一种基于角色的分布式动态服务组合方法*

刘必欣+, 王玉峰, 贾 焰, 吴泉源

(国防科学技术大学 计算机学院,湖南 长沙 410073)

A Role-Based Approach for Decentralized Dynamic Service Composition

LIU Bi-Xin+, WANG Yu-Feng, JIA Yan, WU Quan-Yuan

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

+ Corresponding author: Phn: +86-731-4575821, Fax: +86-731-4512504, E-mail: bxliu@nudt.edu.cn, http://www.nudt.edu.cn

Received 2004-10-20; Accepted 2005-06-02

Liu BX, Wang YF, Jia Y, Wu QY. A role-based approach for decentralized dynamic service composition. *Journal of Software*, 2005,16(11):1859−1867. DOI: 10.1360/jos161859

Abstract: Web service composition is an important technology for agile inter-enterprise application integration. Centric composition engines are widely adopted to enact composite services in many Web service composition research projects. Such a centralized architecture, however, results in problem of scalability, message exchange efficiency and autonomy. A role-based decentralized approach for service composition is proposed in this paper, which partitions the global process model of a composite service into local process models according to the participant roles so as to distribute the control logic of a composite service and corresponding execution load into multiple nodes. An algorithm of generating the local process models is presented in detail with deployment and execution mechanism introduced. Simulation results indicate that this approach can support highly concurrent requests and large volume of data more effectively than the centralized architecture, so it is helpful to improve the scalability of composite services.

Key words: Web service; dynamic service composition; role; global process model; local process model; model partition

摘 要: 服务组合是开放环境中实现跨组织敏捷应用集成的重要技术.许多研究采用集中的服务组合引擎管理组合服务的执行,在系统的可伸缩性、消息传输效率及自治性等方面存在局限.针对集中结构的上述问题,提出一种基于角色的分布式动态服务组合方法,通过划分组合服务的全局流程模型产生各个角色的本地流程模型,从而使得组合服务的控制逻辑及执行负载能够对等地分布到多个结点.讨论了本地流程模型的生成算法及部署与执行机制.模拟实验结果表明,与集中式结构相比,该方法能够更有效地支持大规模并发访问以及大数据量的消息转输,有助于提高组合服务的可伸缩性.

关键词: Web 服务;动态服务组合;角色;全局流程模型;本地流程模型;模型分解

* Supported by the National Natural Science Foundation of China under Grant No.90412011 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant Nos.2003AA115210, 2004AA112020 (国家高技术研究发展计划(863))

作者简介: 刘必欣(1977－),女,江苏建湖人,博士生,主要研究领域为中间件,工作流,Web 服务技术;王玉峰(1977－),男,博士生,主要研究领域为分布计算与应用服务器技术;贾焰(1960－),女,博士,教授,博士生导师,CCF 高级会员,主要研究领域为数据库,分布式计算;吴泉源(1942－),男,教授,博士生导师,主要研究领域为人工智能与分布式计算.

© 中国科学院软件研究所 http://www.jos.org.cn

中图法分类号: TP311 文献标识码: A

随着 Web 服务技术的发展,面向服务的计算[1]逐渐成为开放异构环境中复杂分布应用的主流计算模型,基于 Web 服务的动态服务组合[2]作为实现灵活、快速集成的重要方法,成为新的研究热点.Web 服务组合是指为了特定的业务目标将多个独立自治的 Web 服务(称为基本服务)按照其语义及逻辑关系“拼装”起来,以实现高层次的功能聚合(称为组合服务).目前已有大量研究工作从不同角度切入这一主题,就服务组合的建模与执行控制技术而言,许多研究继承来自工作流领域的成果,采用集中式的体系结构[3,4],即位于系统中心位置服务组合引擎负责根据组合服务模型调度各个基本服务依照指定的执行顺序和条件执行,并在它们之间路由数据.然而不同于传统工作流管理系统所面对的企业内部可控环境中的流程调度问题,服务组合面向开放的互联网环境,采用集中结构将存在如下问题:首先,由于参与组合的基本服务数目以及客户并发请求规模均具有难以预知的特征,集中的体系结构在可伸缩性及可用性等方面存在局限性[5];其次,集中结构中基本服务之间需经过中心引擎路由数据,增加了不必要的网络开销和处理时间[6];再者,某些应用(如 B2B 应用)出于商业和安全的目的不允许业务数据流经第三方结点.

本文提出一种基于角色的分布式动态服务组合方法,将组合服务的全局流程模型分解为可分布部署的本地流程模型,从而使得组合服务的控制逻辑能够依据执行活动的角色分布到多个结点;在执行时,这些结点之间建立对等的协作关系,并进行直接的数据交换,避免了单个结点成为系统性能瓶颈和通信中心.模拟实验结果表明,与集中式结构相比,该方法具有更好的可伸缩性,更适合 Internet 范围内大规模的动态服务组合应用.

1 基于角色的分布式服务组合模型

角色是组合服务建模阶段对组合服务参与者的抽象,代表一类能够完成特定功能的具体服务.以移动支付电子文献检索服务为例,该组合服务由移动通信运营商(如中国移动、联通)、中国银联、文献检索内容提供商(如万方、维普)的相关服务组成,则该组合中涉及 4 个角色:移动支付文献检索服务、移动支付服务、银联服务和文献检索服务.由于组合服务本身亦具有独立的提供商,并与其他基本服务之间存在通信关系,因此我们将组合服务本身也视为一个特殊的角色,因此,包含 N 个基本服务的组合服务中具有 $N+1$ 个角色.在组合服务执行过程中,抽象的角色被绑定到具体的服务提供商,从而确定扮演各个角色的基本服务.在组合引擎的协调下,基本服务依据组合服务模型执行相应操作,并通过消息交互实现基本服务之间的状态同步和数据交换.因此,组合服务是多个参与者之间分工协作的过程,每个参与者在组合中扮演特定角色,承担特定的责任.

基于角色的分布式动态服务组合方法基于上述对于组合服务的理解,在建模方法上采用两个层次的模型来描述组合服务:全局流程模型(global process model)定义组合服务各个活动之间整体的约束关系,与常规的建模方法基本一致;而本地流程模型(local process model)则旨在描述组合中某角色的单方行为,所有角色的本地流程模型构成了该组合的分布模型;由全局流程模型可以导出各个角色的本地流程模型.在系统运行结构上,系统中不存在集中控制的组合引擎,运行时刻组合中的角色绑定到具体服务提供商结点,多个结点依据分布的本地流程模型共同协作执行组合服务,每个结点均控制一部分组合服务的执行过程,从而将调度与通信责任布到多个参与者结点,它们在系统中的位置是对等的.

1.1 全局流程模型

全局流程模型定义组合中的活动及其时序关系与通信关系:时序关系构成由基本服务执行的活动之间的控制依赖,通信关系则构成活动的数据依赖.本文采用基于有向图的方法描述服务之间的控制依赖与数据依赖,有向图的结点表示活动,边(分为控制连接子与数据连接子)表示活动之间的控制流和数据流.

定义 **1**(角色(**role**)). 角色抽象地表示组合服务的参与者,定义为二元组 $Role::=(rid,OP)$,其中 rid 为角色标识,OP 表征角色的能力,即支持的操作集合.

定义 **2**(流程变量(**process variable**)). 流程变量是流程使用的消息文档,表征流程的全局状态,定义为向量 $Pvar::=(el_1,el_2,\ldots,el_n)$,其中 $el_i(i=1,\ldots,n)$为消息分量.

© 中国科学院软件研究所 http://www.jos.org.cn

定义 **3**(活动(**activity**)). 活动表示组合服务的一个步骤,定义为四元组 *Activity*::=(*role*,*op*,*input*,*output*),其中 *role* 为执行活动的角色;*op*∈*role*.*OP* 为活动执行的操作;*input*,*output* 分别为活动输入变量和输出变量.

定义 **4**(全局流程模型(**global process model**)). 全局流程模型定义为 *GPM*::=(*ROLES*,*PVS*,*ACTS*,*CLS*,*DLS*),其中:

(1) $ROLES=\{R_0,R_1,\ldots,R_N\}$为角色集合,其中 R_0 表示组合服务本身,$R_1,\ldots,R_N$ 表示组合服务的 N 个参与者;

(2) *PVS* 为流程变量集合,$I,O\in PVS$ 分别为流程模型的输入变量和输出变量;

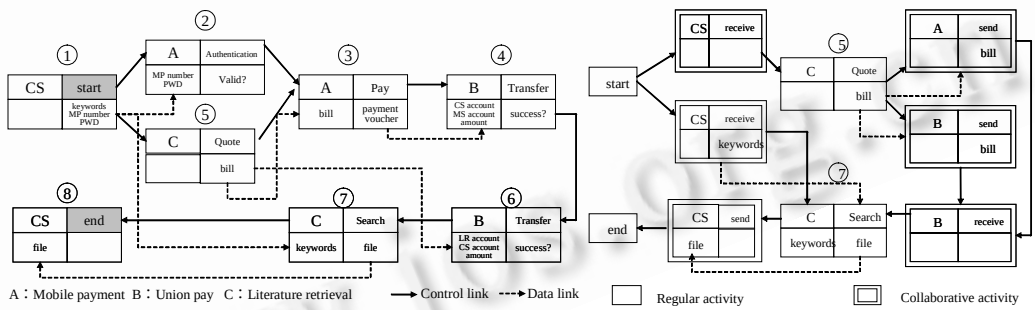
(3) *ACTS* 为活动集合,其中 *start* 和 *end* 为两个特殊的活动,分别表征流程的开始与结束,$start=\langle R_0,null,null,I\rangle$,$end=\langle R_0,null,O,null\rangle$;

(4) $CLS\subseteq ACTS\times ACTS\times C$ 为控制连接子(control link)的集合.对于 $e=\langle A_i,A_j,c\rangle\in CLS$,$A_i$ 与 A_j 分别为 *e* 的源活动与目标活动,定义在 $A_i.output$ 上的布尔函数 *c* 为转移条件,并记 $A_i\prec_c A_j$(如果不关心 *c*,则简记为 $A_i\prec A_j$);其语义是,源活动完成后,若 *c* 被满足,则激活控制连接子;

(5) $DLS\subseteq ACTS\times ACTS\times\Phi$为数据连接子(data link)的集合.对于 $d=\langle A_i,A_j,\phi_{i,j}\rangle\in DLS$,数据映射函数 $\phi_{i,j}$ 定义为从 $A_j.input$ 到 $A_i.output$ 的部分函数,并称 A_j 关于 $\phi_{i,j}$ 数据依于 A_i,记作 $A_i\triangleright^{\phi_{i,j}}A_j$;数据连接子表示活动之间的数据流,若 $\phi_{i,j}(el_m)=el_n$ ($el_m\in A_j.input$, $el_n\in A_i.output$),则该数据连接子的语义是,A_j 的输入变量的数据分量 el_m 的值来自于活动 A_i 的输出变量的数据分量 el_n;如果 $A_j.input=A_i.output$,则 $\phi_{i,j}$ 为恒等映射 ϕ_I.

定义 **5**(路径(**path**)). 在 GPM 中若有活动序列 $A_1,\ldots,A_n(n\geq 0)$ 使得 $A_i\prec A_1\prec\ldots\prec A_n\prec A_j$,则称在 GPM 中 A_i 可达 A_j,记作 $A_i\rightarrow A_j$,并称($A_i,A_1,\ldots,A_n,A_j$)为从 A_i 到 A_j 的一条路径.

如图 1(a)所示为一个移动支付电子文献检索服务的全局流程模型.全局流程模型以全局视角定义了组合服务的参与者、流程结构及数据交换关系,与当前主流的组合服务模型定义思想(如 WSFL 的流模型、BPEL 的可执行流程模型)是一致的.全局流程模型可以作为集中式引擎的输入来控制组合服务的执行过程.该引擎管理活动的时序逻辑和流程的内部状态,基于为对承担某角色的服务的远程过程调用实现活动的执行,从而构造基于集中引擎的组合服务系统.



(a) GPM of literature retrieval by mobile payment service (b) LPM of literature retrieval service

(a) 移动支付电子文献检索服务全局流程模型 (b) 文献检索服务的本地流程模型

Fig.1 Examples of global process model and local process model

图 1 本地流程模型与全局流程模型示例

1.2 本地流程模型

服务组合从另一个角度可以理解为多个角色之间的协同过程,每个角色通过说明自身的动作以及与其他角色的交互来表达它在协同中单方面的行为规约,称为“本地流程模型”,从而提供了另一种方法建模组合服务.角色的本地流程模型主要描述组合服务中某角色在与其他角色协同时的动作规约和消息规约.动作规约说明发生在该角色上动作(执行的操作)以及动作之间的约束关系(操作之间的因果依赖与数据依赖).例如,图 1(a)中的文献检索服务的局部动作是首先计算检索费用,之后执行文献查询.本地流程模型的动作仍建模为活动,这些活动均由该角色执行.消息规约说明该角色与组合中其他角色交互消息的内容及其编排.例如,文献检索内容提

© 中国科学院软件研究所 http://www.jos.org.cn

供商在完成费用计算后需要将检索费用发给移动运营商,以确定支付金额.我们将消息交互也建模为一类活动,称为协同活动.相对于协同活动,定义 3 中的活动称为常规活动.

定义 **6**(协同活动(**collaborative activity**)). 协同活动是表达消息交互语义的活动,定义为 $CActivity::=(role,op,input,output)$,其中 *role* 表示协同对方的角色;$op\in\{send,receive\}$表示协同的交互方向;*input* 表示发送的消息,当 *op= receive* 时 *input* 为空;*output* 表示等待接收的消息,*op=send* 时 *output* 为空.

定义 **7**(本地流程模型(**local process model**)). 角色 *R* 的本地流程模型定义为 $LPM_R::=(ROLES,PVS,ACTS,CLS,DLS)$,其中 *ACTS* 划分为两个不相交的子集 $ACTS_A$ 与 $ACTS_C$,分别为常规活动集合与协同活动集合,$\bigcup_{v\in ACTS_A} v.role=\{R\}$ 且 $\bigcup_{v\in ACTS_C} v.role = ROLES\setminus\{R\}$; PVS,CLS 和 DLS 的定义与全局流程模型相同.

定义 **8**. 给定角色 *R* 的本地流程模型 LPM_R 与全局流程模型 GPM,如果下面的关系成立,则称 LPM_R 为角色 *R* 关于 GPM 的本地流程模型.

(1) $LPM_R.ROLES\subseteq GPM.ROLES$;

(2) $LPM_R.PVS\subseteq GPM.PVS$;

(3) $LPM_R.ACTS_A=\{v\mid v\in GPM.ACTS\text{且}v.role=R\}\cup\{start,end\}$;

(4) $\forall A_i,A_j\in GPM.ACTS$ 且 $A_i\prec A_j$,如果 $A_i,A_j\in LPM_R.ACTS_A$,则在 LPM_R 中有 $A_i\prec A_j$;如果 $A_i\in LPM_R.ACTS_A$, $A_j\notin LPM_R.ACTS_A$,则 $\exists A_k\in LPM_R.ACTS_C$, $A_k.role=A_j.role$ 使得 $A_i\prec A_k$;如果 $A_i\notin LPM_R.ACTS_A$, $A_j\in LPM_R.ACTS_A$,则 $\exists A_l\in LPM_R.ACTS_C$, $A_l.role=A_i.role$ 使得 $A_l\prec A_j$;

(5) $\forall A_i,A_j\in GPM.ACTS$ 且 $A_i\rhd^{\phi_{i,j}}A_j$,如果 $A_i,A_j\in LPM_R.ACTS_A$,则在 LPM_R 中有 $A_i\rhd^{\phi_{i,j}}A_j$;如果 $A_i\in LPM_R.ACTS_A$, $A_j\notin LPM_R.ACTS_A$,则 $\exists A_k\in LPM_R.ACTS_C$, $A_k.role=A_j.role$ 且 $A_i\rhd^{\phi_{i,k}}A_k$;如果 $A_i\notin LPM_R.ACTS_A$, $A_j\in LPM_R.ACTS_A$,则 $\exists A_k\in LPM_R.ACTS_C$, $A_i.role=A_i.role$ 且 $A_i\rhd^{\phi_{i,j}}A_j$.

条件(1)和条件(2)要求 LPM_R 涉及的角色和流程变量是 GPM 中定义的角色和流程变量.条件(3)说明 LPM_R 中定义常规活动为 GPM 中所有由 *R* 执行的活动.条件(4)和条件(5)给出 LPM_R 与 *GMP* 中控制流和数据流的一致性要求,即 GPM 中由 *R* 执行的活动之间的时序关系和数据依赖在 LPM_R 中仍然成立;若 GPM 中两个活动不由同一角色执行,则 LPM_R 中相应地存在协同活动在角色之间传递控制和数据.

如图 1(b)所示为"文献检索服务"角色关于如图 1(a)所示的全局流程模型的本地流程模型.

关于组合服务全局流程模型的本地流程模型定义了组合中与特定角色相关的流程状态、活动时序、数据依赖以及该角色与其他角色之间的控制传递和数据传递,且与全局流程模型是一致的.因此,本地流程模型定义了组合服务的一个刻面,是全局流程模型在特定角色上的投影.由于本地流程模型只涉及单个服务的行为控制,可以在服务的所在地独立地被解释执行,从而构成了服务结点之间的相互协作的行为脚本,为组合服务的控制逻辑分布到各个服务结点提供了可能.

1.3 本地流程模型产生算法

从组合服务的全局流程模型可以推导每一个角色关于该全局流程模型的本地流程模型.本节介绍本地流程模型的算法 Partition.其基本思想是:按照执行活动的角色将输入的组合服务全局流程模型中的活动划分为多个子集,子集内部的活动之间保持原有的控制连接子和数据连接子;若控制连接子或数据连接子跨越子集边界,则引入成对的协同活动进行解耦,并重构跨越子集边界的控制连接子和数据连接子.算法描述如下:

算法. Partition.

输入:包含 $N+1$ 个角色$\{R_0,R_1,\ldots,R_N\}$的全局流程模型 $GPM=(ROLES,PVS,ACTS,CLS,DLS)$;

输出:$N+1$ 个本地流程模型 $LPM_{R0},LPM_{R1},...,LPM_{RN}$.

步骤:

(1) 依次考查 GPM 中的控制连接子 $e_{i,j}=\langle A_i,A_j,c\rangle$,若 $A_i.role\neq A_j.role$,,则创建一对协同活动 $N^!_{i,j},N^?_{i,j}$,其中 $N^!_{i,j}.role=A_j.role$, $N^!_{i,j}.op=send$, $N^?_{i,j}.role=A_i.role$, $N^?_{i,j}.op=receive$;创建控制连接子 $\langle A_i,N^!_{i,j},e_{i,j}.c\rangle$,$\langle N^?_{i,j},A_j,T\rangle$ 和 $\langle N^!_{i,j},N^?_{i,j},T\rangle$,删除 $e_{i,j}$.

© 中国科学院软件研究所 http://www.jos.org.cn

(2) 依次考查 GPM 中的数据连接子 $d_{i,j}=\langle A_i,A_j,\phi_{i,j}\rangle$, 若 $A_i.role\neq A_j.role$:

(2.1) 如果 A_i 到 A_j 的路径上有常规活动,则创建一对协同活动 $N^!_{i,j}$ 与 $N^?_{i,j}$,其中 $N^!_{i,j}.role=A_j.role$, $N^!_{i,j}.op=send$, $N^?_{i,j}.role=A_i.role$, $N^?_{i,j}.op=rec$,并创建控制连接子 $\langle A_i,N^!_{i,j},e_{i,j}.c\rangle$,$\langle N^?_{i,j},A_j,T\rangle$ 和 $\langle N^!_{i,j},N^?_{i,j},T\rangle$; 否则,记 A_i 与 A_j 的路径上的一对协同活动为 $N^!_{i,j}$ 与 $N^?_{i,j}$.

(2.2) 令 $N^!_{i,j}.input=N^?_{i,j}.output=A_j.input$,创建数据连接子 $\langle A_i,N^!_{i,j},\phi_{i,j}\rangle$ 及 $\langle N^?_{i,j},A_j,\phi_I\rangle$, 删除 $d_{i,j}$.

(3) 依次考查每一对常规活动 A_i 与 A_j,如果 $A_i.role=A_j.role$,A_i 可达 A_j 且它们之间的路径上不存在常规活动 A_k 使得 $A_i.role=A_k.role$,则对从 A_i 到 A_j 的路径上的协同活动按照它们在路径中出现的顺序进行排序,分别记为 $N^!_0,N^?_0,...,N^!_p,N^?_p$, 创建控制连接子 $\langle N^!_o,N^?_p,T\rangle$.

(4) 删除每一对相邻的协同活动 $N^!_{i,j}$ 与 $N^?_{i,j}$ 之间的控制连接子 $\langle N^!_{i,j},N^?_{i,j},T\rangle$.

经上述步骤,由组合服务全局过程模型被分割为 N+1 个子图,记为 $(V_i,E_i,D_i),i=0,...,N$. 在此基础上构造各个角色的本地流程模型 LPM_{Ri}=($ROLES_i$,PVS_i,$ACTS_i$,CLS_i,DLS_i),i=0,…,N,其中:

(1) $ROLES_i=\{v.role\mid v\in V_i\}$;

(2) $PVS_i=\{v.input,v.output\mid v\in V_i\}$;

(3) $ACTS_i=\begin{cases}V_i, & i=0\\ V_i\cup\{start_i,end_i\}, & i=1,...,N\end{cases}$;

(4) $CLS_i=\begin{cases}E_i, & i=0\\ E_i\cup\{\langle start_i,v,T\rangle\mid v\in V_i\setminus ACTS,v.op=rec\text{且}\neg\exists e\in E_i,target(e)=v\}\cup & \\ \quad\{\langle v,end_i,T\rangle\mid v\in V_i\setminus ACTS,v.op=send\text{且}\neg\exists e\in E_i,source(e)=v\}, & i=1,...,N\end{cases}$;

(5) $DLS_i=D_i$.

定理 **1**. 算法 Partition 的输出 LPM_{Ri} 是 *Ri* 关于输入 GPM 的本地流程模型.

证明:即证明 LPM_{Ri} 满足定义 7 和定义 8 即可,详细证明略. □

定义 **9**. 全局流程模型或本地流程模型(*ROLES*,*PVS*,*ACTS*,*CLS*,*DLS*)是结构正确的,如果以下条件满足:(1) 每个活动均位于一条从 *start* 到 *end* 的路径上;(2) 对于任意 $A_i\triangleright^{\phi_{i,j}}A_j$, $A_i\to A_j$.

定理 **2**. 如果输入 GPM 是结构正确的,算法 Partition 产生的 LPM_{Ri}(i=0..N)是结构正确的且为 GPM 的等价分解.

证明: LPM_{Ri} 的结构正确性.由 GPM 是结构正确的得到图(*ACTS*,*CLS*)为连通图.由于算法 Partition 在处理控制连接子时不会添加孤立的协同活动,也不改变活动之间的可达性,因此图 $(V_i,E_i)(i=0,...,N)$ 亦为连通图,而 LPM_{Ri} 中控制连接子的构造过程保证每个活动都位于从 *start* 到 *end* 的一条路径上(详细证明略).另一方面,对于 LPM_{Ri} 中的数据连接子 $d_{i,j}=\langle A_i,A_j,\phi_{i,j}\rangle$, 如果 A_i 与 A_j 均为常规活动,则 $d_{i,j}$ 也是 GPM 的数据连接子,由 GPM 结构正确得到在 GPM 中有 $A_i\to A_j$,由本地流程模型的定义得到在 LPM_{Ri} 中 $A_i\to A_j$ 也成立;否则,由于算法只在有控制依赖的协同活动与常规活动之间创建数据连接子,因此 $A_i\to A_j$ 亦成立.定义 9 的两个条件满足,结构正确得证.

等价性.令 GPM′ 为算法步骤(3)得到的模型,只需证明 GPM 与 GPM′等价即可.首先证明控制流等价,即证 $\forall A_i,A_j\in GPM.ACTS$,在 GPM 中 $A_i\to A_j$ 成立当且仅当 GPM′中 $A_i\to A_j$ 成立:根据 GPM′的生成过程,充分性成立是显然的;事实上,算法仅在步骤 (2.1) 中增加了常规活动之间的路径,但是由于步骤(2.1)仅在数据依赖的活动之间增加控制路径,GPM 的结构正确保证有数据连接子的活动之间必存在路径,因此步骤 (2.1) 重构数据连接子加入的控制连接子不会为 GPM′增加新的可达关系,因此如果在 GPM′中活动 A_i 可达 A_j ,那么必然在 GPM 中它们也可达,必要性得证.接下来证明数据流等价.由 GPM′的构造过程可知,GPM 中的数据连接子 $\langle A_i,A_j,\phi_{i,j}\rangle$ 被 GPM′中数据连接子 $\langle A_i,N^!_{i,j},\phi_{i,j}\rangle$ 和 $\langle N^?_{i,j},A_j,\phi_I\rangle$ 取代,由 $\phi_{i,j}=\phi_{i,j}\circ\phi_I$ 得到 $\forall el\in A_j.I$, $\phi_{i,j}(el)=(\phi_{i,j}\circ\phi_I)(el)$,即对于任意常规活动而言,其输入变量的值在 GPM 和 GPM′中均来自相同的流程变量.数据流等价得证. □

© 中国科学院软件研究所 http://www.jos.org.cn

下面举例说明算法 Partition 的执行过程.为了直观表示每个活动的执行者,对图 1(a)重新布局,如图 2(a)所示;图 2(b)中描绘了经过步骤(1)的控制连接子的变化情况;图 2(c)示意了步骤(2)对结点 1 与结点 7 之间数据连接子的处理;图 2(d)说明了经步骤(3)、步骤(4)后最终形成的各个本地流程模型.

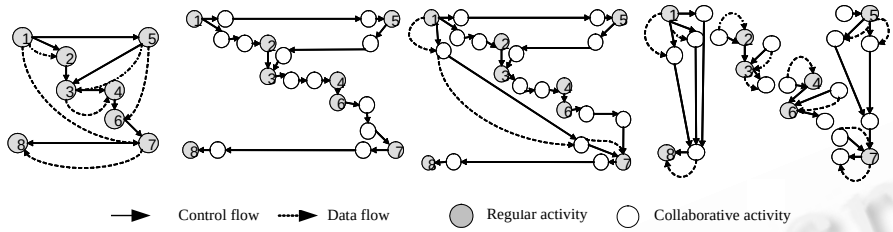


Fig.2 Example of generating LPM from GPM

图 2 本地流程模型生成示例

2 基于角色的分布式动态服务组合系统

基于角色的分布式服务组合系统模型如图 3 所示.服务集合Σ部署运行在N个自治的结点上,每个结点运行一个组合管理器,负责管理本结点上部署的服务、解释流程模型、管理流程实例、调度本结点上的服务操作,并通过 SOAP 消息与其他结点的组合管理器交换执行状态和业务数据.它们构成了一个服务提供网络.网络中每个结点都可以发起一个组合服务,成为组合服务的提供者,同时也可以作为参与者参与其他结点所发起的组合服务.

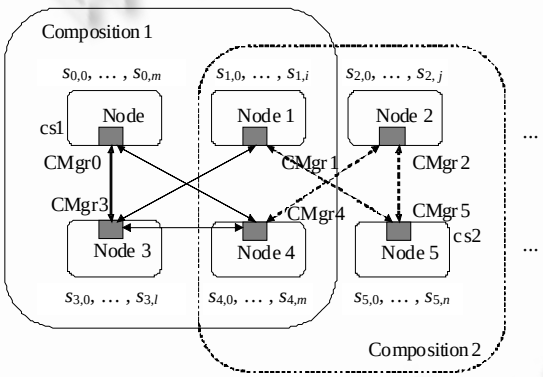


Fig.3 Decentralized service composition system

图 3 分布式服务组合系统

基于角色的分布式动态服务组合的实施由建模、部署与执行这 3 个阶段组成.

在建模阶段,组合服务的开发者在建模工具的辅助下开发组合服务的全局流程模型:确定参与组合的各个角色及各个角色所承担的活动,设计活动之间的控制约束与数据交换关系.之后,工具执行第 1.3 节算法 Partition 自动生成各个角色的本地流程模型.

在部署阶段,系统首先根据某些业务/系统指标选取服务 $s_1,...,s_N \in \Sigma$ 为角色 $R_1,...,R_N$ 的服务实例,确定 $s_1,...,s_N$ 的宿主结点的组合管理器 $CMgr_j(j=1,...,N)$;同时,确定组合服务的宿主结点 $Node_0$ 及其组合管理器 $CMgr_0$;进而,将各个角色的本地流程模型分发给相应的组合管理器,并为各个组合管理器初始化角色端点映射表 $\{R_i \to CMgr_i(j=0,...,N)\}$.

部署完成后,组合服务进入就绪状态.当请求到达组合服务宿主结点 $Node_0$ 时,$CMgr_0$ 载入 LPM_{R0} 并执行.在 LPM_{R0} 的执行过程中,$CMgr_0$ 与其他组合管理器的通信将依次激活 $CMgr_j$(j=1,…,N)解释执行 LPM_{Ri}(i=1,…,N).流程模型的解释执行采用事件驱动的模式.在组合服务的执行过程中,各个结点都存储一部分组合服务的模型信息,承担一部分调度任务并维护一部分组合服务的状态.它们在系统中的地位是对等的,其行为是协作的.

3 性能模拟

我们对如图 1 所示的组合服务在集中式结构和基于角色的分布式结构下的性能进行了模拟.在服务空间中存在两个移动支付服务、两个文献检索服务和一个银联服务.当组合服务执行时,以相等的概率分别在移动支付服务和文献检索服务之间进行选择.实验设计的目的是分析调度与通信开销对系统性能的影响,故使得每个服务操作被调用时立即返回,并取活动的平均调度开销为 0.1s,网络带宽为 200Kbps.

© 中国科学院软件研究所 http://www.jos.org.cn

模拟实验 1 用于比较随着并发访问的增加集中结构与基于角色的分布式结构的性能.取服务的输入输出数据量分别为 20kb,当请求的平均到达间隔从 5s 减小到 0.5s 时,系统利用率和平均响应时间的变化如图 4 所示.系统利用率标识了活动调度对结点造成的负载情况,从图 4 中可以看出,集中结构下引擎负载明显高于分布结构下各个结点的负载,且很快接近饱和,分布结构中各结点的负载增长较为缓慢,有备选服务的结点表现更为明显.平均响应时间反映了组合服务执行时等待调度与执行调度的总开销,在系统轻载条件下,集中结构的平均响应时间较优,这是因为使用分布式结构时协同活动的调度增加了调度时间.但是随着并发访问的增加,集中引擎成为调度的瓶颈,大量调度任务的排队而使得平均响应时间迅速增大,而分布式结构则表现出比较平缓的曲线.

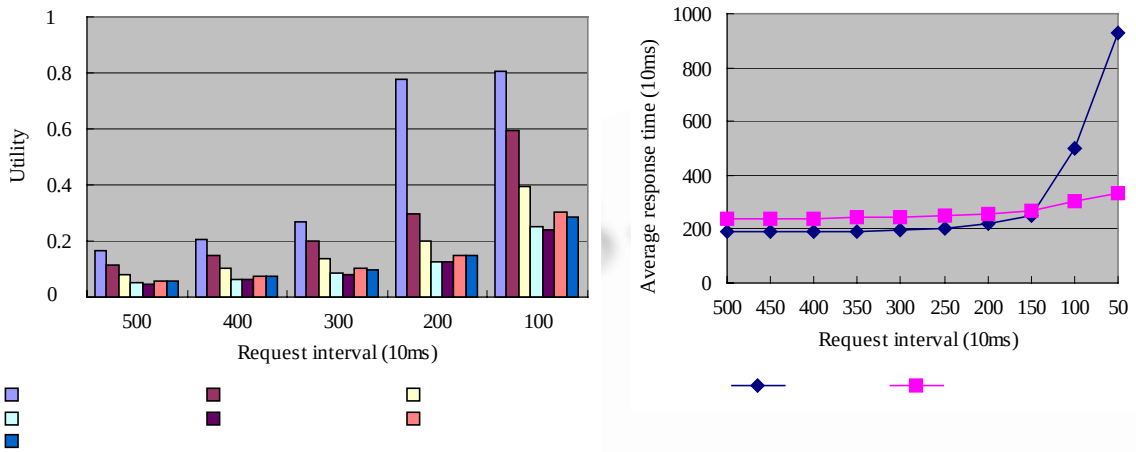


Fig.4 Performance comparison with respect to the number of concurrent requests

图 4 并发访问量对系统性能的影响

模拟实验 2 用于比较基本服务输入/输出消息大小增加时组合服务平均响应时间的变化情况.取请求到达间隔为 5s,网络带宽同实验 1,结果如图 5 所示.实验表明,随着消息大小的增长,采用集中结构时组合服务的平均响应时间的增长率大于采用本文所述的基于角色的分布式结构.分析可知,在分布式结构中,由于基本服务之间直接交换数据,故在最好的情况下,即在数据依赖只发生在那些直接相邻的活动之间时,基本服务消费和产生的数据在结点之间传输且仅传输一次,通信量可减为集中模式的 1/2.

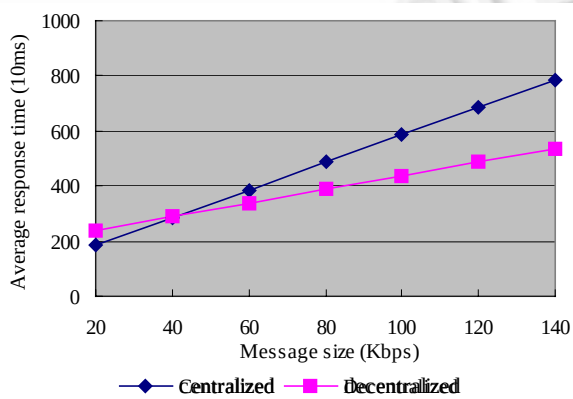


Fig.5 Performance comparison with respect to message size

图 5 数据传输量对系统性能的影响

上述模拟实验说明,与具有中心引擎的集中式结构相比,本文提出的基于角色的分布式服务组合方法有助于提高组合服务的可伸缩性.这是因为,一方面,由于组合服务调度责任分布到所有参与组合的结点,因而能够

© 中国科学院软件研究所 http://www.jos.org.cn

有效地缓解大量并发请求对组合引擎带来的负载,改善系统的总体性能;另一方面,由于业务数据在结点之间直接传输无须经过中心引擎的转发,从而降低了网络通信开销,更适合于具有较大数据量的组合服务应用.

4 相关工作

服务组合技术是新兴的综合性课题,基于流程建模组合服务是当前主流的组合服务建模思想,主要的建模语言有 WSFL[7],BPEL4WS,DAML-S 等.本文采用基于有向图的方法对组合服务进行抽象建模,因此本文所述的全局流程模型及本地流程模型可以使用上述语言进行描述.WSFL 提出的流模型(flow model)和全局模型(global model)分别描述组合服务内部基本服务执行操作的顺序和它们之间的消息交互,是平行的两个建模侧面.本文提出两级流程模型具有不同的含义,它们均是定义组合服务的实现流程,本地流程模型是全局流程模型在特定角色上的投影,它们之间是全局与局部的关系.

在研究项目和原型系统方面,基于集中结构的组合服务系统主要包括:eFlow[3]是一个支持 e-service 的组合描述、执行和管理的系统,使用有向图描述组合服务流程,采用集中式的流程引擎执行组合 e-service;SCET[4]使用 WSFL 语言静态建模组合服务,并编译产生组合流程的 Perl 执行代码,由集中的控制器加载执行,控制器使用管道-过滤器的模式,按照组合服务描述连接活动的输入与输出;Meteor-S[8]项目侧重于使用语义技术提高服务组合的自动程度与灵活性,其底层执行机制依赖于其早期工作流系统 Meteor;类似的工作还有文献[9].为了提高服务组合系统的可伸缩性,文献[10,11]将分布式体系结构引入服务组合系统.Self-Serv[10]是一个使用状态图建模的分布式组合服务系统.它通过状态分解产生每个活动的协调器并为之配置路由表,执行时部署了协调器的各个结点根据其路由表决定下一步骤的执行地点.Orisis[11]是一个基于数据库的分布式服务组合系统,它通过模型、服务及运行时数据的复制实现组合服务的分布执行.与上述工作不同,本文采用模型划分的方法实现组合服务的分布执行,不需要产生额外的协调代码,也不需要各个节点之间进行模型和执行状态的拷贝.IBM 印度实验室 symphony[6,12]是最新的研究项目,其研究思想与本文最为接近.它将一个基于 BPEL 的流程分解为多个 BPEL 流程,并在多个 BPEL4J 引擎上验证了分布执行的性能,所得到的结果与本文类似.在模型分解方法上,symphony 采用基于同步分析[12]的方法以实现最大并行化,本文的工作基于通用的有向图流程模型,模型分解则基于活动之间的依赖关系,实现基于角色的划分.

5 结 论

本文针对开放环境对大规模服务组合的可伸缩性及自治性的需求,提出基于角色的分布式动态服务组合方法.该方法通过将全局定义的流程模型划分为各个角色的本地流程模型,从而使得组合服务的控制能够分布到多个服务结点,并在服务结点之间直接交换数据.模拟实验结果表明,与集中结构相比,使用该方法构造的组合服务系统具有更好的可伸缩性.下一步的工作将主要从以下两方面展开:(1) 本地流程模型产生算法的优化:本文给出的算法在协同模式上采用基于增量同步的策略[13],可以进一步分析活动之间数据和控制的依赖关系,通过弱同步实现本地模型之间协同消息数量的优化;(2) 分布执行在带来性能优化的同时必定伴随着管理的复杂化,尤其在有错误发生的情况下,因此下一步的工作将基于组合服务的异常建模,分析异常处理逻辑的划分及分布式错误恢复技术,并进一步分析分布式结构带来的开销问题,从而进一步完善基于角色的分布式动态服务组合方法.

References:

[1] Papazoglou MP, Georgakopoulos D. Service oriented computing. Communications of the ACM, 2003,46(10):25−28.

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

[3] Casati F, Ilnicki S, Jin L, Krishnamoorthy V, Shan M. Adaptive and dynamic service composition in eFlow. In: Wangler B, Bergman L, eds. Proc. of the Int'l Conf. on Advanced Information Systems Engineering. LNCS 1789, Stockholm: Springer-Verlag, 2000. 13−31.

© 中国科学院软件研究所 http://www.jos.org.cn

[4] Chandrasekaran Senthilanand. Composition, performance analysis and simulation of Web services [MS. Thesis]. Georgia: University of Georgia, 2002.

[5] Chen Q, Hsu M. Inter-Enterprise collaborative business process management. In: Proc. of the 17th Int'l Conf. on Data Engineering (ICDE). Heidelberg: IEEE Computer Society, 2001. 253−260.

[6] Chafle G, Chandra S, Mann V. Decentralized orchestration of composite Web services. In: Proc. of the 13th Int'l World Wide Web Conf. New York: ACM Press, 2004. 134−143.

[7] IBM Corporation. Web services flow language (WSFL) version 1.0. www-3.ibm.com/software/solutions/webservices/pdf/WSFL.pdf. 2001

[8] Patil A, Oundhakar S, Sheth A, Verma K. METEOR-S Web service annotation framework. In: Proc. of the 13th Int'l World Wide Web Conf. New York: ACM Press, 2004. 553−562.

[9] Du ZX, Huai JP, Wang Y, Zhang Y. Research and implementation of composite Web service supporting system. Journal of Beijing University of Aeronautics and Astronautics, 2003,29(10):889−892 (in Chinese with English abstract).

[10] Benatallah B, Dumas M, Sheng QZ, Ngu AHH. Declarative composition and peer-to-peer provisioning of dynamic Web services. In: Proc. of the 18th Int'l Conf. on Data Engineering (ICDE 2002). San Jose: IEEE Computer Society, 2002. 253−260.

[11] Weber R, Schuler C, Neukomm P, Schuldt H, Schek HJ. Web service composition with O'Grape and Osiris. In Proc. of the 29th VLDB Conf. Berlin: Springer-Verlag, 2003. 1081−1084.

[12] Nanda MG, Karnik N. Synchronization analysis for decentralizing composite Web services. In: Proc. of the ACM Symp. on Applied Computing (SAC). Melbourne: ACM Press, 2003. 407−414.

[13] Muth P, Wodtke D, Weissenfels J, Dittrich A, Weikum. From centralized workflow specification to distributed workflow execution. Journal of Intelligent Information Systems, 1998,10(2):159−184.

附中文参考文献:

[9] 杜宗霞,怀进鹏,王勇,张煜.组合 WEB 服务支撑系统的研究与实现.北京航空航天大学学报,2003,29(10):889−892.

© 中国科学院软件研究所 http://www.jos.org.cn