Web services 合成关键技术分析*

孙海燕+, 王晓东, 周 斌, 贾 焰, 邹 鹏

(国防科学技术大学 计算机学院,湖南 长沙 410073)

Key Technologies Analysis of Web Services Composition

SUN Hai-Yan+, WANG Xiao-Dong, ZHOU Bin, JIA Yan, ZOU Peng

(School of Computer, National University of Defense Technology, Changsha 410073, China)

+ Corresponding author: Phn: +86-731-4573694, E-mail: shyfirst@hotmail.com, http://www.nudt.edu.cn

Received 2004-05-10; Accepted 2004-07-16

Sun HY, Wang XD, Zhou B, Jia Y, Zou P. Key technologies analysis of Web services composition. *Journal of Software*, 2004,15(Suppl.):124~129.

Abstract: Web services Composition technology is gaining more and more attention. In this paper, the key technologies in Web Services Composition are surveyed, followed by brief introduction to each open problem. This paper also discusses some traditional problems, such as QoS, fault-tolerance and transaction, that are facing new challenges in Web services composition.

Key words: Web services; Web services composition; QoS

摘 要: Web services 合成作为一项新技术得到了工业界和学术界的广泛关注.在介绍 Web services 合成背景、基本概念的基础上,分析了 Web services 合成的关键技术并指出 QoS、容错及事务等传统问题在 Web services 合成中所面临的新的挑战.

关键词: Web services;Web services 合成;服务质量

新的在线经济为企业的发展带来了巨大潜力和激烈竞争,为了在信息计算带来的革命中生存,各类企业纷纷采用 Internet 和 Web 技术,将它们的主要业务搬到 Web 上以得到更强的自动性、更迅速的交易和更广阔的视野.目前,Web services 为 Internet 上 Web 应用的主要形式之一.Web services 是一种新型的 Web 应用程序,具有自包含(self-contained)、自描述(self-describing)以及模块化、松耦合等特点,可以通过网络发布、查找和调用[1].

随着 Web services 的广泛应用,另一项新技术——Web services 合成(Web services composition)技术得到工业界和学术界的广泛关注.Web services 合成就是将多个自治的 Web 服务根据需要,经过服务发现以及接口集成等以提供新的、功能更强的 Web 服务[2].Web services 合成的应用非常广泛,可以应用到商业、生活以及军事等各个领域.

在商业领域中,企业和企业之间存在频繁的业务来往.企业之间可能需要形成联盟以提供更好、功能更强

* Supported by the National Natural Science Foundation of China under Grant No.90104020 (国家自然科学基金)

作者简介: 孙海燕(1976－),女,山东莱阳人,博士生,主要研究领域为分布计算,数据网格;王晓东(1973－),男,博士,副研究员,主要研究领域为协同工作,移动计算;周斌(1971－),男,博士,副研究员,主要研究领域为分布计算;贾焰(1960－),女,博士,教授,主要研究领域为分布计算,数据库;邹鹏(1957－),男,博士,教授,主要研究领域为分布计算.

大的服务,或者企业之间可能需要共享技能、资源等,这种企业间的来往即 B2B(business-to-business)技术,如美国 Ford 汽车公司离不开能源公司提供的能源、钢铁公司提供的钢铁材料以及各类销售商为其销售产品.而采用 Web services 合成技术就能够适应现代商业中动态多变的业务需求,能够在数量众多、分布广泛的企业间自动构造并自动执行企业业务.

又如在军事领域中,信息战成为现代战争中一种重要的甚至是决定性的作战形式.不可避免地,信息战的过程中需要不同单位、不同兵种所提供的服务联合作战来共同完成任务.任何一场战斗的胜利都需要进攻系统、防御系统和后勤系统等多方的共同协作.如何有效组织现有的信息服务,来准确、迅速地完成任务成为一个重要课题.采用 Web services 合成技术就能够在广域网的范围内灵活协调各方以共同完成目标.

本文分析了现有的信息集成技术不足以支持现代企业业务集成,并为此提出了 Web services 合成技术,并在介绍 Web services 合成基本概念的基础上,分析了 Web services 合成的关键技术并指出 QoS、容错及事务等传统问题在 Web services 合成中所面临的新的挑战.在本文中,Web services 主要指 Web services 技术或者 Web services 体系结构,而 Web service 即 Web 服务,特指基于 Web services 相关协议实现的、面向应用的具体服务.

1 相关技术比较

与 Web services 合成类似的信息集成技术包括 ERP(enterprise resource planning),EDI(electronic data interchange),EAI(enterprise application integration)和跨组织的工作流(cross-organizational workflows)等,它们主要应用在企业信息集成领域,并分别适合不同的应用环境[3].

ERP 系统是一个有效组织、计划和实施企业"人"、"财"、"物"管理的系统,它依靠计算机技术和手段保证其信息的集成性、实时性和统一性.比较著名的 ERP 公司有 SAP 和 PeopleSoft.但现有的 ERP 系统非常庞大复杂,不能够满足现代企业动态变化的需求.

最早出现的 EDI 技术主要是为满足跨组织的应用与应用之间传递业务文档.传统的 EDI 试图通过 VAN(value added networks)连接企业和他们的客户、供应商和合作伙伴.美国的 GM 汽车公司建立了命名为 EDSNET 的网络,GM 公司在该网络上通过 EDI 技术与 2 000 多个提供商进行通信.但 EDI 具有昂贵、耗时及灵活性差等缺点,不能够满足现代企业间动态协同的需求.

在 20 世纪 90 年代提出的 EAI 主要是解决企业内部分散系统的集成问题,其典型系统为 IBM 的 MQSeries 系统.EAI 主要用于满足企业内部的一些集成需求,尽管 Inter-EAI 可以解决企业间业务集成的部分问题,但其仅能为少量企业的集成提供支持,且在集成过程中,要求企业间的合作关系是静态的.

跨组织的工作流的目的是在不同系统中自动处理业务.最早期的项目包括 InterWorkflow,WISE,Flow-Jet 等,它们主要关注少量的、紧耦合业务过程之间的集成.其特点是仅仅适合少量业务的应用集成,且要求企业之间的合作关系为可预测的.

通过上面的分析,我们发现以上几类技术在满足现代企业集成要求方面都存在问题.针对上述问题,企业界和学术界提出了 Web services 合成技术.Web services 合成的目标是提供一种更自动、灵活、有效的机制来支持广域网范围内企业间复杂业务的自动集成和协同.

2 Web services 合成基本概念

根据开发过程,Web 服务(Web service)可以分为基本服务(elementary service)和合成服务(composite service)[3,4].

基本服务:指预先存在的服务或者本地服务,对于其他服务或客户而言,这类服务是一个黑盒.

合成服务:由一组其他服务(可能是基本服务也可能是合成服务)集成,该组服务作为合成服务的组件服务存在.

Web services 合成过程可以分为静态合成和动态合成两类[4].

静态合成:在设计时,由设计者根据需要选择组件服务的提供者,并按照一定的顺序组合服务,该类合成称

为静态合成.其典型系统包括微软的 Biztalk Server 和 Bea WebLogic Integration.

动态合成:指在运行时刻由系统根据客户的需要,自动选择组件服务的提供者,并生产新的 Web 服务.其典型系统包括 HP 的 eFlow 和 Stanford 大学的 SWORD 系统等.

服务合成选择静态合成还是动态合成取决于应用的特点和用户的要求.如果被合成服务的组件服务存在内在关系,且其各方之间的关系和约定不会经常发生变化,则静态合成就可满足需要.如果需要根据环境做出动态变化,则需要采用动态合成方式.动态合成以静态合成技术为基础,但动态合成的实现难度远远大于静态合成.在本文中,我们主要关注动态合成.

Web services 动态合成过程如图 1 所示.动态合成过程包括下面 4 步:

(1) 任务分解:将用户提交的任务分解为小的、更容易解决的简单任务;

(2) 简单任务组合:将简单任务按照一定的逻辑组合在一起,形成抽象服务,每个简单任务作为抽象服务的子服务;前两步的实现目前大多借鉴于人工智能的解决方案;

(3) 服务发现:通过服务发现过程为抽象服务中的每个子服务定位合适的服务提供商,形成具体服务;

(4) 服务执行:调度具体服务执行,得到最终结果.

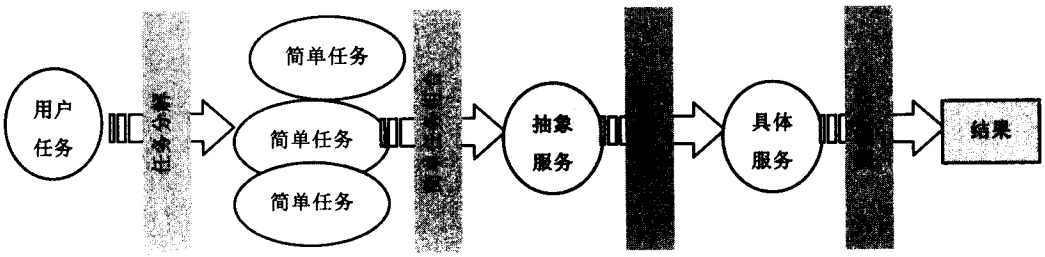

图 1 动态服务合成过程

3 Web services 合成关键技术

由 Web services 合成过程可以看出,Web services 合成的具体实现涉及人工智能等传统问题.本文仅仅关注与 Web services 领域关系比较密切的服务发现、服务执行等问题,并不详细讨论人工智能问题.本节将详细分析服务发现以及服务执行相关技术及 QoS、容错及事务等传统问题在服务合成中所面临的新的挑战,并指出保证服务合成过程中的 QoS、容错和事务等特性所需的研究工作.

3.1 服务发现

服务发现是在广域网、动态环境下实现服务合成时首先要考虑的问题.服务发现即在服务合成过程中根据需要到注册中心找到合适的服务提供者.服务发现技术包括服务的描述、服务发现引擎以及服务匹配算法等关键技术:

(1) 适当的服务描述:关于 Web 服务的描述应该包括语法(What does it look like?),语义(What does it mean?)和 QoS(How well does it perform?)等方面的信息;目前 WSDL(Web services description language)[5]为广泛使用的 Web services 描述语言.WSDL 对 Web services 的接口描述提供支持,给出了服务可能提供的操作、网络地址等一系列语法信息.DAML-S(Darpa Agent markup language for services)[6]是一种基于本体论的服务描述语言,既可以用来描述语法又可以描述 Web services 的语义.DAML-S 同时对服务的 QoS 描述提供一定支持.

(2) 自动的服务发现引擎:发现引擎根据给定的 Web services 相关信息,到服务注册中心定位相应的服务实体;UDDI(universal description,discovery and integration)[7]为基于 XML 和 SOAP 之上定义的新规范,UDDI 规范使得不同企业采用相同的描述方法描述所能提供的服务,并能查找其他企业提供的服务.UDDI 主要提供语法方面的支持,而对语义、QoS 等目前并没有提供支持.

(3) 适当的服务匹配算法:服务注册中心返回的可能是一组满足条件的服务,客户需要根据自己的要求选择最适合的服务.

3.2 服务包装

因为合成服务中的 Web 服务由多个提供商提供,故其开发过程是独立的,其结构是异构的.因此,有必要隐藏每个 Web 服务的异构性并将其特性抽象,以保证 Web 服务间的互操作[8].服务包装层的目的就是将预先存在的服务规范与其实现分开,以实现服务间的互操作.服务包装层主要包括下面几个部分:

(1) 通信 manager:Web 服务可能采用不同的通信方式,而该 manager 的功能就是实现异构通信协议之间的转换;

(2) 安全 manager:Web 服务的访问可能需要通过防火墙和其他安全措施,安全 manager 负责访问的授权及认证;

(3) 内容 manager:Web 服务可能采用不同的信息格式和语法,如内部的应用采用 xCBL 格式表示,而客户需要的是 cXML 格式的文档,此时需要由内容 manager 进行格式转换;

(4) QoS Manager:需要对 Web 服务的 QoS 等进行监测;

(5) 状态 Manager:监测 Web 服务实例的状态.

3.3 Web services合成服务描述

Web services 合成服务一般采用工作流的方式描述 Web 服务的执行顺序以及 Web 服务间的信息流动,也可以采用其他方式,如 Stanford 大学开发的 SWORD 系统采用规则系统描述合成服务.

IBM 的 WSFL[9](Web services flow language)和微软公司的 XLANG[10]为影响较大的服务合成描述语言,这两者都基于并扩展了 WSDL 语言.其中,WSFL 既支持静态的服务合成也支持在运行时刻到服务注册中心查找服务,而 XLANG 要求服务是静态配置的.BPEL4WS[11](business process execution language for Web services)为新出现的 Web services 合成语言,其集成了 WSFL 面向图形的过程表示和 XLANG 基于结构构成过程的特点,形成一个统一的描述合成服务的标准.除此之外,"DAML-S"[6]为一组标记语言,该语言用清晰的、计算机可以理解的形式来描述 Web services 合成服务.

WSFL,XLANG,BPEL4WS 和 DAML-S 等语言的特点见表 1.

表 1 几种服务合成语言的特点比较

	WSFL	XLANG	BPEL4WS	DAML-S
过程建模方案	支持基于图的过程建模	支持基于构造的过程建模	支持基于图和构造混合的过程建模	支持基于构造的过程建模
是否支持语义	不支持语义	不支持语义	不支持语义	支持基于语义的服务描述和服务发现
是否支持QoS规范	使用WSEL扩展元素描述QoS	不支持QoS规范	不支持QoS规范	通过Service Profile类支持QoS规范
与WSDL的关系	基于WSDL之上	基于WSDL之上	基于WSDL之上	提供一种可以比WSDL支持更多特征的语言

3.4 服务合成引擎

服务合成引擎为服务合成的核心部分,负责分解用户的任务、生成合成服务,并调度合成服务中各组件服务的执行.在合成引擎的设计中,必须考虑下面关键问题:

3.4.1 合成服务的执行

合成服务的执行分为两种方式:集中控制模式和 Peer-to-Peer 模式.

(1) 集中控制模式:基于 Client/Server 体系结构.该模式存在一个调度者,由调度者集中控制合成服务的执行.调度者根据合成服务的描述,激活相应的服务实例并得到其运行结果.HP 公司的 eFlow 系统就采用集中控制模式.

(2) Peer-to-Peer 执行模式:在该模式下,相关的 Web 服务通过协调和共享执行上下文来实现分布执行.每个 Web 服务实例中都存在一个协同组件,通过与别的服务实例的协同组件协作以实现服务合成.该模式可以解决集中控制模式中调度者的瓶颈问题,更适于大型应用.澳大利亚新南威尔士大学和昆士兰科技大学开发的 SelfServ 系统中采用该执行模式.

3.4.2 QoS

随着 Web services 的广泛应用,服务质量将变成一个判定服务提供者是否满足客户需求的重要因素.QoS 决定服务的可用性和实用性,而这两方面都会影响到服务的应用.

QoS 对于服务合成过程非常重要,这是因为:

(1) 需要根据任务的 QoS 目标和要求定义合成服务;

(2) 基于 QoS 来选择和执行组件服务;

(3) 监控合成服务的执行,以满足任务的 QoS 要求;

(4) 根据 QoS 来评价各种可选方案.

在服务合成中,要实现基于 QoS 的 Web 服务描述、选择,其关键技术包括:

(1) Web 服务的 QoS 描述;

(2) Web 服务 QoS 的监视与评估;

(3) 基于 QoS 的服务匹配算法;

(4) 合成服务的 QoS 实时计算;

(5) 合成服务的 QoS 描述等.

3.4.3 事 务

事务为保证系统可靠性的有效途径.在基本事务模型中,主要是在计算过程中保证事务的 ACID 这 4 个特性.保证 ACID 特性的基本策略有:(i) 在事务的执行过程中封锁资源,如两阶段锁协议;(ii) 多步提交事务,如两阶段和三阶段验证协议.然而,普通模型并不适合 Web services 合成过程中的事务处理,主要基于以下两个原因:

(1) 在 Web services 合成过程包括不同的 Web services 系统,而不同 Web services 系统可能属于不同的服务提供商,因此它们有可能属于不同的管理域,Web 服务实例之间会产生不兼容或者某些 Web 服务不愿意放弃本身的自主性,而这就影响了事务的处理;

(2) 在嵌套事务结束之前封锁访问资源的策略在 Web services 合成中也不可行,这同样是因为他们的自主性以及每个 Web 服务可能存在大量并发客户,而这些客户不愿意忍受因为资源封锁而带来的额外延迟.

由上面的分析,可以看到传统的 ACID 模型并不适合 Web Services 合成过程的要求.因此必须提出一种放松的事务模型,以满足 Web Services 合成的需要.

3.4.4 容 错

容错是保证系统可靠性和可用性的关键机制,使得系统在有失效发生时仍然能够继续正确执行.一般系统的容错手段包括冗余配置、失效检测和恢复等.

但是在 Web Services 合成过程中,其容错与一般系统相比有其特殊性[12],这是因为:

(1) 每个单独的 Web 服务都是独立开发、维护的,服务消费者(或者合成引擎)对其使用的服务的服务质量及可用性几乎无能为力.如在使用异地服务的过程中发现一个错误,消费者没有办法使其得到尽快改正.

(2) 在动态合成中,服务消费者不可能事先知道所要用到的服务,更不能要求每个服务提供商都为其服务提供冗余支持.

因此,需要在服务合成平台中提供一种适应 Web Services 合成特点的失效检测机制和失效恢复机制.通过失效检测机制能够尽快检测到失效的发生;而失效恢复机制可基于采用替代服务,或者是采用其他合成途径来实现.

4 小 结

Web services 合成技术作为 Web services 技术的关键研究领域之一,在很大程度上促进了 Web services 技术的应用与发展.本文在介绍 Web services 合成概念的基础上分析了其关键技术:

(1) 服务描述、发现与匹配为动态服务合成的基础;

(2) 服务包装的目的是实现跨企业的 Web services 合成;

(3) Web services 合成服务描述为合成引擎调度组件服务执行的基础;

(4) 合成引擎是服务合成的核心,负责调度组件服务的执行;由于 Web services 合成的松耦合性,使得 QoS、事务及容错等传统问题具有新的特点并面临新的挑战.

我们目前正在开发基于 StarWebServices(我们自主研发的 Web services 平台)的服务合成系统——StarWSCoP(Star Web services composition platform),在该系统中我们实现了基于 QoS、容错、事务的 Web services 合成.

References:

[1] Web services architecture overview. 2000. http://www-106.ibm.com/developerworks/web/library/w-ovr/?dwzone=ibm

[2] Ponnekanti SR, Armando F. Sword: A developer toolkit for building composite Web services. In: Proc. of the 11th Int'l World Wide Web Conf., WWW 2002. Honolulu: ACM Press, 2002.

[3] Benatallah B, Dumas M, Fauvet M, Rabhi F. Towards patterns of Web services composition. In: Patterns and Skeletons for Parallel and Distributed Computing. Springer-Verlag, 2003. 265~296.

[4] Chandrasekaran S, Miller J, Silver G, Arpinar IB, Sheth A. Composition, performance analysis and simulation of Web services. The Int'l Journal of Electronic Commerce and Business Media (EM),2003,13(2).

[5] Web services description language (WSDL) 1.1. 2001. http://www.w3.org/TR/wsdl

[6] DAML-S: Semantic markup for Web services. 2001. http://www.daml.org/services/daml-s/2001/10/daml-s.html

[7] UDDI technical white paper. 2000. http://www.uddi.org

[8] Benatallah B, Dumas M, Fauvet M-C, Rabhi FA, Sheng QZ. Overview of some patterns for architecting and managing composite Web services. ACM SIGecom Exchanges, 2002,3(3):9~16.

[9] Web services flow language (WSFL 1.0). 2001. http:// www-3.ibm.com

[10] Thatte S. XLANG Web services for business process design. Microsoft Corporation, 2001.

[11] Curbera F, Goland Y, Klein J, Leymann F, Roller D, Thatte S, Weerawarana S. Business process execution language for Web services, Version 1.0. 2002. http://www.ibm.com/developerworks/library/ws-bpel/

[12] Mikalsen T, Tai S, Rouvellou I. Transactional attitudes: Reliable composition of autonomous Web services. In: Proc. of the Dependable Systems and Networks Conf. 2002.