文章编号:1005－0523(2006)04－0090－03

Facade 模式在 EJB 中的应用研究

梁冠敏,刘觉夫

(华东交通大学 信息工程学院,江西 南昌 330013)

摘要:探讨了典型的 Facade 设计模式,并介绍了设计和使用 Facade 模式的一些规则.结合 Facade 模式的特点,将其应用到实例中,表明应用该模式能够缩短软件的开发时间,并能提高软件的复用性、可维护性、和可移植性.

关 键 词:设计模式;Session Facade;EJB

中图分类号:311 **文献标识码**:A

1 引言

模式是在工作中为了解决经常出现的问题而用到的可以重复使用的方法.它能使所生成的系统体系结构更加精巧、简洁和易于理解.设计模式提供了设计不同的系统、不同的应用经常发生的问题的解决方案,并以此形成了可服用的面向对象软件的基础.J2EE 是 Sun 公司提出的开发和运行企业级 Web 应用的标准,用于开发大型、多层次、分布式的企业级 Web 应用系统.它为组件开发提供了广泛的支持,便于开发模块化、可重用和平台独立的业务逻辑.EJB 技术是 J2EE 的重要组成部分.EJB 组件是为企业级应用设计的 JAVA 组件模型,是基于标准分布式对象技术、CORBA 和 RMI 的服务器端的组件门面模式是 EJB 中常用的一种设计模式.

2 Facade 模式的原理和结构

EJB 组件一般分为三种核心类型的 BEAN:会话 BEAN、消息驱动 BEAN 和实体 BEAN;会话 BEAN 和消息驱动 BEAN 是业务对象;实体 BEAN 是数据对象;在开发 web 应用程序的时候,很多人采用的方法是客户端直接访问实体 EJB 如图 1 所示,这种结构有着重大的缺陷

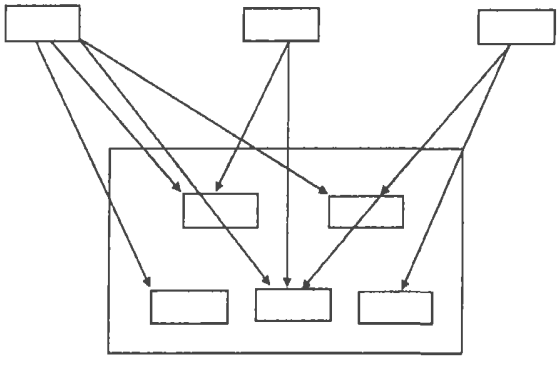

图 1 未采用 Facade 模式结构

(1) 高的网络开销:在没有使用 Facade 模式时将有多个网络调用,这将会降低性能.

(2) 差的并发性:如果客户端离服务器很远(如一个 applet 或 application 和远程的 EJB 系统交互,甚至可能通过 internet 或一个防火墙),这个事务将会持续一个很长的时间.这将导致超额锁定(excess locking),增加冲突和死锁的可能性,并且降低了其他客户端存取同一个 entity bean 实例的并发性.

(3) 高耦合:我们的客户端直接写一个 entity bean 的 API,这将使客户端和 entity bean 紧密的耦合在一起.如果 entity bean 层需要在将来变化,我们也

收稿日期:2006－05－12

作者简介:梁冠敏(1979－),男,河北邯郸人,华东交通大学研究生.

必须改变客户端.

(4) 差的可复用性：执行“传送存款”的用例的商业逻辑直接被嵌入到客户端.那么它将被有效的套在

了那个客户端中.其他类型的客户端(Java application，applet，servlet，等等)不能重用这个商业逻辑.这个把表示逻辑和商业逻辑混在一起是一个差的为任何重要的发布的应用设计.

(5) 差的可维护性：JTA 的使用导致完成事务的中间件逻辑和应用逻辑交织在一起.通过宣称(declarative)事务分离这两个会更干净，所以我们能拧(tweak)和调(tune)我们的中间件，而且不会影响我们的商业规则.

(6) 差的开发角色的分离：一个通用的在大型项目的实践把表示层逻辑程序员的开发任务(如 servlet 和 jsp 开发者)和商业逻辑/中间件程序员(EJB 开发者)分离开来.如果商业逻辑和客户端/表示层一起编码，一个清楚的角色区分是不可能的.

为了解决以上问题，我们在客户端和业务对象之间加上一层(Sesssion Facade 层)，这就引入了设计模式中的 Facade 模式.Facade 的定义是为子系统中的一组接口提供一个一致的界面，很显然我们需要为这些 bean 提供一个统一的对外界面.Facade 定义了一个更高层次的接口，使子系统更容易使用.图 2 给出了一个 EJB－session facade 的例子.

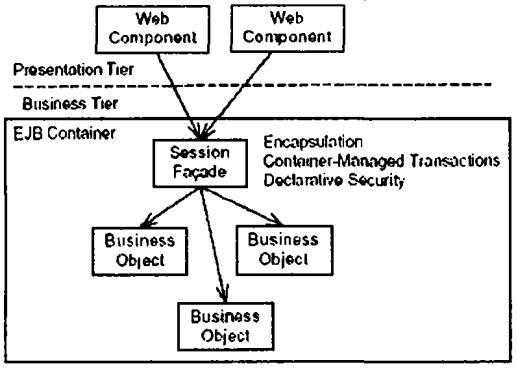

图 2　Facade 模式的结构

在这种设计模式中，表示层可以通过远程来调用 facade(相当于 session 的一个高级接口)，EJB 容器从 facade 中得到这个调用，并验证调用者的权限，然后开始一个业务处理.这个时候 facade 调用底层的业务对象，而这些业务对象负责实现具体的业务逻辑.等 Facade 返回后，EJB 容器提交业务处理或者让该业务处理循环等待.

3　Facade 模式在 EJB 的中的实现

在 EJB 应用中，有两个端点，这一端是用户端，另外一端是 EJB，通常在这两个端点间会增加一层，用来松散两个端点之间的耦合，比如在宠物店例子中，考虑到不同身份的用户有不同的操作流程，比如顾客注册进入后，需要浏览目录，下订单，而商店管理者进入后需要确认或者否定订单，或者检查库存.这些功能需要借助 Session bean 和 Entity bean 完成.

但是如果用户端直接和这些 bean 互动，会有以下问题：

(1) 用户端必须注意和这些 beans 的所有有联系或互动的事情，无法阻止用户端可能不恰当的使用这些 beans.

(2) 如果 EJB 的 API 改动，那么用户端的一些代码也要修改.无疑扩展性很差.

(3) 即使这些 beans 都在同一台服务器上，用户端还是用 remote 方式来调用它们，造成网络无故拥挤.

那么我们使用 Facade 模式来解决这个问题，Facade 的定义是为子系统中的一组接口提供一个一致的界面，很显然我们需要为这些 bean 提供一个统一的对外界面.如图 3 所示

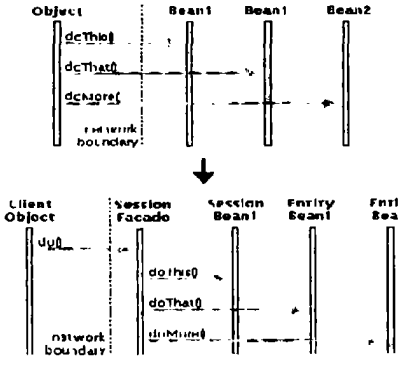

图 3　Facade 模式的实现

4　在 EJB 中使用 Facade 模式的规则

Facade 模式可以使我们开发出高效的，可扩展的 EJB 应用程序，但是很多开发者有滥用、误用 Facade 模式的现象，以下是使用 Facade 模式的一些规则：

(1) 设计合适数量的 Facade，不应设计太多的 Facade.经常能发现 session facade 被误用的项目：创

建一个 session bean 巨类(God－class)·常常开发者把系统中所有的用例都放在一个 session bean 中·这导致臃肿的 session bean 并降低开发生产力,因为所有的开发者都需要用这一个类·session bean 应该被分成相关的用例的组群(house groupings)·

(2) 避免把领域(domain)逻辑放在 session bean 中·一个设计得好的面向对象领域模型应该包括所有的应用中的商业/用例逻辑·大多数 session facade 方法应该只是代理(delegate)合适的 entity bean,除非用例包含需要跨不同 bean 而且不是直接有联系的工作流(workflow)逻辑·

(3) 避免跨 facade 的商业逻辑的重复·随着项目的增长,常常 session bean 方法包含重复代码,这个解决方案是增加一个服务层(作为一个 session bean 或普通 java 类来实现),来封装这个可重用、用例独立(use－case－independent)的商业逻辑·这个服务层是对客户端隐藏的·当项目的体积变大时,用常规重构(refactoring) sessions(重复逻辑被发现和抽出(extracted))是很有用的·

5 结束语

在 EJB 层应用 Facade 模式可以将客户端,商业逻辑层和数据访问层分离,有效地增强了程序的可扩充性和易维护性,从而减少了网络拥挤,提高了速度·另外将 Facade 模式应用在 EJB 中能够有效地提高开发的效率,并且弥补了 EJB 在执行效率等方面的不足,使得 J2EE 体系更加的完整、灵活,提高了可维护性和代码复用率·

参考文献:

[1] 阎宏·Java 与模式[M]·北京:机械工业出版社,2002·

[2] Erich Gamma、Richard Helms、Ralph Johnson & John Vlissides: Design Patterns: Elements of Reusable Object-Oriented Design, Addison-Wesley, 1995, 193－196·

[3] William Brown、Ralph Malveau、Hays McCormick & Thomas Mobray: AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis, John Wiley and Sons, 1998, 72－75·

[4] ErichGamma, RichardHelmRalph1 设计模式:可复用面向对象软件基础(英文版)[M]·北京:机械工业出版社,2003·

Principle and Application of Facade Design Patterns

LIANG Guan-min, LIU Jue-fu

(School of Information Engineering, East China Jiaotong University, Nanchang 330013, China)

Abstract: Facade－the typical software design pattern is discussed, and some rules in designing and using facade pattern are presented· According to the advantages of this design pattern, it can shorten the time of software development and raise the software performance and software maintainability·

Key words: design patterns; session facade; EJB