一个面向服务的应用案例研究*)

郭长国[1,3] 周明辉[2] 刘东红[1] 尹 浩[1]

(国防科技大学计算机学院 长沙 410073)[1] (北京大学信息科学技术学院软件研究所 北京 100871)[2]

(中国电子设备系统工程公司 北京 100039)[3]

摘 要 本文从一个现有的信息系统出发,通过分析因为业务增长和环境变化而带来的问题,提出需要进行面向服务的改造。根据面向服务系统的基本原则,从请求之间的状态、请求方式、接口的多样性和体系结构等方面阐述了该系统的改造。本文所描述的SOA改造的方法和体系结构设计以及面向服务改造后带来的优点具有普遍的意义。

关键词 面向服务的体系结构,CORBA

A Case Study of SOA Application

GUO Chang-Guo[1,3] ZHOU Ming-Hui[2] LIU Dong-Hong[3] YIN Hao[3]

(School of Computer Science, National University of Defense Technology, Changsha 410073)[1]

(Institute of Software, School of Electronics Engineering and Computer Science, Peking University, Beijing 100871)[2]

(China Electric Equipment and Systems Engineering Ltd., Beijing 100039)[3]

Abstract Starting from an existent information system and analyzing problems that the system are confronted due to the growing business and the changing environments, this paper points out that information system should be reconstructed from the perspective of the service-oriented architecture. According to the basic rules of constructing SOA system, the paper describes the reconstructions of a representative case of information system from the aspects of the request states, the diversities of request means and interfaces, and software architectures. The methods of SOA reconstruction and software architecture design proposed in this paper and the advantages brought by SOA reconstructions have universal significance.

Keywords SOA, CORBA

1 引言

随着网络技术的快速发展,传统的以数据为中心的软件应用被越来越多的松耦合软件应用系统替代,它们在广泛分布的Internet环境下运行,成为一种普遍的软件开发方法[1]。这种方法就是面向服务的体系结构(Service Oriented Architecture, SOA)。SOA是"抽象、松散耦合和粗粒度"的软件体系结构,它将IT资源通过服务这样一个在业务上有重要涵义的概念来提供和共享,可以根据需求通过网络对松散耦合的粗粒度应用构件(即服务)进行分布式部署、组合和使用[2,3]。

SOA不是抛弃现有的基础设施,而是在充分利用现有基础设施的基础上进行"松耦合"的改造。无论是Web Services[4,6],或是CORBA(Common Object Request Broker Architecture)都可以是SOA的一种技术实现方式,都可以成为SOA的技术实现架构,但也都不足以构成SOA的全部。

通过透视SOA的本质,我们认为,实行SOA需要遵循如下几个原则:首先,SOA不是把现有的信息系统重新实现一遍,而是进行SOA封装,SOA非常强调重用自己的资产及其伙伴的业务功能,来构造新的应用程序;其次SOA中服务的粒度比通常对象和模块中方法的粒度要大,一般情况下SOA服务的方法是没有状态的,这也是"松耦合"的一个具体的体现。当然,粗粒度和无状态不是机械的功能组合,粗粒度不仅要给客户带来易于访问的效果,而且要给服务带来易于部署、易于扩展的能力。第三,SOA中,客户一般使用最基础的网络设施即可以访问服务,不需要安装特殊软件,这种访问的便捷性和灵活性是松耦合的另一个具体体现,而且在SOA中,服务功能的变化不应该影响到客户的访问。在下文的案例中,可以看到这些原则的具体体现。

本文从一个现有的信息系统出发,通过分析因为业务增长和环境变化使得它所面对的问题,提出需要进行面向服务的改造。根据面向服务系统的基本原则,从请求之间的状态、请求方式、接口的多样性和体系结构等方面阐述了该系统的改造。本文所描述的SOA改造的方法和体系结构设计以及面向服务改造后带来的优点具有普遍的意义。

本文内容组织如下:第2节介绍了现有的一个信息交换系统,从描述它的应用需求出发阐述了它的技术发展过程:从直接使用socket进行数据交换到采用CORBA技术,再到对面向服务的迫切需求;第3节针对SOA构造原则,对该信息系统进行改造,并描述了改造后的体系结构及其优势;最后对全文作出总结。

2 现有信息系统介绍

如图1所示,现有的信息系统是一个广域网范围内的信息交换系统。

*)本文得到863课题的资助(2006AA01Z198)。**郭长国** 博士后,主要研究方向为分布计算和实时计算;**周明辉** 博士后,主要研究方向为分布计算、软件工程和可信计算。

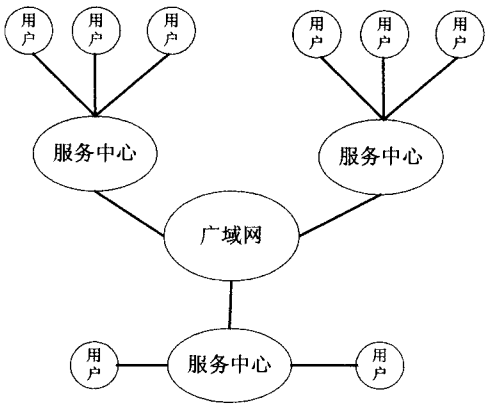

图 1　现有信息系统结构图

每一个客户通过自己的服务中心向其它用户发送消息，或者接收其它用户发送给自己的信息(一般情况下信息量比较大)，然后进行处理。虽然实际的业务逻辑十分复杂，进行抽象后，客户和服务器之间的交互可以简化为如图 2 所示的逻辑。客户首先向服务器注册，注册成功后，就获得了一个服务器的会话对象，其后的操作都在该会话对象上发生，直至客户注销了本次会话。在该信息系统中，客户和服务器之间的接口经过了多次的演化和发展。最初它们之间没有明晰的接口，直接使用 socket 进行数据交换，这时的接口就是双方对传送的数据格式的相互约定。这样带来的问题是开发困难：双方不仅需要对数据的语义严格约束，而且对数据的打包、解码和编码格式都要严格约定。同时，每一个功能扩展都要对这些约定进行修改，代价很高。

在第二阶段，针对上述问题引入 CORBA 技术进行解决。CORBA 定义了 IDL 来描述客户和服务之间的接口，规范了双方关于数据的打包、编码和解码格式等，提供 ORB(Object Request Broker)实现客户和服务的通信，并采用代理方式实现远程调用：远程对象在本地的代理对象是 stub 程序，客户程序通过访问 stub 程序实现对远程对象的完全透明的访问；服务端使用 skeleton 程序接收 stub 程序的调用请求，并派发给真正的对象实现，执行相应的方法。这种方式抽象了底层分布环境(网络、主机、操作系统、编程语言)的复杂性和异构性，解决了异构网络环境下分布应用软件之间的互连、互通和互操作问题[7,8]，给开发带来了极大的便利。

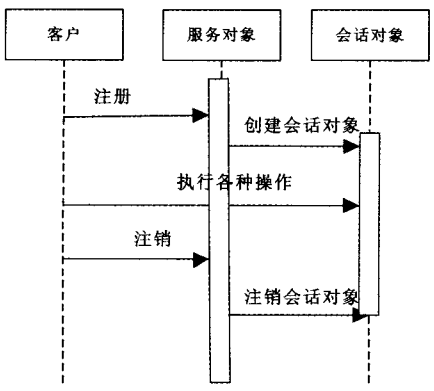

图 2　信息系统的抽象辑

但是随着业务系统的成长，对信息系统的改造越来越迫切。首先，用户分布的地域越来越广泛，用户和服务器之间的信道不仅仅是局域网，而是涉及电话、手机、GPRS、卫星等通讯信道。这些信道发生断线的概率很高，而且数据传输速度有限，从而导致客户常常在执行操作时断线，这时服务方会认为会话异常终止。而客户方要想重新执行操作，也必须重新从注册开始。这大大影响了电话、手机和 GPRS 客户的正常使用。其次，随着业务拓展，很多其它新生的用户或者系统也需要使用该业务系统。这时就必须为这些新的用户安装客户软件，比如 CORBA ORB 等等。这一方面限制了用户的发展，另一方面也制约了服务的成长，因为客户使用 IDL 接口访问服务，而服务接口的任何扩展都需要客户方做出相应的变化。而且对客户来说，使用 IDL 访问服务的开发难度也较大。这些都要求业务系统能够进行面向服务的改造。面向服务改造应该达到的效果是支持不同链路的客户的访问，特别是能够支持频繁断线的客户，甚至客户可以离线处理各种操作，而且最好客户无需安装特殊的软件，即使安装也最好能够和服务的升级和扩展相独立。针对以上需求和目标，我们对该信息系统进行了面向服务的改造。

3　信息系统面向服务的改造

根据面向服务系统的基本原则，我们对上节的信息交换系统进行了改造。3.1 节重点阐述改造内容，3.2 节描述改造后的体系结构；3.3 节对改造后的系统进行分析。

3.1　面向服务改造的内容

请求的无状态改造

原有的信息系统中，真正的业务操作是有状态的。一是在业务操作之前必须进行注册，所有注册之后的操作都在一个会话当中；二是两个业务操作可能具有顺序关系，必须先完成前者。这样，每次操作失败(由于链路和通讯引起的失败)都需要对状态进行恢复(比如，任何中间的失败都必须重新从注册开始等等)，难以保证在广域网和恶劣信道下的使用。

因此，面向服务改造的第一步就是消除业务操作之间的状态联系，让每一次操作都是互相独立的。以抽象的信息系统逻辑为例，假设要业务操作 f_i 和 f_j 具有顺序关系，并且执 f_j 行之前需要先执行 f_i，并且 f_i 的参数为 p_i，f_j 的参数为 p_j，即 $f_i(p_i)$，$f_j(p_j)$。假设注册为 L，参数为 p_l，注销为 M，参数为 p_m，即 $L(p_l)$，$M(p_m)$。这时改造成 f，参数为 p_l，p_i，p_j，p_m，即 $f(p_l,p_i,p_j,p_m)$。这时 f 由多个服务器的本地操作组成，即 $f=L(p_l)\cap f_i(p_i)\cap f_j(p_j)\cap M(p_m)$，即为先执行 $L(p_l)$，再依次执行 $f_i(p_i)$ 和 $f_j(p_j)$，最后执行 $M(p_m)$。这时，这些内部的操作都是在服务方一次完成，而不是由客户发送多次请求(这时仅仅发送一次请求)。这里没有考虑业务操作的返回值，如果业务操作有返回值，可以对上述方法进行简单修正即可，不影响该改造方式的一般性。实际的业务系统中是有返回值的。这种消除状态的改造是最基本的面向服务的改造之一。经过无状态的改造后，对那些频繁断线的客户提供了一种有效的解决方案。

无状态改造有一个效率问题，即从单个请求来看，传递的参数变得多了，每一个请求都需要携带注册和注销的参数 p_l 和 p_m，而且服务方的负担看来也增加了不少，因为每一个操作都要进行注册和注销操作。但是实际上，在广域环境下，尤其是对那些通讯链路差的客户来讲，主要的瓶颈不是服务方的执行速度，而是网络速度和链路不稳定导致的请求多次尝试的代价。无状态改造后，从完成业务的角度来讲，客户发送请求的次数减少了，这些组合的多个操作实际上是在服务端执行的，不需要客户多次请求，服务方也不需要多次和客户进

行会话。因而对这些客户来讲，无状态解决了他们的主要瓶颈，并且不会带来性能的损失。

请求的方式和接口

如前所述，现在的业务系统使用 CORBA 进行通讯。这要求在客户上安装系统，至少也要安装 ORB，因为我们不能假设所有的客户系统中都有 ORB，但是我们现在可以假设客户系统上基本都有浏览器，都有编写文本的工具和 FTP 客户端以及收发送 EMAIL 的能力（通常的操作系统都缺省支持）。这就为我们的改造提供了新的思路：即让客户尽量少甚至不安装软件就能够访问服务。在面向服务的改造中，我们提供了多种服务的接入方法：首先，客户可以通过多种方式请求服务，包括 CORBA（使用 IIOP 协议），SMTP，FTP 和 HTTP。对于客户通过 CORBA 访问，毋庸多说，对 SMTP，FTP 和 HTTP 来讲，首先要制定访问服务的数据格式。以 SMTP 为例，我们规定如果 EMAIL 的主题是“request for X400 service”（X400 service 是我们的服务名称），我们就认为该 EMAIL 是对服务的一次请求，而请求的具体描述，包括请求名称、参数等等都在 EMAIL 的正文之中。我们使用 XML Schema 规定请求体的格式。对 FTP，客户使用发送符合我们制定的 XML Schema 格式的文件来请求服务，对 HTTP 客户则发送相应的表单（FORM）来提交对服务的请求。而应答则以 XML 文件的格式返回给客户端。经过对请求方法的面向服务的改造，客户在现有的设施下，无需安装任何特定的软件就可以访问服务，并且客户的处理可以是异步的，而且对网络链路恶劣的客户可以充分发挥网络基础设施的作用，比如 FTP 客户端就可以使用断点续传能力等等。同时，服务方的升级和接口的变化都不会对这些客户端造成影响，真正做到了面向服务的承诺。

同时，关于请求的方式，我们还允许使用队列系统来访问我们的服务，比如使用 IBM 的 MQ 以及微软的 MSQ 等等。这些是为了保证请求的可靠性。

3.2 体系结构

综上所述，面向服务改造后的信息系统如图 3 所示。

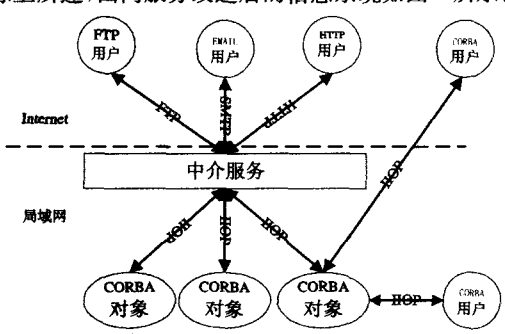

图 3　面向服务的体系结构

中介服务是服务的入口，它可以接收 FTP 请求、SMTP 和 HTTP 请求。在验证请求的合法性后，将这些请求转化为内部的 IIOP 请求。请求 CORBA 对象获得结果后，再将结果转化为合适的格式返回给客户。同时，在局域网内部或者外部，客户依然可以直接使用 IIOP 协议访问服务对象。前提是这些客户安装了客户所需要的软件，并且具有较好的网络状况。

3.3 负载均衡

在经过面向服务的改造后，服务方的负载均衡可以更加方便地实施。在未进行改造时，任何业务请求均在某个会话中，所以一旦某个 CORBA 服务响应了客户的注册请求，该客户在注销之前的所有操作都必须在该 CORBA 服务上执行，因为其它 CORBA 服务没有该客户的会话信息。在进行面向服务的改造后，每一个请求都是独立互不联系的，所以对每个请求都可以进行负载均衡，即使是同一个客户发送的请求也是如此。这样就给系统的可扩展带来了极大的便利。如图 3 所示，中介服务在接收到请求后，可以派发给任何一个空闲的 CORBA 对象进程处理，因此中介服务同时又是负载均衡器，并且 CORBA 对象还可以动态的增加和减少，极大地便利了系统的扩展。这些都是面向服务改造带来的优势。

结论　面向服务虽然增加了请求的粒度，某种意义上增加了服务的计算量（比如该案例中每次服务请求都需要注册和注销）。但是从“服务”的意义上来讲做到了灵活的为客户提供服务的能力。使客户能够随时随地的利用已有的网络基础设施流畅的享用服务，在 Internet 下的主要矛盾不是服务器的计算量，可以说，面向服务的思想会在 Internet 下大放其彩，可以预见各种信息系统都面临着面向服务的改造。

本文从一个实际的系统出发，通过对业务逻辑的抽象，给出了面向服务改造的一个案例。改造后的面向服务的系统已经得到了很好的运行，取得了不错的效果。通过该案例，我们可以看到，面向服务不是对现有系统的重新开发。在系统中，核心的功能部分没有变化，而是对获得服务的方式进行了改造，使得客户能够更容易获得服务，服务的变化和客户不直接相关。同时，服务本身的可扩展（比如负载均衡）也得到了很大的提高，充分体现了“松耦合”的本质。这些方法和取得的效果具有普遍的指导意义。

然而，功能完整的 SOA 不仅仅是将应用开发成服务，还需要安全性、工作流程和管理等标准才能确保 SOA 的扩展能力和可靠性，但这些内容仍然处于开发阶段。这些也是我们信息系统，特别是核心信息系统面向服务改造过程中需要进一步解决的问题。

参 考 文 献

1 Tsalgatidou A, Pilioura T. An Overview of Standards and Related Technology in Web Services. Distributed and Parallel Databases, 2002, 12(2-3): 135

2 Sillitti A, Vernazza T, Succi G. Service Oriented Programming: A New Paradigm of Software Reuse. LNCS 2319, 2002. 269～280

3 Fei Y K, Wang Z J. A Concept Model of Web Components. In: Proc. of IEEE International Conference on Services Computing, 2004. 159～164

4 http://www.webservices.org/index.php/ws/content/view/full/466

5 Ge S, Hu C M, Du Z X, et al. WebSASE: A Web service based application supporting environment. In: Huh S Y, Lee B T eds. Proc of the 5th Northeast Asia Symposium. Seoul, 2002. 67～76

6 胡春明，怀进鹏，孙海龙. 基于 Web 服务的网格体系结构及其支撑环境研究. 软件学报，2004, 15(7): 1000～9825

7 IEEE. http://dsonline.computer.org/middleware/index.htm

8 Object Management Group. The Common Object Request Broker: Architecture and Specification. http://www.omg.org