第 29 卷 第 7 期
2006 年 7 月
计 算 机 学 报
CHINESE JOURNAL OF COMPUTERS
Vol. 29 No. 7
July 2006

基于 Petri 网的 Web 服务自动组合研究

钱柱中 陆桑璐 谢 立
（南京大学计算机软件新技术国家重点实验室 南京 210093）
（南京大学计算机科学与技术系 南京 210093）

摘 要 根据 Web 服务中消息和行为的关系，Web 服务被描述为面向消息的基于行为 Petri 网模型（Moap）. 该模型包括消息域和服务过程，前者是服务协同以及和用户通信的通道，后者是基于 Petri 网的 Web 服务行为过程描述. Moap 支持组合服务的重用，可以很好地描述可用服务的并行调用，另外元消息机制也有利于服务的自动组合. 在 Moap 基础上，定义了服务组合系统和服务的协同；给出了服务自动组合规则，并证明了其有效性和完备性. 最后，通过一个实例直观描述了服务组合系统的工作方式.

关键词 Web 服务；Petri 网；Web 服务自动组合
中图法分类号 TP311

Automatic Composition of Petri Net Based Web Services

QIAN Zhu-Zhong LU Sang-Lu XIE Li
(*State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing* 210093)
(*Department of Computer Science and Technology, Nanjing University, Nanjing* 210093)

Abstract Automatic Composition of Web services is one of the most important issues in the research of service-oriented computing (SOC). According to the relationship of messages and activities, Web services are defined as message oriented activity based Petri net model (Moap). The model is characterized in terms of message domain and service process. The formal is used to the service cooperation and the communication with client; the latter is the description of Petri net based service process. Moap supports the reuse of the composite services. Compared with automata based models, Moap can describe parallel composition and the meta-message mechanism benefits the automatic composition. Based on Moap, the problem of service composition is presented including the definition of composite Web service system and the service cooperation. Then, the rules of automatic service composition are proposed and the soundness and the completeness are proven. Finally, an example also proofs the usability of Moap.

Keywords Web services; Petri net; automatic Web services composition

1 引 言

近年来，面向服务的计算（SOC）成为了软件领域最热门的研究内容之一，它的核心理念是在交互的软件成分之间，构建松耦合的协同软件体系. SOC 以 Web 服务（或者称为 e-服务）作为基本组成成分，并采用了一系列标准化的协议进行交互. 关于

收稿日期：2006-02-17；修改稿收到日期：2006-05-15. 本课题得到国家“九七三”重点基础研究发展规划项目基金（2002CB312002）、国家自然科学基金（60402027）和江苏省自然科学基金创新人才项目（BK2005411）资助. 钱柱中，男，1980 年生，博士研究生，主要研究方向为面向服务的计算、分布式计算、形式化方法. E-mail：qianzhuzhong@dislab.nju.edu.cn. 陆桑璐，女，1970 年生，教授，博士生导师，主要研究领域为分布式计算、并行处理. 谢 立，男，1942 年生，教授，博士生导师，CCF 高级会员，主要研究领域为分布式计算、信息安全.

SOC 的研究涉及很多方面，其中一个重要的问题就是 Web 服务组合. 服务组合描述的是，一个复杂的需求无法由单一 Web 服务完成，而可以由多个可用 Web 服务组合起来完成. Web 服务组合主要涉及到两个方面的问题[1]：(1) 服务合成(synthesis)；(2) 组合服务的编制执行(orchestration). 本文工作集中于前者，研究如何将给定的一组可用 Web 服务组合成为一个复合服务，满足用户的需求.

目前，包括工作流、Pi 演算、Petri 网、智能计划、自动机等在内的诸多理论和技术都应用于 Web 服务的组合研究. 由于 Petri 网具有异步和并发的特点[2]，适用于像 Web 服务这样的分布式软件系统，因此本文将 Petri 网理论用于服务组合. 如今，已经有一些基于 Petri 网的服务组合的研究成果. 文献[3]定义了一个 Petri 网模型用于描述 Web 服务，并分析了服务的模拟、验证和自动组合. 这样的模型描述的服务是无状态的，服务组合不可重用. 文献[3]中定义了顺序、并行、选择等 6 种基本复合方式，其自动组合的主要思想是，先由可用 Web 服务根据它们的交互关系，产生所有可能的组合网，然后在这些 Petri 网中找出能够到达对应目标的标识的一种组合. 这种方法的复杂度取决于 Petri 网的空间个数和可能的服务组合方式. 而其它更多的工作是利用 Petri 网的性质对这样的面向服务的软件系统进行分析和验证. 文献[4]将 Web 服务的行为定义为一个 Petri 网，同时给出了顺序、选择、循环和并行等多个基本组合操作所对应的 Petri 网的表示方式. 文献[5]定义规则将 BPEL4WS 转化为面向服务的 Petri 网(BPWS-Net)，并利用这样的网模型分析该 Web 服务组合的正确性. 文献[6]建立了一个面向服务结构模型 WS-Net 来描述 Web 服务，包括接口、通信和互操作 3 个层次，都采用 Petri 网来定义. 每个层次都包括两个通道，“Enqueue”接收数据，“Dequeue”接收服务请求并处理. 这样的模型有利于考察面向服务软件的可靠性，能够使用 Petri 网理论来分析系统. 类似于文献[3]，该模型并不描述服务内部行为，而且其通信和互操作等服务协同都是人为预先定义，并不能自动化完成服务组合. 另外，文献[7]使用了有色 Petri 网对服务组合系统行为进行验证.

不同于以上这些基于 Petri 网的服务组合研究，本文定义了一个面向消息的基于行为 Petri 网的 Web 服务模型(Moap). 同以上方法相比较，Moap 具有两个特点：(1) 消息分解. 将其视为 Petri 网的资源来推动服务行为，有利于自动服务组合；(2) 服务组合可以重用. Moap 将原子 Web 服务和服务组合系统都描述为一系列的行为序列和状态的转换，这样组合服务可以作为一个可用服务供更复杂系统使用. 在 Moap 基础上，本文提出了服务组合系统的描述和服务组合的目标，制定自动服务组合规则，并证明了这种自动组合方式的有效性和完备性.

本文第 2 节给出 Web 服务模型的定义；第 3 节叙述基于 Moap 模型的 Web 服务组合，定义组合服务系统，描述服务之间的协同；第 4 节详细说明自动服务组合，通过定理说明服务组合的可用性和完备性；第 5 节通过一个简单实例说明 Moap 进行服务组合的方式和步骤；第 6 节中比较 Moap 模型和其它 Web 服务模型的特点；最后是结论和下一步的工作.

2 Web 服务模型

Petri 网系统由表示状态的 S 元素(又称为位置，place)和表示变化的 T 元素(又称为变迁，transition)构成. 它适合于描述异步并发系统，当系统中的变迁正确描述后，Petri 网理论能够保证整个网络系统是可实现的，而且具有系统描述的性质. 因此，Petri 网适用于 Web 服务这种松散耦合的分步式系统，并且能够严格控制其行为. 另外，Petri 网中没有任何形式的固有控制，任何形式的控制必然会与系统中其它的物质资源和信息资源一样出现在系统的 Petri 网中[2]. 而 Web 服务的协同是通过消息来控制触发行为的，因此，我们可以将消息定义为网内的资源，Web 服务的行为不受系统外因素控制，从而有力地支持了服务的自动组合.

本节介绍这个 Web 服务模型，该模型包括两个方面：消息域和服务过程. Web 服务互操作的基础是消息，而 SOC 的理念是一种松耦合的协同软件构架，Web 服务之间的消息传递是异步的，因此消息域的引入是必要的，它能够作为“消息栈”[8]，用于暂存接收到的消息. 服务过程是基于消息基础上的 Web 服务行为描述，它定义了一个 Web 服务在不同消息触发下所能够完成的任务. 服务过程是一个 Petri 网模型，因此本文中定义的 Web 服务模型被称为面向消息的行为 Petri 网模型(Message oriented activity based Petri net model，Moap). 下面我们从元消息开始，给出这个模型的完整定义.

定义 1(元消息 Schema). 元消息 m 为六元组 $(ID, Type, Value, Status, Source, Target)$，其中：

ID 用于标示不同的元消息；

$Type \in D$ 指消息的类型，D 是消息类型的集合；

$Value$ 是元消息的值；

$Status$ 是元消息当前的状态，$Ready$ 表明接收到该消息，等待相应的处理；$Processed$ 表明该消息已经触发相应的服务行为，即已经处理完毕；

$Source$ 和 $Target$ 表明元消息的来源和目标.

一个元消息只包含一个信息单位，描述 Web 服务的 WSDL 文档中定义的调用该 Web 服务的一个参数. 在实际系统中，多个元消息组成一个服务请求消息或服务响应消息. 例如，机票预定服务 flight_book(date, flight)，触发该服务的行为的服务请求消息分解为"机票服务"、"日期"和"航班"，这 3 个元消息. 根据"$Target$"，元消息分为内部状态消息 m_i 和外部交互消息 m_c，前者在本 Web 服务内部流动，表示当前的状态或者内部的中间结果，后者用于同外部实体的通信.

定义 2(消息域). 消息域是一个关系数据表 R_{MD}，元素为元消息.

Web 服务接收到的所有消息都被分解为元消息，记录于消息域，并被置为"$Ready$"状态，表明服务需求；服务提供者在空闲状态下，获取相应元消息，并置为"$Processed$"，表明接受服务请求并触发服务行为. 消息域的作用范围为本 Web 服务，为服务的请求者和提供者之间提供了一个异步的消息传递管道.

定义 3(消息). 消息 M 是一组相关元消息(包括 m_i 和 m_c)的集合，$M(m_1, m_2, \cdots, m_n)$ 表示由一组元消息 $m_1, m_2, \cdots, m_n$ 组成的消息 M.

消息是实际系统中服务请求者和服务提供者之间交换的信息单位，如上例中的机票预定请求消息. 在 Moap 中我们之所以采用更细粒度的元消息作为消息域的基本单位，是出于下面两个原因：(1)有利于支持自动消息分解；(2)有利于支持可用服务的并行处理. 例如，"旅行业务"预定，旅行者给出消息包含"起讫地"、"起止日期"和"旅行路线"(表示为〈日期，地点〉这样的二元组集合). 可用服务"机票预定"需要参数"起讫地"和"起止日期"，而"旅馆预定"则要求给出"日期"和"地点". 分解为元消息后，两个服务分别可以自主获取所需参数形成服务调用消息 M 并触发行为. 以会话模型为基础的 Web 服务协同机制，采用一个调用消息作为数据传递单位，导致一次调用一个服务，而且必须获得应答结果后才发出下一个信息调用，因此难以并行处理. 而以元消息作为单位，服务调用者可以给出尽可能多的信息，由可用服务自主选择，一旦满足其触发要求，就可以执行某个行为，提高了服务执行的效率. Moap 中定义的服务过程则正是基于这样的元消息的一个标记[9]的有色 Petri 网(Colored Petri Net, CPN).

定义 4(服务过程). Web 服务过程 $\mathcal{W} = (P, T; F, \Sigma, h, C, I, S^0, G_f)$，其中

$(P, T; F)$ 为有向网，称为 $\mathcal{W}$ 的基本过程，满足：$P \cap T = \varnothing$，$P \cup T \neq \varnothing$，$F \subseteq (P \times T) \cup (T \times P)$，且 $dom(F) \cup cod(F) = P \cup T$；

Σ 是有限字符集；

$h: T \rightarrow \Sigma$ 为标识函数；

$C: P \cup T \rightarrow \mathcal{P}(D)$，$\mathcal{P}(D)$ 为消息类型集 D 的幂集；

$I: (P \rightarrow T) \cup (T \rightarrow P)$，任意 $I(p,t) \in [C(t)_{MS} \rightarrow C(p)_{MS}]_L$① 且 $I(p,t) = 0$ 的充要条件是 $I(p,t)$ 不属于 F；

$S^0: P \rightarrow D_{MS}$ 是服务过程 $\mathcal{W}$ 的初始标识，满足 $\forall p \in P: S^0(p) \in C(p)_{MS}$，即 $S^0(p)$ 是 p 的消息类型集合上的多重集；

$G_f \subseteq R(S^0)$ 为终态集合.

位置 P 表明了 Web 服务的状态，变迁 T 表示 Web 服务的一次行为. 流关系 F 连接 P 和 T，表明了消息这种资源的流动情况. 一方面说明变迁发生的基本前提，即行为的可能性，即什么样的消息触发什么样行为；另一方面指出了行为的结果，即一个行为完成后，可以产生什么样的消息(包括表示内部状态的 m_i 和同外部请求者通信的 m_c). $\mathcal{W}$ 的基本过程指明了 Web 服务不同行为之间的基本逻辑顺序. 标记函数 h 的引入将现实世界的行为 T 映射为 Σ 上的抽象符号，这样的映射本质上是一种语义的匹配关系. 由于可用的 Web 服务并不是统一开发的，虽然遵循了 Web 服务的标准，但是语义上的异构性仍然存在，例如，同一个行为在不同的服务描述中可能有不同的命名. 同样的，元消息的异构性也会导致服

① $C(t)_{MS}$ 表示集合 $C(t)$ 上的所有有限多重集组成的集合. 多重集是允许一个元素多次出现的集合. 如 $\{a,a,b\}$ 是 $\{a,b\}$ 的多重集，其对应 $f: \{a,b\} \rightarrow IN_0$，其中 $f(a)=2$，$f(b)=1$. 本文只考虑有限多重集，即只在集合 S 的有限多个元素上值不为零. $[C(t)_{MS} \rightarrow C(p)_{MS}]_L$ 则表示从 $C(t)_{MS}$ 到 $C(p)_{MS}$ 上的所有线性函数的集合.

务组合的失败，为此，文献[10]提出了基于文法的服务间异构数据匹配，在此基础上我们提出了基于本体的异构消息处理机制[11]. 关于行为和消息的语义匹配的研究已经超出了本文的研究范围，在下文中将直接使用 Σ 中的抽象行为，不再说明 T 中的实际行为通过 h 映射到 Σ 中的抽象行为；关于元消息则简单认为它们共享了同一个消息字母表.

元消息 m 对应网内流动的权标(Token)①，"有色"体现在消息具有不同的类型，D 表示所有消息类型的集合. 在服务过程中，消息是行为的触发者，行为是消息的消耗者，同时也是产生者. 函数 I 是消息和行为关系的纽带，表明行为发生对消息的影响. 根据对消息的消耗和产生关系，我们将 I 进一步划分为 I_- 和 I_+ 两种关系，分别称为 $P\times T$ 上的负函数和正函数. I_- 对应于 F 中状态(p)到行为(t)的流关系，表示对消息的消耗，也类似于 OWL-S 中行为的先决条件(pre)，表示当且仅当出现 Web 服务接收到这些消息，行为才会被触发，同时这些消息消亡；I_+ 对应 F 中行为(t)到状态(p)的流关系，表示行为完成后产生的消息②，类似于 OWL-S 中的后续条件(post). 需要注意的是，I_+ 只产生内部状态 m_i 消息，而 m_c 则在行为过程中，即 t 中定义结果返回动作，将消息送给服务请求者. Moap 中，用 ${}^*t=I_-(p,t)$ 表示触发行为 t 所需要的一组元消息. 消息在服务过程中是一种资源，函数 I 通过对消息的消耗和产生的计算，实质上体现了 Web 服务过程中消息对流程的控制.

$\mathcal{W}$ 中的标识用于说明服务过程中所有位置的消息情况，从而表明服务过程的当前状态. Moap 中，我们用 S^0 表示服务过程中，触发服务行为所有可能的消息交互的位置和需要的各类型消息的个数. 请注意，S^0 并不是指服务启动时，必须一次性全部给出这些消息，而是指服务过程中潜在需要接收消息的位置和接收消息的类型及其个数. 定义中的多重集表明，在同一个位置，可能需要同种类型的多个消息，这与实际应用环境是一致的. 例如，查询账单的服务中，需用设定查询范围的起始日期和终止日期这两个相同类型的消息. S^0 体现了在服务过程中，服务请求者和服务提供者之间的交互，一个消息系列激活一个行为序列. 我们用"$S_i[\sigma>S_j$"来表示 $\mathcal{W}$ 在标识 S_i 经过一个行为序列 σ 后到达标识 S_j.

G_f 是服务过程 $\mathcal{W}$ 的终止标识集，$R(S^0)$ 表示 S^0 经过任意行为序列之后的标识集. 当服务过程达到满足 G_f 中的某个标识时，该 Web 服务被认为正常结束，可以返回最终结果，并等待下一个调用.

不同的消息触发不同的行为，而由于消息的异步性使得消息获取是不确定的[12]，同时由于并发行为的存在，这样导致行为的发生次序是不定的. 但是根据 Petri 网的性质[2]，如果经过一个系列的行为能够到达某个状态标识，那么该序列的并发行为的先后顺序并不会改变状态标识. 因此，从整体行为过程来看，我们可以将 S^0 视为服务的初始标识，表示服务请求者的消息，$\mathcal{W}$ 在初始标识下，通过一个行为序列达到 G_f 中的某个标识. 最后，我们将这些行为序列称为 $\mathcal{W}$ 的行为语句，引入关于行为语言的定义[9].

定义 5(服务行为语言). 设 $\mathcal{W}=(P,T;F,\Sigma,h,C,I,S^0,G_f)$ 是一个服务过程，则 $L(W)=\{h(t)\in\Sigma^*\mid(t\in T^*)\wedge(S^0[t>S)\wedge(S\in G_f)\}$ 称为 $\mathcal{W}$ 的服务行为语言.

一个 Web 服务是一个消息的接收器，根据消息触发相应的行为，一次服务的过程，就是由消息触发的一个行为序列，而这个行为序列是属于服务行为语言的一个语句. 因此，服务行为语言描述了 Web 服务的行为特性，体现了该 Web 服务的能力. 需要注意的是，$\mathcal{W}$ 描述 Web 服务的行为，因此正确的过程描述应当保证 $\mathcal{W}$ 中的每个行为都是可能发生的，同时也不存在死锁等情况.

3 Web 服务组合

3.1 组合 Web 服务系统

在 Moap 的基础上，我们将服务需求构建为一个中介 Web 服务[13]作为组合 Web 服务的接口，请求者只与这个中介服务交互. 于是，组合服务系统对服务请求者是透明的，它属于服务的实现部分；另外，组合 Web 服务和普通 Web 服务在使用方式上的一致，支持了组合服务的重用.

定义 6(Web 服务组合系统). 服务组合系统 $\mathcal{C}$ 是一个松散耦合的软件功能实体，表示为三元组 $(T,G;\mathcal{A})$，其中

T：$(R_{MD},\mathcal{W})$ 是一个中介 Web 服务，表示新的服务需求；

G 是一个关系数据表，元组为元消息，记录全局

① Token 在 Petri 网中用于表示资源，有色 Petri 网中的 Token 通过定义不同的"颜色"，表示不同种类的资源.

② 有的行为只是对消息取值的修改，甚至仅仅起了"催化"作用，而没有任何改变. 但我们仍然视为消耗一个消息，并产生一个相同类型的消息.

状态；

$\mathcal{A}$是一个可用 Web 服务的集合，表示实现服务请求者功能需求的 Web 服务.

中介服务 T 在服务组合系统中具有特殊的意义，一方面体现了服务请求者的需求，另一方面它又是一个虚拟的服务，其功能由$\mathcal{A}$中的 Web 服务来实现. 因此，T 是服务组合的目标，在本文中称这个中介服务 T 为组合服务系统的目标服务. G 记录系统接收和发出的消息以及系统内部 Web 服务之间的消息，它是组合系统的消息域. T 和 G 构成了组合服务的核心，协调$\mathcal{A}$中的可用 Web 服务实现用户需求.

定义 7(可实现性).　称一个 Web 服务组合系统$\mathcal{C}(T,G;\mathcal{A})$是可实现的，当且仅当 $\forall\sigma\in L(T)$，$t\in\sigma$，$\exists A_i\in\mathcal{A}$，满足：

(1) $\exists\sigma_i\in L(A_i)$：$t\in\sigma_i$；

(2) 令 $\Gamma_{T\to A_i}(\sigma)$是从 σ 中删除 $\Sigma_T-\Sigma_{A_i}$ 中所有字符后的剩余子串(投影序列)，则 $\Gamma_{T\to A_i}(\sigma)\in L(A_i)$.

可实现性的第 1 条规定了目标服务 T 中的所有行为在$\mathcal{A}$中都有对应的服务能够完成这样的行为，即存在将 T 中行为指派给$\mathcal{A}$中可用服务来完成的可能性；第 2 条说明了这样的指派是有条件的，即$\mathcal{A}$中可用服务完成行为的次序在 T 中也必须保持，另外也要求$\mathcal{A}$中可用服务所完成的功能确实是 T 所期望的行为，而没有其它额外的行为. 满足这两个条件表明 T 和$\mathcal{A}$中可用服务协同工作的行为是一致的，因此$\mathcal{C}$是可实现的；反之，$\mathcal{C}$是可实现的，那么 T 中的行为是满足这两个条件的.

3.2　Web 服务的协同

Web 服务组合系统是要将可用的 Web 服务合理组织起来，协同完成复杂任务，服务的协同特性是组合 Web 服务系统的核心. 组合 Web 服务的协同包括可用 Web 服务之间的互操作以及可用 Web 服务和用户的交互；前者将多个可用服务构建为一个整体，后者则真正用于满足服务请求者的服务请求. 从消息传递角度，上述的两种协同可以统一为服务请求者和服务提供者所建立的会话①(session)[7]，在本系统中，这样的会话是通过对 G 的操作来实现的.

如图 1 所示，所有的消息都通过全局状态 G 来传递. 这种中介模式的服务组合有两个主要优点：(1)体现接口和实现的分离. 服务请求者只和组合服务系统的消息处理接口通信，而不关心实际调用哪个服务；(2)可用 Web 服务之间的通信都通过 G，于是 T 可以监控服务的协同，确保组合服务行为的正确性.

为了实现这样的消息传递，消息处理机制包含以下 3 个操作：$Query(Type)$，通过消息类型，查询是否有相应类型的元消息处于“$Ready$”状态；$Get(M)$，从 G 中获取一组元消息，并将这些元消息的状态转变为“$Processed$”；$Out(M)$，将一组元消息 M 写入 G，并置状态为“$Ready$”.

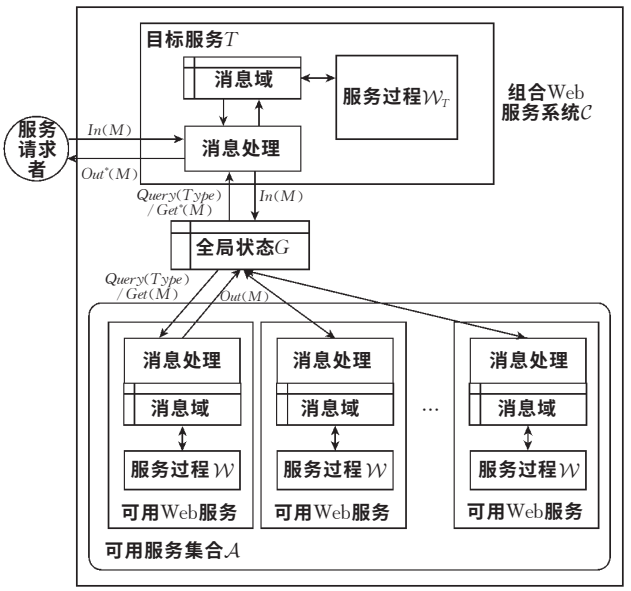


图 1　Web 服务消息的传递

Out 操作对应系统中产生新消息的过程，包括服务请求者发出的服务请求消息、服务提供者的响应消息. Out 将这些元消息写入 G，等待处理. $Query$ 和 Get 操作对应服务主动获取消息的过程. 当$\mathcal{A}$中的一个 Web 服务处于某个状态，即该 Web 服务的消息域中有部分 m_i 处于“$Ready$”状态，可用服务通过 $Query$ 操作主动查询 m_c，一旦满足触发条件，调用 Get 操作获得这一组 m_c，同相应的 m_i 组成一个消息 M，从而触发一个行为. 由于目标服务 T 的特殊性，一方面它不实际执行任何行为，因此并不真正消耗 G 中的元消息；另一方面它的消息处理要负责组合服务系统和用户之间的通信. 为此加入操作 $In(M)$，接收系统外消息，并分解为元消息写入 G. 另外，$Get(M)$ 改写为 $Get^*(M)$，它从 G 获取一组元消息，但不改变这些元消息的状态；$Out(M)$ 操作改写为 $Out^*(M)$，这个操作在 G 中查询相应的元消息，直到这些元消息都在 G 中出现(表明可用服务

① Web 服务之间协同可以认为是一个 Web 服务作为请求者调用其它的 Web 服务.

已经完成相应操作,返回结果或产生新的调用),则读取这组元消息 M,将它们发送给用户.

当一个服务实例启动,$In(M)$ 接收用户请求消息,写入 G;可用服务通过 $Query$ 和 Get 操作查询和获得当前 G 中可用的元消息,触发某个行为.当可用服务的一个行为完成后,Out 操作返回结果,或者发出新的请求消息.Get 操作并不从 G 中移除元消息,只是将其属性置为"$Processed$",因此 G 记录下一个组合服务实例运行过程中的所有传递的消息;当一个服务过程完成后,消息域 G 将被清空.

Moap 中服务的协同建立在消息触发行为的基础上,服务请求者的请求和服务提供者的结果都置于 G 中.不同服务主动获取 m_c,体现了 $\mathcal{A}$ 中可用服务"有能力者提供服务"的原则.也就是说,如果某可用服务可以完成这个任务,那么就获取这个服务请求消息,并触发相应的行为;同时,服务请求者也根据所需获取相应结果.Web 服务能够这样自主协同的基础是消息和行为基于同一字母表;必须保证服务过程 $\mathcal{W}$ 的描述是正确的.

4 自动服务组合

设组合服务系统 $\mathcal{C}(T,G;\mathcal{A})$,其中 $\mathcal{A}=\{A_1,A_2,\cdots,A_n\}$,$P=P_t\cup P_1\cup P_2\cup\cdots\cup P_n$ 为组合服务系统的位置集,$\Sigma=\Sigma_t\cup\Sigma_1\cup\Sigma_2\cup\cdots\cup\Sigma_n$ 为抽象行为集,作如下两个约定:

(1) $\forall t_1,t_2\in\Sigma,{}^*t_1={}^*t_2\rightarrow t_1=t_2$,即一个服务请求消息 $M(m_1,m_2,\cdots,m_n)$ 触发一个确定的服务行为;

(2) 当一个行为 t 满足触发条件,Web 服务即通过 $Get(M)$ 获取相应的消息,同时该行为就可以被触发,而不再受其它外部条件控制①.

请注意,约定(1)规定相同消息触发相同行为,但是相同行为并不一定对应相同消息,这是由于不同 Web 服务的服务过程中可能包含相同的行为,但触发这些行为的消息是不同的.如对于"预定确认"这个行为,在"旅馆预定"和"机票预定"两个服务过程中,分别由"旅馆预定"和"机票预定"的两个不同的确认(或放弃)消息来触发.

服务组合规则如下.

规则 1. $T.Get^*(M)\Leftrightarrow\exists A_i\in\mathcal{A}:A_i.Get(M)$;

规则 2. $S_T\in G_f^T\rightarrow\forall A_i\in\mathcal{A}:S_{A_i}\in G_f^{A_i}$.

规则 1 说明,目标服务 T 获取一组元消息 M 意味着 $\mathcal{A}$ 中必然有某个可用 Web 服务也获取了相同的一组元消息 M,反之亦然.规则 2 表明,当目标服务 T 中止,所有参与的 Web 服务都到达中止状态.

构造目标服务过程 $\mathcal{W}_T$ 的可达图,那么服务组合过程就是从初始标识 S^0 出发遍历达到所有终止状态的标识序列.因此,其时间和空间复杂度与 $\mathcal{W}_T$ 相关.

引理 1. 令 $t\in\Sigma$,则有 $S_T[t>S_T'\rightarrow\exists A_i\in\mathcal{A}:S_i[t>S_i'$.

证明. 根据约定(1),触发行为 t 的消息是确定的,于是设 ${}^*t=M$.目标服务触发了行为 t,并导致状态标识改变,因此必然有 $T.Get^*(M)$ 操作.又由组合规则(1)可知,在 $\mathcal{A}$ 中必然存在一个 Web 服务(设为 A_i)也获取了这个消息 M.根据约定(2),当 A_i 获取这个消息后,那么就会触发行为 t,从而也导致其状态标识的改变.

引理 1 说明根据这样的组合规则,目标服务 T 的所有行为都由 $\mathcal{A}$ 中的 Web 服务来实现.

定理 1. 一个 Web 服务组合系统是可实现的,当且仅当满足上述两条组合规则.

证明. (概要)设任意一个 Web 服务组合系统 $\mathcal{C}(T,G;\mathcal{A})$,证明组合规则的完备性和有效性.

完备性($\Rightarrow$):

(1) $\mathcal{C}$ 是可实现的,设 $\sigma_T\in L(T)$,$t\in\sigma_T$.根据定义 7,存在一个可用服务 A_i,使得 t 存在于 A_i 行为语言的一个语句 σ_i 中,并且 σ_i 中的其它行为都在 σ_T 中出现并保持相对顺序的一致.因此,T 在触发行为 t 之前必然具有和 A_i 一致的关于 t 的本地元消息.由约定(1),当出现满足行为 t 的 m_c 后,那么 T 和 A_i 都可以组成触发 t 的消息,根据约定(2),T 调用 $Get^*(M)$,A_i 调用操作 $Get(M)$ 获取相同的 m_c.规则 1 得证.

(2) 设任意的 $h(\sigma_T)\in L(T)$,由定义 5 可得,$S_T^0[\sigma_T>S_T$ 且 $S_T\in G_f^T$.根据定义 7,$\exists A_i\in\mathcal{A}$ 满足 $h(\Gamma_{T\rightarrow A_i}(\sigma))\in L(A_i)$,即 A_i 的某行为序列是构成 σ_T 的一个部分,同样的,由定义 5 得,$S_{A_j}^0[\Gamma_{T\rightarrow A_i}(\sigma)>S_{A_j}$ 且 $S_{A_i}\in G_f^{A_i}$.由于 σ_T 的任意性,所以可得 T 中止时所有参与的服务都中止.规则 2 得证.

有效性($\Leftarrow$):

设任意的 $h(\sigma)\in L(T)$,$t\in\sigma$,并且 $\exists S_T$ 和 S_T'

① Web 服务异步通信方式表明,当触发某一行为的原子消息都已经到达了,那么说明触发该行为的外部条件已经满足,服务提供者是否立即触发该行为还与 Web 服务内部因素有关,如系统的负荷、系统内部资源占用情况等.但这对于服务请求者而言,可以认为 Web 服务已经接受了请求并准备执行.

满足$S_T[t>S'_T$. 需要证明存在一个可用服务，满足定义 7 的两个条件. 根据引理 1，存在可用服务能够执行该行为，令服务 A_m 能够执行该行为，则条件(1)成立. 对于投影序列，设有 $S^0_{A_m}[\Gamma_{T\to A_m}(\sigma)>S_{A_m}$. 根据规则 2，$S_{A_m}\in G_f^{A_m}$. 所以由定义 5，可得 $h(\Gamma_{T\to A_i}(\sigma))\in L(A_i)$.　　证毕.

由定理 1 可知，这样的 Web 服务组合是有效的且完备的，从而保证了使用这种方式进行服务组合的正确性.

5　实　例

本节通过一个简单实例给出服务组合系统直观的组合方式. 给定可用服务“旅馆预定(B_h)”和“机票预定(B_p)”，目标组合服务“旅行业务(travel)”，图 2 分别描述了服务过程$\mathcal{W}_h$，$\mathcal{W}_p$和$\mathcal{W}_t$，并在图例中详细说明了各个抽象消息和行为所表示的语义. 从状态(位置)到行为(变迁)转换弧上的标识消息表明完成这个行为所需要获得的消息(对应 I_-)，从行为到状态弧上的消息则表明完成一个行为后产生的消息(对应 I_+). 我们以旅馆预定 B_h 为例来说明服务实例同抽象的服务过程$\mathcal{W}_h$的关系. B_h 包含三个行为：“旅馆预定”t_1，“(旅馆)预定确认”t_2 和一个空闲等待 wait. 整个服务过程分为 4 个阶段：

(1) B_h 处于就绪状态，等待服务请求信息 M_1，而当它到达后，触发“旅馆预定”行为 t_1；

对应$\mathcal{W}_h$中 $I_-(P_1,t_1)=h$，$I_-(P_2,t_1)=M_1$，即 h 和 M_1 触发 t_1；

(2) 行为 t_1 完成后返回信息 M_2 给请求者，并到达预定准备状态，等待请求者的确认信息 M_3；

对应$\mathcal{W}_h$中 $I_+(t_1,P_3)=h_ready$，即行为 t_1 产生表明服务状态的元消息 h_ready；

(3) 获得 M_3 后触发行为 t_2，完成旅馆的预定，返回 M_4 给请求者表明预定完成，进入服务中止状态；

对应$\mathcal{W}_h$中 $I_-(P_3,t_2)=h_ready$，$I_-(P_4,t_2)=M_3$，$I_+(t_2,P_5)=h_fin$，即 h_ready 和 M_3 触发行为 t_2，而行为 t_2 产生元消息 h_fin 表明服务完成；

(4) 处于中止态的服务，经过一个等待行为 wait 之后，再次进入就绪状态，等待下一轮的调用.

对应$\mathcal{W}_h$中 $I_-(P_5,t_3)=h_fin$，$I_+(t_3,P_1)=h$，即 h_fin 直接触发等待行为 wait，停止等待后即产生 h 进入下一轮服务.

B_h 在整个服务过程中需要接收服务请求者 M_1 和 M_3 两组元消息，分别用于触发 t_1 和 t_2 这两个行为；另外，h 表明服务的启动. 于是，根据第 2 节定义 4 所述，其初始标识为 $S_h^0=(h,M_1,0,M_3,0)$；而易见其终态标识为 $G_f^h=(0,0,0,0,h_fin)$.

类似地，机票预定 B_p 也有 4 个阶段，就绪阶段等待消息 M_2，此后触发“机票预定”行为 t_3，返回预定情况的消息 M_6，并等待请求者确认，最后完成预定，返回预定完成消息 M_8，经过等待行为 wait 后，进入下一轮. 而目标服务“旅行业务(travel)”则同时提供请求者旅馆和机票的预定，并且这两者是可以同时进行的，而其中的每个过程则同样需要经过上

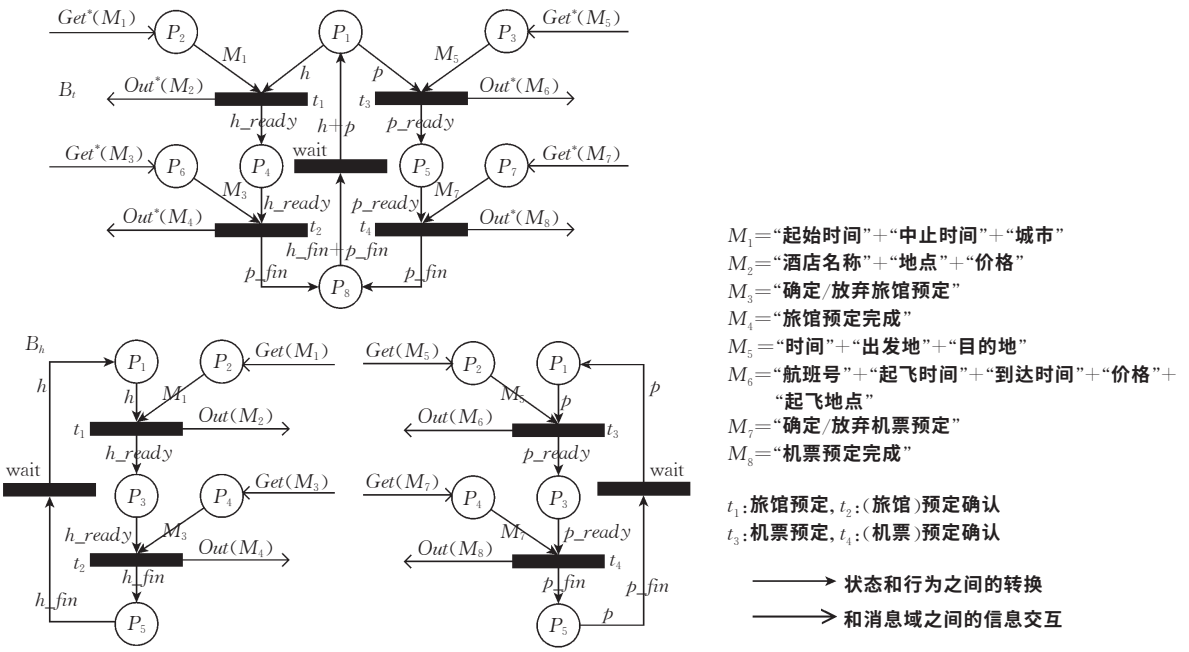


图 2　一个实例：“旅行业务”服务

述的步骤. 机票预定 B_p 和目标服务 travel 的初始标识分别为 $S_p^0=(p,M_5,0,M_7,0)$, $S_t^0=(h+p,M_1,M_5,0,0,M_3,M_7,0)$;终态标识为 $G_f^p=(0,0,0,0,p_fin)$, $G_f^t=(0,0,0,0,0,0,0,h_fin+p_fin)$.

旅行者使用"旅行业务",并传入信息"起讫地"、"起止日期"和"旅行路线"(表示为〈日期,地点〉这样的二元组集合). *In* 操作将这些信息以参数为单位分解为元消息写入 G,它们可以组成 M_1 和 M_5,于是目标服务通过 Get^* 获取 M_1 和 M_5,而 B_h 和 B_p 通过 *Get* 操作分别获取 M_1 和 M_5,并触发相应 t_1 和 t_3. t_1 和 t_3 的执行没有依赖性,可以并行执行,在目标服务中也体现出了这种并行调用的关系. t_1 完成后,通过 *Out* 操作将结果 M_2 写入 G,此时目标服务 Out^* 探测到 G 中的 M_2 后,将它发送给旅行者. B_h 产生内部消息流入 P_3,并等待用户的确认. 当旅行者返回确认消息后,t_2 被触发,返回最终结果,产生内部消息 h_fin,并达到终态标识. 当 B_h 和 B_p 都完成,目标服务亦达到终态. 请注意,目标服务 P_8 是对两个服务的同步,只有当两个服务都完成才能进行下一轮的旅行预定.

6 讨 论

目前,用于 Web 服务自动组合的其它模型主要有 3 种[14]:OWL-S 模型、会话模型和 Roman 模型. 我们对这几个模型分别进行讨论.

OWL-S[15]是对 Web 服务的基于语义的描述,它使用服务刻画(profile)来描述服务的基本信息,指明服务能够做什么;服务模型(model)用于定义该服务的实现方式,通常被定义为一个过程;服务基础(grounding)说明服务的访问方式. 引入服务语义可以帮助实现服务的自动组合. 在 OWL-S 模型中,虽然服务模型定义了一个过程,包含顺序、并行、选择和循环等基本控制模式,但是这个过程是服务的内部流程,过程中的状态是 Web 服务的内部状态,外部观察者不可见. Web 服务一旦启动,外界不能干预其内部行为. OWL-S 模型是一种基于语义的 Web 服务模型,通过定义 Pre 和 Post 条件来表明服务的语义. 基于 OWL-S 的 Web 服务组合系统采用工作流、Pi 演算和智能计划等技术来实现服务的自动组合. 不同于其它模型,这种服务组合是不支持复用的[14],也就是说,目标组合服务在完成服务请求者的一次服务请求之后,不能再应用于更复杂系统的组合. 而 Moap 模型是支持复用的服务组合模型,两者具有不同的应用场合. OWL-S 侧重于语义,其服务组合侧重于智能,用户提出相对模糊的请求,基于 OWL-S 的组合系统智能地分解为多个任务来实现用户请求,适用于需求多变的门户系统. 而 Moap 则需要用户提出明确的需求,定义严格的行为特性,组合服务系统会生成一个服务组合规范,严格完成需求,同时可以发布这个组合 Web 服务,适用于构建复杂的软件应用系统,具有较长的生命周期.

会话模型主要考虑 Web 服务之间的消息传递,每个 Web 服务都被看成一个对等的通信实体(peer),它们通过米利机(Mealy Machine)模型来表示[8]. 每个 Web 服务都有一个先进先出的消息接收栈,每次 Web 服务从中取出一个消息进行处理. 会话模式中的全局行为捕捉所有相关服务之间的通信,得到一个全局的消息序列. 对于一个目标 Web 服务,其全局消息序列是确定的,可以用正则语言来描述. 服务组合的任务就是要找到一系列的对等实体,它们之间通信的整体行为满足该正则语言. 会话模型抓住了 Web 服务通过消息协同的特征,并把服务之间的互操作定义为一组会话,采用了类似 CSP 模型的通信方式来定义 Web 服务实体之间的协同. California 大学的研究小组对此做了较深入的研究[8,12,16],但是主要工作集中在考查组合服务行为的确认和验证,并没有涉及自动服务组合. Moap 也同样考虑消息作为服务协同的方式,由消息去触发服务的行为;不同之处在于,Moap 中采用了元消息作为消息传递的基本单位,并且将服务内部的状态也以消息的形式定义,从而建立了一个通过消息自动触发行为的组合服务系统. 同时,Moap 也能够反应服务的行为过程,能够有效地应用于服务自动组合.

所谓 Roman 模型[17]是指采用基于行为的状态转换模型来定义 Web 服务,可以用状态转换图来表示. 在转换图中,每一次转换用抽象的行为来标记,一个给定状态中的出边表明了到达该状态后,下一步可以进行的行为. 基于 Roman 模型的服务组合采用了命题动态逻辑(Proposal Dynamic Logic,PDL)技术[18]. 目标服务中的每一次状态转换,对应于某可用 Web 服务的一次状态转换,即目标服务的行为,由某可用 Web 服务实现;当目标服务达到终止状态时,所有可用 Web 服务也都到达终止状态. Roman 模型能够有效地应用于自动服务组合,但是模型中并没有考虑消息在服务协同过程中的重要作用. 不同的 Web 服务过程中可能包含相同的行

为，这时需要组合服务系统来选择正确的服务，尽管文献[19]对这个问题做了研究，并提出了“k-向前看(k-lookaheads)”的方法，但是这个方法需要人为给出预先读入的行为个数，并且没有说明当出现多次行为冲突、k 值不同时如何处理. Moap 模型充分考虑了消息和行为的关系，在不同服务过程中的相同行为，实质上是由不同的消息触发，体现了服务请求者不同的意图. 因此，Moap 中不会出现这样的服务选择问题，于是其自动组合更为有效. 另外，Petri 网适用于并行处理，Moap 能很好支持并行地调用可用服务来实现目标服务；而基于状态转换的并行组合将导致内部状态的大量增加[20]，因此它只适用于串行的服务过程.

值得一提的是，文献[21]提出了 Colombo 模型，它结合了会话模型和 Roman 模型，试图建立一个统一的 Web 服务模型. 文献[22]详细表述了该模型，并定义 PDL 规则用于自动服务组合. Colombo 和 Moap 的服务自动组合都是中介方式的，即根据用户的需求产生一个满足要求的 Web 服务，而该 Web 服务实际上是由给定的可用服务实现的. Colombo 采用了状态转换自动机来描述服务过程，定义全局状态(world state)，记录组合服务系统行为，本地栈用于暂存消息实现服务间的异步通信，使用“连接(link)”作为消息的通道，是一个比较完备的 Web 服务模型. 但是这样的模型中服务过程只体现了服务之间的消息交互，而消息和行为的联系并没有很好体现，需要通过制定 PDL 规则来确定消息触发一个行为. Moap 模型中将消息作为一种资源，行为作为资源消耗者，自然地建立消息触发行为的关系，并且在服务过程确定之后，不需要任何附加的控制约束，其本身就可以控制流程的正确执行. 另外，由于 Colombo 中没有像 Moap 模型分解消息，因此，组合服务系统的消息交互必须和可用服务的消息交互严格一致，这样并不是很灵活. 而 Colombo 模型的优势在于服务之间通过“连接”交互，它更接近于“中心控制(broker)”方式，这种拓扑结构使得数据可以直接在服务之间传递而不必经过中介服务，更适合有较多数据交换的系统.

7　结束语

本文提出了一个面向消息的行为 Petri 网(Moap)模型用于 Web 服务的描述，它包含消息和基于消息的服务过程. Moap 模型能够很好地描述 Web 服务的特征，同时也能有效地用于可重用的自动服务组合. 基于 Moap 模型，我们构建了 Web 服务组合系统，并提出了一种新的 Web 服务自动组合方式，并证明了该方法的可用性和完备性，最后通过实例验证了该方法的可用性.

Web 服务自动组合的研究还在进行中. 下一步的工作主要有两个方面：(1)Moap 模型的进一步完备化，主要工作集中于进一步细化 Web 服务内部状态和服务互操作传递的消息，明确各自相关联的行为特征；(2)关于自动组合的研究，根据本文中提出的服务组合规则，建立一个模型，从而能够更好地分析服务组合过程.

参　考　文　献

1　Alonso G.，Casati F.，Kuno H.，Machiraju V.. Web Service Concepts，Architectures and Applications. Springer，2004

2　Yuan Chong-Yi. Theory and Application of Petri Net. Beijing：Publishing House of Electronics Industry，2005(in Chinese)
(袁崇义. Petri 网原理与应用. 北京：电子工业出版社，2005)

3　Narayanan S.，McIlraith A.. Simulation，verification and automated composition of Web services. In：Proceedings of the 11th International World Wide Web Conference(WWW2002)，Honolulu，USA，2002，77～88

4　Hamadi R.，Benatallah B.. A Petri net-based model for Web service composition. In：Proceedings of the 14th Australasian Database Conference，Adelaide，Australian，2003，191～200

5　Sun J.，Jiang C. J.. Web service composition based on BPWS-Net. In：Proceedings of the Content Computing，Advanced Workshop on Content Computing，Zhenjiang，China，2004，303～313

6　Zhang J.，Chang C. K.，Chung J. Y.，Kim S. W.. A Petri-net based specification model for Web services. In：Proceedings of the IEEE International Conference on Web Services(ICWS'04)，San Diego，USA，2004，420～427

7　Yang Y. P.，Tan Q. P.，Xiao Y.，Liu F.，Yu J. S.. Transform BPEL workflow into hierarchical CP-Nets to make tool support for verification. In：Proceedings of the 8th Asia-Pacific Web Conference，Harbin，China，2006，275～284

8　Bultan T.，Fu X.，Hull R.，Su J. W.. Conversation specification：A new approach to design and analysis of e-service composition. In：Proceedings of the 12th International World Wide Web Conference (WWW2003)，Budapest，Hungary，2003，403～410

9　Jiang Chang-Jun. Behavior Theory and Application of Petri Net. Beijing：Higher Education Press，2003(in Chinese)
(蒋昌俊. Petri 网的行为理论及其应用. 北京：高等教育出版社，2003)

10　Wang Xiao-Ling，Guo Zhi-Mao，Zhou Ao-Ying. Grammar-

based message processing in Web service composition. Chinese Journal of Computers, 2005, 28(4): 478～485(in Chinese)
(王晓玲,郭志懋,周傲英. Web 服务组合的基于文法的消息处理. 计算机学报, 2005, 28(4): 478～485)

11 Qian Zhu-Zhong, Lu Sang-Lu, Xie Li. Ontology based heterogeneous message transformation in Web service composition. Acta Electronica Sinica, 2005, 33(12A): 2472～2475(in Chinese)
(钱柱中,陆桑璐,谢 立. 基于本体的 Web 服务组合异构消息的转换. 电子学报, 2005, 33(12A): 2472～2475)

12 Fu X., Bultan T., Su J. W.. Analysis of interacting BPEL Web services. In: Proceedings of the 13th International Conference on World Wide Web(WWW 2004), New York, USA, 2004, 621～630

13 Hull R., Renedikt M., Christophides V., Su J. W.. E-services: A look behind the curtain. In: Proceedings of the 22nd ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, San Diego, USA, 2003, 1～14

14 Hull R., Su J. W.. Tools for composite Web services: A short overview. Sigmod Record, 2005, 34(2): 86～95

15 Bechhofer S., Harmelen F., Hendler J., Horrocks I., McGuinness L., Patel-Schneider F.. OWL Web Ontology Language Reference, W3C, 2004

16 Aysu B. C., Bultan T., Fu X.. Design for verification for asynchronously communicating Web services. In: Proceedings of the 14th International Conference on World Wide Web (WWW 2005), Chiba, Japan, 2005, 750～759

17 Berardi D., Calvanese D., Giacomo D., Lenzerini M., Mecella M.. Automatic composition of e-services that export their behavior. In: Proceedings of the International Conference of Service Oriented Computing 2003, Trento, Italy, 2003, 43～58

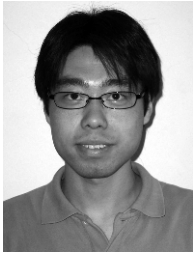
18 Kozen D., Tiuryn J.. Logics of programs. Handbook of Theoretical Computer Science: Formal Models and Semantics, ESP, 1990

19 Gerede E., Hull R., Ibarra H., Su J. W.. Automated composition of e-services: Lookaheads. In: Proceedings of the 2nd International Conference on Service Oriented Computing, New York City, USA, 2004, 252～262

20 Traverso P., Pistore M.. Automated composition of semantic Web services into executable processes. In: Proceedings of the 3rd International Semantic Web Conference (ISWC2004), Hiroshima, Japan, 2004, 7～11

21 Hull R.. Towards a unified model for Web services composition. In: Proceedings of the Advances in Computer Science-ASIAN 2005, Kunming, China, 2005, 1～10

22 Berardi D., Calvanese D., Giacomo D., Hull R., Mecella M.. Automatic composition of transition-based semantic Web services with messaging. In: Proceedings of the 31st International Conference on Very Large Data Bases (VLDB 2005), Trondheim, Norway, 2005, 613～624



QIAN Zhu-Zhong, born in 1980, Ph. D. candidate. His research interests include service oriented computing, distributed computing and formal methods.

LU Sang-Lu, born in 1970, Ph. D., professor. Her research interests include distributed computing and parallel computing.

XIE Li, born in 1942, professor, Ph. D. supervisor. His research interests include distributed computing and information security.

Background

Service Oriented Computing (SOC) is the computing paradigm that utilizes Web services as fundamental elements for realizing distributed applications. One of the hottest research topics behind SOC is the Web services composition which integrates the single services into a more powerful composite service. Until now, different approaches of service composition are proposed based on different Web service models. However, to the best of our knowledge, the relationship between the messages and the activities are not well defined in the current models, and there is still lack of a unified methodology for automatic services composition. The authors' research currently focuses on the message, activity and the cooperation characteristics of Web services. Employing Petri net theory, the authors are trying to building a unified Web service model to achieve the automatic service composition and to solve other related pending problems.

This work is partially supported by the National Natural Science Foundation of China under grant No. 60402027, Jiangsu Province Natural Science Foundation under grant No. BK2005411 and the National Basic Research Program (973 Program) of China under grant No. 2002CB312002.