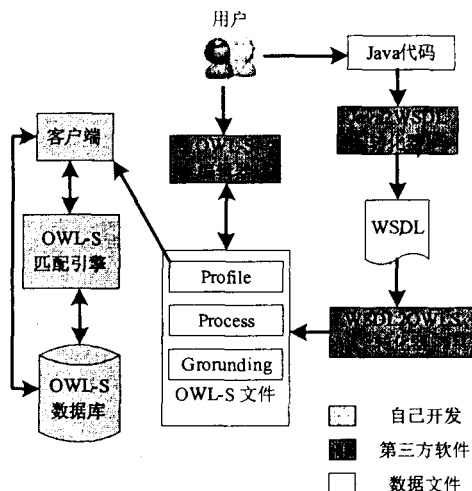


图 2 系统体系结构图

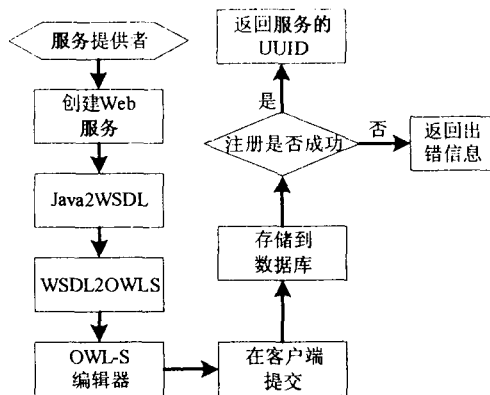


图 3 服务提供者发布服务流程图

图 4 是服务请求者查找服务时的执行流程图:首先通过 OWL-S 编辑器生成请求服务的 OWL-S 描述;接着在客户端提交,OWL-S 匹配器通过比较请求服务和数据库中已注册的服务,返回匹配结果,请求者可以查看结果中的每一个服务的详细信息,如果满意则选择此服务实际

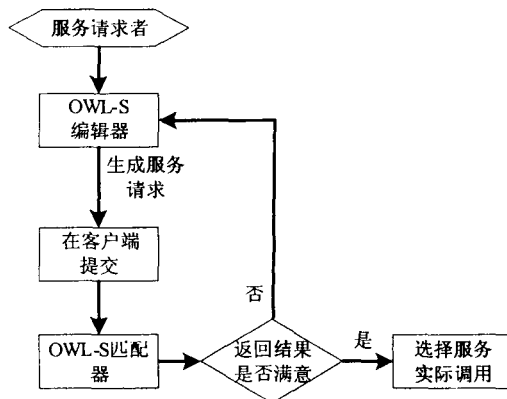


图 4 服务请求者请求服务流程图

调用,否则继续编辑查询条件重新查找,直到找到满足其需要的服务为止。

3 关键技术和算法

3.1 存储发布服务的 OWL-S 信息到数据库

通过 OWL-S 编辑器生成发布服务完整的 OWL-S 描述,在客户端提交后,系统采用数据库进行存储。首先利用卡耐基-梅隆大学开发的 OWL-S API 对 Profile 文件进行解析,以得到 Profile 文件中各部分的具体信息,如服务的联系信息、输入/输出参数信息以及服务的性能参数等。数据库表结构的设计包含存储这些信息的表,如输入参数表里面有存储输入参数 URI、参数类型等信息的字段。同时,OWL-S 本体中定义了很多概念,这些概念具有某些属性,概念与概念之间具有等价关系或某个概念是另一概念的子概念等关系,同样属性之间也存在着不同的关系,每个属性都有自己的定义域和值域。因此设计如图 5 所示的数据库表结构来存储 OWL-S 本体信息。

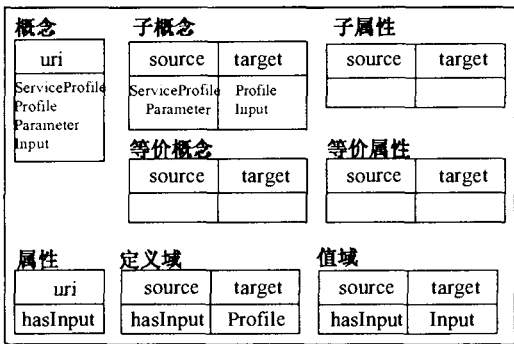


图 5 数据库表结构

在把 OWL-S 描述的语义信息映射到数据库时,使用推理机对本体进行推理。系统采用开源的 OWL-JessKB^[5]进行本体的推理。OWLJessKB 是针对 OWL 语言而设计的描述逻辑推理机。OWLJessKB 提供 Java API 以支持读取 OWL 文件并对 OWL 文件进行信息检索。OWLJessKB 底层设计采用 Jena 和 Jess。OWLJessKB 支持基本的推理任务如包含关系、等价关系等的推理,因而系统可以利用它来推理出 OWL-S 所描述的语义信息。

3.2 服务匹配算法

服务匹配是 Web 服务发现的一个重要部分,服务匹配就是根据用户对服务接口的描述去寻找能够满足要求的服务。

文中采用文献[6]中提出的服务匹配算法的基本原理,并对其加以改进。此算法的核心是基于语义匹配和性能搜索,并且由于精确匹配请求服务的服务并不一定总是存在,所以允许服务请求者输入其可以接受的最低匹配程度,即匹配结果可以按匹配程度划分为不同的等级。算法的基本原理是当请求服务的所有输出被发布服务的输出匹配,并且发布服务的所有输入被请求服务的输入匹配时,发布服务就匹配了请求服务。此原理确保了匹配结果满足请求者的需要,并且请求者能提供给匹配结果所有正

确的输入。匹配的主循环是请求服务与数据库中已注册的每一个服务相比较,以返回满足请求服务要求的。两个服务之间的匹配分为 4 个不同的阶段,每个阶段都独立于其他阶段,最后的匹配结果是这几个阶段匹配结果的综合。4 个阶段分别为服务输入参数的匹配、服务输出参数的匹配、服务 Profile 分级的匹配和服务质量的匹配。

下面详细介绍服务匹配的各个阶段。输入输出匹配阶段的匹配过程以及匹配结果的分级非常相似,因此文中就以服务输入参数的匹配为例来重点介绍。在匹配服务输入参数时,对于发布服务的每一个输入,算法都试图在请求服务输入中找到一个匹配。这是因为要想正确调用服务,发布服务的每一个输入必须被匹配。在 OWL-S 中每一个输入都是服务 Profile 实例的 hasInput 属性的属性值,所以输入的匹配包括属性匹配和输入属性值的类型匹配。

文中定义了 3 个不同的属性匹配等级:

* 等价。两个输入属性指的是同一个属性。

* 子属性。请求服务的输入属性是发布服务输入的子属性。

* 失败。以上都不是则失败。

同样定义了 4 个不同的类型匹配等级:

· 等价。两个参数类型指的是同一个概念。

· 包含。请求服务输入参数类型包含发布服务输入参数类型。

· 反包含。发布服务输入参数类型包含请求服务输入参数类型。

· 失败。以上都不是则失败。

输入匹配结果综合了属性匹配和类型匹配。表 1 显示了不同的输入匹配结果分级。等级越高,两个输入参数匹配的越好。

表 1 输入匹配结果分级表

等级	属性匹配结果	类型匹配结果	注 释
0	失败 任意	任意 失败	发布服务中至少有一个输入没有找到匹配 发布服务的每一个输入都被匹配,输入对的属性匹配结果要么是等价要么是子属性,但至少有一个是子属性,并且类型匹配是反包含
1	子属性	反包含	发布服务的每一个输入都被匹配,输入对的属性匹配结果要么是等价要么是子属性,但至少有一个是子属性,并且类型匹配是包含
2	子属性	包含	发布服务的每一个输入都被匹配,输入对的属性匹配结果要么是等价要么是子属性,但至少有一个是子属性,并且类型匹配是等价
3	子属性	等价	发布服务的每一个输入都被匹配,每个输入对的属性匹配都是等价,但至少一个的类型匹配是反包含
4	等价	反包含	发布服务的每一个输入都被匹配,每个输入对的属性匹配都是等价,但至少一个的类型匹配是包含
5	等价	包含	发布服务的每一个输入都被匹配,每个输入对的属性匹配都是等价,但至少一个的类型匹配是等价
6	等价	等价	发布服务的每一个输入都被匹配,每个输入对的属性匹配都是等价,类型匹配也都是等价

如果发布服务的所有输入都被匹配,带有最低匹配等级的输入将决定最终的匹配结果。比如表 1 中,匹配等级 5 代表所有输入对的属性匹配都是等价,类型匹配要么是等价要么是包含,但只要有一个输入对的类型匹配是包含,将决定了匹配等级低于所有类型匹配都是等价的等级。

输入参数匹配的步骤如下:对于发布服务的每一个输入,都试图找到和其具有最高匹配等级的请求服务输入。最高的匹配等级大于 0 则找到了一个匹配。发布服务的输入和具有大于 0 的最高匹配等级的请求服务的输入形成一个输入对。

当发布服务的输入列表为空时,输入参数匹配等级返回为 6,因为此时发布服务不需要任何输入参数,没有必要进行输入的匹配。

输出参数的匹配与输入一样,这里不再赘述,但值得注意的是匹配顺序的颠倒,对于请求服务的每个输出,系统都试图在发布服务输出中找到一个匹配。

在 OWL-S 中,可以使用 OWL 本体中描述的某种分层方法来对服务进行分级。在这个本体中,每个具体的服务或者是类 Profile 的实例或者是类 Profile 某个子类的实例。在进行服务 Profile 分级的匹配时,系统试图在发布服务中找到最匹配请求服务需求的服务。假设 reqServiceHie 表示请求服务分级,advServiceHie 表示发布服务分级。reqServiceHie 和 advServiceHie 均为某个服务分层本体中定义的概念。两者进行类型匹配的结果见表 2。

表 2 服务分级匹配结果

等级	匹配结果	注释
0	失败	两者彼此没有任何关系
1	未分类	两者中只要有一个未分类,即直接是 Profile 类本身的实例
2	包含	advServiceHie 被 reqServiceHie 所包含。即 advServiceHie 代表一个比 reqServiceHie 更具体的概念。既然发布服务的分级是请求服务分级的一个子类,意味着发布服务提供比请求服务需求更具体的功能
3	匹配	reqServiceHie 和 advServiceHie 是等价的。这是对请求服务分级的最好匹配

最后一个匹配阶段是服务质量的匹配,服务请求者提

出的服务质量和发布服务的服务质量进行语义匹配以检验服务质量要求是否满足。

4 总结和展望

文中介绍了语义 Web 服务的描述语言 OWL-S,设计并实现了一个基于 OWL-S 的 Web 服务发现系统。针对 OWL-S 描述的 Web 服务的发布和查询过程进行了介绍,重点介绍了 Web 服务的 OWL-S 信息在数据库中的存储结构以及服务匹配算法。文中只是实现语义 Web 服务发现的一个初步模型,存在一些不足之处,如系统并没有考虑与现有 UDDI 注册中心的结合,另外文中所介绍的 OWL-S 本体到数据库的映射完全取决于系统采用的推理机,推理机的局限性会导致信息存储的不完整,这些问题将在进一步研究工作中予以解决。

参考文献:

- [1] Berners-Lee T, Hendler J, Lassila O. The Semantic Web[J]. Scientific American, 2001, 284(5): 34-43.
- [2] OWL Web Ontology Language Guide[R/OL]. W3C Candidate Recommendation 18 August 2003. <http://www.w3.org/TR/2003/CR-owl-guide-20030818/>, 2003.
- [3] Martin D. OWL-S: Semantic Markup for Web Services[R/OL]. Technical report, Daml consortium, <http://www.daml.org/services/owl-s/1.0/owl-s.pdf>, WRSP Primer Working Draft 0.3., 2004-02.
- [4] Paolucci M, Kawamura T, Payne T R, et al. Importing the Semantic Web in UDDI[A]. In Proc of Web Services, E-Business and Semantic Web Workshop, CAiSE 2002[C]. [s. l.]: [s. n.], 2002. 225-236.
- [5] Kopena J, Regli W. DAMLJessKB: A tool for reasoning with the semantic web[J]. In IEEE Intelligent Systems, 2003, 18: 74-77.
- [6] Paolucci M, Kawamura T, Payne T, et al. Semantic matching of web service capabilities[A]. In Proceedings of 1st International Semantic Web Conference (ISWC2002)[C]. Berlin: Springer-Verlag, 2002. 333-347.

(上接第 154 页)

- [3] Lindell Y. A simpler construction of CCA2 - secure public-key encryption under general assumptions[A]. Eurocrypt 2003 [C]. LNCS 2656. [s. l.]: Springer-Verlag, 2003. 241-254.
- [4] Fujisaki E, Okamoto T. Secure integration of asymmetric and symmetric encryption schemes[A]. Crypto'99 [C]. LNCS 1666. [s. l.]: Springer-Verlag, 1999. 537-554.
- [5] Desmedt Y, Frankel Y. Threshold cryptosystems[A]. Crypto'89 [C]. LNCS 435. [s. l.]: Springer-Verlag, 1989. 307-315.
- [6] Anderson R. Two remarks on public key cryptography[EB/OL]. Invited Lecture. ACM-CCS '97. <http://www.cl.cam.ac.uk/ftp/users/rja14/forwardsecure.pdf>, 1997.
- [7] Bellare M, Yee B. Forward security in private-key cryptography[A]. CT-RSA 2003 [C]. LNCS 2612. [s. l.]: Springer-Verlag, 2003. 1-18.
- [8] 王 滨, 汪和松. Diffie-Hellman 密钥建立协议的前向保密性研究[J]. 长沙电力学院学报(自然科学版), 2004, 19(3): 15-17.
- [9] 王 滨, 张少武, 杨 颢. 密码协议的前向保密性研究[J]. 计算机工程与应用, 2004, 40(25): 157-160.
- [10] Cheon J H, Lee D H. Diffie-Hellman Problems and Bilinear Maps[EB/OL]. <http://eprint.iacr.org/2002/117/>, 2002.