一种基于服务覆盖网络的业务运行平台①

叶俊鹏 王 雷 (中国科学技术大学 自动化系 安徽 合肥 230027)

摘 要: 针对分布式业务的动态性和可扩展性需求,设计并实现了业务运行平台 BEPSON(Business Execution Platform Based On Service Overlay Network)。该平台基于服务覆盖网络,采用分层架构,提出并实现了具有 QoS 保证的动态服务发现算法以及分布式的执行策略。最后,通过实例验证了平台的有效性和灵活性。

关键词: 服务覆盖网络;动态服务发现;Web 服务;QoS

Business Execution Platform Based on Service Overlay Networks

YE Jun-Peng, WANG Lei

(Department of Automation, University of Science and Technology of China, Hefei 230027, China)

Abstract: To meet the dynamic and scalability of distributed business, a business execution platform—BEPSON (Business Execution Platform Based On Service Overlay Network), is proposed and implemented. The platform is based on service overlay network, using a layered architecture. A QoS guaranteed dynamic service discovery algorithm and a distributed execution strategy are proposed and implemented on the platform. Example shows that the platform is flexible and efficient.

Keywords: service overlay network; dynamic service discovery; web services; QoS

1 引言

随着 SOA 的广泛应用， 以服务为基本元素进行业务开发，通过组合现有服务实现服务的增值，是下一代互联网发展的趋势。

基于 Web Services 和 BPEL 进行服务组合是当前研究的热点，如何实现广域动态环境下 Web Services 的组合和运行一直是 SOA 应用面临的巨大挑战。现有的 Internet 并不支持灵活、动态地完成服务的组合和运行。

服务覆盖网络(Service Overlay Network[1], SON)作为一种应用层通用的框架，能够满足服务组合和执行的需求。但是，目前的研究主要集中于多媒体领域，难以适应高度动态性和可扩展性的应用场景。

目前，对于组合服务，多数的引擎都是集中式的，如 JOpera[2] 和 eFlow[3]，由于参与组合的服务是分散和动态变化的，集中式的执行模式存在着可扩展性差，维护代价高、单点失败等缺点。分布式的执行机制中典型的有 HOSS[4]和 Self-Serv[5]。HOSS 提出了一个基于 SON 的服务组合框架，引入了协议来刻画服务间复杂的交互关系，实现了按需的服务组合和分布式地运行，但是运行时系统采用一个全局的实体 Portal 来选择服务实例，容易成为系统的瓶颈。Self-Serv[5]提出了一个基于状态图的组合语言和一个 P2P 的组合服务执行模式，但是当前的系统实现，采用了集中式的 UDDI 来发现服务，也容易成为系统的瓶颈。

本文在上述研究的基础上，针对分布式业务的动态性和可扩展性的需求，提出并实现了 BEPSON——一个基于服务覆盖网络的业务运行平台，BEPSON 的主要目的在于广域分布式环境下，动态发现服务组件，

① 基金项目:国家 863 课题 “新一代业务管控协同支撑环境(2008AA01A317)”
收稿时间:2010-03-02;收到修改稿时间:2010-04-03

根据业务逻辑组合服务，完成业务的执行。针对这一目标，BEPSON 需要解决两个问题：一是动态地发现服务组件；二是业务流程的分布式执行。在本文中，我们将讨论 BEPSON 的结构以及基于该平台的动态服务发现算法和分布式执行策略。

2 BEPSON结构

为了适应网络的异构性，业务运行平台采用了一个混合式的结构，分为业务代理层和业务执行层。业务执行层是业务运行平台的核心，业务执行节点在广域环境下组织成一个服务覆盖网络，提供服务的注册、发现和执行。业务代理节点部署在网络的边缘，形成业务代理层，负责接收业务访问信息。下面我们介绍业务运行平台中的几个概念：

业务(Business)：业务是由 BPEL 描述的业务流程，通过组合现有的 Web Services 形成的新的 Web Services，面向终端用户或其他应用。

服务(Service)：服务的形式是 Web Services，通过调用物理资源来完成具体的操作，用来构建业务，多数情况不会暴露给外部用户直接调用。

业务逻辑链路表(Business logical Link Table)：业务逻辑链路表是解析业务流程 BPEL 文档得到的抽象级的服务路径，对应于组合逻辑，包含有逻辑判断条件，独立于任何的服务实例。

业务执行链路表(Business Execution Link Table)：业务执行链路表是一条可执行的业务路径，对应于业务实例的一次执行过程，由具体服务连接而成。

2.1 体系结构

BEPSON 采用了分层的结构，主要分为四层，分别是业务层、服务覆盖网络层、结构化 P2P 网络层、物理网络层，其结构如图 1 所示。

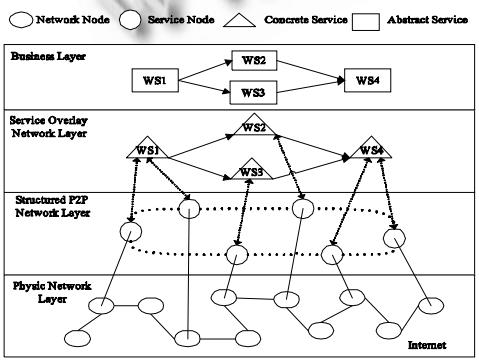

图 1　业务运行平台的体系结构

业务层(Business layer)的功能是根据用户或者应用程序对业务的请求，搜索对应的业务流程文档，将业务流程解析成相应的业务逻辑链路表。

服务覆盖网络层(Service overlay network layer)的功能是将业务逻辑链路表映射成业务执行链路表。根据业务逻辑链路表，查找服务，结合服务自身以及所在节点的 QoS 信息，从多个候选服务中选择出满足用户需求的服务，组合服务路径，形成业务执行链路。

结构化 P2P 网络层(Structured Peer-to-Peer Network layer)的功能是提供业务执行节点的自组织、服务信息的注册和服务的发现。

物理网络层(Physical Network layer)是现有的 Internet 架构，各个物理节点独立地提供服务。

2.2 服务覆盖网络

BEPSON 构建的服务覆盖网络面向广域网络，由多个提供服务的业务执行节点构成，将离散的服务通过覆盖网链接连接起来，形成新的业务。我们选择 Chord[6]作为网络结构，通过分布式哈希表(DHT)构成一个逻辑环结构，将数据和节点映射成 Key，利用 Key 完成数据的存储和查询。Chord 使用 SHA-1 或 MD5 哈希函数，为每个节点和数据分配 m 位的标识符，节点标识符通过哈希节点的 IP 产生，数据标识符通过哈希数据产生。Chord 具有可扩展、鲁棒性、负载均衡和容错性，满足业务执行节点动态加入和离开的需求。此外，chord 使用折半查找方法，系统中的每个节点维护 O(logn)个节点的路由信息，每次查找的最大跳数不超过 O(logn)。

在 BEPSON 中，服务位于业务执行节点上，分散在整个广域网络。为了实现服务的自动注册，每个业务执行节点拥有一张服务表(Service Table)，记录着节点所提供服务的元信息，包括服务功能名称、服务名、服务地址、静态 QoS 信息等。业务执行节点加入覆盖网络时，会读取 Service Table 所提供的服务信息，以服务信息为参数利用哈希函数得到一个 key 值，然后将此服务的元信息存储在 DHT 中。当业务执行节点离开时，会读取 Service Table，利用 DHT 系统提供的操作，将其注册的服务信息从 DHT 中删除。

3 BEPSON的运行机制

为了实现业务灵活、动态地执行，BEPSON 采用

动态服务发现机制和分布式执行策略，区别于其他BPEL执行引擎所采用的静态流程执行机制，也不需要一个集中式的中心来控制业务运行。对于用户的每一个业务请求，BEPSON会根据用户的需求和网络的性能，动态地选择一条业务执行路径，完成业务的执行。

3.1 基于QOS的服务发现

在BEPSON中，同一功能的服务由不同的业务执行节点提供，分散在不同的地点，具有不同的QoS属性。当一个业务执行节点要查找服务时，可以使用服务信息作为key来查询DHT系统，所得结果是一个具有匹配信息的候选服务列表。为了能够区分这些服务，选择出满足要求的服务，我们充分考虑了服务的QoS信息，包括静态QoS信息和动态QoS信息。静态QoS信息指服务的静态属性，存储在DHT中；动态QoS信息指服务所在节点与查询节点间的带宽和延迟等信息，通过网络测量来获得。

这里，我们定义了一个服务选择的度量函数S，选取的主要指标是延迟、带宽以及响应时间。因为在BPESON中，服务发现的主要目的是根据业务逻辑链路表，选择下一跳服务，完成业务的执行。这三个参数充分考虑了服务的属性和业务执行节点提供服务的能力，具有通用性。

我们使用这三个指标参数生成一个度量函数S(i,j)，设i表示查询节点，j表示服务所在节点，D(i,j)，B(i,j)分别表示两节点间的延迟和带宽，R(j)表示服务的响应时间，α (0< α <1)是权重因子，度量函数定义如下：

$$S(i,j) = \frac{B(i,j)}{a*D(i,j)+(1-a)*R(j)} \quad (1)$$

服务发现算法如下：

步骤 1. 查询节点以服务的功能名称为参数，利用哈希函数得到一个key值，使用该key值产生一个查询消息。

步骤 2. 覆盖网使用chord协议，根据各个节点的路由表，定位到负责key值的节点。

步骤 3. 节点将与key值关联的服务元数据列表返回给查询节点。

步骤 4. 查询节点通过计算度量函数，从步骤3所得的服务列表中选出满足要求的服务。

3.2 分布式的执行策略

BEPSON采用了一种纯分布式的运行机制，通过动态发现下一跳服务和有序的传递业务请求消息，依靠多个业务执行节点的协作完成业务的运行。

当业务部署到运行平台时，业务所在节点的业务层会解析该业务流程的BPEL文档，生成该业务的业务逻辑链路表，交由所在区域的业务代理保存。

当客户访问一个业务时，首先将业务请求发送给所在区域的业务代理，业务代理会根据业务请求，选择匹配的业务逻辑链路表，查找到业务的第一个服务，生成业务请求消息，发送给第一个服务所在节点。

为了实现业务执行节点间信息的交互和理解，我们定义了业务通信协议(Business Communication protocol)。业务通信协议的文本如下：

From: <IP Address>/*发送消息的节点IP地址*/

To: <IP Address>/*接收消息的节点IP地址*/

Service Numbers:<Service Numbers> /*业务中剩余服务的个数*/

Service:<Service Name>,<Parameter Numbers> /*服务描述*/

Parameters:<Parameter Name>,<Parameter Value> /*服务的参数描述*/

… …

Service:<Service Name>,<Parameter Numbers> /*服务描述*/

Parameters:<Parameter Name>,<Parameter Value> /*服务的参数描述*/

协议包含了通信双方的地址信息和构成业务的服务信息，其中的服务信息由业务代理解析业务逻辑链路表得到，它是一个有序的服务信息列表。

业务通信协议的实现采用了XML的格式，称为业务请求消息(Business request message)。消息包括两部分，请求头部和请求主体。请求头部包括通信双方的地址信息以及当前服务的执行结果；请求的主体是一个特殊的业务逻辑链路表，包括服务的个数和对应的服务信息。

当业务执行节点收到业务请求消息后，首先，解析请求消息获取服务参数，完成服务的调用，将业务中的剩余服务数减1；接着，根据请求消息获取下一跳服务功能名称，查找下一跳服务；然后，生成新的业务请求消息，发送到下一个服务所在执行节点。新的业务请求消息的头部是通信双方地址信息和执行结果，主体是剩余服务的个数和对应的服务信息。这个

过程重复多次直到业务请求消息中的业务所包含的服务数为 0，最终完成一个业务实例的执行，返回执行结果。

采用这种机制，BEPSON 每一个中间的业务执行节点只需要保存与它交互的执行节点的信息，不需要知道全局的信息，从而可以动态和灵活地完成业务的执行。

此外，为了提高业务执行的容错性，我们提供了一个简单的容错机制，就是每一个执行节点在选择下一个服务节点时，不是选一个，而是选择满足条件的几个作为备选。这样当它把消息发送给下一个服务，没有收到确认消息或者超时的情况，可以重新选择服务节点来完成业务执行。

4　系统实现

目前，我们已经使用 Java 语言实现了 BEPSON 的一个原型系统，采用开源的 Open Chord[7]作为 chord 协议。该原型系统作为一个中间件部署在业务执行节点上，构建了一个分布式的业务运行环境。下面我们使用一个贷款流程业务 Loan Business 来说明 BEPSON 如何完成业务执行。

4.1 业务流程描述

贷款流程包含两个服务：风险评估 (RiskAsse—ssment) 和贷款审批(LoanApproval)服务。“风险评估”服务用来查询个人的信用风险，“贷款审批”服务用来获得专家对贷款申请的审批意见。

首先，客户发送贷款审批请求，包括客户个人信息和贷款金额(amount)。其次，使用风险评估服务，对于低于 10 000 的贷款并且是低风险的客户，将同意贷款。对于高额贷款和信用不确定的客户，使用贷款审批服务提供的功能。最终，客户将得到是否同意贷款的结果。我们使用 ActiveVOS Designer 设计了该业务流程，如图 2 所示。

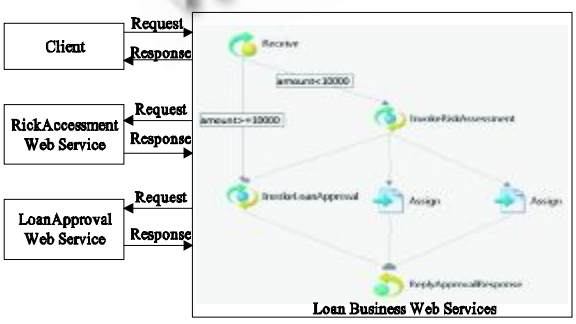

图 2　贷款业务流程

4.2 执行过程

图 3 给出了 Loan Business 的执行过程。

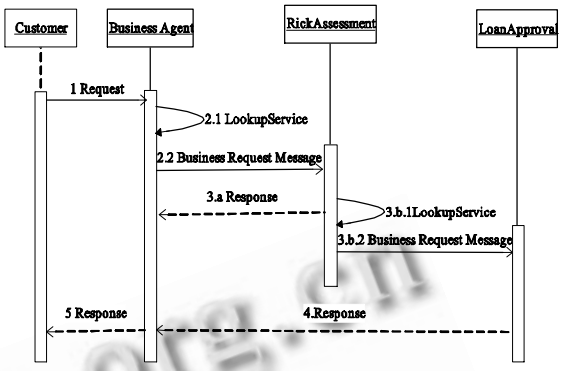

图 3　Loan Business 的执行过程

客户的业务请求消息首先到达业务代理，业务代理根据业务逻辑链路表，查找第一个服务 RiskAsse—ssment，发送业务请求消息给 RickAssessment 所在执行节点。接着，RickAssessment 所在执行节点，对用户信用进行评估并判断用户输入的金额(amount)，如果 amount<10000 且用户信用良好，同意贷款，将结果返还业务代理；如果 amount>=10000 或用户信用不能确定时，查找下一个服务 LoanApporval，发送业务请求消息给 LoanApproval 所在节点，完成贷款审批，同时检查消息中的业务逻辑链路表，发现自己是最后一个服务，将执行结果返回给业务代理，并由后者呈现给客户。

5　总结

本文提出的分布式业务运行平台 BEPSON，以服务覆盖网络为基础，采用分层架构设计，在动态服务发现算法中考虑了 QoS 因素，并采用了分布式的执行策略，实现了在广域环境下，业务的动态和分布式运行。运行实例验证了平台的有效性和灵活性。

在进一步的工作中，我们将以 BEPSON 平台为基础，考虑语义与 QoS 的结合，并引入到服务描述和发现机制中。

参考文献

1 Duan Z, Zhang ZL, Hou YT. Service overlay networks: SLAs, QoS and bandwidth provisioning. In: Proceedings of IEEE 10th International Conference on Network Protocols (ICNP), 2002.334—343.

(下转第 14 页)

(上接第 4 页)

2 Pautasso C, Alonso G. JOpera: a Toolkit for Efficient Visual Composition of Web Services. International Journal of Electronic Commerce (IJEC), 2004/2005, 9(2):104—141.

3 Casati F, Ilnicki S, Jin L. Adaptive and dynamic service composition in e-Flow. HP Labs Technical Report, HPL-200039. Palo Alto: Software Technology Laboratory, 2000:13—31.

4 李扬,怀进鹏,郭慧鹏,杜宗霞.一个基于服务层叠网的分层服务组合框架.软件学报, 2007,18(12):2967—2979.

5 Benatallah B, Dumas M, Sheng QZ, Ngu AHH. Declarative composition and peer-to-peer provisioning of dynamic Web services. IEEE Computer Society. San Jose, 2002:297—308.

6 Stoica I, Morris R, Liben-Nowell D, Karger DR, Kaashoek MF, Dabek F, Balakrishnan H. Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications. IEEE/ACM Transactions on Networking, 2003,11(1):17—32.

7 Open-Chord. http://sourceforge.net/projects/open-chord. 2009-06-07.