EJB组件开发模型及实现技术

袁国勇 （北京中央财经大学 100081）

摘要：本文提出了一种基于EJB组件的企业分布式多层应用体系结构，分析了其与传统应用体系结构相比的优点。同时就模型中涉及的服务器端EJB组件开发技术和流程作了一些简单介绍。

关键词：EJB Client/Server Client/Network 分布式计算 企业应用

1 EJB组件开发模型

任何良好的软件设计都是采用分层的思想，每个层面相互独立，只需知道下层通过接口向本层提供何种服务，本层向上一层提供什么服务接口。分层结构中任何一层只要其提供的接口不改变，实现发生任何变化，对其他层没有任何或最少影响。分层后不仅使各层功能变得简单且易实现，而且具有更好的可维护性和可扩展性。

软件分层有逻辑分层和物理分层的区别。在逻辑上，应用常被划分为表示层、商务逻辑层和数据层三层。每层分别完成不同的任务，并由一个或多个组件来实现。表示层主要包括处理系统与用户交互的GUI组件，如JSP、Java Servlet等；商务逻辑层是表示层与数据服务层交互的媒介，主要包括解决业务问题的组件，如EJB组件、CORBA组件和DCOM组件等；数据层负责向商务逻辑层提供数据，通常由一个或多个数据库系统组成。

在物理上软件分层又存在基于两层的客户/服务器模式和基于N层的客户/网络模式之分。传统上企业应用软件大都采用两层体系结构，就实现方式上说，其经历了将表示层和商务逻辑层组合成一个层面，向将商务逻辑层的部分放入数据层的演变，其结构分别如图1和图2所示。这种演变通过将商务层逻辑的数据访问功能转移到数据库的存储过程中，一定程度上减轻了数据访问造成的网络堵塞，但并没有在软件可维护性、可扩展性和可重用性等方面得到改进，特别是很难适应Internet发展的需要。

N层体系结构是在两层体系结构的基础上增加了一层或者更多层，从而将每个逻辑层在物理上进行进一步的分割，以便更加独立地完成其功能，其结构如图3所示。基于N层体系结构的客户/网络模式与仅把应用处理与数据管理分开（单层次连接）的客户/服务器模式有较大的区别，因为它兼顾了主机集中式和客户/服务器分布式这两种应用模式的优点：既能使信息高度分散，而实现资源共享，又可使管理高度集中而降低成本。

本文根据N层体系结构的理论模型提出一种基于EJB组件的企业应用模型。表示层包括两种客户交互方式：WEB客户和GUI应用客户。前者由客户通过浏览器从一个或多个WEB服务器下载静态的HTML页面或由JSP和Servelt动态生成的HTML页面实现，后者由包括流程控制和接收用户输入并返回响应结果的Java应用程序实现。商务逻辑层由包含了实体Bean的会话Bean组成，所有的EJB都在应用程序服务器/容器构建的空间里运行，并利用其提供的中间件服务（包括分布式通信、事务控制、安全服务、持久性和负载平衡等）来完成对底层数据库的访问和响应客

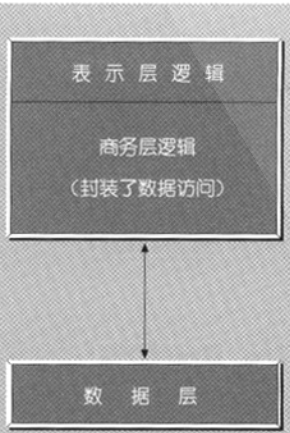


图1 组合了表示层和商务层的两层体系结

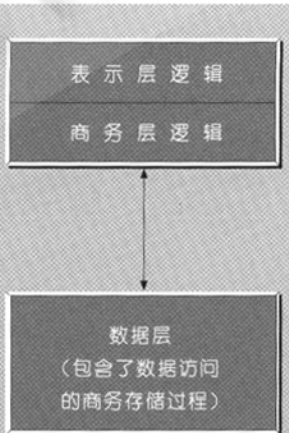


图2 将包含数据访问的商务逻辑放入数据层

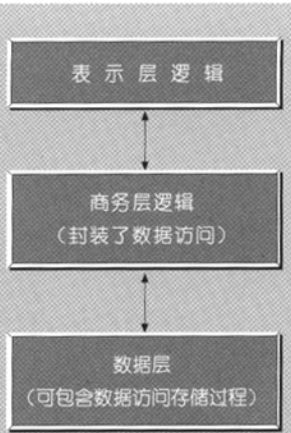


图3 N层体系结构

户端的请求。数据层由数据库系统、ERP和历史遗留系统组成，其中可能包括存储过程及数据存取相关的模块。该模型具有以下优点：

(1) 在网络要求方面：由于数据都在应用服务器上进行分析处理，客户端只是起个显示结果的作用，所以传送数据量小，对网络要求不高。充其量只要求在应用服务器与数据服务器间采用高速网络连接。

(2) 在可维护性方面：由于数据访问都放到应用服务器中，更改数据源或数据库，只需修改服务器端部署文件，而无须对数量繁多的客户端重新配置。另一方面由于商务逻辑都由组件实现，因而只要其发布的商务接口不改变，其商务内容发生变化，仅需修改商务组件的实现，而客户端无须做任何修改。这就大大减少了维护的工作量。

(3) 在可扩展性方面：如果应用服务器严重过载，可方便地通过增加新的应用服务器，而不用通过硬件升级的方法来实现。

(4) 在系统稳定性方面：能使故障被限定在一个层面上，而不发生扩散，其他层的应用常包含了正确处理这种紧急情况（例外）的代码或将应用转移到其他计算机的能力。

(5) 在系统安全性方面：既可通过防火墙技术限制外部用户对重要的商务数据的访问，也可通过应用服务器提供的安全管理服务实现到商务方法级的安全访问。

2 EJB技术简介

2.1 EJB定义和角色

SUN公司对EJB的定义是用于开发和部署多层结构的、分布式的、面向对象的Java应用系统构件体系结构。使用EJB结构编写的应用程序具有跨平台性、可扩展性、交互性以及多用户安全特性。这些应用只需写一次，就可以发布到支持EJB规范的服务器平台上。

从EJB的定义可知其是采用"分而治之"的方式解决服务器端应用问题。EJB将一个项目的完成分成六个组成部分，每个部分分别对应一个角色，完成不同的任务。这六个角色分别为组件供应商、容器供应商、服务器供应商、应用集成商、系统部署员和系统管理员。

(1) 组件供应商：提供可重用的商务组件或他们相应产品的EJB连接器，常由应用领域的专家担任。

(2) 容器供应商：提供EJB应用运行所需的底层运行环境。

(3) 服务器供应商：提供应用服务器来管理、容纳和部署商务组件。容器供应商和服务器供应商可能是同一家公司。

(4) 应用集成商：利用EJB组件产品构造一个完整的应用。

(5) 系统部署员：根据应用集成商的建议和部署文件将EJB组件部署到一个或多个应用服务器中。

(6) 系统管理员：维护已经部署好的应用系统。

2.2 EJB组件类型

EJB组件有三种类型：会话Bean(Session Bean)、实体Bean(Entity Bean)和消息驱动Bean(MessageDriven Bean)。

会话Bean代表商务过程对象，它执行商务逻辑、算法、规则和工作流程，是具有商务过程逻辑的可重用组件。在某一时刻，会话Bean仅对一个客户可用，不能和其他客户共享。

根据多方法请求时是否维持客户状态，会话Bean又可分为无状态会话Bean和状态会话Bean。无状态会话Bean在每次方法调用后，

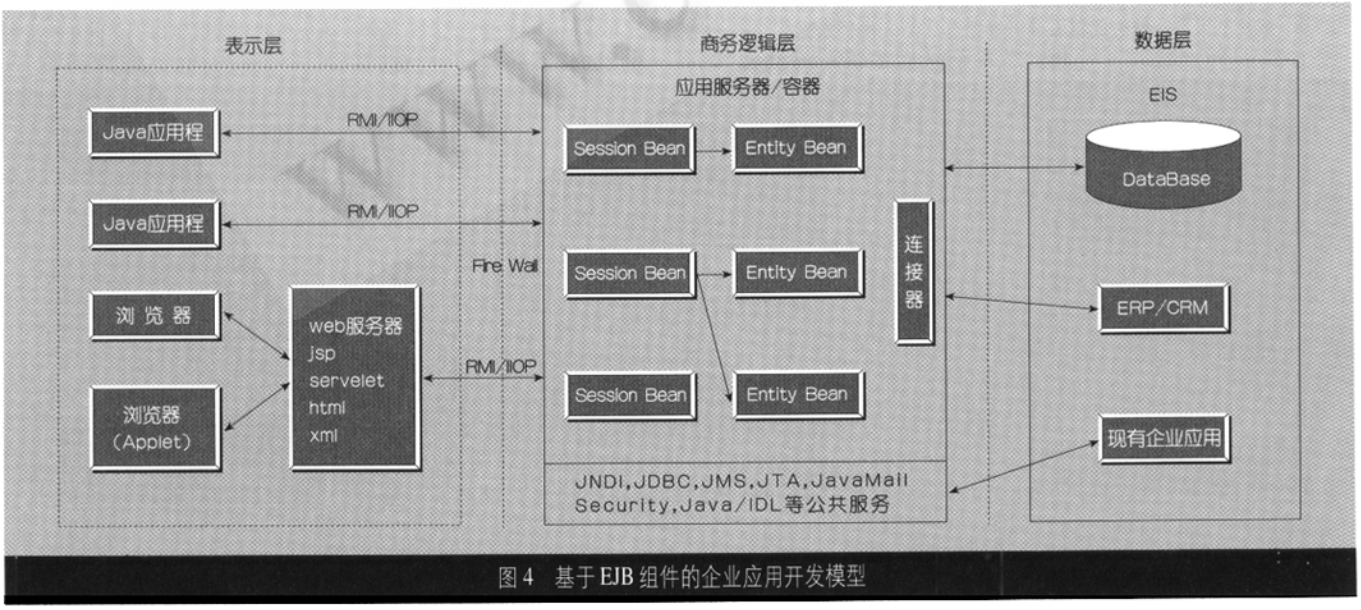


图4 基于EJB组件的企业应用开发模型

就清除属于前次方法调用的信息，而不保存方法之间的对话状态。无状态会话Bean在方法调用完成之后，立即被释放到缓冲池中，从而有更大的伸缩性和支持大量的客户。状态会话Bean是为处理多个方法请求和事务处理的商务过程而设计的。状态会话Bean在会话期间保留代表某个用户的状态，当一个方法调用中状态发生了变化，状态会话Bean能保证在后面针对这个用户的方法调用使用变化后的状态。为了提高系统的效率，状态会话Bean可以在一定的客户空闲时间后被写入二级存储设备(如硬盘)，在客户发出新的调用请求后，再从二级存储设备恢复到内存中。

实体Bean代表数据库或另外一个企业应用系统中的数据对象，如代表数据库的一行记录。实体Bean不包括商务逻辑，它们只是数据模型。其意义在于它将底层数据以对象的形式映射到内存中，供其他组件使用。从本质上说，实体Bean是在多层体系结构中，执行数据访问逻辑层的功能。

和会话Bean不同，实体Bean是持久的(persistent)，允许共享访问。因为它代表底层数据的映像，会和数据库记录保持同步，所以，即使当应用服务器崩溃或停止运行，实体Bean的状态还会保存在数据库中，不会丢失。

根据持久性划分，实体Bean又分为自我管理的实体Bean(Bean-Managed Bean)和容器管理的实体Bean(Container-Managed Bean)。前者由开发者自己完成实体Bean和数据库记录间的同步，后者由容器实现实体Bean和数据库记录间的同步。

在实体Bean和会话Bean中，客户端是以同步和阻塞的方式调用Bean，即调用者必须等待服务器返回一个响应后，才能进行下一步工作。消息驱动Bean则被设计用来处理异步的JMS消息，它无须收到响应，可以继续进行下一步工作。消息Bean将JMS和EJB相集成，是一种全新的企业Bean，它通常配置成是一个特别的主题（Topic）或队列的客户端，作为消息的使用者，消息Bean使用的消息可以来自其他Bean（会话Bean、实体Bean或消息Bean）、非EJB的Java应用程序、或者甚至非Java的应用程序（如果其供应商符合JMS）。例如，遗留应用程序可能使用IBM的MQSeries向队列发送消息，而该消息既可以由其他遗留应用程序使用，同样可以由消息Bean使用。

2.3 EJB体系结构

EJB体系结构中的主要应用程序对象如下：

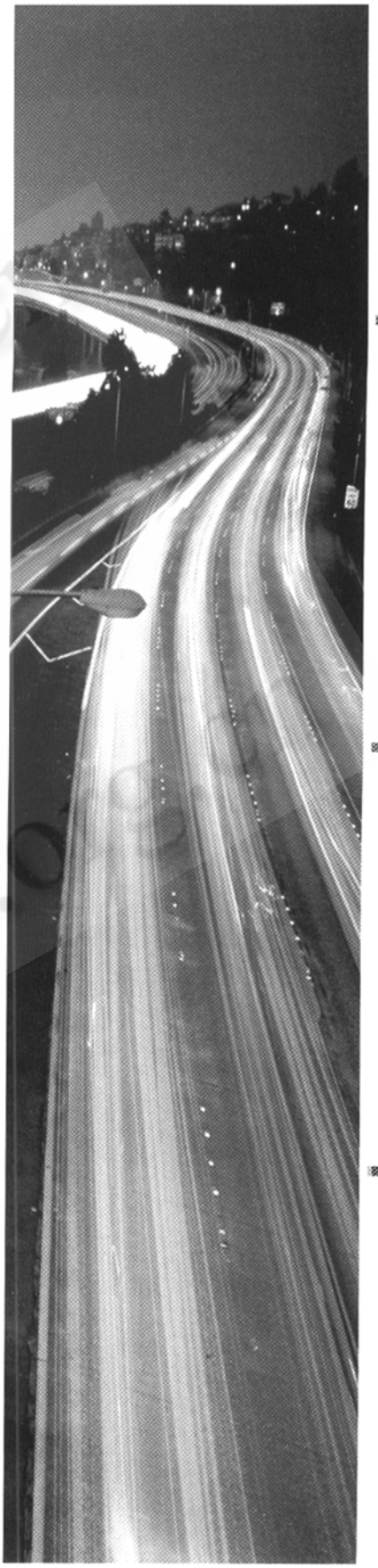
(1) EJB客户：包括WEB客户和非WEB客户两种，它们将使用JNDI来查找对本地接口（Home接口）与远程接口（Remote接口），并且使用EJB本地接口获得对EJB远程接口的引用，从而获得对EJB各种功能的调用。

(2) EJB本地接口（与存根Stub）：EJB本地接口为客户创建、删除与查找EJB远程接口对象的句柄提供了相应的操作。底层存根编码和发送本地接口请求到远程Skeleton，并解码接收到的响应。

(3) EJB远程接口（与存根Stub）：EJB远程接口为EJB客户提供了特定的业务接口方法。

(4) EJB实现：EJB实现是由开发者所实现的实际EJB应用程序组件。用来提供各种特定业务方法的具体实现以EJB规范所要求的各种容器调用方法，如创建、删除、查找、激活和钝化EJB对象的方法及存储和加载数据以保持实体Bean和数据库记录的同步。

(5) 容器EJB实现（与骨架Skeleton）：由容器提供商所实现的Home对象等，用来提供各种容器管理方法的具体实现。骨架用来对



与客户之间所传输的数据进行编码和解码。EJB体系结构中，客户端与EJB组件交互模型和处理流程如图5所示。

① 客户端利用命名与目录服务JNDI功能检索（调用Lookup方法）Home对象引用。

② 命名服务器根据环境变量和命名上下文向客户端返回Home对象引用。

③ 客户端利用Home对象引用创建（调用Create方法）一个新的EJB对象，对于实体Bean还可调用查找方法以获得一个存在的EJB对象。

④ 利用代理将创建EJB对象请求编码并转发给相应的骨架。

⑤ 骨架接受和解码代理请求，并调用EJB Home对象创建EJB对象。

⑥ 骨架将创建的EJB对象引用按照对象请求传递和执行路径逆向返回给客户。

⑦ 客户端利用EJB对象引用调用其相应的各种商务方法。

⑧ EJB远程对象代理将商务方法请求传达给骨架。

⑨ 骨架接受请求，并利用EJB对象将请求指定给Bean实例池合适的Bean实例。

⑩ 骨架按照对象请求传递和执行路径逆向返回响应给客户。由此一次完整的商务请求调用。

2.4 EJB开发流程

EJB的开发可分为服务器端开发与客户端开发，本文主要说明服务器端开发的基本流程。由于容器提供了大部分对分布式通信、持久性管理、负载平衡管理、资源管理与线程管理等底层架构代码编写的支持，因而大大减少了开发人员的编码工作。进行服务器端EJB开发的主要步骤如下：

(1) 编写企业Bean实现，包括容器管理标准接口的实现和特定商务方法的实现。

(2) 为企业Bean中所有商务方法编写客户远程接口（必须扩展EJBOBJECT接口）。

(3) 编写客户本地接口（必须扩展EJBHOME接口），包括用于创建EJB和查找EJB（如果是实体Bean）的应用程序方法。

(4) 编译EJB实现、客户远程接口和客户本地接口。

(5) 编写EJB部署描述文件和清单文件，并将EJB打包成为EJB-JAR模块文件。

(6) 编写应用系统部署描述文件，并将应用系统中所有的EJB-JAR模块打包成为EJB-EAR应用程序。

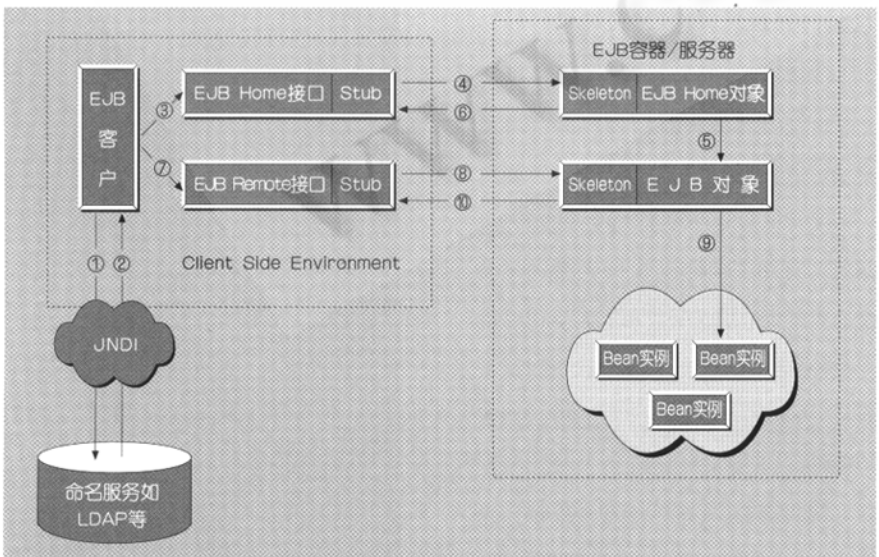


图5 EJB组件与客户交互模型

3 结束语

EJB是服务器端组件开发的基本规范，它采取"分而治之"和"让恰当的专家做恰当的事情"的方式来解决服务器端应用问题。EJB技术不仅使得复杂的分布式多层结构应用系统开发变得容易，而且为进行软件重用和提高软件质量（可移植性、可扩展性等）提供了一条解决途径。EJB作为一种开放的体系结构，仍在不断的发展和完善，如提供特定部署脚本和EJB之间映射的标准化，从而使跨多个供应商工具真正的可移植性和交互性成为可能。可以相信在不久将来EJB必将使我们的开发工作变得更加简单和高效，并且成为分布式多层结构应用系统的服务器端构件模型的首要选择。

参考文献

1 [美] Ed Roman著，《精通EJB》，王进亮等译，电子工业出版社，2002。

2 [美] Paul J.Perrone,et al.著，《J2EE构建企业级系统》，张志伟等译，清华大学出版社，2001。

3 冯博琴等著，《计算机网络》，高等教育出版社，2001。

4 Enterprise JavaBeans Specification, Version2.0.Copyright 1998-2000,Sun Systems,Inc。Aailable at http://java.sun.om/products/ejb.

5 Richard Monson-Haefel.全面研读EJB2.0:Available at http://www.e-keboo.com。

6 李文立等，基于CORBA的分布式计算模型设计，计算机应用，2001，10。

7 曹鸣鹏等，J2EE技术及其实现，计算机应用，2001，10。