摘 要

由于软件的多元化发展，企业内部中充斥着各种异构的应用服务，并且每个系统有着独立的用户登录方式、使用界面和存储系统，使得用户操作和系统维护的成本不断升高，而各个应用系统之间的孤立使得企业资源的共享和复用受到很大的制约。因此，迫切需要一种统一的系统来解决存在于企业应用中的信息孤岛问题，使得各系统框架能够协调工作，从而达到提高企业竞争力的目的。

针对以上问题，本文提出了一个基于 Portlet 的企业信息门户（EIP）框架，并实践构建了武钢设备中心的 EIP 系统。

本文的研究是基于武汉钢铁集团设备中心的综合办公信息系统的基础之上。论文的主要工作包括：首先分析了 Portal 技术的发展现状和 Portlet 的构成，并阐述了企业信息门户的概念；然后基于 JSP 设计模式和 MVC 设计模式，通过 Struts 与 Portlet 技术集成的研究，提出了企业信息门户的 Portlet 应用模型；接着采用 Portlet 应用模型对武钢设备中心 EIP 系统进行了分析，运用面向对象设计思想和 UML 方法，按照数据层、业务逻辑层以及表示层三个层次分别详细设计了其中用户权限管理子系统和设备管理子系统等模块的业务流程；最后，详细阐述了基于该 Portlet 应用模型的 Web 应用整合的构建过程，从而实现了武钢设备中心的 EIP 系统。

一方面，本文通过企业应用与门户的集成对企业资源进行了整合，实现了信息共享。另一方面，采用 Struts Portlet 模型结合 iBATIS 框架为企业信息集成提供了一种更为灵活的解决方案，对于 Portlet 技术在企业信息集成中的应用做了有益的探索，具有一定的参考价值。

关键词：企业信息门户，Portlet，Struts，iBATIS，门户集成

Abstract

As all kinds of heterogeneous applications in enterprise systems come with the multi-development of software, each system has its independent way of Login, user interfaces and storage system, which makes the cost of user's operation and maintenance increase continually. The isolation between the various application systems makes major constraint for enterprise resource sharing and reusing. Therefore, there is an urgent requirements for a unified system to resolve the problem of information isolated island, and it makes system frameworks to coordinate work and improve the competitiveness of enterprises.

In view of the above problems, this thesis introduced an enterprise information portal (EIP) framework based on Portlet, and adopted Portlet technology to achieve the EIP system of WISCO Equipment Centre.

The research of this thesis rooted in the enterprise portal projects of WISCO Equipment Centre. The main tasks of this thesis include: Analyzing the development of the Portal and the composition of Portlet, and expounding the concept of enterprise information portal. Then the thesis presented a Portlet application model of enterprise information portal based on JSP and MVC design pattern through integrated research on Struts Portlet technology. The thesis used the model to analysis the EIP system of WISCO Equipment Centre and used the object-oriented and UML technology to design the business process of user authority management subsystem and the business processes of device management subsystem in accordance with divisions of the data layer, the business logic layer and the presentation layer. Finally, the Portlet application model is used to integrate Web Application, and the centre WISCO equipment EIP system was realized.

In addition, this thesis resolved the issue of information sharing of enterprise resource through integration of enterprise applications and portal.

Meantime, the thesis provided a more flexible solution for enterprise information integration through the combination of Struts Portlet model and the iBATIS framework. It was a worth reference for various application of technical framework in the Portal and was a useful exploration.

Keywords: Enterprise Information Portal, Portlet, Struts, iBATIS, Portal Integration

插图索引

此页若属实，请研究生及导师签名，并装订在学位论文的摘要前。

独 创 性 声 明

本人声明，所呈交的论文是我个人在导师指导下进行的研究工作及取得的研究成果。据我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得武汉理工大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

研究生（签名）：钰鸣 日期 2008.4.5

关于论文使用授权的说明

本人完全了解武汉理工大学有关保留、使用学位论文的规定，即：学校有权保留送交论文的复印件，允许论文被查阅和借阅；学校可以公布论文的全部内容，可以采用影印、缩印或其他复制手段保存论文。

（保密的论文在解密后应遵守此规定）

研究生（签名）：钰鸣 导师（签名）：张丽平 日期 2008.4.5

第1章 绪论

1.1 课题背景

随着企业信息化的应用和普及，数字资源和应用系统迅速增加，这些资源和应用往往来自不同的组织或个人，每个资源或应用都有其专门的访问接口和服务方式，并且可能存放于不同场所，这种分散、异构的信息环境使得普通用户、系统管理员以及系统开发者面临越来越多的障碍和挑战，主要体现在以下方面:

(1) 企业资源分散

随着企业中信息资源的快速增长，异构和分散的资源很容易成为一个个“信息孤岛”[1] 。因此，有效的组织和关联这些资源，将其互联、重组成高效的服务，成为目前企业开发面临的首要问题。

(2) 用户负担繁重

不同应用的权限认证和使用方法各不相同，对于多系统操作用户不得不记忆不同资源的访问口令和使用方法，增加了用户负担，造成用户使用过程中的困扰。

(3) 服务模式单一

传统的无差别的服务模式往往不提供用户选择服务方式的权利，无法满足每个用户对个性化的信息定制和使用偏好的需求。

因此，对于企业信息集成，需要构建一种应用，它能够针对特定需求，将与之相关的各种信息资源、应用和服务组织到一起，消除其地理位置的不同、结构上的差异，并以友好的、易用的、可定制的方式集中呈现给用户，用户可以通过一个中心节点同时访问多个资源，且不需要了解每个资源的位置以及具体访问方法。

由于门户 (Portal) 技术支持单点登录和个性化定制等功能，同时可以实现多个应用系统和企业信息门户的集成。因此采用 Portal 框架便成为目前企业门户集成的首选方案，即企业信息门户(Enterprise Information Portal，简写 EIP)[2] 。而 Portlet 是 Portal 的重要组件，是 Web 数字资源和应用在 Portal 中的封装体。企业信息门户开发的主要工作就是将各个系统的访问接口进行 Portlet 化封装，加入门户中，同时通过统一的用户管理系统结合单

点登录技术（SSO）把分散而独立的用户管理集中起来，从而实现一次登陆，所有系统通用。所以，如何利用开源的门户技术将异构和分散的企业资源封装和转换为统一管理、标准化的门户系统，从而实现资源和服务的重组，成为目前门户技术研究和应用中的主要问题，也是本文研究的主要问题。

1.2 研究现状

门户概念出现于20世纪90年代，从技术发展的角度看，门户的发展目前已经经历了两代的发展[3]。

第一代门户技术将多种资源在界面层聚合到一个页面中，具体实现技术包括嵌入式框架、RSS、内容目录等。第一代门户不仅功能繁多而且结构复杂，其开发重点通常是各种功能模块，如用户管理、工作流、内容管理模块等。最初的门户开发商以大型应用服务器提供商为主，例如 IBM、SUN 等，随着门户市场需求的迅速增长，更多大型软件提供商也加入了门户开发的行列，例如 Oracle、Microsoft、BEA、SAP 等。与此同时，一些研究机构如 Jakarta、JA-SIG 也开始参与到门户技术的研究和应用中，并成功推出 JetSpeed, uPortal, eXo, Liferay 等开源门户软件，这些开源项目大大促进了门户的普及。随着门户标准的相继出台（JSR168 规范和 WSRP 规范）、XML 和 Web Service 等技术的发展，促使门户系统的功能和结构发生了较大变化而进入第二代门户阶段[4]。

第二代门户在第一代门户发展的基础上确定了门户框架必需的三大基础服务，即个性化定制服务、单点登录服务和内容管理服务[5]。同时门户系统作为一个基础框架可以接入其他功能模块进行功能扩展，比如工作流引擎、搜索功能等等。同时，随着 JSR168 规范和 WSRP（即远程 Portlet Web 服务）服务标准的出台，这些规范定义了标准的门户组件交互接口，使符合接口标准的门户组件得以在不同开发商的门户产品中重用，实现了门户组件资源的共享，节省了开发投入，加速了门户技术的发展[6]。新的整合需求的出现促使门户的系统架构也发生了相应变化，从而促使面向服务(Service Oriented)、面向过程(Process Oriented)的软件架构设计方法被越来越多地应用于门户架构的构建[7]，例如 IBM 公司的 WebSphere 门户、BEA 公司的 WebLogic 门户等。这些架构设计模式的关键在于将门户功能抽象成功能独

立的服务或过程，支持即插即用的扩展方式，从而满足整合动态增长的信息服务、实现按需服务的需求[8]。

第二代门户应用侧重于将门户作为一种整合框架来研究和应用，强调门户软件架构的开放性、扩展性和可定制性。同时，随着 Portlet 概念的引入和 Portlet 规范的出台，Portlet 开发方法以及将传统 Web 资源、应用封装成 Portlet 的方法也成为目前门户技术研究和应用的重要问题。

从应用的角度上看，企业信息门户发展呈现了阶梯式的发展，状态大致分为四个阶段。第一个阶段是 Internet Web 阶段，即企业通过建立 Web 网站来发布企业信息，供网络用户或客户浏览，功能比较简单，主要侧重于企业信息发布。第二个阶段是 Intranet Web 门户阶段，这个阶段借助于管理信息系统的自动化普及，使得企业从单一部门的单机系统到多个部门的网络阶段再到整个企业的集成，并进而发展到 Intranet 系统（企业内部互联网系统）。第三个阶段是企业信息门户（EIP）阶段，EIP 的基本作用是提供全方位的企业信息，它强调对结构化和非结构化数据的收集、加工和无缝集成。第四个阶段是企业应用门户（EAP）阶段，它是完全面向应用的企业门户，及对企业业务流程的集成，它以商业流程和企业应用为核心，把商业流程中功能不同的应用模块通过门户技术集成在一起。从总体上看，门户的应用发展呈现由简单到复杂，由表及里的趋势。

从市场的角度上看，2002 年 6 月，Gartner Group 估计门户市场将从 2001 年的 709 万美元上升到 2006 年的 2 亿美元。就在当月 IDC 研究也表明门户市场将从 2001 年的 550 万美元上升到 2006 年的 3.1 亿美元。根据 Delphi Group 的保守估计，门户市场拥有 20%的增长率。在这种大环境下，国内企业信息化完善的公司或那些敢于吃“螃蟹”的 CIO 们已经在摸索中开始了门户建设，通过调查企业门户建设的状态和水平，发现有 35.8%企业门户建设处于萌芽和启动阶段，24.1% 的企业正在进行门户建设的规划制定工作，更有 7.3 %的企业部署了企业门户软件。所以有接近 6 成的企业对企业门户的关注和投入在增加。2007 年度中国互联网调查统计数据显示：2007 年中国互联网综合门户市场保持健康增长，市场规模达到 123.5 亿元人民币，较 2006 年增长 22.3%。这一组调查数据表明企业门户将会得到广泛的应用。虽然现在还无法估计国内的门户市场究竟有多大，但是根据调查结果，我们对于门户应用的趋势不可阻挡性还是十分乐观的。

1.2.1 Portal 研究综述

门户 (Portal) 是把各种应用系统、信息资源和互联网资源集成起来，并可以根据用户的使用特点和角色定制个性化的应用界面[9] 。Portal 通常作为资源和服务的单一入口，支持分布、异构资源的“一站式”操作，提供个性化和定制服务。具体表现在：将多种资源、服务聚集在同一门户页面中，实现资源的有效组织，使用户能迅速发现并使用与其信息需求相关联的资源；每个用户都能根据自身偏好设置个性化的服务模式、定制个性化的资源集合、更改界面布局和显示风格；用户可在门户中访问各种资源、调用各种服务，而无需分别登录到各个系统，以减轻用户负担。

目前门户的开发分为两大技术流派，一种是采用微软的.NET 技术流派，其代表产品是 Share Point Portal Server，它依靠微软强大的操作系统平台和办公软件的支持，内置了许多微软体系的组件，如 Office 组件、Exchange 组件、MSN 组件等，其优点是能够快速构建统一的门户系统[10] ，但是缺点也很明显，那就是与主流企业软件不易集成、数据库单一、技术支持和更新缓慢。另外一种是基于 J2EE 的技术流派，比如 IBM、SUN、BEA 公司的门户软件产品，他们在功能上都很相似，支持门户个性化定制、活动目录、单点登录、权限管理和 Portlet 间的通信，并且还提供了各种数据库的灵活选择。

由于 Share Point Portal Server 只能运行在 Windows 平台，而且缺乏对第三方组件的广泛支持，限制了它在企业级应用中的推广和普及；而基于 J2EE 技术的门户产品，由于它的平台无关性，使其不管是在 Windows 平台还是 Unix 平台均可以顺利部署，并且能支持众多的第三方的组件，因此就成了目前门户产品的主流。

完整的 Portal 通常由 Portal 服务器、Portlet 容器、Portlet 构成[11] 。下面本文就分别对这三部分做详细阐述。

1.2.2 Portal 服务器

Portal 服务器是容纳 Portlet 容器并支持 Portlet 呈现的普通或特殊 Web 服务器。Portal 服务器通常会提供个性化设置、单点登录、内容聚合、信息发布、权限管理等功能，支持各种信息数据来源，并将这些数据信息放在网页中组合而成，提供个性化的内容定制，不同权限的浏览者能够浏览不同的

信息内容。通常，Portal 提供单点登录 (SSO) 、权限控制、内容管理、信息发布、文件管理四个功能[12] 。

1.2.3 Portlet 容器

Portlet 容器提供 Portlet 执行的环境，包含很多 Portlet 并管理它们的生命周期，保持 Portlet 的定制信息。

一个 Portlet 容器接收到来自 Portal 的请求后，接着将这个请求传递给存在的 Portlet 容器的 Portlet 执行，Portal 负责组合 Portlet 产生的信息内容并呈现给用户。Portlet 和 Portlet 容器可以放在一起视为同一个系统的组件，或者分开成为两个独立的组件[13] 。

Portlet 容器是普通 Web Service 容器的扩展，所以一个 Portlet 容器可以构建于一个已经存在的 Servlet 容器或者可能实现全部 Web Servlet 容器的全部功能。通常情况下，Portlet 容器扩展自普通的 Servlet 容器，本文的运行环境采用了 Servlet2.3 规范。

1.2.4 Portlet 构成

Portlet 作为 Portal 最重要的组件之一也被称为门户组件或门户元件，是 Web 数字资源和应用在 Portal 中的封装体[14] 。我们通过开发和部署一个个 Portlet 而将资源和应用整合到门户系统同样，门户管理上也是以 Portlet 为单位进行管理，Portal 用户通过 Portlet 访问后端资源和应用，并进行个性化定制操作。门户部署过程中的很大一部分工作是在现存资源的基础上开发 Portlet，将其改造成符合门户标准、可集成到门户平台的门户组件。所以，如何灵活利用各种 Web 技术，将各种异构或者分散的信息资源和应用快速地封装成 Portlet，成为本文 Portlet 技术研究、应用中的重要问题。

和 PC 桌面一样，每个 Portlet 都占用浏览器的一块区域来显示信息，这些由 Portlet 产生的内容也被称为片段 (Fragment)，而片段是具有一些规则的标签语言 (比如：HTML、XMTML、WML)[15] 。Portlet 的呈现内容可以是一个电子邮件栏，也可以是一个指定的股票走势曲线图。这些一个个 Portlet 内容片段聚合在一起，就形成了 Portal 网页。

从技术角度上看，一个 Portlet 就是一段可以运行在门户服务器上的代码，它通常实现为平台独立的 Java 类，可以被 Web 服务器动态的装入和运行。

从用户角度来看，一个 Portlet 就是门户页面中用户可以个性化定制的一个内容频道或者一个应用，是一个嵌入页面的应用组件，用户通过“遥控器”定制或切换频道。

1.2.5 Portlet 规范

与 Portlet 相关的最重要的两个标准是 JSR168 规范和 WSRP 规范。Portlet 封装和重用的关键在于遵循 Portlet 接口标准（JSR168 规范和 WSRP 规范）来开发具有可重用性的标准化 Portlet，同时 Portal 作为 Portlet 的运行环境，也应能够以标准化的方式接入 Portlet 。

JSR-168 (Java Specification Request)是由 JCP 组织（Java Community Process）制定的适合于 Portlet 开发人员的 Java API 集合，用来提供不同的 Portlet 之间的互通[16] 。JSR-168 规范的优点之一是可以移植，只要开发的 Portlet 遵循 JSR-168，则就可以在所有遵循 JSR-168 的 Portal 上部署执行。JSR-168 规范的出台使得企业门户中可以使用不同厂商的资源服务器，部署即插即用的可视化的门户组件[17] 。

WSRP（OASIS Web Service for Remote Portlet）是由 OASIS 组织（Organization for the Advancement of Structured Information Standards）制定的基于 Web Service 的一种新的商业应用，一种新的标准，WSRP 提供了一个通过 Web Service 联合 Portlet 内容的标准[6] 。WSRP 规范规定了重用远程 Portlet 的机制，允许门户之间、门户与其他集成应用的 Web 应用之间也能以即插即用的方式进行交互的 Web Service 服务。通过 WSRP 返回的信息是相对标记片段，例如 HTML、XHTML 等，可以直接嵌入用户的页面中，而不用像 Web Service 一样开发用户端接口。实现这个规范，Portal 可以跟各式各样的数据源打交道，彻底终结信息孤岛的窘境。

1.2.6 LifeRay 技术综述

Liferay 是 Liferay 公司开发的开源的门户系统，它代表了完整的 J2EE 应用，特别是其前台界面部分使用 Struts 框架技术，基于 XML 的 Portlet 配置文件可以自由地动态扩展，使用了 Web Services 来支持一些远程信息的获取，也使用了 Apache Lucene 实现全文检索功能[18] 。

其主要特点如下：

- 提供单一登陆接口，多认证模式（LDAP 或 SQL）

- 管理员能通过用户界面轻松管理用户、组、角色
- 用户可以根据需要定制个性化的 Portal Layout
- 能够在主流的 J2EE 应用服务器上运行，如 JBoss+Jetty/Tomcat
- 支持主流的数据库，如 Oracle、Sybase ASE、MySQL
- 使用了第三方的开源项目，如 Hibernate、iBATIS、Struts
- 支持包括中文在内的多种语言
- 采用最先进的技术 Java、EJB、JMS、SOAP、XML

1.2.7 企业信息门户综述

企业信息门户（Enterprise Information Portal,简称 EIP），它是用户访问企业信息资源和服务的统一界面。好比企业的大门，透过这扇门，用户可以访问到企业内部的各种信息资源和服务。EIP 本身的价值是透过其所提供的信息资源和服务的价值来体现的。这些信息资源和服务由企业 IT 环境中的各个子系统来提供，比如人力资源系统、文件服务系统、办公自动化系统、生产管理系统等。当这些系统经由门户集成化的展示在用户面前的时候，用户看到的是一个经由企业信息门户包装后的企业信息资源和服务的全貌，而不再是一个个信息孤岛。

(1) 作为企业信息门户，它包括以下的基础服务功能：

- 提供统一的访问控制和身份验证能力
- 能够在不同层面上（表现层、应用层、数据层）整合其他子系统
- 提供企业全网范围内的信息搜索
- 拥有良好的信息资源和服务的分类，使得用户可以按照分类迅速的导航到相应信息资源和服务上去

(2) 此外，企业信息门户还应包含以下的高级服务功能

- 能够根据用户的自定义配置信息（Profile）展现个性化的信息资源和服务，即每个用户能够拥有自己按需定制的“我的门户”
- 用户能够透过门户获得高效的协作功能，例如专家定位、同步/异步沟通、讨论等[19]

1.3 课题研究内容

本文根据作者根据近年来从事应用系统开发的技术经验出发，结合武钢现有应用系统的实际情况，围绕武钢设备中心 EIP 系统的开发进行研究。首

先，全面了解了企业信息门户在国内外的发展现状、技术前沿和研究热点，进而提出了一种基于 Portlet 的企业信息门户的解决方案，并系统地阐述了方案的分析、设计和实现方法。

具体来说，本文主要做了以下方面的工作：

(1) 研究了 Portlet、Struts 和 iBATIS 的相关技术；

(2) 研究了基于 Struts 框架的 Portlet 技术，并研究了基于 LifeRay 的 Struts Portlet 框架模型；

(3) 研究了基于 iBATIS 框架的系统数据层与事物逻辑层的接口技术，并提出了 iBATIS 在 Portal 上的应用模型；

(4) 研究了将 Struts Portlet 模型和 iBATIS 框架整合并在 EIP 上的 Java 技术实现。

1.4 论文结构

本文按照系统分析、系统设计和系统实现的研究思路将论文内容的组织结构如下：

第一章：阐述本文研究的背景和意义，对现有的 Portal 技术做了阐述，并且提出了本文的研究内容。

第二章：从 Portlet 与现有的 Web 应用的区别与联系入手，研究了 Portlet 构成和工作原理，并对 Web 应用 Portlet 化封装的问题从 JSP 到 MVC 的两个层次进行了研究，并研究了 Struts Portlet 的实现模型。

第三章：首先，从方法论角度出发分析了系统开发方法和建模方法，确定了本文采用面向对象的开发方法并用 UML 作为建模语言。其次，对系统的整体功能和结构进行了分析，确立了两个重要子系统并对其结构和业务流程进行了详细的分析。最后，从系统的层次结构的角度分析，确定了系统的三层体系。

第四章：在第三章系统分析的基础上，结合第二章的思想从数据层、业务逻辑层和表示层的角度对武钢设备中心 EIP 系统进行了详细的设计，明确了系统用户权限管理子系统和设备管理子系统的具体业务流程。

第五章：在系统设计的基础上，以 MVC 设计模式作为指导思想，采用 Struts Portlet 和 iBATIS 的框架实现了武钢设备中心 EIP 系统的整合应用。

结论部分对本文进行了总结和未来的研究工作进行了展望。

第2章 Web 应用的 Portlet 化封装

Portlet 是基于 Java 技术的 Web 组件，所以是基于 Web 应用的。到目前为止，业界已经公布了关于 Portlet 的两个规范，即 JSR168 标准和 WSRP 标准，它们为 Portlet 提供了统一的开发标准，并允许将 Web 应用系统引入到门户系统中来。但是，这两个规范并没有明确指出一个 Web 应用应该怎样被部署成为一个 Portlet。因此，区别 Web 应用与 Portlet，探讨 Web 应用如何封装成为标准化的 Portlet 就成为本文研究的重点。

2.1 Portlet与Servlet的关系

2.1.1 Portlet 与 Servlet 的区别

Portlet 被定义成为一个新的组件，具有新的明确的界面与行为。为了尽可能与现有的 Servlet 结合达到重复使用的目的，Portlet 的规范利用了 Servlet 的规范，在同一个 Portlet 应用中，他们将分享同一个类加载器(Class Loader)、上下文(Context)及 Session[20] 。

(1) Portlet 和 Servlet 的相似之处如下：

- Portlet 也是 Java 技术的 Web 组件
- Portlet 也是有特定的 Container 在管理
- Portlet 可以动态产生各种内容
- Portlet 的生命周期由 Container 所管理
- Portlet 和客户端的互动是通过 request/response 的机制来实现

(2) Portlet 和 Servlet 的区别如下[21] ：

- Portlet 只产生信息片段，不是完整的网页文件，只能通过 Portal 将所有的信息片段放到一个完整的 Portal 网页而呈现给用户
- Portlet 不会和 URL 有直接的关系，客户端必须通过 Portal 系统的 URL 重写才能和 Portlet 互动
- Portlet 有一些定义好的 request 处理，Action request 以及 Render request
- Portlet 默认定义 Portlet 模式及窗口状态，可以指出在网页中该 Portlet 的哪个功能正在执行及现在的状态
- Portlet 可以在同一个 Portal 网页中存在多个

- Portlet 接受请求的类必须继承自 javax.portlet.Portlet，而不是 Web 应用中 Servlet 的 javax.servlet.http.HttpServlet。

2.1.2 Portlet 的生命周期

一个 Portlet 有着良好的生命周期管理，定义了怎样装载，实例化和初始化，怎样响应来自客户端的请求及怎样送出服务。这个 Portlet 生命周期由 Portlet 接口的 init、processAction、render 和 destroy 方法来表达[22] 。

具体过程如下：

(1) 载入和实例化：Portlet 容器负责载入和实例化 Portlet。当 Portlet 容器运行 Portlet 应用或者接收到使用者的请求时，Portlet 就会被载入并实例化。载入 Portlet 类后，Portlet 类随即被实例化。

(2) 初始化：Portlet 类实例化后，Portlet 容器还需要初始化 Portlet，以调用 Portlet 去响应客户端的请求。Portlet 容器调用 Portlet 接口中的 init 方法初始化 Portlet。通过 PortletConfig 的扩展类可以读取定义在部署描述文件中的初始化参数，以及 Resource Bundle。

(3) 响应用户请求：一个 Portlet 对用户有两个请求方式，processAction 用来处理类似提交 form 之类的用户事件，而 render 则用来处理有关显示的方法。当 processAction 处理完之后，Portlet 容器仍会调用 render 方法。

(4) Portlet 的关闭：当 Portal 服务器终止运行后，调用此方法表示 Portlet 的生命周期结束[23] 。

2.2 Portal的工作原理

一个 Portal 系统根据需要由一个或者多个 Portal 页面组成，每个 Portal 页面包含零个或者多个的 Portlet。每个 Portlet 呈现自己的信息内容，以此实现内容聚合。通过定义每个 Portlet 的可用权限，实现个性化的桌面信息定制。

2.2.1 Portlet 样式以及窗口状态

JCP 组织提出的 JSR-168 规范定义了 Portlet 的实现标准。每个 Portlet 对外表现为一个小窗口，有自己的默认样式和窗口状态。从各种数据来源提取的信息以 Portlet 内容的形式呈现在 Portlet 中。

Portlet 样式指出 Portlet 正处于什么模式，Portlet 通常会根据所处的模式而执行不同的工作并产生不同的内容。

Portlet 模式让 Portlet 决定它该显示什么内容和执行什么动作。调用一个 Portlet 的时候，Portlet 容器会提供一个 Portlet 模式给那个 Portlet。当在处理一个请求动作时，Portlet 的模式是可以用程序来改变的。

2.2.2 Portal 页面

每个 Portal 页面包含零个或者多个 Portlet 小窗口，构成一个完整的信息呈现页面。Portal 在启动之后根据 Portlet 配置文件等信息，给 Portlet 的标题等属性赋值，赋予 Portlet 编辑、关闭等各种控制按钮，使 Portlet 成为一个标准的 Portlet 窗口。Portal 合并这些 Portlet 窗口，组成一个完整的文档，即 Portal 页面。每个 Portlet 都处于相应的布局当中，呈现事先定义的内容。而且 Portlet 可以在不同的布局之间切换。Portlet 响应客户端的请求，并将请求提交到相应的 URL 进行逻辑处理。

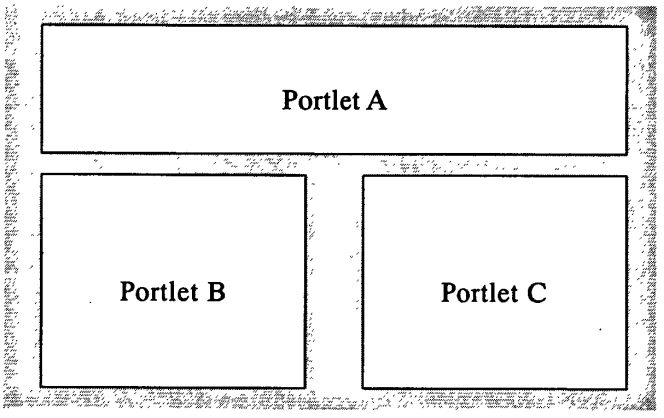

图2-1 Portlet 窗口样式

Portlet 开发完毕之后，部署到 Portal 服务器，由 Portal 服务器负责组织、权限控制和呈现。Portal 页面创建过程如下：

Portlet 在 Portlet 容器内执行，Portlet 容器接收 Portlet 产生的内容。通常 Portlet 容器将这些内容提交给 Portlet 服务器，Portlet 服务器依照这些内容建立 Portal 页面，然后将它传给客户端呈现[24] 。

对于 Portal 页面的请求，使用者经由客户端设备（例如浏览器）存取

Portal，Portal 根据接收到的请求决定哪些 Portlet 需要被执行以满足需求。Portal 通过 Portlet 容器呼叫 Portlet，然后由 Portlet 产生的片段建立 Portal 页面，再传回客户端呈现给使用者，数据流动方向和页面产生过程相反。

2.3 JSP Portlet 的研究

2.3.1 JSP Portlet 的主要接口和类

JSP Portlet 即是将基于 JSP Model 1 的 Web 应用程序实现为 Portlet 的过程，实际上是直接采用 Portlet API 进行 Portlet 开发[25] 。这个过程中所用到的主要接口和类分别是：

javax.portlet.Portlet 是所有 Portlet 都要实现的接口；

javax.portlet.PortletConfig 接口提供对 Portlet 相关信息的只读方式的访问，这些信息来源于 Portlet 的部署描述文件；

javax.portlet.PortletContext 接口定义了 Portlet 容器的 Portlet 视图，同时也使得 Portlet 能获得资源，Portlet 使用 PortletContext 可以访问 Portlet 日志和获得对资源的 URL 引用；

javax.portlet.GenericPortlet 提供了对 javax.portlet.Portlet 接口的默认实现，提供了 Portlet 实例类应该具有的方法。如图 2-2 Portlet 的类图所示。

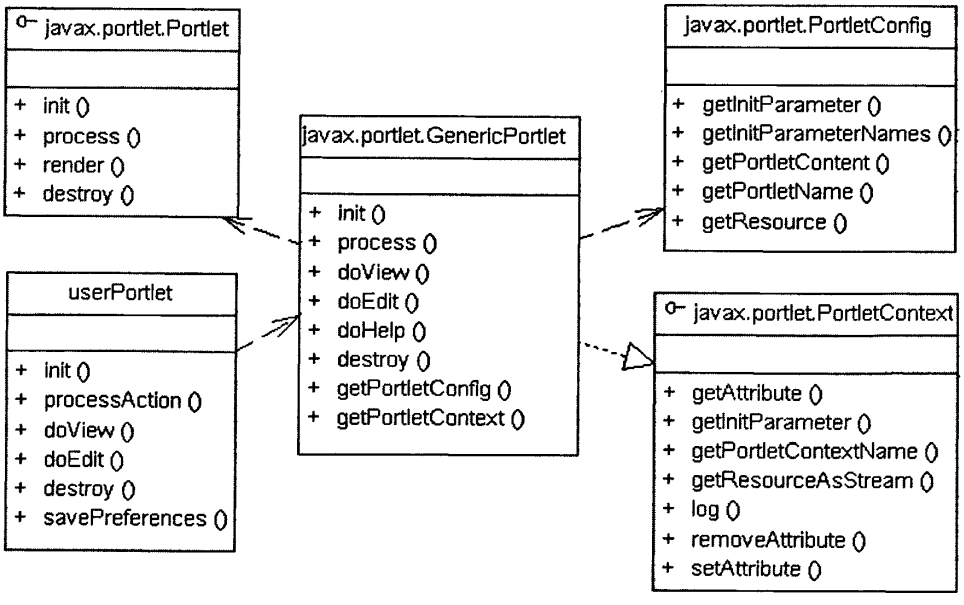

图2-2 Portlet 的类图

从 Portlet 类图上我们可以看到，像 Servlet 一样，Portlet 的开发也必须直接或者间接的扩展基类 GenericPortlet。这是由 JCP 针对 Portal 提出的 JSR-168 规范定义的。只要扩展自规范的 GenericPortlet，所有的 Portlet 都可以在支持 JSR-168 规范的 Portal 服务器上运行。

在 Portlet Portal 运行的时候，doView、doEdit、doHelp 三个方法分别被调用，用以生成 Portlet 标记。同样也可以调用 Servlet 生成 Portlet 标记，或者不调用 JSP 或者 Servlet，直接在方法中得到 PrintWriter 然后用最简单的 pw.println()打印出内容[26] 。

与 Servlet 类似，可以使用 getInitParameter(String s)得到配置文件中 Portlet 的初始值，只不过 Servlet 在 web.xml 中，而 Portlet 在 portlet.xml 中。

2.3.2 JSP Portlet 的对象

JSR-168 给 Portal 定义了几个特别的对象，用来操作 Portal 特有的信息。这些对象跟 Servlet 的对象有点类似，又有点区别。这些对象都封装在{PORTAL_HOME}/common/lib/ext/portlet.jar 包中，具体支持实现要视 Portal 服务器而定。

(1) Request 对象

Portlet 中的 Request 与 Servlet 的 Request 一样接受客户端发送的请求，但是与 Servlet 不同，Portlet 的 Request 分为 RenderRequest 和 ActionRequest 两种类型，因此 Portlet 接口中定义了两种方法用来处理不同的 Request。分别是 processAction(ActionRequest request, ActionResponse response) 和 render(RenderRequest request , RenderResponse response)，分别用以处理 Action Request 和 Render Request。某种意义上来讲，render 方法类似 Servlet 中的 service 方法，doView，doEdit，doHelp 方法又类似 doGet，doPost 方法。RenderRequest 和 ActionRequest 分别由 renderURL 和 actionURL 来触发。actionURL 适用于有确实的 Action（行为）的情况下。renderURL 通常用来处理 Portlet 的导航。

(2) Response 对象

与 Request 对象一样，Response 对象也有两种：RenderResponse 和

ActionResponse，分别用来封装对应的 RenderRequest 和 ActionRequest 的返回信息，比如重定向、窗口状态、Portlet 模式等。他们两者的父类 PortletResponse 拥有 setProperty 和 getProperty 两个方法，用来传递信息给 Portal 容器。ActionResponse 主要用来处理重定向、改变窗口状态、Portlet 模式和传递 parameter 参数到 RenderRequest 中去。RenderResponse 主要用来设置 ContentType、得到 OutputStream 和 Writer 对象、Buffering 缓冲、设定 Portlet 的标题。

(3) PortletConfig 对象

和 ServletConfig 对象类似，PortletConfig 对象提供对 Portlet 初始化信息以及 PortletContext 对象存取的方法。

(4) Session 对象

由于容器不同，Portal 的 Session 对象与 Servlet 的 Session 对象略有不同。由于 Portlet 处于 Portal 服务器的缘故，Portlet 的 Session 分为 Application Scope 和 Portlet Scope。Application Scope 范围的 Session 中保存的对象，对于同一个 Portlet 应用范围内的所有 Portlet 都是可用的。而 Portlet Scope 范围的 Session 中保存的对象，只对本 Portlet 可用，其他 Portlet 即使在同一个应用中，也不可用。

(5) Preference 对象

Preference 对象被设计用来实现用户的个性化设置，可以帮助用户对 Portlet 进行符合用户需求的显示定制和行为定制。Preference 对象对于配置信息采用键－值的形式存取[27] 。

通过上面对 JSP Portlet 的分析，我们可以看到，由于 JSP Portlet 的 Web 应用没有将视图层和控制层做很好的划分，而是直接采用 Portlet API，如果在实际开发中完全采用 JSP Portlet 方式开发将会步履维艰。所以，现在的 Portal 开发商大都开始采用 MVC 的方式来开发产品，而 Struts 结构的应用就成为了最恰当的选择。但是了解基础的 JSP Portlet 的开发仍然是必要的。

2.4 Struts Portlet 的研究

2.4.1 MVC 概述

MVC 是 Model-View-Controller 的简称，即模型-视图-控制器。MVC 是

Xerox PARC 在 20 世纪 80 年代为编程语言 Smalltalk-80 发明的一种软件设计模式，现在已经被 J2EE 平台广泛使用，而成为软件设计的主流模式。

MVC 强制性的把应用程序的输入、处理和输出分开。MVC 把应用程序分成三个核心模块：模型、视图和控制器，它们分别担负不同的任务[28] 。

(1) 视图

视图是用户看到并与之交互的界面。视图向用户显示相关的数据，并能接收用户的输入数据，但是它并不进行任何实际的业务处理。

(2) 模型

模型是应用程序的主体部分。模型表示业务数据和业务逻辑。一个模型能为多个视图提供数据。由于同一个模型可以被多个视图重用，所以提高了应用的可重用性。

(3) 控制器

控制器接受用户的输入并调用模型和视图去完成用户的需求。当 Web 用户单击 Web 页面中的提交按钮来发送 HTML 表单时，控制器接收请求并调用相应的模型组件去处理请求，然后调用相应的视图来显示模型返回的数据。

这几个模块各自的功能和它们的相互关系如图 2-3 MVC 设计模式所示。

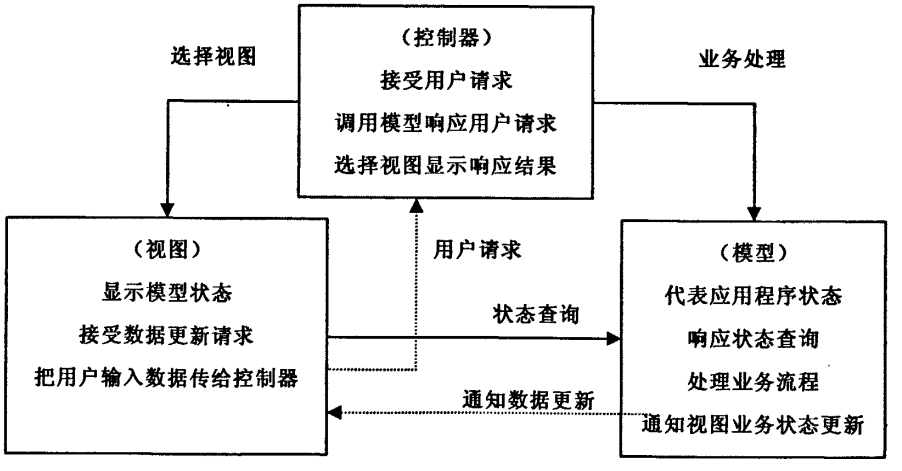

图2-3 MVC 设计模式

由于使用 MVC 需要精心的计划，由于它的内部原理比较复杂，所以并

不适合小型应用程序的开发。相反，对于开发中存在大量用户界面，并且业务逻辑复杂的大型应用系统，MVC 将会使应用软件在健壮性、代码重用和结构方面上一个新的台阶。

2.4.2 Struts 框架与 Struts Portlet 的关联

基于以上的分析可以看到，采用 MVC 设计模式的 Web 应用更加适合于企业信息集成中的 Portlet 开发，而本文则采用了基于 MVC 的 Struts 框架与 Portlet 的技术融合来实现企业信息门户系统。

(1) Struts 框架概述

Struts 框架作为 MVC 设计模式的一种具体实现，是基于 JSP Model2 的，JSP Model2 是一种联合使用 JSP 与 Servlet 来提供动态内容服务的方法，它吸取了 JSP 和 Servlet 两种技术各自的突出优点，用 JSP 生产表示层的内容，让 Servlet 完成深层次的处理任务。而在 Struts 框架中，模型由实现业务逻辑的 JavaBean 或 EJB 组件构成，控制器由 ActionServelt 和 Action 来实现，视图由一组 JSP 文件构成[29] 。如图 2-4 Struts 实现的 MVC 框架所示。

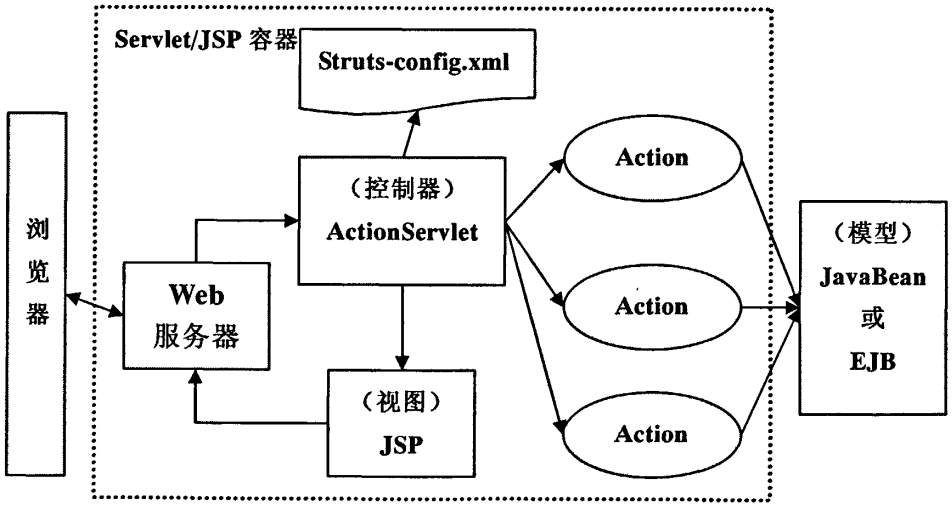

图2-4 Struts 实现的 MVC 框架

(2) Portlet 的 MVC 实现框架

对于 Portlet 来说，当用户提出 request 后，根据不同的 request（Action requests 和 Render requests）分发给对应的控制器（processAction 和

render)。其中 Action requests 通过重定义 processAction 来处理，然后通过 actionResponse 返回信息，主要用来改变 Portlet 的状态或者用于重定向；Render requests 通过 GenericPortlet 提供的 render 方法来处理，该方法根据用户的 Portlet 的模式状态请求，通过轮流调用 doDispatch 来分配 doView、doEdit 或者 doHelp 方法，然后通过 renderResponse 返回。这样 processAction 和 render 构成了 Portlet 的控制层，负责接收用户请求并调用模型组件中的业务逻辑，并将结果生成格式化的 fragment 在 container 中生成 JSP 页面标记。由此，通过 processAction 和 render 方法，将模型和视图分开，构成了 Portlet 的 MVC 框架，使得我们可以处理更加负责的业务逻辑和界面视图。如图 2-5 Portlet 的 MVC 实现框架所示。

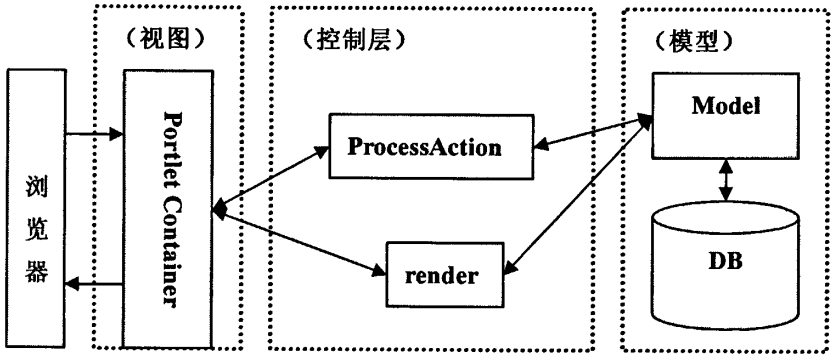

图2-5 Portlet 的 MVC 实现框架

(3) Struts 框架与 Portlet 的关联

由此可见 Struts 框架和 Portlet 的 MVC 设计模式和工作原理大体相同，我们将两者结合就形成了 Struts Portlet 的框架，只是对于 Struts Portlet 来说，要求控制层返回的是 fragment 标记，视图层通过这些标记来显示相应的 JSP 页面。所以共同的设计模式决定了 Portlet 可以重用 Struts 应用的很多功能[30] 。

2.4.3 Struts Portlet 模型的构建

基于 Struts 框架与 Portlet 的关联，我们通过 Struts 框架和 Portlet 技术的结合，主要在控制层上面做二次开发，通过对于 processAction 和 render 的继承和扩展，打包生成 Struts Portlet 的接口，从而得到 Struts Portlet 的开

发模型。

当用户提出的 HTTP 或者 WAP 的 request 后，所有的请求都通过 MainServlet 响应。本文通过 Struts 基类 ActionServlet 的扩展实现了 MainServlet，MainServlet 处理所有的请求，使得每个请求都被合适的 PortletAction 处理。portlet-ext.xml 用于部署 Portlet 自身的相应信息，而此时的页面流管理通过 struts-config.xml 来定义，对于视图的呈现，通过 tiles-defs.xml 定义页面的布局。如图 2-6 Struts Portlet 的 MVC 开发模型所示。

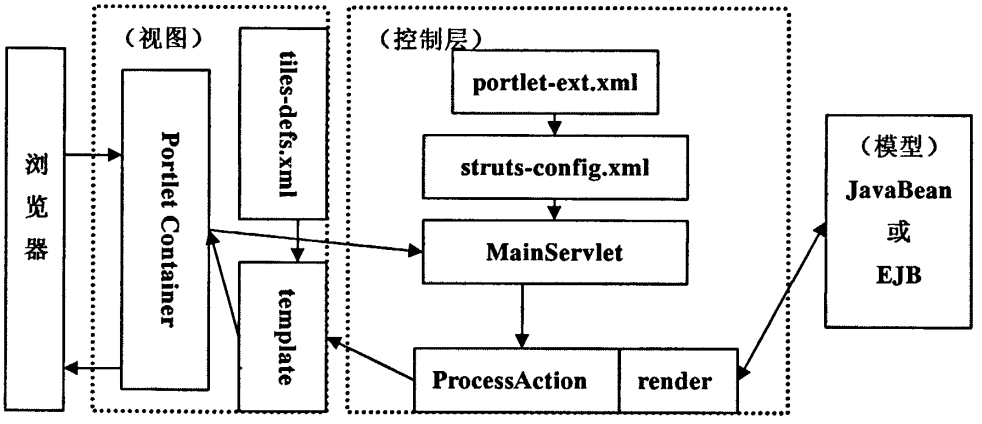

图2-6 Struts Portlet 的 MVC 开发模型

通过 Struts 与 Portlet 的结合，可以将任意一个遵循 Struts 标准的应用实现为一个 Portlet。我们通过 MainServlet 将原有的 Struts 应用和 Portal 联系起来，从而可以在该应用中引用 Portlet API。对于原有系统的封装，只需要创建 portlet-ext.xml 和 tiles-defs.xml，并对 struts-config.xml 进行改写，具体步骤将在后面的系统实现中给出。整个系统对于标签库的声明和引用体现在配置文件*.xml 中，而将旧有的 J2EE 应用集成与 Portal 之中不需要对原有系统进行大幅度的修改，只用在 JSP 文件中引用相应的 Portlet 标记库。

总之，本文通过 Struts Portlet 模型的应用，实现了企业信息门户的应用集成，给目前的企业信息的门户集成提供了一个有效的解决途径。

2.5 本章小结

本章主要阐述了 Web 应用的 Portlet 化封装的技术。第一节通过 Portlet 与 Servlet 的关系比较，着重分析了 Portlet 的生命周期。第二节分析了基于 Portlet 的 Portal 的工作原理。第三节分析了基于 JSP 技术的 Portlet 的技

术。第四节通过与前一节 JSP 技术的 Portlet 的比较，着重分析了采用 MVC 模式的 Struts 框架与 Portlet 的整合，并分析了的 Struts Portlet 的实现模型。

第3章 武钢设备中心 EIP 的分析

3.1 系统建模方法分析

目前，面向对象编程（Object Oriented Programming，简称 OOP）的分析方法已经成为软件工程的主要手段，因为通过 OOP 方法设计出的应用组件可以方便的实现软件重用，而这正是大型应用系统开发所需要的。

OOP 的主要特征包括抽象(Abstract)、封装(Encapsulation)、继承(Inheritance)和多态(Polymorphism)。抽象是把事物共同点抽取出来,以统一的方式进行概要描述的过程；封装是指将对象的属性和对象的操作组合起来，然后封装成对象，将一些属性隐藏，只提供公共方法供用户调用功能；继承是通过存在的类型定义新类型的机制，通过继承可以实现代码重用；多态即是一个名称多种形式，一个类中的方法重载就是一种多态，多态在调用方法时根据所给对象的不同自动选择不同的处理方式[31] 。它的这些特征使得在采用 OOP 方法将客观世界模型转换为计算机语言时给我们带来了极大的便利。

UML 是一种标准建模语言，它只定义了一些图以及它们的含义，而它的思想与方法无关，它是面向对象分析与设计的一种标准表示。UML 由图和元模型组成，图是 UML 的语法，而元模型则给出图的意思，是 UML 的语义。UML 通过模型来描述系统的结构或静态特征、行为或动态特征，它从不同的角度为系统的架构建模，形成系统的不同视图，包括：用例视图（User Case View）、逻辑视图（Logical View）和组件视图（Component View）等[32] 。UML 尽可能的结合了世界范围内面向对象项目的成功经验，因而它的价值在于它体现了世界上面向对象方法实践的最好经验，并以建模语言的形式把它们打包，以适应开发大型复杂系统的要求。

基于以上分析，采用面向对象开发方法具有可重用性和易于维护的特点。采用 UML 语言进行建模能够充分体现面向对象的分析过程，它贯穿在系统开发的五个阶段，分别是：需求分析、系统分析、系统设计、系统构造、系统测试[33] 。下面将从系统分析开始，建立动态模型，然后根据业务流程的需要来建立系统的静态模型，包括用例图、逻辑视图等，从而构造整个武钢设备中心的系统结构。

3.2 系统分析

在武钢设备中心 EIP 的构建中，用户权限管理子系统是核心部分，它为企业用户提供统一身份认证和基于 Web 访问的授权应用。企业用户最终都必须和此系统进行交互，用户的所有授权也必须通过该系统显示给用户，并能响应用户登录应用的请求，实现自动登录应用的过程。其他子系统均凌驾于该系统之下，分别是设备管理、OA 管理和生产管理等，而这些系统一些需要新的开发设计，一些仍将保留原有系统并通过接口开发将其整合到企业信息门户中。如图 3-1 武钢设备中心 EIP 的系统功能结构图所示。

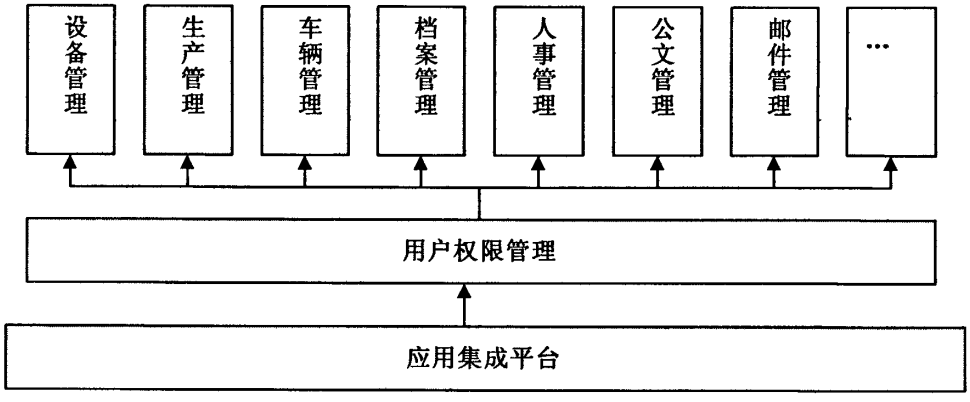

图3-1 武钢设备中心 EIP 的系统功能结构图

3.2.1 EIP 的用户权限策略

本文 EIP 的用户权限策略采用“用户-用户组-角色-Portlet”的关联方式来实现用户权限的管理[34] 。它们的相互关联如下：

用户隶属于用户组，也可以单独存在

用户组具有某种或多种角色。

角色分配给分属于不同的用户组，也可以直接分配给用户，角色直接分配给用户的功能是作为考虑系统的灵活性而做出的让步，但是操作不当会极大的增加用户管理的复杂性，所以我们建议是只把权限分配到用户组，而不是具体用户。

而操作某个 Portlet 需要具有其控制权限的角色。所以首先要定义的是用户组和角色，而对于 Portlet 来说，每个 Portlet 都有特定的 Id。

3.2.2 用户权限管理子系统分析

(1) 用户分类

对于用户权限管理子系统来说，按照用户在系统中承担的任务把用户组划分为一般用户组（Users）、应用管理员组（Application Admins）和高级管理员组（Administrators）。一般用户组是默认用户组是企业门户中最普通的用户，只是通过该系统访问自己的日常办公的应用系统，不具有任何管理操作，但是一般用户可以自定义自己的私有信息以及自己门户的 Portlet 桌面布局。应用管理员组除了访问他们授权的应用以外，通常还负责某些应用的管理工作。而高级管理员组则总体负责定义用户组、角色，并具有用户授权操作的权限。

- 一般用户组（Users）

一般用户组的功能如图 3-2 一般用户组的用例图所示。

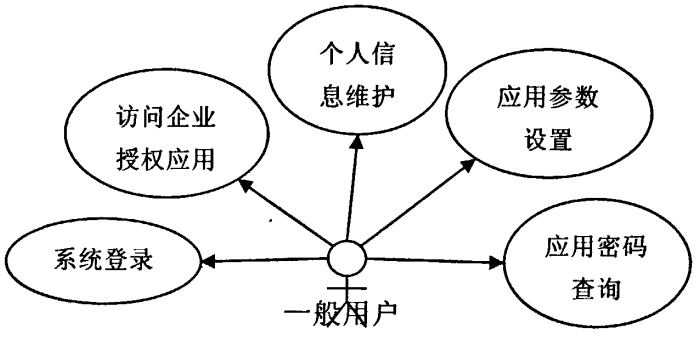

图3-2 一般用户组的用例图

a) 系统登录

一般用户通过浏览器，在登录窗口输入单点用户名和密码，如果通过验证，就可以进入系统并进行相应的操作。

b) 访问授权的企业应用

一般用户进入系统后，授权的企业应用以列表呈现给该用户，用户先选择应用类别，在对应的应用类别中只要点击需要访问的应用链接，即可进入应用。

c) 个人信息维护

已经登录的一般用户可以修改自己的个人资料，包括名字、邮件地址、

生日、图片等。

d) 应用参数设置

应用参数设置里面，包括语言的设置、时区设置、分辨率设置以及每个用户拥有的 Portlet 的设置等。

e) 应用密码查询

应用密码查询主要用于具体的应用子系统的密码的查询，由于长期使用单点登录的功能，用户可能忘记相应应用系统的密码，在这里只要输入正确的单点密码即可查询应用系统的密码。

- 应用管理员组

应用管理员相对于一般用户还拥有应用授权的管理权限，包括应用的分配和回收。如图 3-3 应用管理员组的用例图所示。

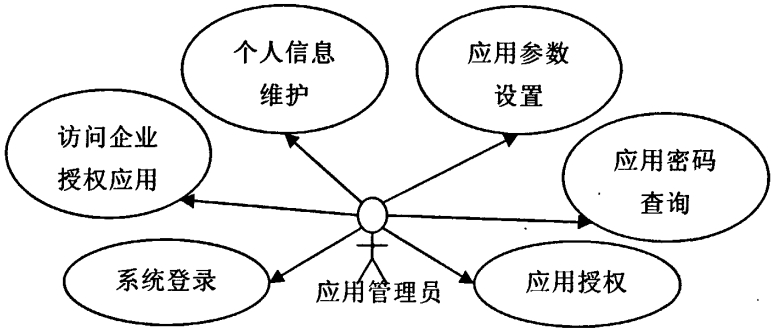

图3-3 应用管理员组的用例图

- 高级管理员

高级管理员相对于应用管理员还具有用户角色定义的功能，即可以新建、修改或删除用户以及角色，来维护系统的权限策略。如图 3-4 高级管理员组的用例图所示。

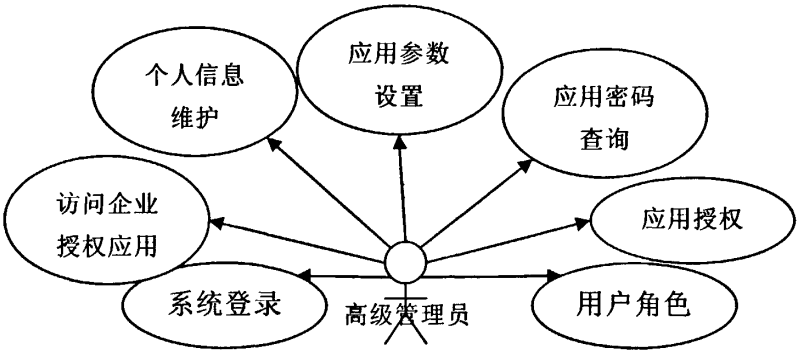

图3-4 高级管理员组的用例图

3.2.3 设备管理子系统分析

鉴于其他子系统的分析和应用都比较类似，在集成到 Portal 的过程中并不存在根本性区别，所以只选用设备管理子系统为例，其系统分析如下：

(1) 设备分类

在设备分类里面，本文只取计算机相关设备做论述。

把计算机及其周边设备按照组成和管理方法的不同分为 5 个大类，分别是：计算机类（包括单机附带的打印机和音箱等小型外部设备）、大型打印/复印机类、网络设备类和服务器类、光缆类。

计算机类包含兼容机、品牌机，其中兼容机或者品牌机设备由显示器、主机、键盘/鼠标和外设组成，外设包含打印机或者音柱。

所有计算机类都是由具体的零部件类组成，在设计的时候是一对多的关系。详细零部件分类即定义见表格 3-1 零部件分类明细表：

表格 3-1 零部件分类明细表

名称	编号	品牌	型号	状态	主机编号
显示器	编号	品牌（自定义）	型号（手工输入）	状态	主机编号
CPU	编号	类别（自定义）	型号（手工输入）	状态	主机编号
内存	编号	品牌（自定义）	型号（自定义）	状态	主机编号
硬盘	编号	品牌（自定义）	型号（自定义）	状态	主机编号
显卡	编号	品牌（手工输入）	型号（手工输入）	状态	主机编号
网卡	编号	品牌（手工输入）	型号（手工输入）	状态	主机编号
光驱	编号	品牌（手工输入）	型号（手工输入）	状态	主机编号
电源	编号	品牌（手工输入）	型号（手工输入）	状态	主机编号
打印机	编号	品牌（手工输入）	型号（手工输入）	状态	主机编号

大型打印/复印机类包含多功能一体机、大型复印机和大型打印机。

网络设备包含路由器、交换机、集线器等，其基础信息和大型打印/复印机类相似。

考虑到实际的设备维护中，维修员并不对大型打印/复印机、网络设备和服务器做具体硬件维修工作所以只对整体设备进行跟踪，而普通台式机跟服务器电脑不同，对于内部的每个配件都要做到具体跟踪，所以单独分为一个大类，而它本身又包含一套详细的零配件，每个零配件都有详细的编号，用于跟踪。

(2) 用户分类

对于设备的使用和管理，将用户分为三种类型，分别是：设备使用者、设备维修员和设备主管。使用者使用设备，并可以提出设备维修申请；维修员负责管理计算机设备具体维修、维修情况记录和报废申请；主管负责管理计算机设备台帐、零部件台帐、计算机设备配置变更、计算机设备报废。

(3) 计算机设备管理业务流程

整个管理流程如图 3-5 设备管理流程所示。

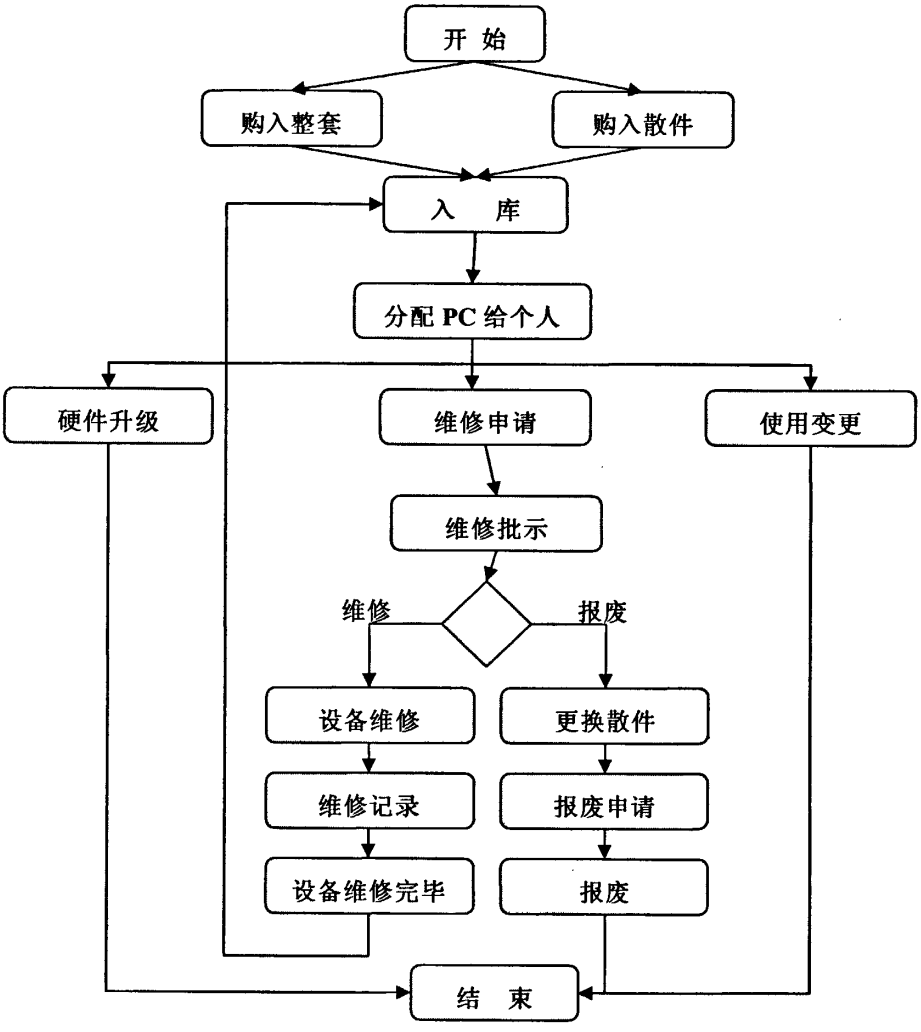

图3-5 设备管理流程

其中计算机设备维修分为两种：硬件设备维修（包括散件维修）和软件维修（光缆维修与软件维修相同）。其中硬件维修，通过维修员对设备状态作出鉴定，如果是可维修则进入维修流程，如果不可维修进入报废流程，报废申请获得主管批准后，同时执行新设备替换和旧设备报废的流程。所有维修完毕后，均进入维修情况记载，维修后的设备自动入库。

3.3 系统结构分析

从系统的层次角度来分析，我们把武钢设备中心 EIP 系统划分为硬件平台（即计算机、网络设备等）、系统软件平台、Java 虚拟机和 API、门户 Struts Portlet 框架、EIP 的 Portlet 容器、数据库系统和 iBATIS 框架层。如图 3-6 EIP 系统整体结构图所示。

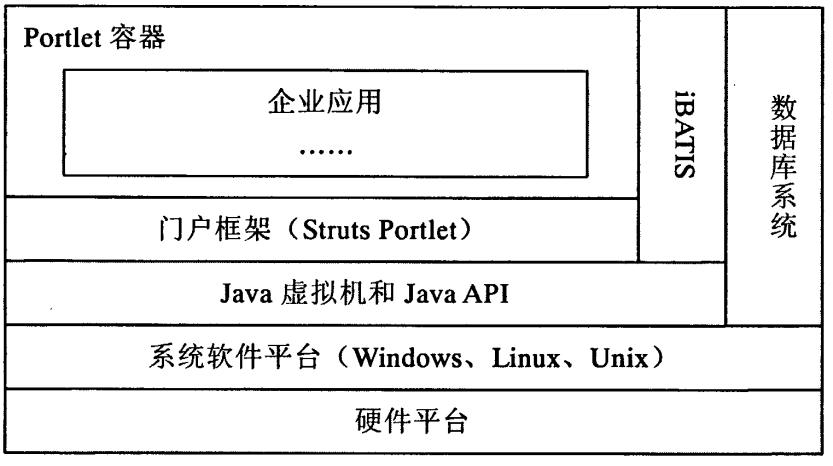

图3-6 EIP 系统整体结构图

整个系统我们采用了 B/S/D(浏览器-服务器-数据库)的分层模式[35] 。浏览器作为瘦客户端工作，所有的处理和实现都在服务器端完成。因为采用了 J2EE 的平台，所有的应用都必须是基于 Java 语言的，这也为跨平台的部署和移植提供了极大的便利。由于可以充分利用开源的组件做开发，也在经济上极大的压缩了开发成本和开发时间。

在 J2EE 应用服务器领域，像 JBoss、Tomcat 和 Geronimo 已经是商业领域的领跑者和技术领域的先行者。而这三者中，Tomcat 虽然作为一种轻量级的解决方案，并不包括 Java EE 5 的所有特性，但正是由于它的轻量级，才使它对内存的占有量比较少，并且比其它两种服务器运行起来更快。根据实际的应用开发来看，Tomcat 和 Sun Java 的结合是最稳定的，所以本系统将采用 Tomcat 作为底层 JSP 的容器[36] 。

而对于 Portlet 的选择来说，目前流行最广泛的四个开源框架分别为：Pluto、Liferay、eXo 和 Jetspeed[37] 。作为一个开源 Portal 产品，Liferay Portal 和 JetSpeed 一样帮助多个组织实现更有效的合作。与其他商业的

Portal 产品相比，Liferay Portal 提供了对 JSR-168 的全面支持，而且它还提供对多个独立系统的内容集成，有着一系列的优良特性，而且开源，所以便成为在设计实施武钢设备中心 EIP 的首选。

在数据开发层，相对于 Hibernate 和 Apache OJB 等全自动的 ORM 解决方案而言， iBATIS 的着力点在于 POJO 与 SQL 之间的映射关系。也就是说，iBATIS 并不会为程序员在运行期自动生成 SQL 执行语句。具体的 SQL 需要程序员编写，然后通过映射配置文件，将 SQL 所需的参数，以及返回的结果字段映射到指定 POJO。相对 Hibernate 等"全自动"ORM 机制而言，iBATIS 以 SQL 开发的工作量和数据库移植性上的让步，为系统设计提供了更大的自由空间，更适合于应用于新旧系统的混合集成[38] 。

其他方面，浏览器采用了 IE6.0 和 firefox2.0 作为目前通用操作系统上主流的客户端测试环境。而采用微软的 Visio2003 作为 UML 设计工具，在开发工具上，选择了 Eclipse＋Dream weaver＋Edit Plus 作为应用开发的编程工具，使用 Power Designer 和 PLSQL Developer 作为数据库设计和开发的工具。

3.4 本章小结

本章从系统建模开始对武钢设备中心 EIP 系统进行了详细的系统分析。第一节阐述了系统建模方法，提出采用面向对象的方法和 UML 的设计语言对系统进行设计。第二节对系统的用户策略以及两个设备子系统进行了详细的分析。第三节对系统的层次进行了划分，并对每个层次确立了具体的技术实现。

第4章 武钢设备中心 EIP 的设计

4.1 系统整体设计

经过前面章节的分析，考虑到系统的总体要求，在系统设计过程中采用了表示层/业务逻辑层/数据层的三层体系结构。表示层负责处理与用户的交互。业务逻辑层处理用户所需要的信息，所有的应用逻辑和控制都在这里实现，是系统中最复杂的部分，而数据层中用于存储所有的数据信息和数据逻辑，所有与数据有关的安全、完整性控制、数据一致性和并发处理等都在此实现[39] 。三层体系结构在本系统中的实现如图 4-1 三层体系结构所示。

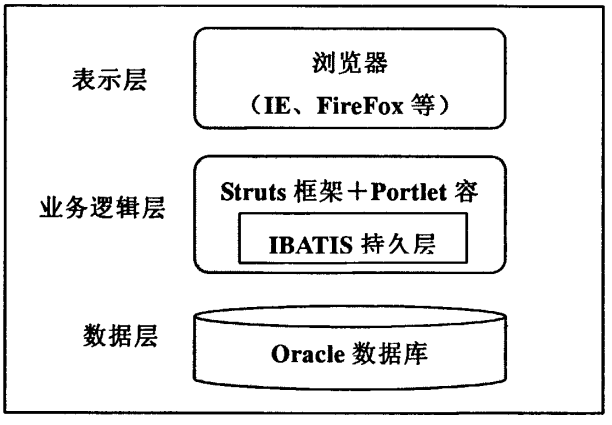

图4-1三层体系结构

表示层实现的功能为 Portlet 的布局显示和具体的应用系统的界面的显示。

业务逻辑层实现的功能包括：系统页面的导航控制、用户权限管理控制、业务逻辑的实现与数据库的通信和操作。

数据层的实现功能主要体现在系统数据库对持久层数据的存取操作管理。

4.2 数据层设计

4.2.1 数据库的选用

随着数据的不断增长，企业在长期的数据积累的同时需要更多的分析功

能，企业的运营决策越来越多的决定于原有数据的分析。所以在数据层设计实施的开始阶段，就要为未来的发展战略做好准备，数据库管理系统的选用显得尤为关键。

目前的主流数据库主要有四种，分别是：SQL Server、DB2、Sybase ASE、Oracle。相对于其他数据库，Oracle 是基于 Java 技术开发，能在所有主流平台上运行（包括 Windows），完全支持所有的工业标准，采用完全开放策略，可以使客户选择最适合的解决方案，Oracle 的并行服务器对各种 UNIX 平台的集群机制都有着相当高的集成度，而且在诸多数据库中性能最高，是最佳选择[40] 。

4.2.2 用户权限管理子系统数据库的设计

根据用户权限策略“用户-用户组-角色-Portlet”的关联方式，对应数据库的设计如下：

(1) 概念结构设计

根据策略“用户-用户组-角色-Portlet”，设计出用户权限子系统的实体联系模型，如图 4-2 用户权限管理系统 E-R 模型图所示。

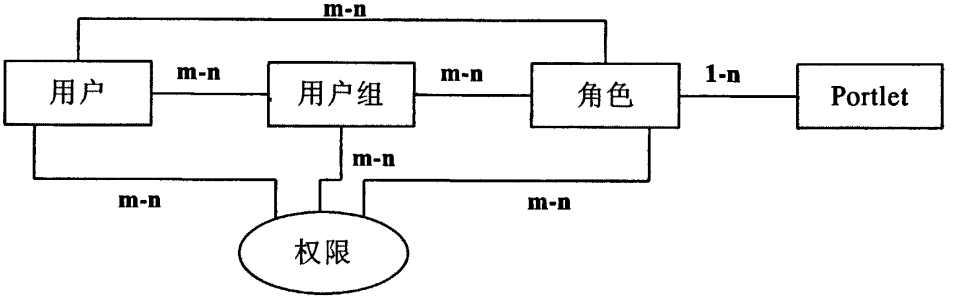

图4-2 用户权限管理系统 E-R 模型图

(2) 逻辑设计

根据以上 E-R 图的模型分析，得到如下图 4-3 用户管理子系统数据库关系模型图所示。

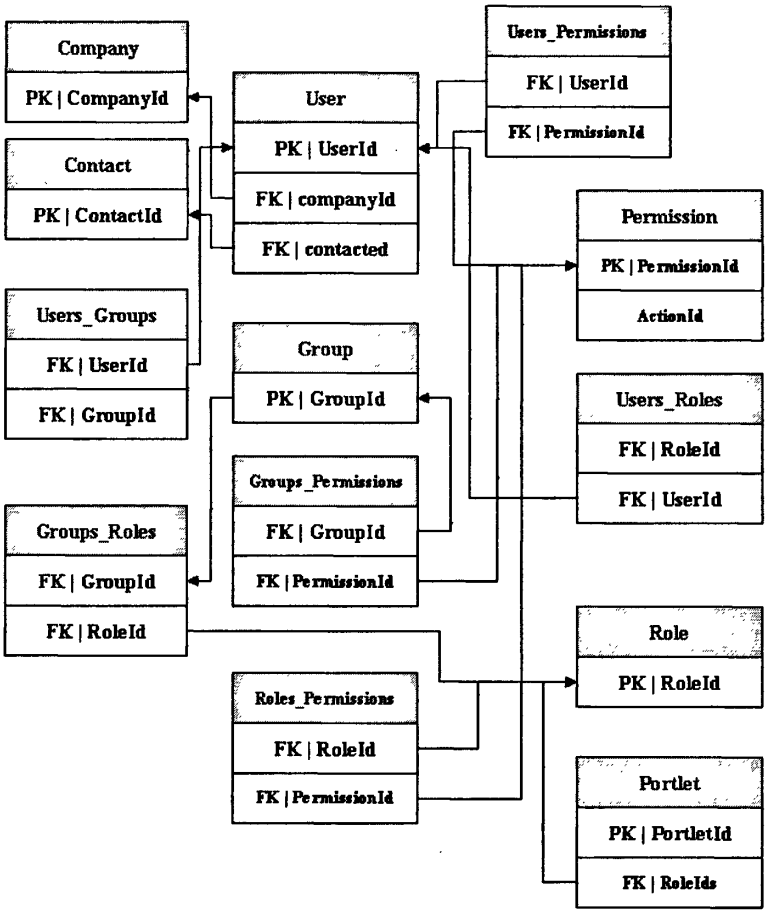

图4-3 用户管理子系统数据库关系模型图

4.2.3 设备管理子系统数据库的设计

(1) 概念结构设计

通过系统的概念结构设计，建立该子系统的 E-R 关系模型，即实体－联系模型。通常把每一类对象个体称为“实体”，而每一类对象个体的集合称为“实体集”，把实体集之间存在的各种关系称为“联系”。在本系统中，实体集有 PC 机、零部件、使用者、维修员，而实体间的联系中，PC 机和零部件属于包含关系，一台 PC 机包含多个零部件，而 PC 机和使用者为一一对应的关系，PC 机通过维修与维修员建立关联，如图 4-4 设备管理子系统 E-R 模型图所示。

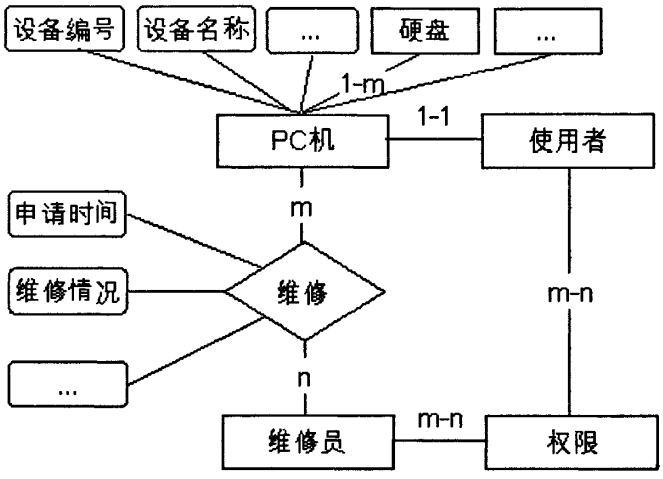

图4-4 设备管理子系统 E-R 模型图

(2) 逻辑设计

根据以上的 E-R 模型，在确定了各个数据表主键字段的基础上，根据各个实体集的属性以及它们之间的相互关系，我们构建数据库表的关系模型。如图 4-5 设备管理子系统数据库关系模型图所示。

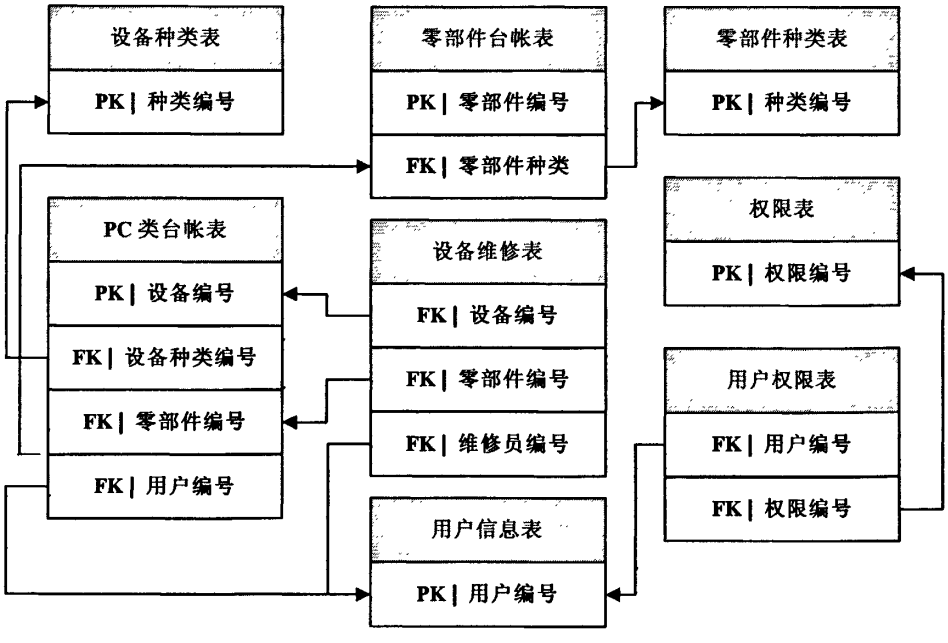

图4-5 设备管理子系统数据库关系模型图

其中设备维修表和 PC 台帐表有两个外键均指向用户信息表，表示一个为 PC 使用者，一个为维修员，他们的角色由权限表所决定。PC 类台帐表和零部件台帐表，实际上是一对多的关系，PC 类台帐中会引用到多个零部件台帐的主键作为外键。

4.3 事务逻辑设计

4.3.1 数据库连接的设计

我们在武钢设备中心 EIP 中采用了基于 JNDI（Java Naming and Directory Interface）的数据库连接池技术。数据库连接池的解决方案是在应用程序启动时建立足够的数据库连接，并将这些连接组成一个连接池，由应用程序动态地对池中的连接进行申请、使用和释放[41] 。连接池技术尽可能多地重用了消耗内存的资源，大大节省了内存，提高了服务器的服务效率，能够支持更多的客户服务。如图 4-6 数据源的运行机制所示。

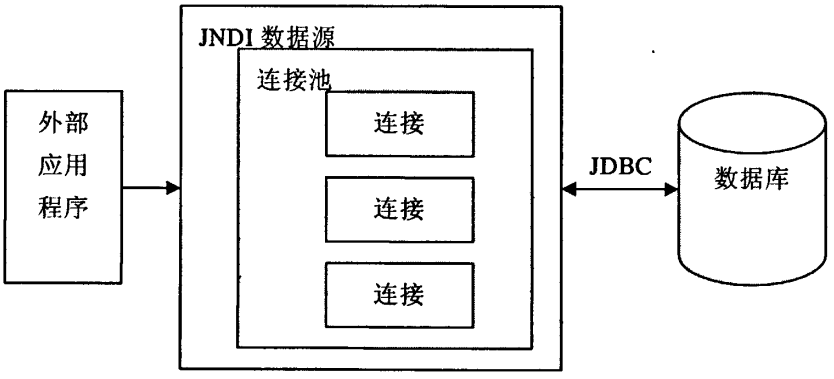

图4-6 数据源的运行机制

4.3.2 基于 iBATIS 的持久层设计

iBATIS 是一个功能强大实用的 SQL Map 工具，不同于其他 ORM 工具（如 hibernate），它是将 SQL 语句映射成 Java 对象，而对于 ORM 工具，它的 SQL 语句是根据映射定义生成的。iBATIS 以 SQL 开发的工作量和数据库移植性上的让步，为系统设计提供了更大的自由空间[42] 。

在 J2EE 开发过程中，使用数据访问对象(DAO)设计模式把底层的数据访问逻辑和高层的商务逻辑分开。一个典型的 DAO 实现有下列几个组件：

一个 DAO 工厂类、一个 DAO 接口、一个实现 DAO 接口的具体类、数据传递对象(值对象)。具体的 DAO 类包含了从特定的数据源访问数据的逻辑。

iBATIS DAO 是由 Apache 主持的开源框架项目，它允许在工程中以 DAO 模式为基础建立应用[43] 。这样我们通过建立一个 XML 文件，并声明 XMLContactDAO.java 是 ContactDAO 的实现类的方法来与关系表进行交互。

通过 Struts 和 iBATIS 的结合，将表现层和持久层分开来，通过 JavaBean 在它们之间传递信息。如图 4-7 基于 iBATIS 框架的系统架构图所示。

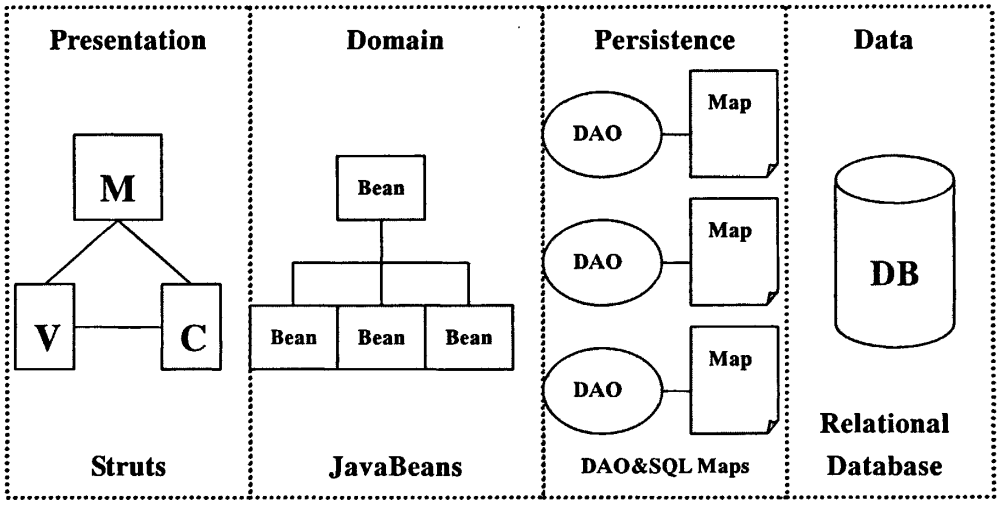

图4-7 基于 iBATIS 框架的系统架构图

通过 Portlet 接收到的请求，传递到 Struts 时，产生的对应 Action 又通过 iBATIS 的方法将所要做的动作与执行数据库动作的 SQL 语句映射到达数据库，即 request→action→JavaBean→SQLMapDao→SQL→data，在这里方法接口与 SQL 语句的映射称为 SQL Map，SQL Map 提供了一个简洁的框架，使用简单的 XML 描述文件将 JavaBean、Map 实现和基本数据类型的包装类映射成 JDBC 的 PrepareStatement。图 4-8 iBATIS 执行流程图描述了 SQL Map 的生命周期。

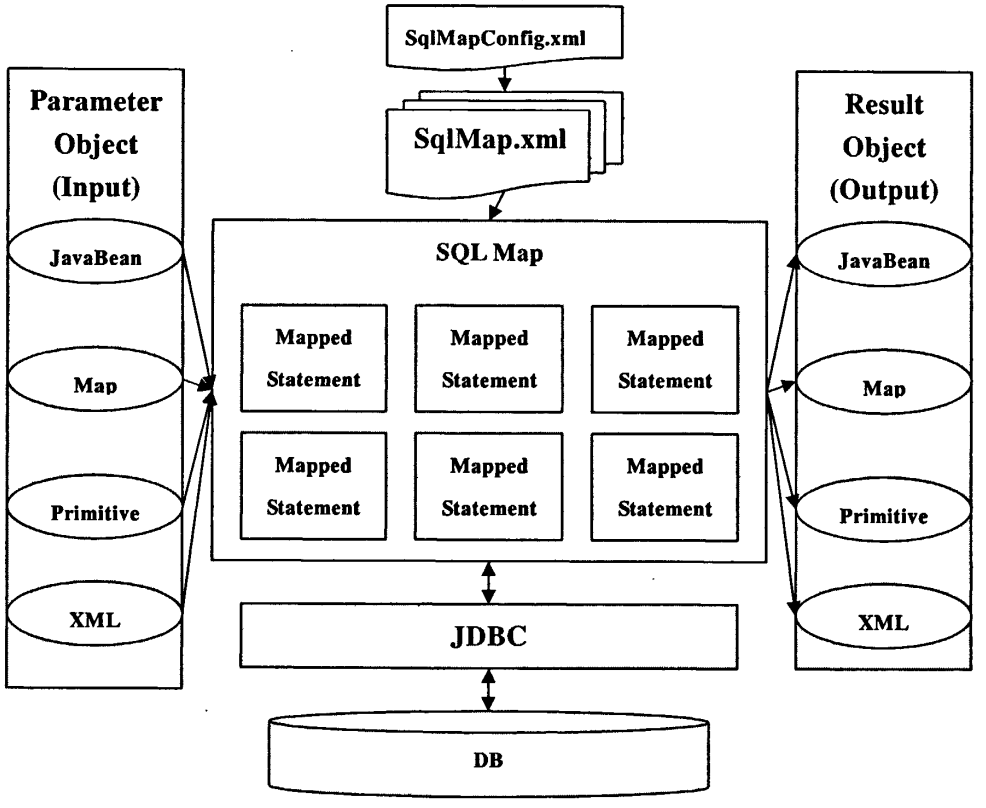

图4-8 iBATIS 执行流程图

按照 iBATIS 的设计流程，方法的实现通过 DAO 来配置，生成与数据库表字段对应的 JavaBean，然后，提取数据库操作的方法设计 Java 接口，接着在实现类中继承接口方法，并同时扩展 DaoManager，这样就将逻辑层的方法与数据库的 SQL 语句操作对应起来，从而实现了数据层的访问。具体的设计步骤如下：

(1) JAVA 类的设计

- 设计 JavaBean

在该类中定义和数据库表中栏目名对应的名称作为属性，然后生成 getter 和 setter 方法用于读取和写入该属性，JavaBean 充当着数据承载的功能。

- 设计读取数据的方法接口

该接口中将对数据库的读写操作分解称为通用的方法，每个方法都将在

下面的 SQL Map 配置中有对应的 SQL 语句来执行。但是在接口中只定义方法，并不实现它，而把具体的实现内容交给继承该接口的实现类。

- 设计该接口的实现类

实现类完全继承接口的所有方法，通过 DaoManager 来调用对应的 SQL 来操作数据库。UserDaoImpl 部分实现方法类见如下代码：

```
public class UserDaoImpl extends BaseSqlMapDao implements UserDao {
  public UserDaoImpl (DaoManager daoManager) {
        super (daoManager);
  }
  public User getUserById (String userId) {
        return (User) queryForObject("getUserByUserId", userId);
  }
.....
```

图 4-9 User 的接口及实现显示了 User 表的接口和实现类的类结构。

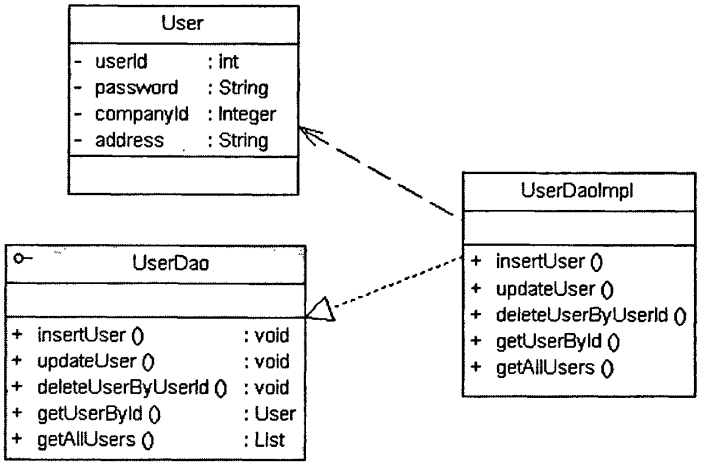

图4-9 User 的接口及实现类

(2) SQL Map 的配置

- dao.xml 的配置

从上面接口及其实现我们看到 UserDaoImpl 继承自 UserDao，而它们的联系是通过 dao.xml 的映射来完成的，通过 interface 与 implementation 的对接，将接口和实现类联系起来，部分实现方法如下：

```
<dao interface="com.wisco.portal.user.RoleDao"
```

```
implementation="com.wisco.portal.user.RoleDaoImpl"/>
```

dao.xml 对应的 iBATIS 的 SQL Map 通过以下代码在 dao.xml 中指定：

```
<transactionManager type="SQLMAP">
<property name="SqlMapConfigResource" value="sql-map-config.xml"/>
</transactionManager>
```

- sql-map-config.xml 的配置

sql-map-config.xml 文件指定了连接数据库的用户名、密码和数据库连接地址以及 SQL Map 的 SQL 执行语句。数据库的连接，通常单独定义一个 database.properties 来放置连接数据库的用户名、密码、数据库连接地址以及连接用到的驱动，而从 sql-map-config.xml 配置中读取该文件，并通过 Manager 建立连接，在这里每个连接都是作为一个完整的事务处理的。

而 SQL Map 的连接配置，通常情况下，我们会为每一张数据库建立一个配置文件，并在 sql-map-config.xml 配置中包含进来。

- database.properties 的配置

database.properties 为一个二进制文本，其中包括驱动包名称（driver）、数据库连接地址（url）、连接用户名（username）、连接密码（password）。

- User.xml 的配置

User.xml 的配置中定义了所有在 UserDaoImpl 实现类方法中调用的 SQL 语句并严格对应实现类中每个方法的输入和输出类型，部分代码如下：

```
<sqlMap namespace="User">
<typeAlias alias="user" type="com.wisco.portal.user.User"/>
<select id="getUserByUserId" resultClass="user"
parameterClass="java.lang.String">
SELECT * FROM portalAdmin.Portal_user
WHERE userId = #value#
</select>
</sqlMap>
```

4.3.3 Struts Portlet 设计

Struts Portlet 的定义首先要通过 portlet-ext.xml 和 liferay-portlet-ext.xml 两个基础的配置文件进行设置，从而将逻辑层与持久层框架 iBATIS 进行连

接。然后通过定义页面流 struts-config.xml 和布局流 tiles-defs.xml，从而实现与表示层的连接。Struts 在 struts-config.xml 中提供了集中的页面流管理，这使得它具有高度可升级性，并且使得流程代码更加模块化。通过使用 Struts，可以使用一些最佳实践并集成在 EIP 框架中。

(1) 定义 Portlet

portlet-ext.xml 的定义了 Portlet 的初始化信息，在配置文件中添加的 Portlet 节点就代表一个 Action 的配置映射。在 Portlet 节点中定义了用户管理窗口的 Portlet，对应的 portlet-class（com.wisco.portal.userManage），Portlet 的初始化路径（/ext/library/view），Portlet 的操作模式（edit）以及 Portlet 的使用权限（用户组 power-user 和 user）。

将 Liferay 的组件包信息添加到配置文件到 liferay-portlet-ext.xml 中，用于定义 Portlet 的模板、实例总数、是否重复定义等，代码如下：

```
<portlet>
    <portlet-name> userAdmin </portlet-name>
    <struts-path>ext/user</struts-path>
    <use-default-template>false</use-default-template>
</portlet>
```

struts-path 这个选项只用于 Struts Portlet 框架。如上面代码所示 struts-path 的值是“user”，表示 portal 中所有的路径为“user/*”的请求都在这个 portlet 领域的考虑范围之内。如果用户访问到这个 portlet 的话，只有路径匹配“user/*”的用户请求才被允许访问。

(2) 定义页面流

在页面流中页面的跳转需要得到一个 URL 或 Uri（Uniform Resource Identifier，即统一资源标识），这些是通过 Struts 控制器传送的，由控制器决定哪个页面应该被显示出来。如图 4-10 Portlet 页面流的跳转方式图所示。

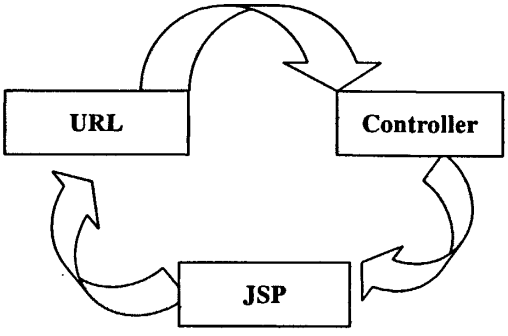

图4-10 Portlet 页面流的跳转方式图

struts-config.xml 定义页面流，在这里定义具体的跳转页面，或者 Servlet，部分配置代码如下：

```
<struts-config>
    <action-mappings>
        <action path="/ext/user/userAdmin"
            forward="/portlet/ext/user/userAdmin" />
        <action path="/ext/user/addUser "
            type="com.ext.portlet.user.action.AddUserAction">
          <forward name="portlet.ext.library.error"
            path=" portlet.ext.library.error " />
          <forward name=" portlet.ext.library.success "
            path=" portlet.ext.library.success " />
        </action>
    </action-mappings>
</struts-config>
```

(3) Action 的设计

在 struts-config.xml 配置的控制器中我们可以看到 action path 定义了一个添加用户的动作，当 JSP 页面提交一个 ActionForm 时，那么这个动作对应的 Action 类 AddUserAction 便被调用执行，以下是 AddUserAction 的部分代码：

```
public class AddUserAction extends PortletAction {
  public void processAction(
```

```
ActionMapping mapping, ActionForm form,PortletConfig config,
ActionRequest req, ActionResponse res)throws Exception {
    String username = req.getParameter("userName");//从 JSP 页面的 form 中来
    if ( null == userName || "".equals(userName) ) {//检测 userName 是否提交
        setForward(req, "portlet.ext.library.error");//如果没有提交或格式错误
    } else {//如果提交信息正确
        setForward(req, "portlet.ext.library.success");
 }
}
public ActionForward render(ActionMapping mapping, ActionForm
form,PortletConfig config, RenderRequest req,RenderResponse res)
throws Exception {
    if (getForward(req) != null && !getForward(req).equals("")){
        return mapping.findForward(getForward(req));
    } else {
        return
        mapping.findForward("portlet.ext.library.view");
    }
 }
}
.....
```

4.3.4 用户权限管理子系统设计

(1) 系统管理流程设计

系统管理模块主要是实现管理员对“用户-用户组-角色-Portlet”权限系统的管理，在这里涉及到数据库的交互，图 4-11 用户管理活动模型显示了用户管理的活动模型。

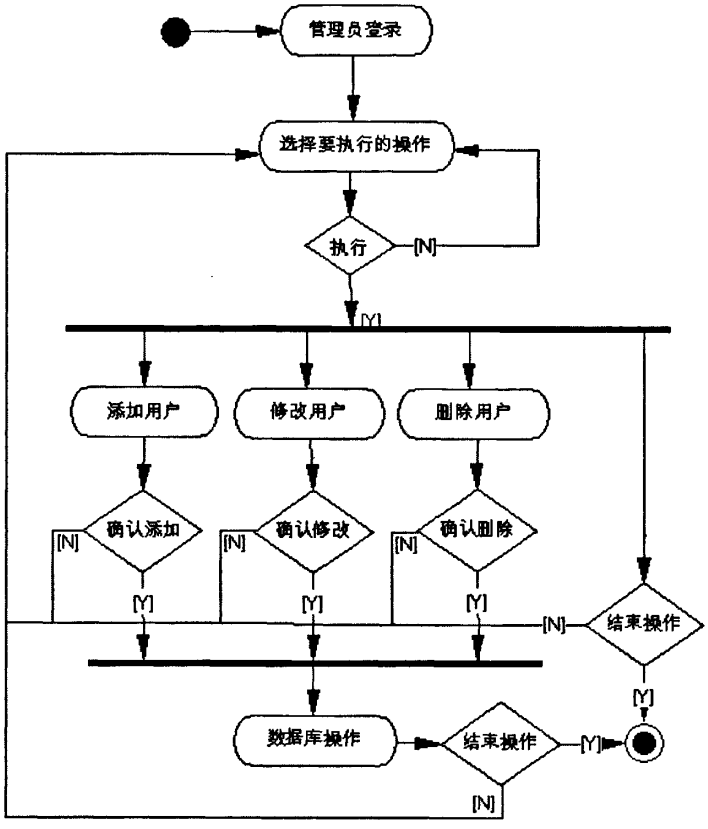

图4-11 用户管理活动模型

(2) 系统类图

以上的活动模型反映了设备中心 EIP 的用户权限管理子系统的业务流程，根据需求，设计出系统实现所需要的类，并通过类图来描述系统组件和它们之间的相互管理。如图 4-12 用户权限管理子系统主要系统类图所示。

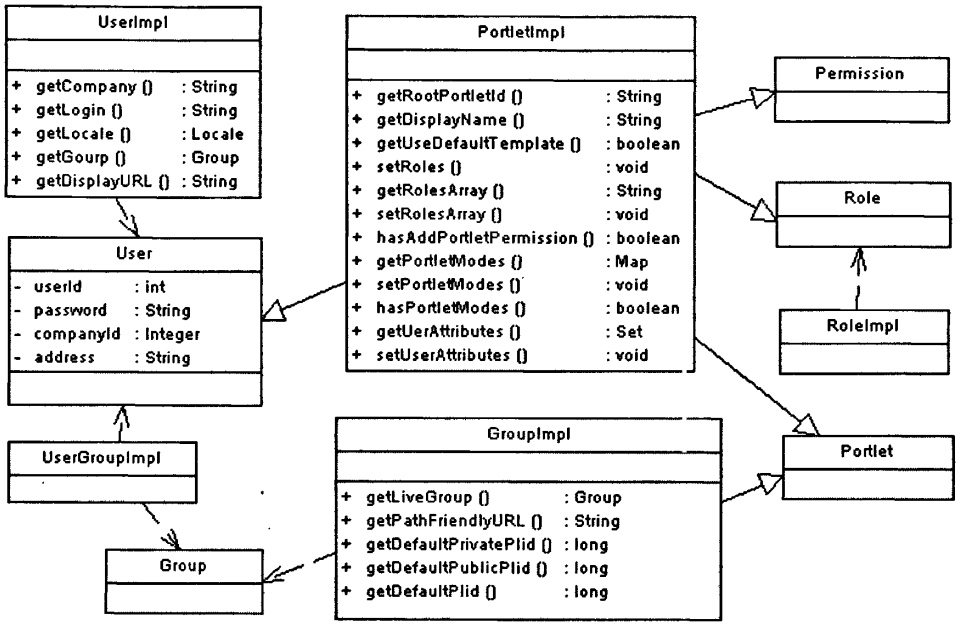

图4-12 用户权限管理子系统主要系统类图

从系统类图可以看到，PortletImpl 是系统中所有活动的中心，通过集中的功能定义，大部分的功能实现都在这里体现出来。

4.3.5 设备管理子系统设计

(1) 业务流程设计

设备管理子系统主要涉及三个角色，即普通使用者、设备维修员和设备管理员，普通使用者对设备有设备使用、设备查询和申请设备维修的动作；而设备维修员有设备维修、申请设备报废的动作；设备管理员有管理设备使用者、管理设备维修员、设备入库、分配设备给使用者、批准报废、使用变更、设备升级的动作。对此，用 UML 进行业务流程设计，将这三个角色分别放置到不同泳道，如图 4-13 设备管理子系统主要业务活动模型图所示。

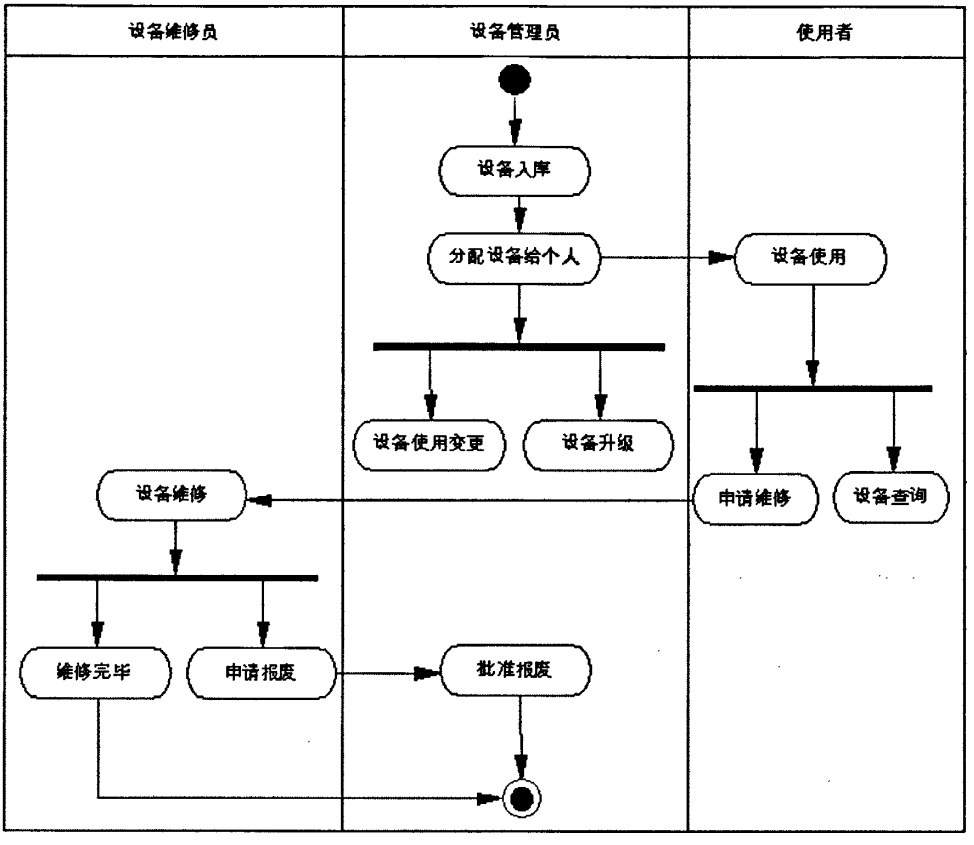

图4-13 设备管理子系统主要业务活动模型图

(2) 系统时序图

根据活动模型给出的工作流程，可以根据角色的不同而总结出数据在整个系统中的流动，下面我们通过时序图将数据的流动根据功能模块，按照时间顺序把它们展现出来。如图 4-14 设备管理子系统数据流时序图所示。

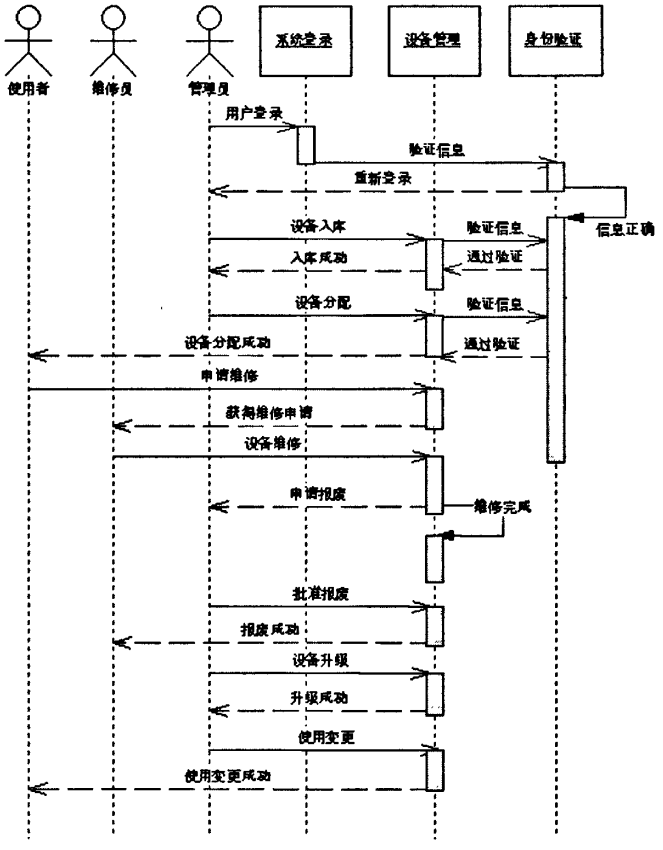

图4-14 设备管理子系统数据流时序图

4.4 表示层设计

在表示层的设计中，采用 tiles 的方式，使得一个简单的模板可以被用于定义页面布局。只要模板改变，则所有页面将会依次改变。通过定义 tiles-defs.xml 配置文件，将设计的页面模板和页面流跳转的目标绑定起来，从而实现页面模板化，如图 4-15 加入布局模板后的页面跳转模式所示。

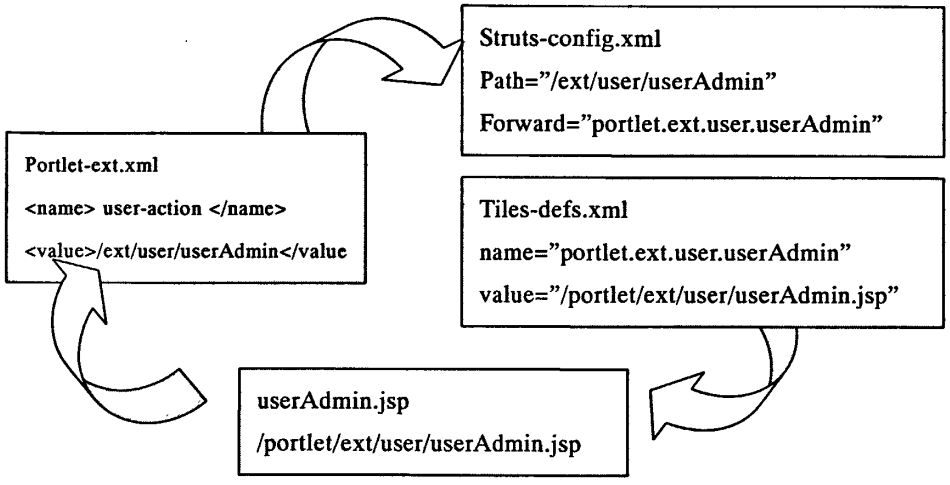

图4-15 加入布局模板后的页面跳转模式

Tiles-defs.xml 的配置文件代码如下：

```
<definition name="portlet.ext.user" extends="portlet" />
<definition name="portlet.ext.user.userAdmin" extends="portlet.ext.user">
    <put name="portlet_content" value="/portlet/ext/user/userAdmin.jsp" />
</definition>
```

4.5 本章小结

本章对武钢设备中心 EIP 系统进行了详细的设计。第一节分析了系统的三层结构设计。第二节详细研究了数据层的设计，并对两个子系统的数据库结构做了详细的设计。第三节详细研究了业务逻辑层中的数据库连接方法设计、Struts Portlet 的模型设计和两个子系统的设计，这一节的设计是本文整个系统设计的核心部分。第三节对表示层设计做了系统阐述。

第5章 武钢设备中心 EIP 的系统实现

5.1 武钢设备中心EIP的构成

按照第四章对武钢设备中心 EIP 的系统设计，将武钢设备中心 EIP 的系统分为持久层、逻辑层和表示层。而这些应用层之间的联系，详细的配置以及开发的方法，在第四章已经进行了详细的分析。最后总结整个系统的设计构成，制作出武钢设备中心 EIP 系统的整体运行部件图，其具体构成如图 5-1 武钢设备中心 EIP 系统结构图所示。

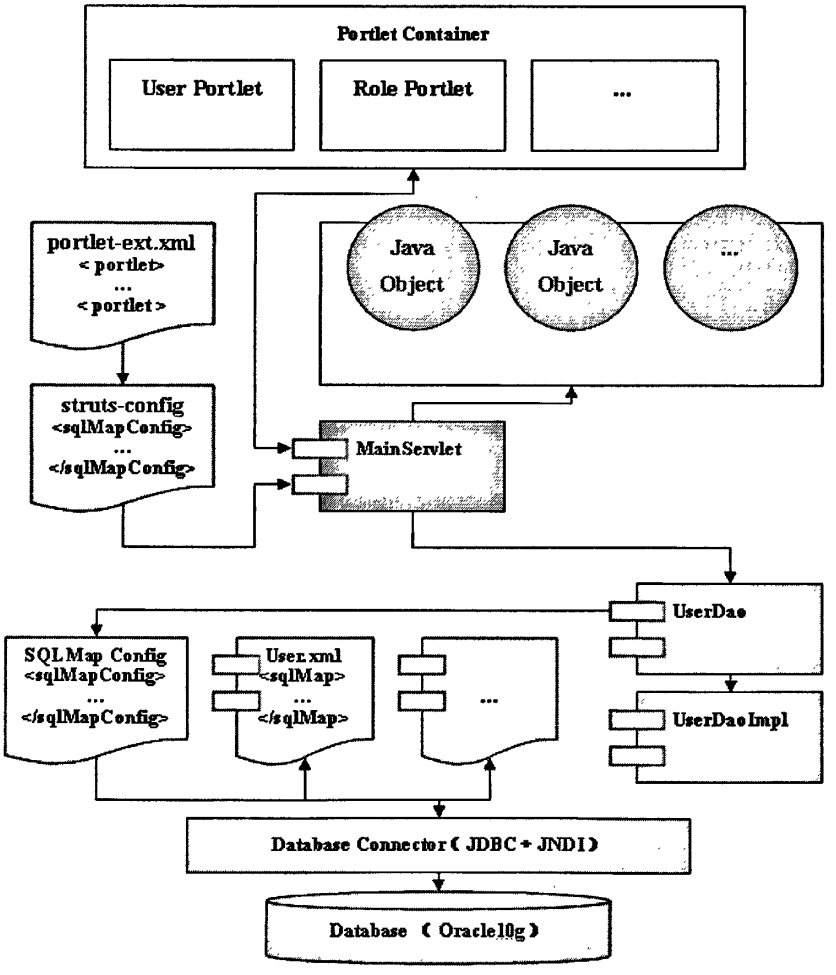

图5-1 武钢设备中心 EIP 系统结构图

5.2 系统开发环境与运行平台的部署

根据武钢设备中心 EIP 系统的结构，依次对系统由下而上的进行部署，详细的部署内容包括系统开发环境、系统运行平台组件和门户平台的搭建。

5.2.1 系统开发环境

(1) 操作系统：Windows 2003 Server

(2) 软件开发环境：jdk5-4.4.1

(3) Web 服务器：Tomcat5.5.20

(4) Portlet 容器：LifeRay4.4.1

(5) 数据库服务器：Oracle10g

(6) 开发工具：Eclipse3.2、Dream weaver CS3、EditorPlus2.12、Photoshop9.0、PowerDesigner12

5.2.2 系统运行平台组件

由于该系统是运行在 J2EE 的环境中，所以设计安装的软件有：J2EE 系统开发环境、Web 服务器、Portlet 容器以及其他实现系统功能所需要的包，详细情况见表格 5-1 系统平台组件列表所示。

表格 5-1 系统平台组件列表

软件名称	软件版本	软件下载名称
J2SE SDK	1.5.0	jdk-1_5_0-windows-i586.exe
TOMCAT	5.5.20	apache-tomcat-5.5.20. exe
LifeRay	4.4.1	Liferay-4.4.1. war
Struts	2.0.9	Jakarta-struts-2.0.9.zip
iBATIS	2.2.0.638	iBATIS_DBL-2.2.0.638. exe
Oracle	10g	Oracle10g

其中，J2SE SDK 是 J2EE 的开发环境，tomcat 是 Web 服务器，而 LifeRay 是开源的 Portlet 容器。

5.2.3 门户平台的搭建

(1) Java 运行环境的安装与配置

以上所用的应用 Web 服务器、Portlet 容器以及数据库管理系统都必须在 Java 虚拟机环境下运行，所以，安装 JRE 和 J2SE SDK 是系统部署的第一步。

JVM 虚拟机的安装，下载 jre-1_5_0_04-windows-i586-p.exe，即对应 JDK 的 jre 运行环境，点击安装。然后运行 jdk-1_5_0-windows-i586.exe 安装 Java 系统开发环境，安装目录为 c:\jdk15，然后在系统环境中配置 Java 环境变量，参数如下：

JAVA_HOME：c:\jdk15

PATH：%JAVA_HOME%\bin

CLASSPATH：%JAVA_HOME%\lib\tools.jar; %JAVA_HOME%\lib\dt.jar

(2) Web 服务器的安装与配置

在安装完成 JDK 后，点击 apache-tomcat-5.5.20. exe 安装 tomcat 的 Web 服务器到路径 c:\tomcat55 下，安装完成后并对系统环境变量做如下配置：

CATALINA_HOME：c:\tomcat55

PATH：% CATALINA_HOME %\bin

CLASSPATH：% CATALINA_HOME %\lib; % CATALINA_HOME %share\lib

(3) Portlet 容器的部署与配置

部署方式有手动部署和自动部署，本文采用自动部署的方式发布系统。自动部署要用到 Ant，Ant 是网站自动化部署中最常用的工具，本项目中要经常更新文件，并在服务器上调试，所以使用 Ant 能极大的提高部署效率。一个 Ant 部署代表一个 project，这个 project 可能包含许多 target 即子任务，本项目中 Ant 的主要任务就是将 web 文件打包成 war 包，并且拷贝到 Liferay 的网站目录下。具体部署文件代码实现如附录 A 所示。

(4) iBATIS 持久层与数据库的部署与配置

iBATIS 的配置文件包括 dao.xml、sql-map-config.xml 和对应各张数据库表与 JavaBean 的映射文件比如 User.xml，以及包括一个 iBATIS 的 DAO 控制器类 DaoConfig.class。而与数据库的连接配置文件还包括一个 database.properties 文件。

5.3 将武钢设备中心EIP实现为Portlet

5.3.1 Struts Portlet 模型的应用

本文中 EIP 系统包括两个子系统，即用户权限管理子系统和设备管理子系统，这两个子系统是作为两个 Struts 应用来实现的。在完成编码工作后，就可以根据第四章所的 Struts Portlet 设计模型，将它们分别实现为两个

Portlet 并集成到企业信息门户中，具体部署步骤如下：

(1) 添加 Liferay Portlet 所需要的包到 tomcat 的 lib 库中

将用于提供 Struts 应用与门户接口的 portal-bridges-struts-1.2.7.jar、用于提供 Portlet 基础 API 接口的 portlet.jar 包复制到 tomcat 的文件夹下面的 \common\lib\ext 中或者 portal 应用的/WEB-INF/lib 目录下面。

(2) 部署配置文件到 tomcat

将 portlet-ext.xml、liferay-portlet-ext.xml、struts-config-ext.xml 和 tiles-defs-ext.xml 四个配置文件复制到 portal 应用的/WEB-INF 目录下面，并检查文件是否部署正确。

(3) 添加 Portlet 监听器

与其他普通 Web 应用相比，Liferay Portal 的 Portlet 应用还需要在 web.xml 中添加监听器，这样通过监听器监听所有跟 Portlet 有关的信息，并将监听到的内容交给 Liferay Portal 的 Portlet 容器处理。添加代码如下：

```
<listener>
    <listener-class>com.liferay.portal.servlet.PortletContextListener</ listener-class>
</ listener>
```

(4) 设置 Struts Portlet 标签

将 struts-1.2.7-portlet-1.0.tld 复制到 portal 应用的/WEB-INF/tld 目录下面

(5) 对已经采用 Struts 框架实现的两个子系统的 JSP 文件进行修改。

主要是标签库引用的修改,在 Tomcat 服务器中，JSP 文件中采用了 Struts 标签，而在实现为 Portlet 的时候，应该用 Struts Portlet 标签来代替 Struts 标签。具体为：在 web.xml 中添加标签

```
<taglib>
        <taglib-uri>/WEB-INF/tld/struts-portlet.tld</taglib-uri>
        <taglib-location>/WEB-INF/tld/struts-1.2.7-portlet-1.0.tld</taglib-location>
 </taglib>
```

(6) 用 JDK 命令 jar 把文件打包部署

即把 class 文件、jsp 文件、jar 包、JSP 标签库文件、Web 应用部署描述文件 web.xml、Struts Portlet 配置文件 portlet-ext.xml、liferay-portlet-ext.xml、struts-config-ext.xml 和页面布局模板文件 tiles-defs-ext.xml 打包成 web archive(WAR)文件并复制到 Liferay 的\WEB-INF\deploy 目录下，重启动

Tomcat，即可完成部署。

5.3.2 用户权限管理子系统的实现

本文中用户权限管理子系统的具体实现是以 Portlet 的形式集成在了门户中，其实现页面包括了系统管理和用户个人信息管理两部分，而系统管理又细分为用户管理、用户组管理、角色管理、Portlet 管理和权限管理等。

(1) 个人信息管理

用户登录后，进入用户权限管理模块，点击左边导航列表中的用户信息按钮即可进入个人信息管理页面，在这里用户可以修改自己的用户名、密码以及其他个人信息等。具体实现页面如图 5-2 个人信息管理页面所示。

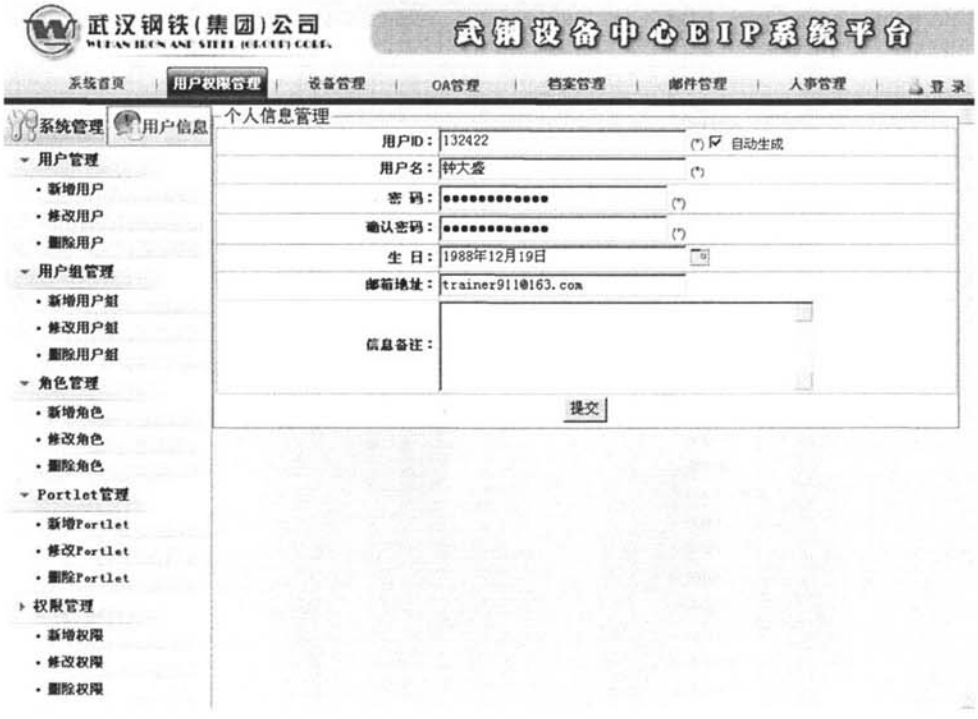

图5-2 个人信息管理页面

(2) 用户管理

在系统管理中，可以对所有用户信息进行管理，为用户分配用户组，也可以直接对用户进行角色分配。如图 5-3 新增用户信息页面所示。

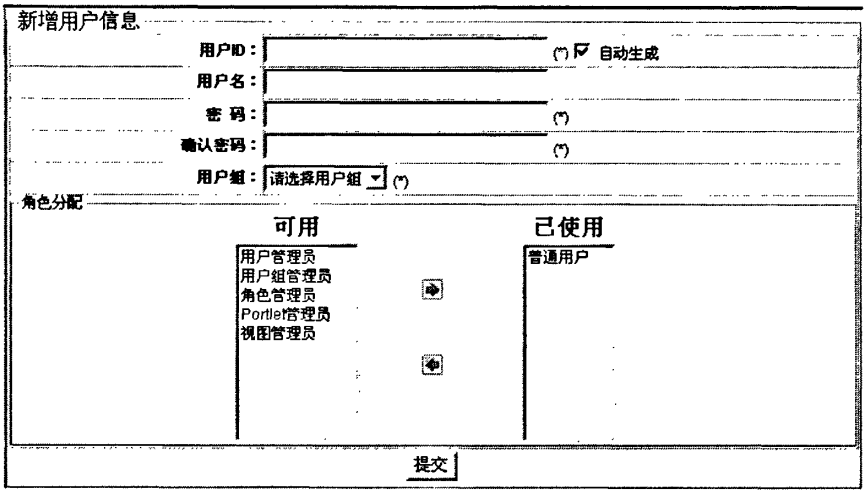

图5-3 新增用户信息页面

(3) 用户组管理

在用户组管理中，主要操作是为每个用户组分配对应角色。如图 5-4 新增用户组信息页面。

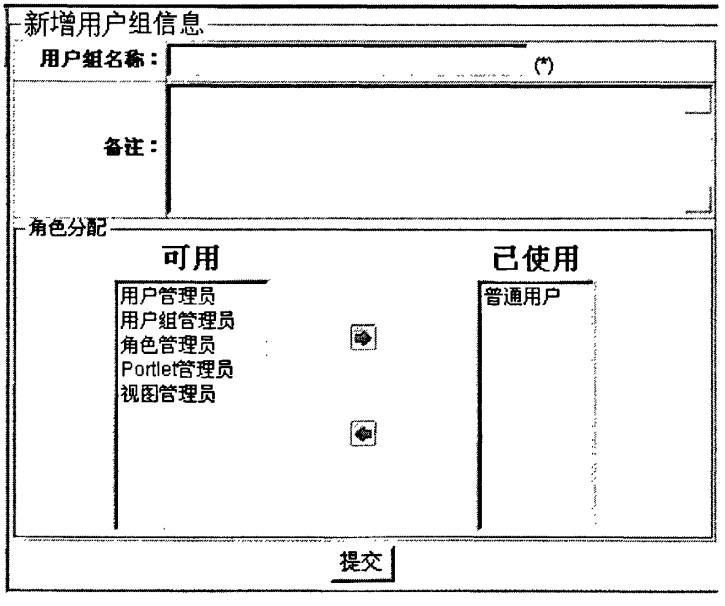

图5-4 新增用户组信息页面

(4) 角色管理

角色管理中，除对角色可以设置基础的系统权限以外，还可以对角色分配具体的 Portlet 的权限，其中每个 Portlet 的操作权限又分为编辑、浏览和帮助这三种。

角色与 Portlet 的对应关系存在于 portlet-ext.xml 的配置文件中，我们通过 linkRoles()方法将角色和角色对应 Portlet 的建立连接，部分实现代码如下：

```
public List linkRoles() {
    List linkedRoles = new ArrayList();
    Iterator itr = _unlinkedRoles.iterator();//取得未连接的角色
    while (itr.hasNext()) {//依次取出角色
        Role unlinkedRole = (Role)itr.next();
        //从已建立连接的 Portlet 中查找该角色
        Role roleLink = (Role)_roleMappers.get(unlinkedRole);
        if (Validator.isNull(roleLink)) {//如果该角色不存在
            if (_log.isDebugEnabled()) {//记录日志
                _log.debug(
                    "Linking role for portlet [" + getPortletId() +
                        "] with role-name [" + unlinkedRole +
                            "] to role-link [" + roleLink + "]");
            }
            linkedRoles.add(roleLink);//插入该角色
        }
        else {//如果该角色存在则报错
            _log.error(
                "Unable to link role for portlet [" + getPortletId() +
                    "] with role-name [" + unlinkedRole +
                        "] because role-link is not null");
        }
    }
    return linkedRoles;
}
```

(5) Portlet 管理

在 Portlet 的管理中，可以对每个 Portlet 做修改和删除的动作，也可以

对 Portlet 分类，然后统一对每个 Portlet 类别进行操作。对 Portlet 的管理是整个用户权限管理的核心部分，涉及到 Portlet 的模式设置、对 Portlet 的权限的操作。下面的详细代码实现了获取 Portlet 支持的所有模式的方法：

```
public Set getAllPortletModes() {
    Set allPortletModes = new TreeSet();
    Iterator itr1 = _portletModes.entrySet().iterator();//获取该 Portlet 支持的模式
    while (itr1.hasNext()) {//将模式格式化，插入 allPortletModes 中
        Map.Entry entry = (Map.Entry)itr1.next();
        Set mimeTypeModes = (Set)entry.getValue();
        Iterator itr2 = mimeTypeModes.iterator();
        while (itr2.hasNext()) {
            String portletMode = (String)itr2.next();
            allPortletModes.add(portletMode);
        }
    }
    return allPortletModes;//返回 allPortletModes
}
```

(6) 权限管理

在权限管理中，将权限分为系统权限和 Portlet 权限，系统权限用于用户权限管理系统本身，而 Portlet 权限又分为默认权限和独立权限，默认权限针对一类 Portlet 进行预设置，而独立权限是单独定义的权限，在对具体 Portlet 进行设置时，将权限分配给 Portlet。

5.3.3 设备管理子系统的实现

在设备管理子系统中，根据系统的流程，在预设了基础数据字典（包括各种设备分类、零部件分类、设备状态分类、设备所属部门分类、人员分类等等）后，遵循的操作步骤为：设备入库和分配、设备报修、设备维修、设备升级和变更、设备报废。此外，系统还具备设备跟踪和设备统计等功能。如图 5-5 设备管理子系统的实现页面所示。

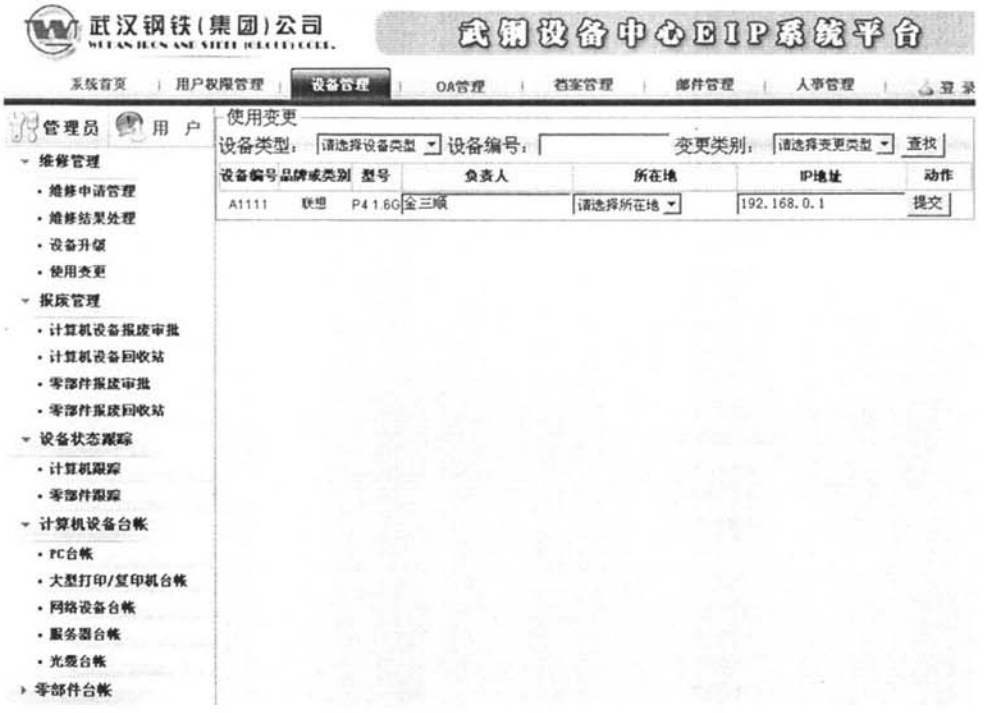

图5-5 设备管理子系统的实现页面

(1) 设备入库和分配

设备入库分为 PC 台帐入库、大型打印/复印机入库、网络设备入库、服务器入库、光缆入库和零部件入库等。

(2) 设备报修

当使用者的设备出现故障后，提出设备维修申请上报给维修员，维修员接受申请单，并实施维修动作。

(3) 设备维修

维修员受理维修申请，并对设备做出处理，然后在系统中记录维修结果。如图 5-6 维修结果处理的实现页面所示。

维修结果处理

序号	设备编号	品牌或类别	型号	故障描述	申请人	所在地	状态	动作
1	P2014	大型打印机	6300	卡纸	刘静	档案室	已受理	结果记录
2	C3445	联想电脑	X61	黑屏	张良	技工办	已受理	软件故障 | 硬件故障
3	N257	网络设备	TpLink2100	断电	吴波	网络中心	已受理	结果记录
4	F271	光缆	ADSS	无信号	张学伟	网络中心	已受理	结果记录

[首页 | 上一页 | 下一页 | 末页] 页数: 1

图5-6 维修结果处理的实现页面

(4) 设备升级和变更

管理员可以根据使用者或者维修员的申请对设备进行升级，或使用者的变更。如图 5-7 设备升级的实现页面所示。

设备升级

设备编号： 查找设备

大型设备升级（更换）

设备类型	原设备编号	品牌	型号	设备负责人	所在地	升级设备编号	动作
大型打印机类	P2300	联想	FP91	刘涛	档案室	查找库存同类型设备	确认升级

PC设备升级(零部件添加或更换)

CPU	P4 1.6G W211122		查找库存同类型设备
内存1	HY 400X 512M		查找库存同类型设备
内存2	HY 400X 512M		查找库存同类型设备
硬盘1	希捷 7200 80G		查找库存同类型设备
硬盘2			查找库存同类型设备
显卡	七彩虹 256M		查找库存同类型设备
光驱	SONY 52X		查找库存同类型设备
音响			查找库存同类型设备
打印机			查找库存同类型设备
网卡	TPLIINK 100M		

确认升级

图5-7 设备升级的实现页面踪和统计，包括维修记录跟踪和统计、设备升级跟踪和统计、设备使用变更记录跟踪和统计。

通过以上可以看出，应用本文提出的设计模型和框架，成功地运用本文所设计的 Struts Portlet 和 iBATIS 框架将用户管理权限子系统和设备管理子系统集成到了企业信息门户中。

5.4 系统性能分析

本系统的开发基于 J2EE 平台的多层体系结构，基于 MVC 设计模式，并使用 Struts Portlet 实现表示层和业务逻辑层，iBATIS 实现持久层。下面本文将运用软件工程的分析方法分别从扩展性、维护性、移植性和系统性能几个方面来分析本系统的性能。

(1) 扩展性

本系统采用组件式开发方式，利用 Struts Portlet 可以轻松的实现各业务组件的动态装配，因此，系统具有很强的可扩展性，便于系统的二次开发以及日后的系统升级和扩展。

(2) 维护性

本系统采用基于 Struts Portlet 整合架构的多层结构设计，使用 MVC 模式分离了表示层和业务逻辑层，隐藏了业务逻辑，使得各层之间松散藕合，独立的修改而不影响对方，提高了可维护性。Struts Portlet 框架的使用将 J2EE 层次结构中的业务层分离为业务逻辑层和数据持久层，这样业务逻辑便交给 Struts 处理，而数据访问则交给 iBATIS 处理，使得层次结构更加清晰，也有利于系统的维护。

(3) 移植性

在跨平台方面，由于 Java 语言本身的平台无关性及 J2EE 标准的平台无关性，我们在不同操作系统之间切换只需要简单的修改数据库连接参数即可。在数据库方面，由于本系统采用 iBATIS 实现对数据的存取，iBATIS 在设计上实现了 JavaBean 与数据库的完美映射，具有良好的隔离性和维护性，同时也 iBATIS 提供了对不同数据库的良好支持。

(4) 系统性能

对于复杂的业务管理信息系统而言,性能的瓶颈在数据库的读取操作上面。iBATIS 的持久层框架提供了优秀的性能优化机制，比如事务管理、数据库连接池等，而我们只需要修改 iBATIS 的配置文件即可实现，对于灵活定制的企业业务和系统性能优化的特殊性，iBATIS 具有无可比拟的优势。

另外，由于采用了轻量级的 Tomcat 服务器，配置运行小巧灵活，较其他 EJB 的容器 Weblogic. WebSphere 等，在系统运行性能上，也有很大优势。

5.5 本章小结

本章主要分析了武钢设备中心 EIP 系统的实现方法和步骤。第一节详细地阐述了系统开发环境、系统平台的部署方法以及 iBATIS 框架的部署。第二节分析了武钢设备中心 EIP 系统的整体构成。第三节分析了 Struts Portlet 框架在用户权限管理子系统和设备管理子系统中的实现。第四节对 Struts Portlet 框架的在武钢设备中心 EIP 系统的应用进行了性能分析。以上内容是基于 Struts Portlet 和 iBATIS 框架的武钢设备中心 EIP 系统的技术实现。

第6章 总结与展望

6.1 总结

EIP 系统的开发是软件开发中遇到的最为复杂的问题，是因为应用系统的种类繁多，架构各异。本文通过利用现有的技术和服务，在充分分析现有应用系统架构的基础上，进行了基于 Portlet 的武钢设备中心 EIP 系统的研究和实现尝试。通过查阅了大量的资料、文献和参考部分成功案例后，完成了整体架构的设计和详细的实现方案，本文的主要研究工作总结如下：

(1) 通过 JSR-168 规范和 WSRP 规范，分析了 Portlet 的运行机制，从而明确了 Web 应用和 Portlet 之间的区别和联系。

(2) 对基于 JSP Model1 和 MVC 设计模式的 Web 应用 Portlet 化封装问题分别进行了研究，并设计了 Struts Portlet 的实现模型。

(3) 根据面向对象设计思想，采用 UML 分析和设计了武钢设备中心 EIP 系统。

(4) 采用 Struts Portlet 技术，并且通过与 iBATIS 框架的整合，实现了武钢设备中心 EIP 系统。

6.2 展望

从技术角度看，本文中所实现的 EIP 系统可以作为 Web 应用运行在 IE 浏览器中，也可以作为 Portlet 运行在除了 LifeRay 以外的任何一个遵循 JSR168 规范的门户平台中，实现了 Portlet 作为门户组件所具有的可拔插性。但是从应用角度来看，本系统在以下方面还有待进一步的扩充：

(1) 在用户权限管理中对统一用户认证和单点登录的设计和实现没有考虑，需要进一步的完善。

(2) 在数据层采用的 ORM 技术并未考虑数据分析的高级功能，如果能对系统中的数据构建数据仓库，采用数据挖掘技术对数据进行统计分析，则可以更好的为企业管理者提供决策支持服务。

(3) 本系统的设计是基于 Java 技术，对于其他技术开发的系统的整合仍然缺乏足够的支持。

EIP 已经成为企业信息化建设战略规划、选型、实施与运营中必须考虑的重要组成部分。从未来的 EIP 技术发展来看，越来越多的技术整合将走向成熟，所有的企业信息门户产品开发商都在利用现有流行技术在新一代的 EIP 产品研发中做出尝试。而对于企业来说也在系统的整合中不断的积累着对 EIP 的实施经验，而且也热切的期待着日臻完善的 EIP 产品来提升企业的管理能力和效率。

参考文献

[1] 常立志. 企业信息化建设之系统整合. 浙江冶金, 2006,(03): 16～17

[2] 车颖. 企业信息门户与办公自动化的集成应用. 广东科技, 2007,(11):59～61

[3] Collins H. Enterprise knowledge portals :next generation portal solutions for dynamic information access, better decision making, and maximum results. New York :: American Management Association,, 2003. 5～11

[4] 陈重威. JSR168和WSRP在企业门户的应用. 哈尔滨师范大学自然科学学报, 2004, 20(04): 49～55

[5] Shum J,Chow A,Mar R,et al. Liferay Portal Content Management System Guide. , 2007

[6] Thompson R, Clark C J B. OASIS Web Services for Remote Portlets (WSRP) TC , http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrp. , 2008

[7] Hong-tao Y,Yao-wu W, Yue-song Z. Process-oriented organizational structure andits application in real estate development enterprises in China. Journal of Harbin Institute of Technology, 2006,(1): 54～56

[8] Idc . Enterprise Portal Adoption Trends in the Process Manufacturing Industry , 2004. 14

[9] 陈志凌, 陈刚 . 基于Portlet的网格Portal的介绍与应用. 中国科学院研究生院学报, 2006, 23(04): 442～446

[10] Microsoft . SharePoint Portal Server 2003 概述 , http://www.microsoft.com/china/office/sharepoint/prodinfo/overview.mspx. , 2003

[11] Buckner T, Fischer P, Hesmer S, et al. Portlet Development Guide: Second Edi-tion. : WebSphere Portal Development Team, 2003

[12] 王申源, 董传良, 刘英丹. 基于Portlet的校园信息门户的设计与实现. 计算机仿真, 2005,(3): 103～106

[13] 刘美荣, 刘建, 马殿富. 支持Portlet互操作的容器. 计算机集成制造系统, 2007, 13(6): 1241～1243

[14] 吴铮, 陈福生. Portlet间的协作框架. 计算机辅助工程, 2006, 15(4): 23～26

[15] 王清心, 胡建华, 丁家满. JPOSE Portlet的内部机理及其规范实现. 计算机工程与设计, 2006,(6): 48～51

[16] 柴晓路. 架构Web Service: 交互界面 ，Web服务定义的核心 ，http://www.souzz.net/html/edu/net/net9/3046.html. , 2005

[17] Hepper S, Hesmer S. Introducing the Portlet Specification , http://www.javaworld.com/javaworld/jw-08-2003/jw-0801-portlet.html. , 2008

[18] 闫明, 马玉祥. 基于J2EE企业架构的解决方案. 现代电子技术,2005, 28(01): 21～23

[19] 涂曙光, 熊明锋, 陈曦, et al. SharePoint Portal Server 2003 深入指南. 北京: 电子工业出版社, 2006. 2～8

[20] Bea Systems I. Portlet Developer's Guide, http://edocs.beasys.com/wlac/portals/docs/portlet.html. , 2007

[21] 邹承明, 钟珞, 童琪薇. 基于Jetspeed的Portlet开发探讨. 武汉理工大学学报, 2004, 26(06): 71～73

[22] 孙璐, 葛声, 马殿富. 一种面向Web服务的门户组件机制的设计与实现. 计算机技术与发展, 2006,(08): 10～12

[23] Ivanov M . Portlet servant communication, http://www.jasig.org/wiki/display/UP3/Portlet+Servant+API. , 2006

[24] 田昌鹏, 张升平. 用Portlet技术实现数字化校园信息资源整合. 计算机科学, 2007, 34(08): 293～295

[25] Portlet API Document , http://www.bluesunrise.com/portlet-api/index.html , 2003

[26] 李立焰, 谭庆平, 缴名扬, et al. JSR 168 Portlet应用中保持mvc模式带缓存服务调用方案的研究和实现. 计算机与信息技术, 2006,(04): 52～54

[27] 牛向华, 黄永忠, 姜国庆. Velocity与JSP技术在portlet开发中的研究及对比分析. 微计算机信息, 2005, 21(30): 162～164

[28] 孙卫琴. 精通Struts:基于MVC的Java Web设计与开发. 北京:电子工业出版社,2004. 9～16

[29] 鲁鹏, 曹洪军. 基于Struts和EJB的Web Service框架研究. 计算机与信息技术,2007,(03): 12～19

[30] Interface21. Portlet MVC Framework , http://static.springframework.org/spring/docs/2.0.x/reference/portlet.html. , 2007

[31] 孙卫琴. JAVA面向对象编程. 北京: 电子工业出版社, 2006. 2～5

[32] 高柯夫. UML应用建模研究. 武汉生物工程学院学报, 2006,(03): 23～25

[33] 马天蔚. 软件开发工程化——IBM谈Rational发展策略. 每周电脑报, 2003,(36):63～64

[34] Eamoi. Liferay Portal-Portlet Development Guide, http://www.blogjava.net/eamoi/archive/2006/03/30/38178.html. , 2006

[35] Yanfang L, Renlong Z. Based on the Software Architecture of the Layering Mo-de Used in Hitachi ERP System Project. Journal of ChangChun University of Scie-nce adn Technology, 2005, 28(03): 4～8

[36] 刘冲, 张海玥, 张卫东, et al. 配置Tomcat使Apache服务器支持Java动态网页编程. 计算机应用, 2001, 21(Z8): 109～110

[37] Opensource. Java开源门户系统分类 , http://www.open-open.com/17.htm. , 2006

[38] 张佳阳. J2EE项目中数据访问策略的选择. 电脑知识与技术(学术交流), 2007,(11): 32～33

[39] 王先国, 曾碧卿, 舒纲旭. 基于J2EE的三层管理信息系统. 科技咨询导报, 2007, (18): 26～27

[40] 李学强, 罗省贤. 基于ORACLE系统的数据库性能优化设计. 通信与广播电视, 2006,(04): 16～18

[41] 刘庆杰, 高焕芝, 宋晓刚, et al. Web开发中使用数据库连结池技术. 防灾科技学院学报, 2007, 9(03): 115～117

[42] 周相兵, 杨小平. 基于iBATIS的持久性层次框架研究. 科学技术与工程, 2007, 7(16): 4062～4066

[43] 吴建设. 基于数据持久层的iBATIS DAO研究与应用. 黄石理工学院学报, 2007, 23(04): 19～22

附录 A　　Ant 自动部署配置文件

```
<project name="delploy" default="war" basedir=".">
    <property name="auto_deplpy"
    value="E:\liferay-portal-tomcat\webapps">
    </property>
    <property name="war_name" value="device.war">
    </property>
    <property name="tomcat.dir" value="E:\liferay-portal-tomcat">
    </property>
    <target name="deletewar">
        <delete file="${war_name}" />
    </target >
    <target name="war" depends="deletewar">
        <war destfile="${war_name}" webxml ="./device/WEB-INF/web.xml">
            <fileset dir="./device" excludes="**/web.xml" />
        </war>
    <delete file=" ${auto_deplpy}\${war_name}"/>
    <copy todir="${auto_deplpy}" file="${war_name }">
    </copy>
    </targe>
</project>
```

附录 B　在研期间发表的论文和参研的项目

1. 发表的论文

[1] 基于模式识别视频搜索技术的研究. 福建电脑，2007,8

2. 参与研发的项目

[1] 武汉钢铁集团设备中心综合办公信息系统

[2] 武汉钢铁集团职业技能鉴定管理系统

致 谢

在张南平导师的悉心指导下，本论文终于出稿了。回顾这三年的研究生生活，在这段生命历程中，张老师的教导和帮助给予了我宝贵的财富，他为我指引了研究的领域和方向，教导了我学术研究的方法。从选题到论文成稿，张老师为我排解了无数的困惑，提出了许多关键性的意见。师生之间建立的融洽关系也使我在学业上和生活上受益匪浅。在此，谨对张老师的辛勤培养和关怀表示衷心的感谢。

感谢计算机学院各位老师对本文的仔细评阅，他们所提出的中肯意见使我受益匪浅，也使得本文臻于完善，各位老师治学的严谨和对学生的和蔼态度都使我难忘，感谢各位老师为本文所作的一切。感谢武汉理工大学菲旺软件研究中心的所有同学，在这个温暖的大家庭里，各位同学不厌其烦地帮我解决学习中所遇到的困难，他们勤勉刻苦、谦虚好学的精神值得我永远学习。感谢朝夕相处的好友，在整个研究生生活中，我们互相帮助、共同进步，建立了深厚的友谊。和他们一起度过的快乐时光和彼此的真挚友情，都将是我人生旅途中的珍贵记忆。

感谢我的家人，是父母让我懂得要脚踏实地地做事，诚诚恳恳做人，他们的支持和理解是我永远的力量之源，他们这么多年来无比艰辛的付出和默默的关怀成就了今天的我。

最后，衷心感谢各位评委百忙之中审阅、指导鄙文，由于本人水平有限，论文中所出现的缺点和错误在所难免，敬请各位评委老师批评和指正。