基于 J2EE 的知识库管理系统的设计与实现

姜 哲, 陈 鹏
(华中科技大学 工程计算与仿真研究所, 湖北 武汉 430074)

摘 要: Java2 平台中最重要的就是 J2EE 平台。基于层次化组件模式的 J2EE 平台把业务逻辑和底层网络技术分离出来, 具有可伸缩性、可扩展性、易开发和易维护性, 已经成为企业级商业分布式网络计算的事实标准。J2EE 融合了各种网络技术和应用服务技术, 它是仍在不断发展完善的平台。文中结合知识库管理系统的开发介绍了 J2EE 的基本特征, 以及在 J2EE 平台基础上实现一个知识库管理系统的基本框架和相关的实现细节。

关键词: J2EE; EJB; MVC 设计模式; 知识库管理系统
中图分类号: TP311.5 文献标识码: A 文章编号: 1005-3751(2005)05-0031-03

Design and Implementation of J2EE-Based Knowledge Base Management System

JIANG Zhe, CHEN Peng
(Engineering Computation Simulation Inst., Huazhong Univ. of Sci. and Techn., Wuhan 430074, China)

Abstract: J2EE platform is the most important part of the Java2 platform. Hierarchical component based J2EE platform divides business logic from the low-level network technology. It is scalable, extensible, easy to develop, easy to maintain and has become the actual standard in the enterprise business distributed computing area. J2EE platform combines many network technologies and application service technologies. It is still a developing platform. This article introduces basic characteristics of J2EE platform by the development of knowledge base management system, presents a basic framework of knowledge base management system and its related implementation details.

Key words: J2EE; EJB; MVC design pattern; knowledge base management system

1 J2EE 介绍

J2EE(Java 2 Platform, Enterprise Edition)是 SUN 公司定义的一个开发分布式企业级应用的规范。它提供了一个多层次的分布式应用模型和一系列开发技术规范。遵从这个规范的开发者将得到行业的广泛支持, 使企业级应用的开发变得简单、快速, 而且所开发出来的应用程序将是平台独立的, 基于组件的 J2EE 解决方案不会被束缚在任何一个厂商的产品及 API 上面。在 J2EE 平台上开发的系统遵循 MVC 设计规范, 即模型(Model)、视图(View)和控制器(Controller)对象。模型提供应用业务逻辑, 对应 J2EE 中的 EJB 类; 视图则是其在屏幕上的显示, 包括 HTML 页面、JSP 页面、SwingGUI; 而控制器则是 Servlet、JavaBean 类, 它用于管理用户与视图发生的交互, 处于视图和数据之间, 对视图与模型交互进行管理。通过使视图完全独立于控制器和模型, 就可以轻松替换前端客户程序。将控制器和模型分开可以在不影响模型的情况下改变控制器, 也可以在不影响控制器的情况下改变模型。根据 MVC 设计规范, J2EE 平台的体系结构见图 1。

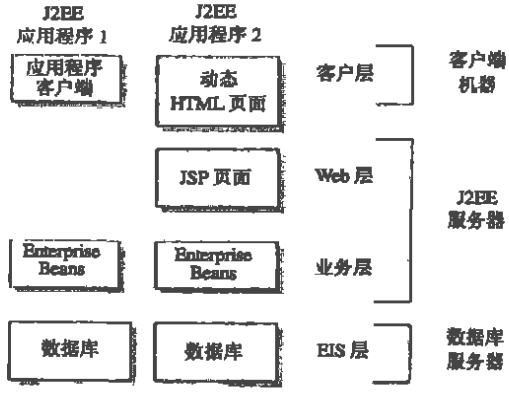

图 1 J2EE 平台结构

根据图 1, J2EE 体系结构大致可以分为 4 层, 即客户层、Web 层、业务层和企业信息系统(EIS)层。基于 J2EE 平台的应用系统可以是基于 Web 的, 也可以是不基于 Web 的一般应用程序。在一个基于 Web 的 J2EE 应用中, 用户的浏览器在客户层运行, 并从一个 Web 服务器上下载 Web 层中的静态 HTML 页面或由 JSP、Servlet 生成的动态 HTML 页面。业务层也被称作 EJB(Enterprise Java Bean)层, 作为解决某个特定业务领域所需要的业务代码一便是

收稿日期: 2004-08-22
作者简介: 姜 哲(1978-), 男, 辽宁抚顺人, 硕士研究生, 主要研究方向为 J2EE 设计与开发、工程仿真计算。

由运行在业务层的 EJB 来执行的。一个 EJB 从客户程序处接受数据,对数据进行处理,再将数据发送到企业信息系统层存储;从企业信息系统层中检索数据,并将数据处理后送回到客户程序。企业信息系统层运行企业信息系统软件,包括企业基础设施系统,例如企业资源计划(ERP)、大型机事务处理、数据统及其它遗留信息系统。

EJB 是 J2EE 平台中最核心的技术,它可以分为实体 Bean(Entity Bean)、会话 Bean(Session Bean),EJB2.0 中还增加了消息驱动 Bean(Message Driven Bean)。一个会话 Bean 代表与客户程序的一个短暂的会话,而且可能执行数据库读写操作。而一个实体 Bean 则代表一个数据库中的数据及作用于该数据的方法,并且实体 Bean 自动支持事务。实体 Bean 与数据库中的数据保持同步有两种方法,即 BMP 和 CMP,BMP 是由 Bean 来管理数据持久性,这时诸如事务、安全、同步等数据库操作要由 Bean 自己来处理;而 CMP 是由 EJB 容器来管理数据持久性的,事务、安全、同步等数据库操作可自动由容器来完成[1,2]。

2 J2EE 技术在知识库管理系统中的应用

知识库管理系统是建立一个物流知识源的系统。其中包括各种文档(字处理系统文件、电子表格文件、演示性文件、PDF 文件、数据库管理系统资料)的上传、下载,其内容按物流中的不同类和主题来划分。两种查询方式有标准查询和详细查询,并且管理员可进行增加、删除、修改。用 J2EE 平台来开发一个知识库管理系统,其优点是:MVC 分层结构清晰,分布式和可移植性更强。客户端可以是基于浏览器的,也可以是基于用各种编程语言开发的应用程序。其体系结构遵循图 1 的结构。

2.1 客户端

应用客户端程序和浏览器是客户端层组件。客户端层组件可以是基于 Web 方式的,即作为 Web 服务器的浏览器,也可以是基于传统方式的(非基于 Web 方式),即独立的应用程序,可以完成瘦客户机无法完成的任务。本系统采用的客户端是基于浏览器的。

2.2 Web 层

基于浏览器的客户端将请求发送到 Web 层,由 JSP、Servlet 处理。JSP 主要用于生成动态的 HTML 页面。Web 层与 Web 浏览器之间一般传递的是 HTML 数据,利用 XML 技术就可以进一步把显示与数据分离,利用 JSP 生成 XML 流发送到客户端,然后客户端用 JavaScript 技术将内容转换为 HTML 显示出来,并在 HTML 中利用 JavaScript 等脚本语言触发事件。其中有个主控制器,它的主要功能是用于将客户端发送来的请求集中分发给不同的 Web 层组件处理,各 Web 组件返回到客户端的数据必须是 XML 流数据。基于 XML 的 J2EE 系统结构可以使得 Web 层不仅仅是可供基于浏览器的客户端访问,还可以供其它应用程序调用,应用程序只需要从 Web 层发送来的 XML 流中分析数据做处理即可,这样就可以扩展 Web 层的功能,并使代码尽可能复用。其结构见图 2[3]。

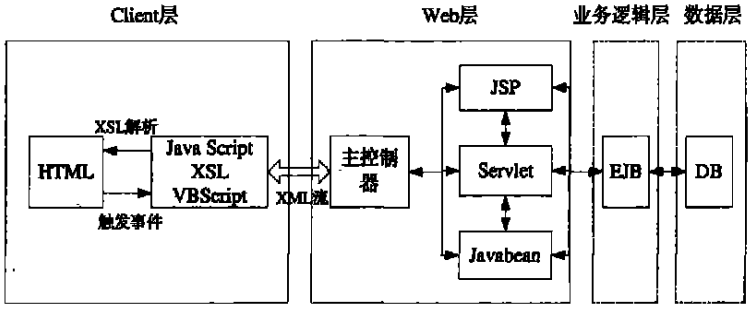

图 2 XML 流结构

其页面流程图见图 3。

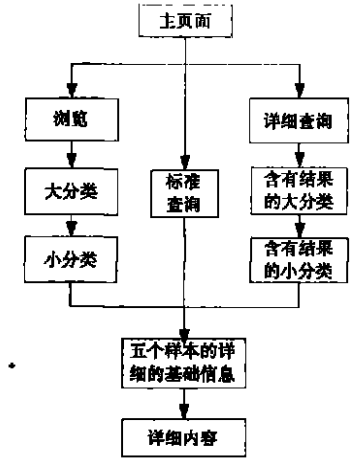

图 3 页面流程

在这个系统中,Web 层并没有直接访问 EJB,而是采用了业务代理模式(Business Delegate),通过访问业务代理类来访问 EJB。在业务代理类中实现了对 EJB 的 JNDI 查找、调用等功能。通过访问业务代理,减少了代码量,优化了代码结构,改善了系统性能。

2.3 业务逻辑层

业务逻辑层是基于 J2EE 平台的应用开发中的关键,业务处理组件的重用可以大大减少编码量,并使以后的维护非常方便。在 J2EE 平台中,只需要开发一个查询处理的会话 Bean 和业务代理类即可。基于 Web 浏览器的应用或其它的应用程序都访问这个会话 Bean 以获得处理结果。在业务逻辑功能模块中将对数据库操作,但是数据库的操作方法不会提供给调用这个 EJB 的客户端,而只向客户端提供业务方法的调用。按 EJB2.0 的规范,CMP(Container Managed Persistence) Entity Bean 是数据层的接口。但是从 EJB2.0 规范发布到现在,业内对 EJB 的争议也是比较多的,最为典型的就是什么时候该使用/不该使用 EJB 的问题上,而争议最大的就是 CMP(Container Managed Persistence)Entity Bean 的使用上。当然不可否

认 EJB 的思想是好的, 但是 CMP 似乎又让人有顾虑, 所以本系统采用了将 CMP 映射到数据库表中。业务数据的增、删、改操作通过 CMP 来实现, 查询是 Session Bean 里封装 JDBC 的方法。这样做业务逻辑既可以被 EJB 容器管理, 又可以避免 CMP 的瓶颈、QL 查询语言的局限性[4, 5]。其部分代码为:

```
import javax.ejb. *;
import com.dlh.info.knowhow.khclassification.ejb. *;
public class KnowHowBean implements javax.ejb.SessionBean{
    private javax.ejb.SessionContext mySessionCtx;
    public void ejbCreate() throws javax.ejb.CreateException {
    }
    public void ejbRemove() {
    }
public Collection getFileExtList() {
      ……
      try{ stat = conn.createStatement();
      String strsql =
      "select topicID, topicName, topicINFO from KHDocinfo";
      rs1 = stat.executeQuery(strsql);
      while(rs1.next()) {
      gInf.setGroupID(rs1.getInt("topicID"));
      gInf.setGroupName(rs1.getString("topicName"));
      gInf.setFimmID(rs1.getString("topicINFO"));
        result.addElement(gInf.clone());
    }
      catch(SQLException e) {
      ……
      } finally {
      ……
      }
```

2.4　数据库层

数据库操作的性能在很大程度上影响整个系统的性能。在基于 J2EE 平台上的知识库管理系统开发中, J2EE 平台已经自动地提供了诸如 JDBC、连接池、JTA 事务支持、JCA 的企业信息系统连接支持、JNDI 数据库的分布式访问支持以及由 CMP 自动维持的数据存取访问。为了提高数据库访问的效率, 系统中就要使用到数据库连接池技术用以共享连接, 减少系统分配连接的时间, 提高存取速度。在基于 J2EE 平台上的系统中, 它本身已经支持连接池, 开发人员只需要输入一些参数来建立这个连接池, 如初始连接数、最大连接数、连接池要访问的数据库、访问数据库的用户名、密码等。利用 J2EE 的 JNDI 服务将连接池加载到 JNDI 上, 可以实现数据库物理位置的无关性。即便是在系统运行过程中改变了数据库的位置, 系统本身也不需要任何改动, 而只要在 J2EE 容器中改变相应的 JNDI 路径即可。分布式事务支持是 J2EE 平台的一大特色, J2EE 中包括 Bean 管理的事务支持和容器管理的事务支持两种形式, 也可以使用数据库系统提供的事务支持。

2.5　下　载

在这个知识库管理系统中下载用的是一个第三方的控件 SmartUpload。SmartUpload 能全程控制下载。利用 SmartUpload 组件提供的对象及其操作方法, 可以获得全部上传文件的信息(包括文件名、大小、类型、扩展名、文件数据等), 方便存取。能对上传时的文件大小、类型等方面做出限制。如此可以滤掉不符合要求的文件。能将文件上传到数据库中, 也能将数据库中的数据下载下来。但是, 因为 SmartUpload 组件是将文件读到服务器的内存中, 所以上传太大的文件(超过 100 兆)可能会出问题。其代码如下:

```
< jsp:useBean id="mySmartUpload" scope="page"
class = "com.jspsmart.upload.SmartUpload" /> <%
mySmartUpload.initialize(pageContext); mySmartUpload.
setContentDisposition(null); mySmartUpload.downloadFile
(((String)session.getAttribute("s_filename"))); %>
```

3　系统性能与伸缩性

由于 J2EE 技术采用基于组件的方法设计、开发、装配和部署企业级应用程序, 因此遵守 J2EE 的规范开发应用系统能有效地保护用户投资, 并使建立可移植应用成为可能。同时这些应用能够完美地与其他应用和系统实现互操作。该系统采用的服务器是 Webshpere, 数据库是 DB2, 开发工具是 WSAD (Websphere Studio Application Developer)。

性能总是应用程序开发的重要课题, 性能表现是多方面的, 例如从客户的角度看到的响应时间, 在企业信息系统中要为许多客户服务, 怎样寻找可接受的响应时间与系统承受的负载即伸缩性之间的平衡是系统设计必须考虑的问题。由于 J2EE 是个分布式体系结构, 调整性能与伸缩性涉及到多个方面(如 Web 服务器、Servlet 容器、EJB 容器及数据库等), 其核心部分是 EJB 性能与伸缩性。实现这些优化, 在开发过程中可遵循如下准则与建议:

1) 将远程方法调用减到最少。分布式系统应用的每个远程调用都有消耗, 使得 EJB 调用造成很大的延迟, 因此为了得到可以接受的性能, 应该将远程方法调用减到最少。需要指出的是, 在 EJB2.0 之前, Enterprise Bean 只有远程接口。EJB2.0 增加了让 Enterprise Bean 定义本地接口的功能, 但本地接口只让在相同 JVM 中的客户通过本地接口访问 EJB。

2) 避免大数据的传输。大量数据的传输肯定会影响系统的性能, 要解决这个问题, 就要解决响应性系统设计, 响应性系统设计的方法将角色、责任及协作行为作为描述对象。

3) 应用服务器的选择。应用服务器及其容器对性能

(下转第 36 页)

和。CORBA 的 IDL 描述如下：

```
module MyCorbaServer
{
  interface IAddTwoNumber;
  interface IAddTwoNumber
  {
  short Add(in short P1, in short P2);
  };
  interface IAddTwoNumberFactory
  {
  IAddTwonumber Create();
  };
};
```

然后可以创建 CORBA 服务程序，同时也创建 CORBA 客户程序，这里创建的 CORBA 客户程序在编写桥接组件时要用到。

再下一步，创建作为桥接组件的一部份 COM 服务程序，可以用 Delphi 提供的 Type Library 来描述 IDL 创建 COM 对象，COM 服务对象建议用进程外组件，这样可以直观地观察到桥接组件的作用。

现在来“改装”这个 COM 服务程序，使它成为桥接组件。注意到 COM 服务程序对象实现部分不是和 FORM 在同一个 Unit，可以在 FORM 的这个 Unit 初始化 CORBA 客户对象，不用手工去编代码，可以直接从先前创建的 CORBA 客户程序中把相应的代码复制过来即可，注意主要是 InitCorba 方法和客户对象的声明，以及 Uses 的内容，还有客户对象的声明也从原来的 protected 成员移到 public 里面，这样以便于在 COM 服务对象中调用[5]。然后在 COM 服务对象的实现的 Unit 里面修改一下即可，其主要意图是 COM 服务对象的实现通过调用 CORBA 的服务对象来实现。

类似的代码如下：

```
function TMyComBridge.Add(Param1, Param2: SYSINT): SYSINT;
begin
    result:= form1.ClientVar.Add(Param1, Param2);//调用 CORBA 服务
end;
```

桥接组件做好了，编译保存，最后在 Windows 里面作为 COM 服务对象注册。

测试运行。ORB 选用 Delphi 自带的 Inprise 公司 VisiBroker，先把其中的 Smart Agent 运行起来，再依次运行 CORBA 服务程序，桥接组件，COM 客户程序，最后测试一下运行结果。

3 结束语

文中给出一个桥接 COM 和 CORBA 的例子。自定义桥接方法最适合那些只需要创建少量的桥接组件的小型系统。在一些大型系统中设计和实现桥接组件要做出极大的努力[1]。大型系统需要大量的 COM/CORBA 的集成，实现大量自定义的桥接组件就不如采用商业化的桥接产品。

参考文献：

[1] Pritchard J. COM 与 CORBA 本质与互用[M]. 徐金梧，张晓彤，屈 蓉，等译. 北京：清华大学出版社，2002.

[2] 钟 灿，钟本善，周熙襄. COM 和 CORBA 的桥接与应用[J]. 电子科技大学学报，2003，32(2)：8－11.

[3] 飞思科技产品研发中心. Delphi7 组件与分布式应用开发[M]. 北京：电子工业出版社，2003.

[4] 万亚平，曾致远. CORBA 设计中的性能分析[J]. 微机发展，2004，14(8)：55－57.

[5] 迟忠先，高永强，张春涛. Delphi6.0 开发务实[M]. 北京：电子工业出版社，2002.

(上接第 33 页)

影响很大，有些容器是针对多客户的伸缩性优化的，有的是对持久性优化的等等。有关比较性能的数据可到相关网站上查询。选择服务器需要综合考虑整体应用和投资效益。有关调整性能与伸缩性还有许多可研究的方法，如数值对象、分布式门面、实例管理算法等等。

4 结束语

J2EE 技术现在正方兴未艾，由于其优越的性能，因而在其它领域，如电信、移动、金融等，越来越多的系统正在或准备转移到 J2EE 平台上来。当然，J2EE 的核心 EJB 还有待改善，就在 2004 年，国内很多 J2EE 开发者也为“轻量级 VS 重量级”的问题争得不可开交。酝酿之中的 EJB 3.0 的规范，完全基于 POJO 的组建模型，简化 EJB 设计，避免对组件的侵入，使用依赖注入、Entity Bean 成为一个真正的 O/R Mapping 工具。这些改变都会使 J2EE 的开发更加简便，更加高效。相信 J2EE 的明天会更好。

参考文献：

[1] 伊小强. J2EE 全实例教程[M]. 北京：北京希望电子出版社，2002.

[2] Johnson R J2EE 设计开发编程指南[M]. 魏海萍，于晓菲，毛 选，等译. 北京：电子工业出版社，2003.

[3] 马仁配. J2EE 平台在图书馆系统中的应用[J]. 现代图书情报技术，2003(4)：29－32.

[4] Roman E. 精通 EJB[M]. 刘晓华，等译. 北京：电子工业出版社，2002.

[5] 李树仁，王锋锐，马 凡，等. EJB 最新技术开发指南[M]. 北京：北京希望电子出版社，2003.