网格环境下的按需服务引擎*)

欧建宏 李 琦 张雪虎 毛 曦

(北京大学数字地球工作室 北京大学遥感与 GIS 研究所 北京 100871)

摘 要 本文探讨了在 Web 服务和网格服务环境下,一种智能的按需服务引擎的技术框架,它能够提供高效的空间服务管理和共享的能力。文章阐述了主动服务发现、智能服务组合、服务搜索、服务替代和服务容错性等问题,并给出原型系统设计及实现。

关键词 服务选择,服务发现,服务组合,服务替代,网格服务

Service Engine on Demand in Grid Environment

OU Jian-Hong LI Qi ZHANG Xue-Hu MAO Xi

(CyberGIS Studio, Institute of Remote Sensing & GIS, Peking University, Beijing 100871)

Abstract This paper gives a service engine on-demand framework, which can effectively manage and share spatial services on the internet scale. Then details about service discovery, intellegent service composition, service search, service substitution and error tolerance are disscussed. We design and implement a prototype system to verify these ideas.

Keywords Service selectiong, Service discovery, Service composition, Grid service

1 引言

Internet 和电子商务实践的发展,带动了大量虚拟组织的出现,在 Internet 范围进行协同作业要求越来越高;另一方面 Web 服务和网格服务的出现,使按需服务成为可能。Web 服务为异构服务的共享和互操作提供了标准,网格为异构的资源共享和协同提供了基础设施[3]。网格环境下服务和资源变化频率高,资源和服务的状态高度异构,传统的基于服务的架构的环境发生了变化[1],本文探讨了在网格环境下,一种智能化的按需服务引擎 CSB(Cyber Service Bus)的设计和实现,CSB 是国家 973 项目"网格环境下空间信息智能服务及应用示范"的一部分,是系统的服务平台,CSB 以一种智能化的方式实现网格环境下的服务管理和组合。本文第 2 节介绍系统的基本假设和总体框架,第 3 节阐述服务发现和查找,第 4 节介绍服务组合原理,第 5 节介绍服务执行引擎,动态服务选择和容错处理,第 6 节介绍原型系统,最后讨论优点和不足。

2 系统框架

服务引擎由服务发现、服务选择、服务组合、服务执行四个主要模块构成,共同支撑服务的搜索、服务组合请求、服务可视化请求,如图 1 所示。

服务发现模地包括服务注册、抓取 WSDL 文件并年取服务信息和服务检索;服务选择模块通过对服务信息建立索引并进行相似度计算,取得候选服务列表;服务组合模块解释用户的组合请求和限制条件,通过匹配引擎获得抽象服务传递给预案生成器,生成的抽象预案经过预案转换适配器成为与执行平台相关的具体的组合预案,如 BPEL[18] 引擎和 COG[19]引擎,它们调度执行具体的最优的可用服务。

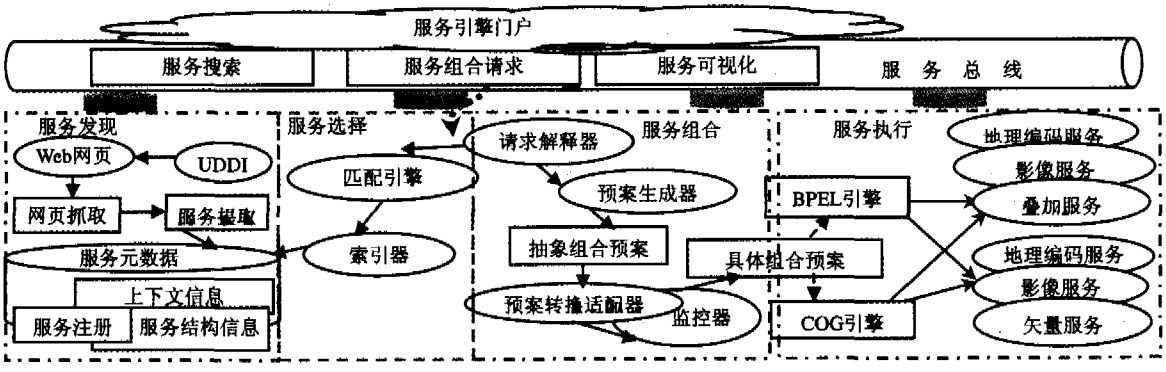


图 1

3 服务发现和选择

服务可分为不可被调用执行的信息服务和可调度执行的计算服务,Web 页面属于前者,Web 服务属于后者,本文仅关注 Web 服务和 Grid 服务。服务发现涉及服务提供者、服务中介者、服务消费者三方面的交互过程,CSB 的服务发现与这

*)国家 973 项目"网格环境下空间信息智能服务及应用示范"(课题编号:2006CB701306)。欧建宏 硕士生,研究领域:分布式计算、数据挖掘、智能 GIS 等;李 琦 教授,博士生导师,研究领域:空间信息科学与技术、数字地球、数字城市等;张学虎 副教授,研究领域:雷达工程、地理编码等;毛 曦 博士研究生,研究方向:人工智能等。

三者对应。

服务提供者通常采用 WSDL 描述服务。WSDL 是一种 XML 文档，它将 Web 服务描述定义为一组服务访问点，客户端可以通过这些服务访问点对包含面向文档信息或面向过程调用的服务进行访问。网格服务目前遵循 OGSA 规范，最新的发展是 WSRF 框架，它是一种有状态的 Web 服务[4]。

服务信息通常发布到一个服务目录以便服务消费者查找调用。传统采用 UDDI 进行服务注册，它存在两方面的问题：中心式 UDDI 容易成为性能瓶颈；存在多个注册中心时服务提供者需要重复注册。在网格中服务信息通过 MDS(Monitoring and Discovery System)对服务和资源建立索引，MDS 之间可以构成层次结构，局部 MDS 中的服务信息向全局 MDS 中汇总。

这两种方式都需要服务在本地进行注册，对于数字城市这样的大范围内应用，需要采用爬虫技术，主动从不同的服务提供者获取各类服务的描述和服务的接口文件(WSDL)，才能适应网格环境下服务高度动态变化的情况，服务内容也更丰富。CSB 的主动服务发现基于 HTTP 协议，采用爬虫技术动态获得 WSDL 文件，并记录服务提供者的信息和 WSDL 的上下文备用，分析器提取服务描述文件得到操作、参数和描述信息。

服务的消费者需要定位到需要的服务，这个过程通常通过浏览分类目录或者查找实现。查找一般有以下方式：通过属性对匹配，基于 schema 模式的匹配，全文检索[5]，前两者均要求对服务本身有一定的先验知识才能实现较好的效果，服务提供者和服务消费者之间通过 Ontology(本体)来获得服务语意的一致理解，但是 Ontology 本身的维护十分困难，不同的领域需要不同的 Ontology。本文结合第一种和第三种方式，以便尽可能地减少对服务先验知识的依赖。通过对服务描述和服务上下文信息建立索引实现全文检索，对服务描述信息提取结构化信息进行模式的匹配实现更精确的服务查找过程，详细算法如下：

服务索引算法

1. 解析 WSDL 文档得到服务元数据 Metadata(input arguments types, output arguments types, keywords, operations)；

2. 计算同类服务，保存可替代服务的索引(算法见第 5 节)；

3. 根据抓取时上下文页面，提取 Context 信息；

4. 对服务描述信息、消息描述信息、上下文 Context 建立倒排索引，建立向量模型[10]；

5. 用户填写语意元数据(提供者，分类…)；

6. 输入服务查询请求，生成查询向量并与服务向量比较，得到候选服务。

4 服务组合

环境的变化经常导致业务逻辑的改变，传统的方式需要开发新的系统进行支撑。在按需服务的框架下，可以通过服务组合快速灵活地形成新的服务实现。服务组合可以分为静态和动态的组合两种，也称为运行前和运行时组合，前者通过在运行之前指定组合中的服务和它们的逻辑关系，运行过程中不再改变，通常需要对服务的先验知识，一般以手动方式形成；后者指运行时根据用户的需求和现有服务形成组合的方案和运行时脚本，可以动态切换，可以自动形成。对应地，可以将组合服务分为具体的组合和抽象的组合服务，具体组合对应的服务是实际存在的服务序列，而抽象组合是一组服务的集合的序列，不具体到单个服务，在运行时刻动态组合形成具体服务。

CSB 采用半自动的服务组合方式，分四阶段形成服务组合方案，如图 2 所示。

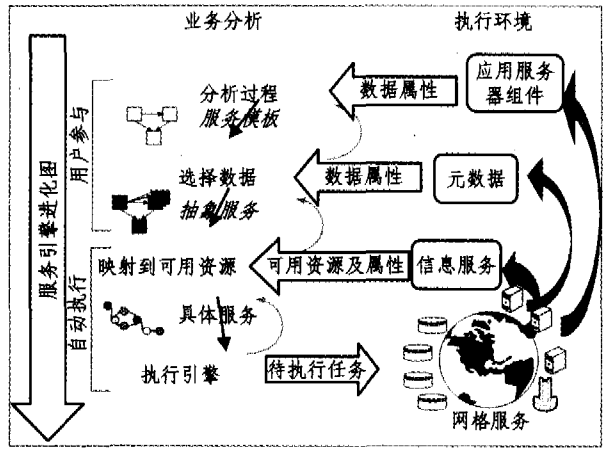


图 2

1. 根据用户服务请求查询，形成候选服务集合(S)；

2. 根据服务集合 S，计算抽象候选方案，用户也可以手动选择服务并进行组合；

3. 根据限制条件映射到可用资源，得到可行预案；

4. 从抽象预案映射到平台相关预案，抽象组合形成具体组合方案并执行。

上述步骤可以半自动实现，下面给出一种基于图论的服务动态组合的算法描述[7]：

1. 根据用户请求，检索得到候选服务列表 S；

2. 生成图 $G(S,L)$，其中 S 为图的节点集合，L 为对应边的集合。检索得到服务元数据 Metadata(input, output arguments types, keywords, operations)计算 S 中服务交互集合 $L=\{L_{i,j}\in L$ 如果 S_i 的输出与 S_j 的输入兼容$\}$，由 S_i 与 S_j 类别和相似度矢量决定(第 5 节)；

3. 根据限制条件(如 CPU 负载、内存大小…)，从 S 中计算选择符合条件的 S'；从 L 选择 L'，得到图 $G'(S',L')$，并生成预案 plan(s)：

a. 指定开始服务 S_{begin} 和结束服务 S_{end}

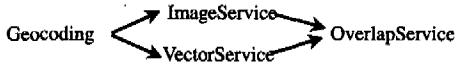
b. 计算每个服务调用的成本 Cost(L')

c. 计算 $G'(S',L')$ 中从 S_{begin} 到 S_{end} 的所有路径 path(s)，得到预案 plan(s)，并计算对应成本 cost(s)，根据成本排序；

4. 生成具体组合：根据用户偏好和引擎状态，生成对应到 BPEL 和 COG 的脚本，提交执行。

基于上述算法，在原型系统中进行检验，输入“地名 影像 矢量 叠加”，得到候选服务列表如下：

给出其他限制条件的情况下，上面的例子直接计算得到可行预案结果：



Geocoding 的输出为 String 型，且输出描述“坐标”与影像服务和矢量服务的输入描述“坐标”高度匹配；同样 Overlap 的服务输入为两个 String，且为影像图和矢量图，这正好与 ImageService 和 VectorService 的输出高度匹配。组合后，用

户输入“北京大学”，通过组合服务，得到北京大学周边的影像与矢量叠加地图。

服务操作名称	输入参数名称	输入参数类型	输出参数名称	输出参数类型	描述
地理编码 Geocoding	Address(地名)	String	Coordinate 坐标对字符串	String	输入地名，转换为经纬度坐标对字符串 x,y
影像服务 ImageService	Coordinate,scale (坐标对字符串，比例尺)	String,int	imageURL 影像地图	String	获得坐标点周围，sacle 比例尺的影像地图
矢量服务 VectorService	Coordinate,scale,buffersize (坐标对字符串，比例尺，缓冲区大小)	String,int,int	vectorURL 矢量地图	String	获得比例尺 sacle，缓冲区 buffersize 的矢量地图
叠加服务 OverlapService	imageURL,vectorURL (影像图，矢量图)	String,String	叠加图	String	实现影像与矢量的叠加

5 服务执行引擎

在执行组合服务之前，预案转换适配器从抽象生成具体预案，目前系统提供了两种适配器，从抽象服务转换为 BPEL 引擎脚本和 COG 引擎脚本。

5.1 执行引擎

COG 引擎采用 Globus 工具中的 Karajan 引擎，不再赘述，下面介绍 BPEL 引擎的工作原理，如图 3 所示。

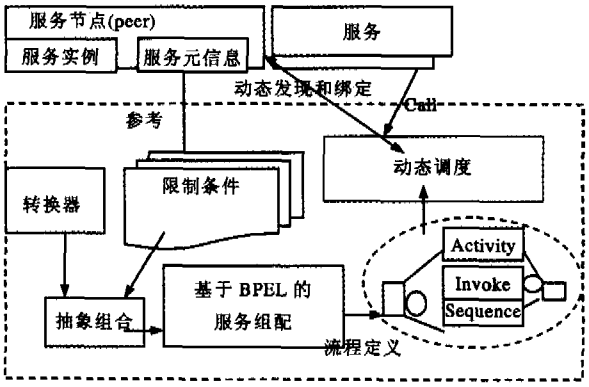


图 3

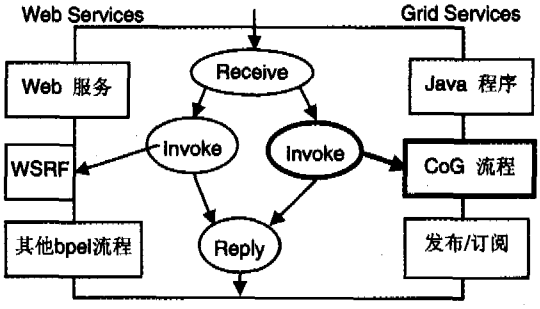


图 4

BPEL 规范并不直接支持网格服务和 WSRF 标准，为了能够无缝集成网格服务，我们对 BPEL 进行了扩展，加入 WSRF 元素和 COG 的元素，以一种统一的方式支持 Web Service 和 WSRF Service 的组合。

扩展后的 BPEL 元素支持基本的 WSRF 服务以及网格基础设施的 gridftp、cog、订阅等机制，同时增加了对 java 类的支持，以充分发挥平台共享计算力的优势。

对扩展的 BPEL 规范建模，目前采用 BPEL 模型树和 BPEL 实例执行环境两者结合的方式，模型树是静态的，执行环境动态生成，实际运行系统中采用对等网络分布化 BPEL 引擎，以避免瓶颈[1,2]。

5.2 服务动态选择和容错

网格环境下，服务经常发生变化，不断有新的服务加入和老的服务退出或者暂时不能使用，服务组合的抽象服务生成到具体服务时，需要根据新的情况选择具体服务；具体服务执行过程中，如果发现原有的服务已经失效，系统动态切换到新的替代的服务。本节讨论服务的相似性、服务的选择以及服务的替代算法。

5.3 服务的相似性

服务的相似性我们指服务的操作(operation)的相似性[6]，因为操作才是可被调用的单元，记操作 1 和操作 2 的相似性为 S(O1,O2)，由输入输出类型和名称、服务描述、输入输出消息描述、操作名称、服务描述文件上下文决定，函数表达如下 S(O1,O2)=f(input types,output types,Service description,input message,output message,operation name,context)，具体算法如下：

1. 计算输入和输出类型匹配程度 s1,s2；

2. 对服务描述、输入输出消息的描述和名称切词，分别建立向量模型，计算各自的相似程度 s3,s4,s5；

3. 计算抓取得到的上下文信息的向量，得到相似度 s6；

4. 计算服务相似度

S(O1,O2)=a*s1+b*s2+c*s3+d*s4+e*s5+f*s6(a,b,c…为权重，可调整)，并根据相似度排序。

5.4 服务的选择

服务选择分为两个阶段，第一阶段通过服务语意信息识别；第二阶段通过分析结构化的 wsdl 文件，得出细致的分析，具体步骤如下：

1. 根据抽象服务的描述信息，通过语意识别(关键词向量)得到候选的服务 S1；

2. 根据 WSDL 文件计算候选服务的相似服务，得到可替代服务集合 S2；

3. 根据 S2 中服务的实时监控信息(负载，网络，内存等)和用户偏好，给出代选服务排序；

4. 执行具体组合服务；

5. 如果有服务不可用，切换到候选服务。

6 原型系统

原型系统基于 Globus 和 COG 的 Karajan 工作流引擎以及自己开发的基于 P2P 模式的 BPEL 引擎[2]，是国家 973 项目“网格环境下空间信息智能服务及应用示范”的一个部分，提供对网格环境下空间服务注册、管理、组合等功能，系统中管理的 Web 服务和 Grid 服务 100 个左右。不仅支持手工方式生成预案，也动态生成服务组合；同时对服务状态和组合服务的执行过程提供监控功能，图 5 是执行过程监控的界面，服务节点以可视化的方式展示，并且可实时监控组合服务的运

行状态。

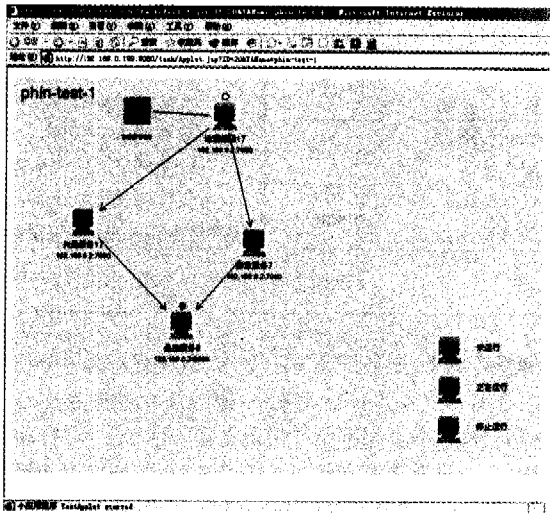


图 5

服务引擎支持数字城市公共卫生应急响应系统、禽流感应急处置系统的高效构建和运转，图 6 是公共卫生应急处置系统执行界面，显示了通过组合服务得到禽流感地区的隔离区域的示意效果。

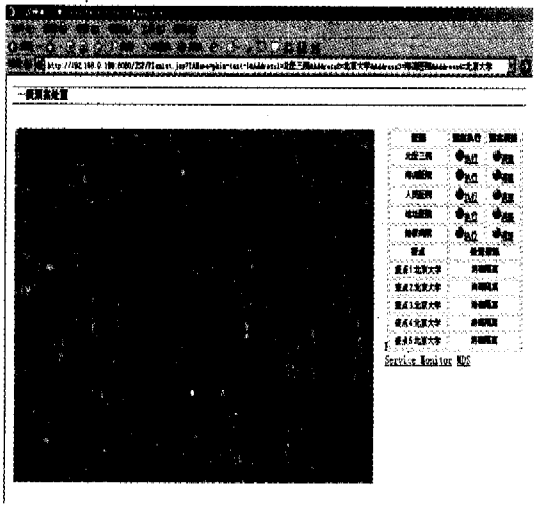


图 6

结论 服务组合在学术界和工业界都得到了很多重视，工业界主要通过 WSDL 结合 BPEL 标准来解决组合问题。学术界出现很多在工作流机制和语意网描述服务语意方面的研究[8~11]，如 Resource Definition Framework (RDF)[13]，OWL-S[14]，也有通过服务语意和目标生成计划得到可执行的 BPELS[12]。网格领域 WebFlow[16] 和 DAGMan[17]，COG 从不同的方面涉及到服务的组合。SWORD[15] 提出一种基于规则的服务组合系统。

本文提出一种在网格环境下建立按需空间服务引擎的技术框架和关键算法，以信息抽取和信息检索为方法，以图论为理论基础，进行服务的自动组合和动态选择，并且在实际应用中得到检验。目前的自动组合算法是在小规模范围内有效，为了增加系统的稳定性和可靠性，需要对服务的相似性和可替换性进行更多的语意理解，建立本体进行推演。另一方面，服务信息现在仅限于 Web 服务和 Grid 服务，对于非结构化的信息服务没有涉及，利用非结构化信息提取技术，可以作为服务组合的数据来源。

参 考 文 献

1 史文勇，李琦．基于 P2P 模式的数字城市服务模式及平台设计．计算机科学，2005

2 史文勇，李琦，林宇．基于 P2P 面向 SOA 的数字城市服务总线设计．计算机科学，2005

3 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，2004，15

4 Foster I, Kesselman C, Nick J, Tuecke S. The Physiology of the Grid: An Open Grid Services Architecturefor Distributed Systems Integration. In OpenGrid Service Infrastructure WG, Global Grid Forum, June 2002

5 Ahmed R, Boutaba R, Cuervo F, Iraqi Y, Li D, Limam N, Xiao J, Ziembicki J. Service Discovery Protocols: A Comparative Study. In: Proceedings of IM, May 2005

6 Dong X, Halevy A, Madhavan J, Nemes E, Zhang J. Similarity Search for Web Services. In: Proc. of VLDB, 2004

7 Agarwal M, Parashar M. Enabling Autonomic Compositions in Grid Environments. In: Proceedings of the 4th International Workshop on Grid Computing Grid, 2003

8 Su X, Rao J. A Survey of Automated Web Service Composition Methods. In: Proceedings of First International Workshop on Semantic Web Services and Web Process Composition, July 2004

9 Hess A, Kushmerick N. Learning to attach semantic metadata to Web services. In ISWC, 2003

10 Deerwester S C, Dumais S T, Landauer T K, Furnas G W, Harshman R A. Indexing by latent seman-tic analysis. JASIS, 1999, 41(6): 391~407

11 Carman M, Serafini L, Traverso P. Web Service Composition as AI Planning a Survey. ICAPS 2003 Workshop on Planning for Web Services, 2003

12 Traverso P, Pistore M. Automated Composition of Semantic Web Services into Executable Processes. In: 3rd Int. Semantic Web Conf., November 2004

13 Resource Description Framework (RDF). http://www.w3.org/RDF/

14 OWL-S: Semantic Markup for Web Services. http://www.daml-s.org/owl-s/1.0/owl-s.html

15 Ponnekanti S, Fox A. SWORD: A Developer Toolkit for Web Service Composition. In: Proc. of the 11th International World Wide Web Conference, 2002

16 Bhatia D, Burzevski V, Camuseva M, Fox G, Furmanski W, PremChandran G. WebFlow : A Visual Programming Paradigm for Web/Java BasedCoarse Grain Distributed Computing. Concurrency: Practice and Experience, 1997, 9(6): 555~577

17 Frey J, Tannenbaum T, Foster I, Livny M, Tuecke S. Condor-G: A Computation Management Agent for Multi-Institutional Grids. In: Proceedings of the 10th IEEE Symposium on High Performance Distributed Computing (HPDC10), San Francisco, CA, August 2001. 7~9

18 [BPEL4WS]. Business Process Execution Language for Web Services Version 1.1. http://www-106.ibm.com/developerworks/Webservices/library/ws-BPEL/

19 von Laszewski G, Foster I, Gawor J, Lane P. A Java commodity grid kit. Concurrency and Computation Practice and Experience, 2001