一种基于网格环境的服务合成模型*)

裴艳琴　杨寿保
(中国科技大学计算机科学技术系　合肥 230026)

摘　要　本文在Web服务合成的基础上,根据抽象工作流和具体工作流责任两种概念,增加了服务的factory机制,加入了服务状态管理器,提出了一个适合于网格环境中的服务合成模型。它采用语义Web中的技术和方法,对服务间的输入输出进行近似的语义匹配,采用动态自适应算法来实现服务的合成。从某种程度上来说,这种服务的合成实质就是对工作流的合成。这个服务合成模型既实现了服务的自动合成,同时又提高了服务合成的容错性和可扩放性。

关键词　网格服务,服务合成,抽象工作流,具体工作流

A Service Composition Model Based on Grid Environment

PEI Yan-Qin　YANG Shou-Bao
(Computer Science Department, University of Science and Technology of China, Hefei 230026)

Abstract　This paper aims at researching how to realize the service composition automatically in grid environment. The research about the service composition techniques in Web environment finds that all these existing methods don't take into account the states of grid service, so it isn't applicable to the grid environment. So a service composition model based on grid environment is proposed. It adds a service factory and divides workflow into abstract workflow and concrete workflow. The input and output between the services are matched semantically and the model uses an adaptive algorithm to compose the services. This model can not only compose the services automatically, but also provide a high degree of fault-tolerance and scalability.

Keywords　Grid service, Service composition, Abstract workflow, Concrete workflow

1　引言

随着因特网技术的日益发展和应用的需求,Web服务[9]已日益成为一种非常流行的技术。不管是工业界还是学术界,都对这种技术的茁壮成长给予了极大的关注。Web服务为应用提供了一种动态交互的统一平台,解决了跨平台应用和系统间无法交互的难题。Web服务已作为一种标准技术应用在大型研究中。OGSI[1]已将Web服务作为一种开发开放网格技术的新标准。Web服务的优点在于网格应用的开发者得到Web服务标准的广泛支持。OGSA[1]不仅将Web服务作为一种网格技术,还对Web服务增加一些新的特征,如实例管理、消息通知机制和网格资源间的状态交互等等,并在现有的网格标准中集成了Web服务技术。简言之,就是OGSI对Web服务进行了一些必要的扩充,这就形成了现在的网格服务(Grid services)[1]。

在网格环境中,常常需要进行大型计算,这就需要调用大量的服务和资源,因此需要多个服务间进行相互协调和合作。但在通常情况下,这些服务处于不同的网络环境中,因而服务之间需要进行大量的消息交互。由于这些服务处于不同的位置上,服务间的通信和调用就成了性能提高的瓶颈,因而需要对服务进行无缝连接,以高效地完成用户提交的任务。这就是所谓的服务合成[2]问题。

本文主要针对一种现有的Web环境中的语义合成方法[2],提出了一种适合于网格环境的语义服务合成模型。本文共分为5部分,其中第2部分介绍Web环境中的语义合成方法以及网格服务的概念;在第3部分中,在Web服务合成模型的基础上进行改进,提出了一种网格环境中的服务合成模型;第4部分给出了一个服务分析工具WSAT[5],主要用来分析服务合成的正确性;最后是全文的总结。

2　相关技术

2.1　网格服务

Web服务是一种部署在Web上的独立实体,它描述了一些操作的接口,使用标准的XML[10]消息传递机制进行通信,可以通过网络访问这些操作。这些接口隐藏了服务实现的细节,因而服务间可以跨软件或硬件平台进行合作和通信,这就使得基于Web服务的应用程序具备松散耦合、面向组件和跨平台实现的特点。但它同时也存在一些局限性:从用户的角度来看,Web服务是无状态且非透明的服务。所谓无状态的Web服务就是指在调用结束时服务无法记住调用结果,因而操作无法重用以前的调用结果。非透明性是指服务的存在周期要比客户的存在周期长,当客户端使用完一个Web服务后,下一个使用者仍可以访问到这个服务保存下来的信息。实际上,当一个用户在使用一个Web服务时,另一个用户也可以同时访问这个服务,这样就会打乱第一个使用者的操作。因此,从这种意义上来说,Web服务是非透明的。OGSI所要解决的就是这些问题。

OGSA把网格服务看作是一种特殊的Web服务,描述的

*)本文得到国家自然科学基金项目(编号60273041)和国家863计划(编号2002AA104560)的资助。**裴艳琴**　硕士研究生,主要研究方向:网格计算、机群计算;**杨寿保**　教授、博士生导师,主要研究方向:网格计算、密码学和网络安全。

是一个网格计算和 Web 服务相结合的计算环境，是一个全新的网格标准，它定义了网格服务的描述，服务实例的创建、发现和管理等所必须遵循的一系列的标准和规范。与 Web 服务相比，网格服务通过使用 factory 的方法克服了 Web 服务的无状态和非透明的难题。factory 用来产生网格服务的多个实例，每个网格服务实例可以被多个用户共享，每个用户也可以使用多个网格服务实例。而且，通过对网格服务实例的生命期进行限制并在使用后进行配置，从而达到了实例的透明化。

2.2 Web 环境中的服务合成方法

这种方法是利用语义 Web 的技术对网格中的服务进行描述。语义合成有三种方法：人工合成、半自动合成和自动合成。所谓语义 Web[14] 是指在网格计算中使用语义 Web 技术，并能使网格中的如工作流的发现、选择、合成和执行等多种任务能够自动执行，从而使自治、异构和分布的应用能够进行无缝的互操作。目前的自动合成技术在复杂的动态环境中进行合成时，仍存在多方面的限制，例如不支持工作流的自治、自适应、重用和共享问题等。因此，语义 Web 的自动合成机制有两个特征：服务的发现和合成是动态和自适应机制；不同层之间进行松耦合集成，以实现重用和共享。

3 网格服务合成

在网格环境中，所有的资源都可以称为服务。网格应用就是将网格中的应用元素综合起来，实现某个特定的功能。在进行分布式计算时，各种资源需要进行相互协作，因此就需要远程过程调用应用元素，这样就形成了网格应用的工作流。因此，对网格服务的合成就是对网格应用工作流的合成。基于以下的几个原因，需要对服务进行合成：首先，对于某个特定的应用，网格环境中存在许多可用的服务，如果要让用户自己来查找合适的服务是很不现实的；其次，需要用户来确保服务间交互的语义和语法的正确性，这增大了用户的工作量，并增加了用户使用网格服务的难度；最后，任何一个服务都不可能提供用户所需的全部功能，要完成一个任务通常需要多个服务的协作。因此，为了解决这些问题，需要设计一种方法，它通过特定的机制不需要用户的参与就能自动管理和查找网格中的资源。这就是所谓服务合成的概念。

3.1 基于网格环境的服务合成模型的设计

Shalil Majithia，David W. Walker，W. A. Gray 提出了一种基于 Web 环境的服务合成模型[7]，在这个模型中将工作流分为抽象工作流 AWF（Abstract Workflow）和具体工作流 CWF（Concrete Workflow）的概念。我们借鉴 Web 环境中服务合成模型的一些思想，提出了一种适合于网格环境的服务合成模型，它加入了服务 factory 机制，采用动态自适应算法实现对服务的合成。这种合成的模型如图 1 所示。

这种合成机制包括八个模块：工作流管理服务系统 WFMS（Workflow Managing Service）、抽象工作流合成器 AWFC（Abstract Workflow Composer）、具体工作流合成器 CWFC（Concrete Workflow Composer）、推理系统 RS（Rule System）和匹配系统 MMS（Matchmaking System）、抽象工作流容器 AWFCR（Abstract Workflow Container）、具体工作流容器 CWFCR（Concrete Workflow Container）、规则库 RB（Rule Base）和服务管理器 SM（Service Manager）。其中，两种流合成器是这个模型的核心，其他的几个模块是这两个核心的支撑模块。我们将分别介绍这些模块的功能。

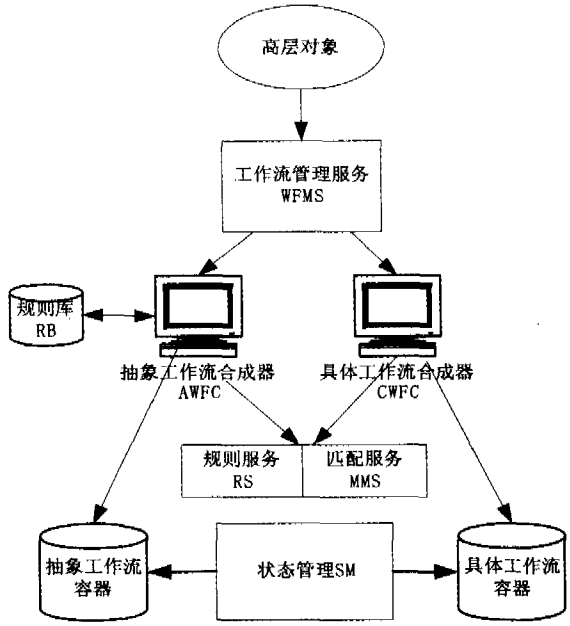

图 1 服务合成的模型结构

3.1.1 工作流管理服务系统（WFMS）

工作流管理服务系统负责协调整个模型中的各个模块的工作，它根据用户的需要将模型中的各个模块按照消息流的流向组织起来。当有合成请求到达时，它先要识别该请求是抽象工作流的合成还是具体工作流的合成，然后用工作流的形式来表示该合成请求，最后根据工作流的类型来决定将合成请求转发给哪一个合成器进行合成。这个模块的功能实际上有两个：一是将合成请求变为工作流的形式；二是决定由哪一个合成器进行合成。

3.1.2 抽象工作流合成器（AWFC）

抽象工作流合成器接收来自 WFMS 的关于抽象工作流合成的请求，在这个模块和 AWFCR 模块中，抽象工作流都使用逻辑标识符来表示。它使用 DAML-S[8] 过程模型来生成抽象工作流。AWFC 使用动态自适应算法，利用以下三种信息资源来进行抽象工作流的合成：

• 抽象工作流容器（AWFCR）。AWFCR 存储以前处理过的服务和抽象工作流的语义描述。AWFC 首先查询 AWFCR 以前是否处理过这样的请求，这种查询通过引用本体来语义匹配 AWFC 提供的工作流名字和 AWFCR 中的工作流名字。如果匹配成功，就将查询到的抽象工作流返回给 AWFC，此时 AWFC 收到的就是合成后的抽象工作流。

• 规则库（RB）。如果匹配不成功，AWFC 就向规则库查询，规则库给出一个用以实现这个合成目标的规则模板。然后，AWFC 再次向 AWFCR 查询，不过这次不是进行工作流的匹配，而是将要进行合成的抽象工作流的输入输出与 AWFCR 中的工作流进行匹配。如果存在相匹配的抽象工作流，就将匹配的工作流返回给 AWFC。否则，AWFC 就根据给定的规则，将工作流分成若干个工作流分量，然后对每个工作流分量进行匹配。这个过程过递归进行，直到每个工作流分量都分解为原子工作流为止。然后，AWFC 向 AWFR 查询其中是否有能与这些原子工作流相匹配的工作流。

• 链接服务。AWFC 使用推理服务按照工作流的输入输出将匹配的工作流连成一个服务链，最后生成的这个服务链就是满足用户要求的合成工作流。

3.1.3 抽象工作流容器(AWFCR)

在这里我们使用了容器的概念,图2给出了一个容器的实例,主要包括两个部分:factories和由此产生的实例,其中每个factory产生一种服务。它将服务封装起来,使得服务与特定的运行环境设置和服务生命周期管理以及服务实例的分发等分离开来。因此,它为它的请求者提供了一种透明服务。若要加入新的服务,只需增加一个factory即可,这样也就提高了该合成机制的可扩放性。

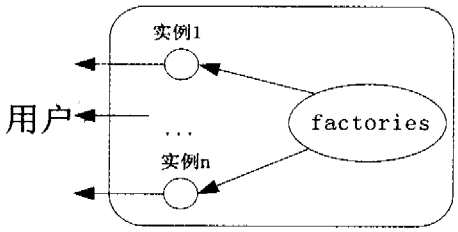

图2 抽象工作流的容器

当有请求到来时,若容器中有该抽象服务的factory,则就产生该服务的实例,然后将产生的服务实例返回给AWFC,否则就给AWFC发送一个匹配失败的消息。具体工作流容器的工作原理与此雷同,这里不再赘述。

3.1.4 具体工作流合成器(CWFC)

具体工作流的合成同样采用的也是动态自适应算法。CWFC使用动态适应算法将抽象工作流中的每个服务映射为网格中的具体可用服务,使用两种服务来实现这种映射:

·匹配系统。将每个抽象工作流中的服务传给匹配服务系统(MMS),MMS会尽量给CWFC返回一个网格中语义匹配的可用服务。CWFC根据匹配的程度选择是否接受这个匹配的服务。匹配包括确切匹配和插入(plug-in)匹配[3]。

·链接服务。CWFC使用推理服务将功能类似的服务链接起来。由于在合成的过程中,某些服务由于种种原因可能会由原来的可用状态变为不可用状态,在这种情况下,WFMS就会调用AWFC,让其提供一个替换的抽象服务。这大大提高了机制的容错性。

3.1.5 推理服务RS

推理服务对抽象合成器和具体合成器提供支持,它使用回溯算法来产生一个服务链,同时也为链接服务提供支持。它使用给定的输入产生要求的输出,其执行步骤如下:

1)对于每个可用的服务,查找一个服务,这个服务的输出与要进行合成服务的输出相匹配。假设该服务为S_n。

2)寻找一个能生成S_n输入的服务,假设该服务为S_{n-1}。

3)该过程循环执行,直到S_{n-x}服务的输入与请求服务的输入相匹配。

4)将S_1到S_n连接起来就形成了服务链。

匹配可以是确切匹配,也可以是插入匹配。这提高了链接算法的健壮性,同时匹配成功的概率也比一般算法要高。

3.1.6 匹配服务MMS

匹配服务提供一种智能匹配机制,它的主要功能是根据两个服务的输入输出和前提以及结果进行匹配。推理服务利用它来匹配每个链接服务的输入和输出。除此之外,匹配服务还用于抽象工作流与具体工作流的匹配。

3.1.7 状态管理器SM(State Manager)

由于网格中的每个服务有一定的状态和生命周期,因此我们在现有模型的基础上增加了一个状态管理器SM,用来管理网格中各种服务的状态,它实际上相当于网格服务的一个大数据库。我们将factory中的状态管理功能提取出来,交由SM统一管理,减轻了容器的工作负担。SM中与AWFCR和CWFCR相连,它采用USI(Universal State Identifier)的方法来为每个服务定位状态。在服务的状态发生变化时,SM会自动在每个服务容器中查找功能类似的服务来代替这个服务;当容器中没有服务可用时,SM就会异步地通知AWFR和CWFR。这样,AWFR和CWFR虽然以前处理过这样的请求,但由于服务的状态发生了变化,因而这个工作流仍然是不可用的,它们要分别通知AWFC和CWFC,要重新合成工作流。

由以上的介绍我们可以看出,这个模型的工作过程如下:WFMS接受来自用户的合成请求,并根据工作流的类型决定将合成请求转发给AWFC或CWFC。若是抽象工作流的合成,AWFC就向AWFCR查询以前是否处理过同样的请求。如果处理过,就将抽象工作流返回给MMS;若无,则将请求提交给RS,再从规则库中返回一个满足要求的合成规则。然后根据规则和输入、输出、前提及结果返回一个能提供类似功能的服务的集合。此过程递归进行,直至生成满足要求的抽象工作流并将其返回给MMS。最后,将合成后的抽象工作流传给CWFC,进行抽象工作流与每个可执行的实际服务间的匹配。如果匹配成功,就可将执行的结果返回给MMS。否则,就向MMS和RS请求返回一个能提供类似功能的服务的集合。整个过程的流程图如图3所示。

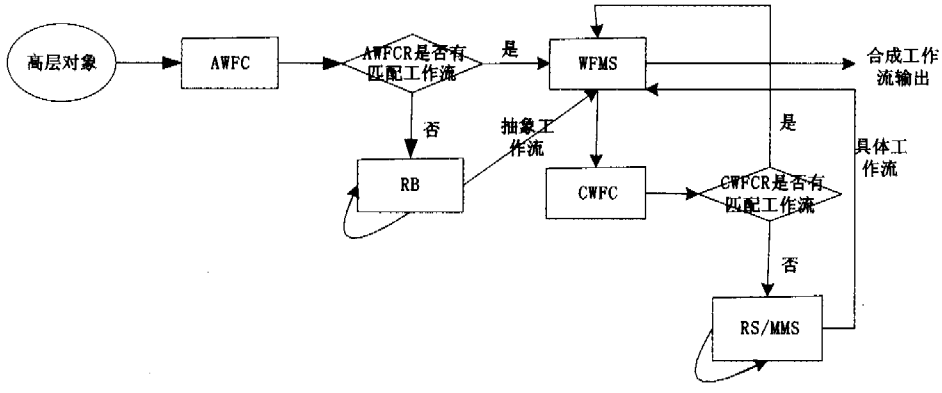

图3 模型服务合成的流程图

4 服务分析工具

4.1 对合成模型的分析

由以上的分析可以看出,这个合成模型满足了自动合成机制以下几个方面的基本要求:

·高度的容错性。模型必须能处理一般性的错误,因为

在网格环境中无法确保特定的服务在特定的时间是可用的。因此，必须有智能机制能调用和发现其他功能类似的服务来进行合成。在这个模型中，链接服务的存在大大提高了机制的容错性。

• 工作流粒度。模型应支持用户生成不同粒度级的工作流(如抽象工作流 AWF 和具体工作流 CWF)，这在网格环境中是非常重要的，因为我们无法确保所有的服务都是可用的。而且，不同粒度的工作流是松耦合的，也就是说它们具有最小的内部依赖性。

• 指这定和提取高级目标。模型允许用户指定和动态提取高级目标，然后再生成工作流。只要对发生变化的工作流子图进行再合成，就能高效地执行"what-if"分析。

• 选择用户的优化标准。这个模型允许用户选择工作流合成/执行的优化标准。例如，用户可以选择总运行时间或昂贵资源使用的最小化为优化标准。

• 可扩放性。随着服务的增加，这个模型是可扩展的。Factory 的采用增加了该合成机制的可扩放性。

4.2 合成模型的性能评估

在这个模型中，我们使用现有的语义 Web 工具，如 RDF、OWL-S 和推理机制来实现模型中的主要模块。使用 OWL-S 来描述服务，MMS 模块则使用 DQL/JTP 服务器来执行包含推理，容器则用来存放 OWL-S 描述的服务。WFMS 是模型的核心部分，它主要用来协调和定位模型中的各个模块，完成用户提交的任务。AWFC 服务用来生成一个抽象工作流，作为 DAML-S 进程模型。然后将这个抽象工作流作为 CWFC 服务的输入，把该工作流中的每个服务与当前可用的执行进行匹配。最后，生成一个包含所执行任务的网络位置和协议细节的 BPEL4WS 文档，并将其提交给 BPEL4WS 执行机执行。

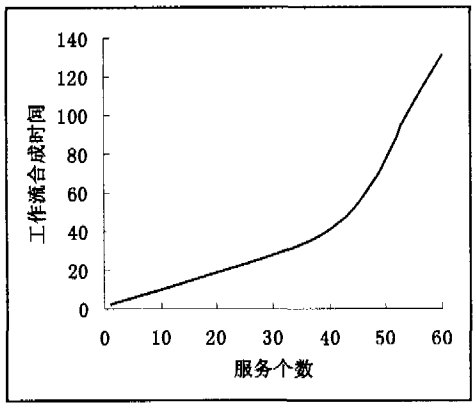

图 4 服务合成时间图

为了验证这个模型，我们在一个配置为 1.4GHz 的 Intel 处理器上和 512M 内存的 Windows 机器上运行了一个蛋白质折叠模拟[9]的实验，它主要是利用计算机模拟快速测定并追踪蛋白质重折叠的全过程，尽可能捕捉折叠过程中的每一个中间状态，直至蛋白质分子处于热力学最稳定、能量最低状态。这个实验包括极值的捕捉和折叠等多个服务。我们首先对具体工作流的生成时间进行了评测，如图 4 所示。从图 4 可以看出，工作流的匹配和合成时间与服务的数量呈指数关系增长。图 5 所示的是我们这个模型和自动回溯合成算法进行工作流合成成功率的比较。从这个图中可以看出，在条件相同的情况下，我们这个模型有 85%的成功率，而回溯算法的成功率仅有 60%。这就证明了这个合成模型有更高的工作流合成概率。

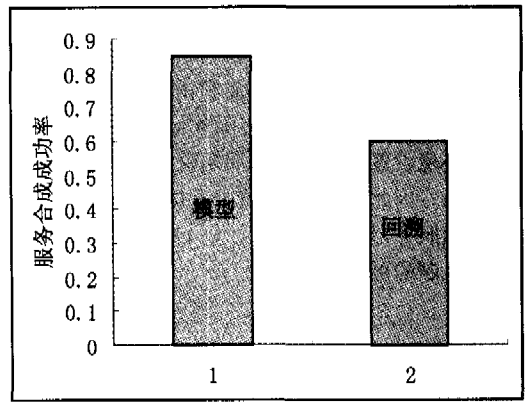

图 5 两种合成方法的比较

4.3 服务分析工具(WSAT)

这里，我们给出一个用于分析服务的工具——WSAT，主要用来分析和验证 Web 服务合成的设计。合成的 Web 服务要么是自底向上，要么就是自上而下。一般说来，对于一个合成的 Web 服务，自底向上的规范就是服务的合成，而自上而下的规范就是服务的会话协议。WSAT 针对不同的规范进行分析的内容不同，对于服务的合成主要进行可同步性分析，对于会话协议则主要进行可实现性分析。但大致来说，这两种分析的步骤是相同的。WSAT 针对不同的 Web 服务语言提供了一个统一的平台，首先要将 Web 服务语言翻译成 GFSA(Guarded Finite State Automata)，然后再进行可同步性(可实现性)分析，最后使用 SPIN 对分析结果的线性时态逻辑进行验证。

结束语 本文主要针对 Web 服务合成中存在的问题以及 Web 服务与网格服务的差异，提出了一种基于网格环境的服务合成模型，并简要介绍了现有的一种服务分析工具。在今后的工作中，我们会逐步完善这个模型，使之能提供更加可靠和实用的服务，然后对我们搭建的瀚海网格平台中的现有服务进行合成，并用 WSAT 对合成后的服务进行可实现性分析。

参 考 文 献

1 What is a Grid Service? http://www.casa-sotomayor.net/gt3-tutorial/multiplehtml/ch01s03.html

2 Srivastava B, Koehler J. Web Service Composition - Current Solutions and Open Problems. In: ICAPS 2003 Workshop on Planning for Web Services

3 0-7695-2095-2/04, Lakhal N B, Kobayashi T, Yokota H. THROWS: An Architecture for Highly Available Distributed Execution of Web Services Compositions. In: Proceedings of the 14th International Workshop on Research Issues on Data Engineering: Web Services for E-Commerce and E-Government Applications (RIDE'04)

4 Xie Yong, Ai Ting, Wang Caixia. Dynamic Grid Service Deployment. March 31, 2004

5 Fu X, Bultan T, Su J. WSAT: A Tool for Formal Analysis of Web Services. In: 16th International Conference on Computer Aided Verification, July 2004

6 Majithia S, Walker D W, Gray W A. Automated Composition of Semantic Web Services. AHM 2004, August 2004

7 http://www.daml.org/

8 0-7695-2051-0/04, Chao Kuo-Ming, Younas M, Griffiths N, et al. Analysis of Grid Service Composition with BPEL4WS. In: Proceedings of the 18th International Conference on Advanced Information Networking and Application (AINA'04) IEEE, 2004

9 什么是 Web 服务? http://www-900.ibm.com/developerWorks/cn/webservices/ws-wsar/part2/index.shtml

10 XMLweb 服务基础. http://www.microsoft.com/china/msdn/archives/library/Dnwebsrv/html/webservbasics.asp

11 岳昆，王晓玲，周傲英. web 服务核心支撑技术：研究综述. 软件学报