基于 J2EE 平台面向服务的体系结构研究与应用

Research and apply of Service – Oriented Architecture based on J2EE Platform

李春霞 高大启 (华东理工大学 200237)
王新华 (上海远程教育集团 200433)

摘要:面向服务的体系结构(service – oriented architecture,SOA)因其固有的松散耦合与互操作性,成为许多企业应用的自然选择。本文针对国家教育资源库863项目中需要耦合多个自成体系的系统,提出了使用 J2EE 1.4 提供的 Web 服务功能进行构建能够访问现有业务系统的 SOA 框架。详细研究了 SOA 框架的构成和相关技术,并分析了在 SOA 框架上进行的几次迭代过程,给出了在国家教育资源库863项目的实际应用。

关键词:面向服务的体系结构(SOA) 服务 动态发现 消息 SOAP

1 引言

作为面向服务的体系架构,SOA 成为一种功能组件化的设计模型。它屏蔽了不同平台、编程语言、操作系统和硬件架构之间的差异,实现了应用程序的简单集成。这意味着 IT 系统的灵活性得到前所未有的提升,同时应用程序的重复应用成为可能,开发成本得到了降低。这就是 SOA 的魅力所在。本文以国家教育资源库项目为例将讨论 SOA 的实现,阐述 SOA 的实现必须遵循的原则,介绍基于 J2EE 平台的 SOA/Web 服务的实现方法以及对粗、细两种粒度实现 SOA 进行了分析。

2 SOA 的原则

SOA 是一种企业架构,因此,它是从企业的需求开始的。但是,SOA 和其它企业架构方法的不同之处在于 SOA 提供的业务敏捷性。业务敏捷性是指企业对变更快速和有效地进行响应、并且利用变更来得到竞争优势的能力。对架构设计师来说,创建一个业务敏捷的架构意味着创建一个这样的 IT 架构,它可以满足当前还未知的业务需求。要满足这种业务需求 SOA 必须遵循以下原则:

2.1 以服务为中心,贯彻业务驱动服务,服务驱动技术的理念

服务是能够通过网络访问的可调用例程。服务公开了一个接口契约,它定义了服务的行为以及接受和返回的消息,在抽象层次上,服务位于业务和技术中间。面向服务的架构设计师一方面必须理解在业务需求和可以提供的服务之间的动态关系,另一方面,同样要理解服务与提供这些服务的底层技术之间的关系。

2.2 通过"动态发现"实现业务的敏捷性是基本的业务需求

服务的接口通常在公共注册中心或者目录中发布,并在那里按照所提供的不同服务进行分类,就像电话簿黄页中列出的企业和电话号码一样。客户(服务消费者)能够根据不同的分类特征通过动态查询服务来查找特定的服务,这个过程称为服务的动态发现。提供响应变化需求的能力是新的"元需求",而不是处理一些业务上的固定不变的需求。从硬件系统而上的整个架构都必须满足业务敏捷的需求,因为,在 SOA 中任何的瓶颈都会影响到整个 IT 环境的灵活性。

2.3 消息是 Web 服务提供的产物,是业务系统耦合的对象

服务消费者或者客户通过"消息"来消费服务。因为接口契约是独立于平台和语言的,消息通常用符合 XML 模式的 XML 文档来构造。作为消息提供者的 Web 服务建立在开放标准和独立于平台的协议的基础之上。Web 服务通过 HTTP 使用 SOAP(Simple Object Access Protocol),以便在服务提供者和消费者之

间进行通信。服务通过 WSDL(Web Service Definition Language)定义的接口来公开,WSDL 的语义用 XML 定义。UDDI(Universal Description ,Discovery and Integration)是一种与语言无关的协议,用于和注册中心进行交互以及查找服务。所有这些特性都使得 Web 服务成为开发 SOA 应用程序的关键选择。

综上所述,在 SOA 中有三种角色,其不同角色间的关系图如图 1。

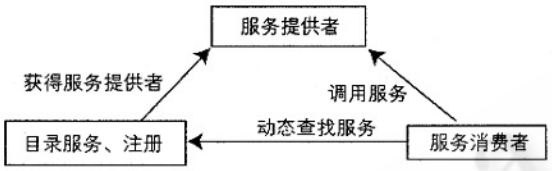

图 1

3 基于 J2EE 平台的 SOA/Web 服务的实现

J2EE 平台通过 JAX – RPC 1.1 API 提供了完整的 Web 服务支持,这种 API 支持基于 Servlet 和企业 bean 的服务端点。JAX – RPC 1.1 基于 WSDL 和 SOAP 协议提供了与 Web 服务的互操作性。J2EE 平台也支持 Web Services for J2EE 规范(JSR 921),后者定义了 Web 服务的部署需求并利用了 JAX – RPC 编程模型。除了几种 Web 服务 API 之外,J2EE 平台还支持 WS – I Basic Profile 1.0。WS – I Basic Profile 标准让 Web 服务克服了不同编程语言、操作系统和供应商平台之间的障碍,从而使多种应用程序之间能够交互这意味着除了平台独立性和完整的 Web 服务支持之外,还提供了跨平台的互操作性。

3.1 实现优势

Web 服务客户可以通过两种方式访问 J2EE 应用程序。

(1) 客户可以访问用 JAX – RPC API 创建的 Web 服务;在幕后 JAX – RPC 使用 servlet 来实现 Web 服务。

(2) 公开无状态的 EJB 作为 Web 服务;Web 服务客户也可以通过 bean 的服务端点接口访问无状态会话 bean。考虑国家教育资源库 863 项目的特点我们采用第二中方案,其优势如下:

① 利用现有的业务逻辑和流程。在教育资源库中,现有的业务逻辑大多已经使用 EJB 组件编写,通过 Web 服务公开它可能是实现从外界访问这些服务的最佳选择。EJB 组件是一种很好的选择,因为它使业务逻辑和服务位于同一层上。

② 并发支持。作为无状态会话 bean 实现的 EJB 服务端不必担心多线程访问,因为 EJB 容器必须串行化对无状态会话 bean 任何特定实例的请求。

③ 对服务的安全访问。企业 beans 允许在部署描述符中声明不同方法级别的安全特性。方法级别角色被映射到实际的 EJB 主体中。使用 EJB 组件作为 Web 服务端,把这种方法级别的安全性也带给了 Web 服务客户。

④ 事务问题。EJB 服务端在部署描述符规定的事务上下文中运行。容器处理事务,因此 bean 开发人员不需要编写事务处理代码。简化了 Web 服务端的开发。

⑤ 可伸缩性。几乎所有 EJB 容器都提供了对无状态会话 bean 群集的支持。因此当负载增加时,可以向群集中增加机器,Web 服务请求可以定向到这些不同的服务器。通过把 Web 服务模型化为 EJB 组件,可以使服务具有可伸缩性,并增强了可靠性。

⑥ 池与资源管理。EJB 容器提供了无状态会话 bean 池。这改进了资源利用和内存管理。通过把 Web 服务模型化为 EJB 组件,这种特性很容易扩展,使 Web 服务能够有效地响应多个客户请求。

3.2 SOA 服务框架的实现

在教育资源库项目中,它的各种子系统之间(计费系统、耦合系统、分布式检索)需要彼此交互,而且还向外界提供服务支持,以便不同的业务系统之间进行交互,还要为各种业务应用程序提供对外的 Web 解决方案。最佳选择就是采用 SOA 框架,通过无状态 EJB 组件把各种服务和业务流程公开为 Web 服务。

下面对 SOA 框架中的组件进行解释:

(1) 客户(Client)。用户通过 Web 浏览器与不同的应用程序交互,浏览器作为应用程序的客户。

(2) 应用程序控制器(Application controller)。应用程序控制器是您的主控制器 Servlet。它负责初始化、委派请求和响应请求处理程序。

(3) 请求处理程序(Request processor)。这是一

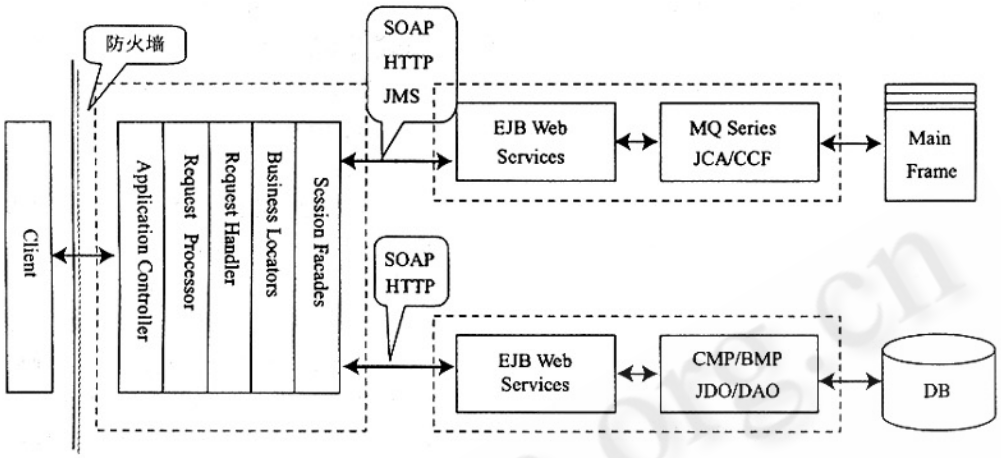

图 2 企业内部的 SOA 框架

个 Java 类，通过调用相应的请求执行程序完成要求的处理，对请求进行预处理。

(4) 请求执行程序(Request handlers)。请求执行程序完成具体请求的活动，比如与服务交互，向不同的企业信息系统(enterprise information systems，EIS)增加或检索信息。请求执行程序依靠业务定位程序发现相应的服务，然后通过这些服务访问需要的 EIS 信息。

(5) 业务定位程序(Business locators)。这些程序负责隐藏查找服务的复杂性，并提供缓存逻辑。业务定位程序可以采用多种形式，比如 Web 服务、EJB 组件、JMS 定位程序。

(6) 会话 Facades(Session Facades)。通过聚合来自多个系统或服务的方法，简化复杂对象的视图。会话 facades 是 EJB Web 服务方法的包装器。

(7) EJB Web 服务(EJB Web services)。根据 EJB 1.4 规范，Web 服务端可以模型化为无状态的会话 bean。

(8) 数据访问接口。使用不同的技术(比如 EJB－CMP、JDO、DAO) 和不同的持久性技术访问 EIS，所使用的访问技术取决于接口需求以及获取、插入或更新的数据量。这一层负责与 EIS 进行交互，并以相应的 EJB Web 服务方法所期望的格式把数据返回给这些方法。

(9) MQSeries/JCA/CCF。现有的基于主机的服务可以公开为 Web 服务，从而向外界展示它们。Web 服务客户使用基于 HTTP 的 SOAP 协议与 EJB Web 服务交互。EJB 方法通过 JMS 协议向 MQSeries 队列发送请求。主机端的 MQSeries 服务器触发相应的基于 COBOL 的程序，后者为与后台系统进行交互提供必要的逻辑。然后这些程序把响应返回到队列中，应用程序逻辑检索这些响应并返回给 EJB 方法。

3.3 两种粒度实现 SOA 服务

通过 SOA 服务框架我们了解到：SOA 帮助企业信息系统迁移到“leave－and－layer”架构之上，这就意味着在不用对现有的企业系统做修改的前提下，系统可对外提供 Web 服务接口，因为它们已经被可以提供 Web 服务接口的应用层做了一层封装，SOA 可以将系统和应用迅速转换为服务。SOA 不仅覆盖来自于打包应用、定制应用和遗留系统中的信息，而且还覆盖来自于如安全、内容管理、搜索等 IT 架构中的功能和数据。因为基于 SOA 的应用能很容易地从这些基础服务架构中添加功能。所以，基于 SOA 的应用能更快地应对市场变化，使企业业务部门设计开发出新的功能应用。可以按基于服务的功能及发送和接收的数据数量来定义服务，如细粒度服务、粗粒度服务或组合服务。

在 SOA 中服务粒度有两种相关的意思，即服务是如何实现的，服务使用和返回了多少数据或多少消息。细粒度服务执行了最小的功能，发送和接收少量的数据。粗粒度服务执行了较大的业务功能，并交换了更多的数据。

细粒度服务是供粗粒度服务或组合服务使用的，而不是由终端应用直接使用的。如果应用是使用细粒度服务建立的，则应用将不得不调用网络上多个服务，并且发生在每个服务上的数据量较少，因而会对对系统整体性带来影响。所以，粗粒度服务的用户不能直

接调用它所使用的细粒度服务。然而，由于粗粒度服务可能使用多个细粒度服务，因此它们不能提供粒度级的安全和访问控制。安全和访问控制必须在细粒度服务中实现。

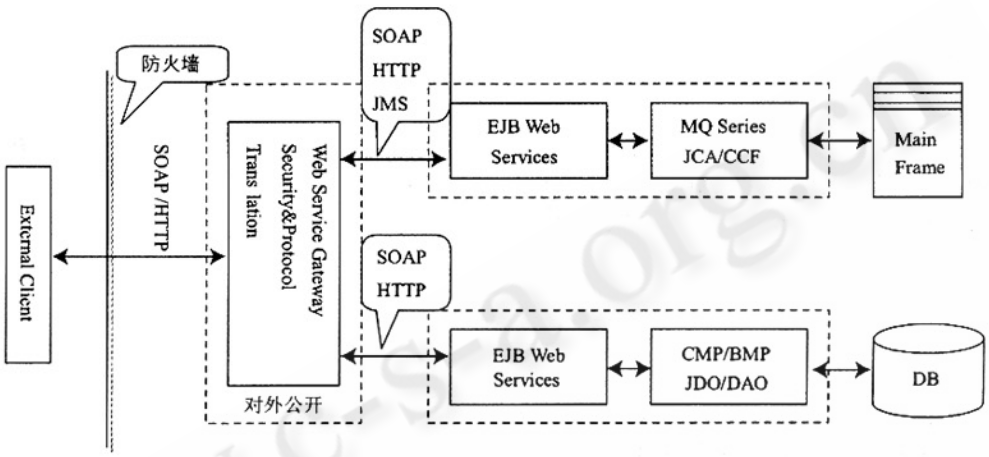

图 3　向外界公布细粒度服务 SOA 框架实现

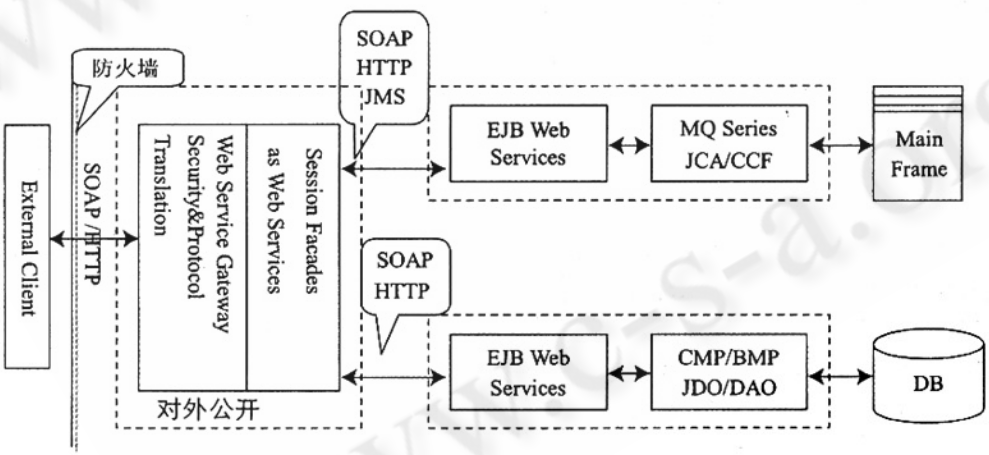

图 4　向外界公布粗粒度服务 SOA 框架实现

组合服务可以使用粗粒度服务和细粒度服务进行组装。数据数量不是粗粒度服务和组合服务之间的区别。粗粒度服务例子，如创建新用户，在这一过程的操作是：需要通过一些外部服务验证对客户进行验证，并在资源库应用系统中创建用户记录。组合服务例子可以是提供一个新的验证基线，这需要一个服务调用来验证用户是否满足在资源库中注册的权限（用户注册之前必须获得系统分配的合法性标识和注册口令）。即所有注册用户必须满足系统服务的安全性验证机制。

通过图 3 和图 4 做对比可以看出细粒度服务通过 Web Services Gateway 向外界提供了一个服务列表。并且每个业务服务作为单个业务过程执行，而粗粒度服务已经公开了 Session Facades 作为 Web 服务端，根据需要，可以实现粗粒度服务和细粒度服务的某种结合，通过调整 Web 服务网关中间件来向外部客户公开两种服务。

4　结束语

本文针对资源库项目中包含的子系统多，结构复杂以及资源的分布存储和系统异构性的特点，给出了 SOA 的实践案例。

SOA 的强大和灵活性将给企业带来巨大的好处。对大多数公司来说，下一步要考虑的不再是点对点的应用，而是 Web services 在企业间以及业务伙伴间更为宽广的应用。这种技术的变迁需要更松散耦合、面向基于标准的服务的架构。这样一个架构要求对 IT 在组织中的角色有新的观点和认识，而不仅仅是一种实现方法。通过对业务的敏捷反应，企业可以得

（下转第 75 页）

（上接第 71 页）

到实实在在的回报，而要达到这一点，面向服务架构设计师的角色非常关键。除此之外，潜在的回报更是不可胜数，而分布计算技术能够保证对业务需求足够灵活的反映，这种业务上的敏捷正是各公司梦寐以求的。

参考文献

1 柴晓路，Web 服务架构与开放互操作技术[M]，北京 清华大学出版社，2002。

2 Li Gong《Inside Java2 Platform Security》—Architecure，API Design，and Implementation，北京 电子工业出版社，2004－9。

3 W3C Note. Simple Object Access Protocol (SOAP)1.1[EB/OL]，http://www.w3.org/TR/WSDL,2000－05.

4 W3C Note. Web Service Description Language(WSDL)1.1[EB/OL].
http://www.w3.org/TR/WSDL,2001－03

5 W3C Working Draft 14，Web Service Architecture[EB/OL].
http://www.w3.org/TR/2002/WD－ws－arch－20021114,2002－11

6 宋善德、王雪飞，基于 Web 服务的企业应用集成方案[J]，计算机应用研究，2003，20(6)：127－129，160。