



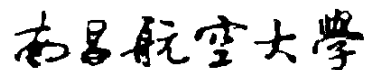
南昌航空大学

毕业设计（论文）

题 目： 电信 UNIX 服务器出租业务管理系统的设计和实现

学 院：	信息工程
专业名称：	计算机科学与技术
班级学号：	10045117
学生姓名：	赖应
指导教师：	杨国为

二〇一四 年 六月



毕业设计（论文）任务书

I、毕业设计(论文)题目：电信 UNIX 服务器出租业务管理系统的设计和实现

II、毕业设计(论文)使用的原始资料(数据)及设计技术要求：

1. 采用基于 Java 语言的面向对象编程技术，熟练掌握和应用面向对象方法学。

2. 采用 java 的 struts 框架、Hibernate 框架、Spring 框架整合技术。

3. 完成软件开发及运行环境的搭建和配置，包括：

(1) JDK：版本 1.7。

(2) 集成开发环境 IDE：MyEclipse10.6 版本。

(3) Java EE 服务器：采用开源版，如 Tomcat 等，完成相关配置。

(4) 数据库系统：ORACLE。

4. 按照软件工程方法学，完成软件系统的分析和设计，包括：系统流程图，数据字典，ER 图，软件结构图，程序流程图等。

5. 根据数据库原理，完成数据库设计。

6. 翻译 1 篇与本专业相关的英文技术文献。

III、毕业设计(论文)工作内容及完成时间：

应用 struts 框架、Hibernate 框架、Spring 框架整合技术，开发一个 Web 应用：用户登录，页面设计，主要的业务逻辑设计和实现，数据库的设计，和后台操作，报表的处理。

第 1 周. 第 3 周 开题（查阅相关资料、学习相关技术）

第 4 周. 第 5 周 需求分析

第 6 周. 第 7 周 概要设计

第 8 周. 第 10 周 详细设计

第 11 周. 第 14 周 系统实现（含调试）

第 15 周. 第 16 周 系统测试、撰写论文

第 17 周. 第 18 周 修改论文、论文答辩

IV、主要参考资料:

-
- [1]. Simache、Kaaniche. Availability Assessment of SunOS/Solaris UNIX Systems based on Syslogd and wtmpx log files: a case study. 11th Pacific Rim International Symposium on Dependable Computing. 2005
-
- [2]. 华文华. 国内电信计费系统现状分析. 人民邮电. 2002
-
- [3]. 李连祥、刘晓亮. 电信计费的内涵与外延. 中国计费网. 2004
-
- [4]. 张海藩. 软件工程导论(第五版). 清华大学出版社. 2008
-
- [5]. 布鲁斯埃克尔. Thinking in java(第四版). 机械工业出版社. 2007
-

信息工程 学院 计算机科学与技术 专业类 100451 班

学生(签名): _____

日期: 自 2014 年 2 月 17 日至 2014 年 6 月 20 日

指导教师(签名):

助理指导教师(并指出所负责的部分):

_____系(室)主任(签名): _____

南昌航空大學

学士学位论文原创性声明

本人声明,所呈交的论文是本人在导师的指导下独立完成的研究成果。除了文中特别加以标注引用的内容外,本论文不包含法律意义上已属于他人的任何形式的研究成果,也不包含本人已用于其他学位申请的论文或成果。对本文的研究做出重要贡献的个人和集体,均已在文中以明确方式表明。本人完全意识到本声明的法律后果由本人承担。

作者签名:

日期:

学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定,同意学校保留并向国家有关部门或机构送交论文的复印件和电子版,允许论文被查阅和借阅。本人授权南昌航空大学可以将本论文的全部内容编入有关数据库进行检索,可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

作者签名:

日期:

导师签名:

日期:

摘要

由于服务器出租业务，使租用单位和个人只需要花很少经费就可获得高配置的 Unix 服务器的高端服务，因此服务器出租业务越来越受欢迎。很显然，服务器出租业务的竞争将变得越来越激烈，因此在 web 上建立一个安全、快捷、有效地电信 Unix 服务器出租业务的管理系统是服务器出租业务企业竞争制胜的法宝。也是企业把业务管理提升为业务智能管理的必由之路。

本文设计开发了一个电信 Unix 服务器出租业务的管理系统。本系统操作的角色：后台管理员。主要功能：用户在 UNIX 服务器上注册账号，然后进行登入使用，该系统会记录用户的登入登出记录，然后根据用户所使用的套餐标准进行计算费用，该系统也可以新建、开通、关闭资费和新建、开通、关闭账户来管理。

本系统采用 STRUTS 2 HIBERNATE SPRING 三大框架整合的技术，基于 MVC 设计模式，STRUTS 2 负责总体架构，HIBERNATE 负责数据持久化，SPRING 负责解耦，优化系统。结合 JSP 页面显示技术，使系统操作更加简便，平台更加通用化，加上 AJAX, JQUERY 等技术，赋予交互更加灵活。

关键词：Struts 框架 Hibernate 框架 Spring 框架 MVC 模式 JSP Unix JAVA

Abstract

Because of the fact that the server rental business enables enterprises and individuals to merely pay a little for the high-end service of high-collocation Unix server, the server rental business is becoming more and more popular. It is obvious that the server rental business competition will become fiercer and fiercer. Therefore, establishing a safe, efficient and effective rental business management system of telecommunication Unix server on the web is not only talisman to success for enterprises operating the server rental business, but also the inevitable way for enterprises to upgrade business management to business intelligence management.

In this paper, a management system of Unix server rental business was designed and developed. Its operation role was a background administrator and its main functions were as follows. Users registered for an account through the UNIX server, then logged in with the system recording users' login and logout, and then calculated the expense based on the packages used by users. Besides, the system can also create, open, close tariffs as well as accounts.

The system used the technology integrating the three framework STRUTS 2, HIBERNATE and SPRING. Based on the MVC design pattern, STRUTS 2 is responsible for overall architecture, HIBERNATE data persistence and SPRING decouple and system optimization. Combined with JSP page display technology, the system was able to operate easier with more universal platform, coupled with AJAX, JQUERY and other technologies making interaction more flexible.

Key words: Struts framework; Hibernate framework; Spring framework; MVC mode; JSP; Unix; JAVA

目录

1 引言	9
1.1 项目背景	9
1.2 国内研究现状	9
1.3 国外研究现状	10
1.4 核心技术介绍	10
1.4.1 SSH 整合技术	10
1.4.2 JQuery 简介	12
1.4.3 AJAX 简介	13
1.4.4 Oracle 简介	13
1.5 核心技术	13
1.5.1 JAVA 简介	13
2 可行性及需求分析	15
2.1 可行性分析	15
2.1.1 技术可行性	15
2.1.2 经济可行性	16
2.1.3 操作可行性	16
2.2 需求分析	17
2.2.1 系统开发的目标	17
2.2.2 功能需求	17
2.2.3 界面需求	18
3 系统设计	20
3.1 概要设计	20
3.1.1 系统功能模块图	20
3.1.2 功能流程图	21
3.2 数据库设计	23
3.2.1 数据库逻辑设计	23
3.2.2 数据库物理设计	23
4 系统实现	29
4.1 项目配置	29
4.1.1 使用到的包	29
4.1.2 SSH 的配置	31
4.2 实体设计	37
4.3 出现的问题及解决方案	39
4.4 功能实现	39
4.4.1 登录功能	39
4.4.2 角色管理	41

4.4.3	管理员	42
4.4.4	资费管理	44
4.4.5	业务账号	45
4.4.6	账单管理	46
4.4.7	报表功能	47
4.4.8	基础功能	49
5	测试.....	51
5.1	测试原则及测试方法	51
5.1.1	测试的任务	51
5.1.2	测试方案	51
5.2	软件测试用例	52
5.2.1	登录测试	52
6	结论.....	53
	参考文献.....	54
	致谢.....	55

1 引言

1.1 项目背景

在电信业务中，有一项业务--Unix 实验室出租。如今为了节约成本和提高效率，传统的 Unix 操作系统会配上固定搭配好的硬件平台一起出售，其价格昂贵，一般人不会去购买的，只有大型企业才会去购买，但是作为 Unix 商用平台，很多的功能能使用得上，因此电信运营商就开了 Unix 服务器出租业务。用户只需在电信运营商那里注册 OS 账号（机器的账号），只需要缴纳一定的费用，即可使用，方便，快捷。任何用户登录电信运营商提供的 Unix 实验室的 Unix 系统时，Unix 系统会记录该 OS 账号的登入和登出信息，这些信息都保存在 Unix 的系统日志文件^[1]中。因此运营商为了更好的管理，运营商想要一个功能齐全，界面友好的管理系统，以帮助拓展这项业务，获取最大利益。

1.2 国内研究现状

近二十年来，国民经济迅速发展，科技迅猛发展，计算机行业也是如此，从手工操作到现在的智能操作。电信竞争越来越激烈，想要在竞争中立于不败的地位，必须提供优质的服务，这是唯一的选择。因此，优质的计费系统是提供优质服务的重要保证，是提高优质服务的支撑，是通讯运营企业的核心系统之一。

目前国内电信运营商的计费系统主要面临的问题有以下几个方面：首先是可扩展性，随着用户数和业务量的增加，传统计费系统的软硬件环境已经无法适应大容量需求，同时随着市场竞争的日益加剧，业务需求变化会越来越快，这就要求电信运营商的计费软件不断更新，从而使开发建设能适应和满足业务发展需求。其次是灵活性，由于各个电信运营商的业务基本上遍及全国，而各省的计费系统分散建设，致使系统不适应新业务的快速变化，再加上计费系统如果不具有很好的灵活性，资费方面的优惠政策就不能顺利实施，从而影响企业的竞争力。第三是实时性，计费系统如果不能支持实时出账，可能会影响预付费业务的开展以及实时信用控制功能的实施，从而导致大量的恶意欠费^[2]。

1.3 国外研究现状

国外的电信计费系统虽然历史比较长,但和国内的计费系统在系统软件(数据库、中间件、开发工具等)、硬件(计算机硬件网络等)方面基本保持同步或领先,但对数据的理解、重视、利用程度上存在很大的差异,比如国外的数据库模型、模型设计扩展性强,对模型的理解统一,多业务计费数据的处理具有综合计费账务系统,数据准确与可用性高,对市场营销、经营分析的支持好^[3]。

1.4 核心技术介绍

1.4.1 SSH 整合技术

Java 语言中 SSH 整合是 Struts 框架、Hibernate 框架、Spring 框架的整合,用它们各自的优点按不同的功能实现一个完整的系统。这种开发模式,有效的处理视图、控制、模型的彻底分离。另外,对业务处理和持久层的分离,使得各个层面互不影响,如果修改任意一层对其他层不会产生影响。而且对数据库也有很好的兼容性,可以换数据库而不改变代码。这种整合大大减少了代码之间的耦合,使得开发和维护效率得到了很大的提升。

SSH 整合的基本流程是:在视图层中,通过显示的 JSP 页面接受(request)和处理(response)请求,然后根据 web 配置和 Struts 配置文件 Struts.xml 将接受到的请求根据 Action 名字找到对应的 Action 处理。在业务层中, Spring 的 IoC 容器负责管理 Action 提供的业务模型和该组件关联的 DAO 对业务进行处理,并且 Spring 的 AoP 容器提供事务处理。在持久层中, Hibernate 会根据 java 映射机制对建立好的模型(java 对象)进行持久化并管理 PO 和实现 DAO。

(1) STRUTS 简介

Struts 框架是 Apache 旗下的开源项目,成立 2001 年,该框架有两个版本: Struts1 是世界上第一个发布的 MVC 框架,得到了广泛的应用,但是 Struts 存在着很多缺陷,比如对页面展示技术单一;和 Servlet API 严重耦合,很难测试;该框架是入侵比较大。因此有了 Struts 的第二个版本——Struts2。Struts2 以 WebWork 为核心设计思想,吸收了 Struts1 的优良设计并得到了很好的发展。

该系统使用的是 Struts2,基于 MVC(模型-视图-控制)模式。模型层(Model)就是我们通过所说的对象,视图层(View)就是页面显示层,控制层(Control)

是负责控制处理业务的。该种设计模式很好的将三层分离，如论在哪一层作了修改都不会影响另外的层。该框架在 SSH 整合中是负责 MVC 分离。

其工作原理可以分为以下步骤：

- 1) 客户端初始化一个指向 Servlet 容器的请求；
- 2) 请求经过一系列的过滤器；
- 3) 接着 FilterDispatcher 被调用，FilterDispatcher 询问 ActionMapping 来决定是否调用某个 Action；
- 4) 如果 ActionMapping 决定需要调用某个 Action，则把请求交给 ActionProxy；
- 5) ActionProxy 通过 Configuration Manager 找到配置文件，找到需要的 Action；
- 6) ActionProxy 创建一个 ActionInvocation 实例；
- 7) 在调用 Action 前后会涉及到相关拦截器的调用；
- 8) 一旦 Action 执行完毕，ActionInvocation 负责根据 struts.xml 中的配置找到对应的返回结果。

原理图如图 1.1 所示：

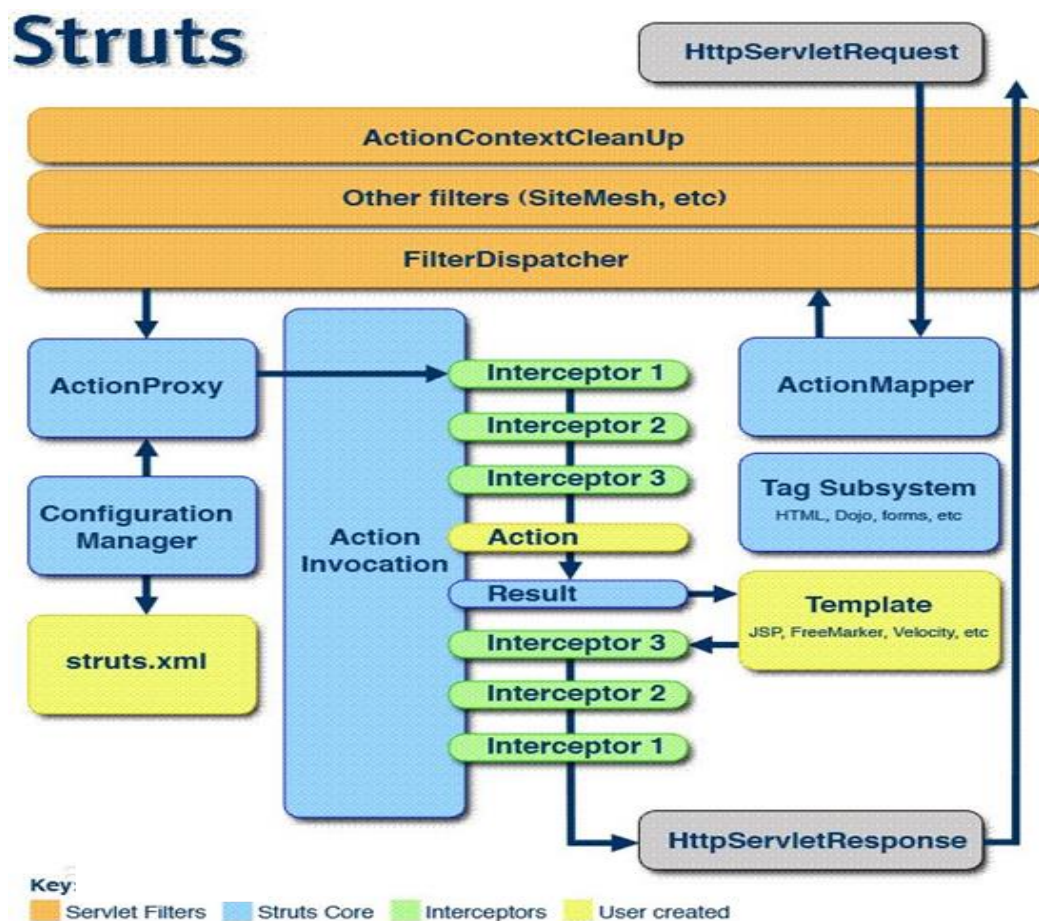


图 1.1 Struts 原理图

（2）Hibernate 简介

Hibernate 框架也是一个免费的开源框架，诞生于 2003 年，它是一个轻量级的 O/RM 框架，对 JDBC 进行了轻量级的封装，且支持很多主流的数据库。Hibernate 框架将数据的关系型很好的转化为人们容易看懂的 Java 对象，这样可以让程序员以面向对象的思想对数据库进行操作，而且 Hibernate 在应用 EJB（Enterprise Java Bean）和 J2EE 架构中取代 CMP。

使用 Hibernate 优点有如下：第一简化了成员对 SQL 的编写。第二把关系型转化成了对象型，可以用面向对象的思想进行编程。第三框架可移植性好，程序员可以方便的换数据库产品。第四实现了对象透明持久化。缺点有：不方便批量修改和不支持一些复杂的数据库操作。

Hibernate 在 SSH 的作用是对持久层提供支持，实现 DAO。

（3）Spring 简介

Spring 是一个免费的开源的轻量级的控制反转 IoC（Inversion of Control）和面向切面 AOP（Aspect-Oriented Programming）的容器框架，诞生于 2003 年，最初的版本作者是 Rod Johnson。在 2004 年 3 月发布了 1.0 版本。Spring 没有要求对程序员对哪个模块进行修改，有很好的解耦性能。因为相对于 EJB 来讲，EJB 使用 JNDI 主动查找需要的对象，而 Spring 是依赖对象注入给相应的类，是被动的，所以称为控制反转。

面向切面编程（AOP）是一种编程模式，就像我们熟知的面向过程编程和面向对象编程一样，但是面向切面编程有一个很大的好处是可以很好的控制事务或者拦截等操作，它是以面或点为单位对程序进行控制。

Spring 框架在 SSH 的作用是解耦和根据需要注入 DAO 和管理 Bean。

1.4.2 JQuery 简介

JQuery 是一个优秀的 JavaScript 框架，由美国的 John Resig 于 2006 年创建。JQuery 本质是 JavaScript 库，兼容 CSS3，并且对各种浏览器有良好的支持。JQuery 对 HTML、动画效果、HTML 事件、页面交互做了一个很好的处理。

它可以根据页面里的 ID 进行查找然后对其进行操作，很方便的可以实现局部页面刷新的操作。JQuery 一个很大的优势是可以在 JSP 页面里让 HTML 代码和 JS 彻底分离，只需要程序员定义 ID 即可。

1.4.3 AJAX 简介

AJAX 全名为 Asynchronous JavaScript and XML，是一个交互设计应用的网页开发技术，它包含 Java 技术、XML 和 JavaScript 技术。其原理是获得浏览器的 XMLHttpRequest 对象，然后使用 XMLHttpRequest 向服务器发出异步请求，在服务器处理请求。AJAX 可实现局部刷新，用户甚至不知道浏览器和服务器通信，页面的地址不会改变。

1.4.4 Oracle 简介

Oracle 是甲骨文公司开发的一款数据库产品，是目前最流行的客户/服务器 (CLIENT/SERVER) 或 B/S 体系结构的数据库之一。2007 年 7 月 12 日，Oracle 公司宣布推出 Oracle 11g，该版本在 Oracle 10g 的基础上增加了 400 多项特性，使 Oracle 数据库变的更加可靠、性能更好、更容易使用和更安全。现在的最新版本为 Oracle 12c。

Oracle 不仅有管理数据库的功能，作为关系数据库，它还是一个完备的产品。它有先进的存储理念，并服务全球。ORACLE 性能好，稳定性好，数据有保障，而且有各个平台的产品，受到很多企业的好评。一般大型企业都使用该数据库。

1.5 核心技术

1.5.1 JAVA 简介

Java 是一门跨平台的计算机编程语言，也是当今主流的编程语言。

Java 语言的雏形是用 sun 公司的詹姆斯·高斯林等人于 1990 年代开发的，最初命名为 Oak，目标是应用在家用电器等小型系统的机器上。起初命名为 Oak，1995 年正式命名为 Java。后来被甲骨文公司并购，成为了甲骨文公司的产品。因为 Java 是一个开放性的语言，吸引了大量的程序员，因此成了现在的主流语言。

Java 语言和 C++ 风格接近。继承了面向对象的思想，抛弃了难学指针改为引用，同时也弃用了运算符重载和多重继承改为接口取代，增加了方便使用的垃圾回收机制。在 Java SE 1.5 版本中引入了泛型编程、类型安全的枚举、不定长参数和自动装/拆箱特性。现在最新版本是 2014 年 3 月 18 日 Oracle 公司发布的 Java SE 8。

Java 平台由虚拟机和 Java 应用编程接口构成。Java 将源代码编译成字节码文件，然后根据不同操作系统平台的 Java 虚拟机进行解释执行。这就是 Java “一次编译到处执行”的原理。Java 语言之所以流行还依赖于另外一个原因，sun 公司把 Java 作为一种开放技术，引来全球的程序员工程师为其开发，使得 Java 得以健壮的发展。

2 可行性及需求分析

可行性及需求分析是软件工程必备的一个步骤,用户根据需要向软件开发人员提供一个人向导。用户将软件运行的环境,需要的功能告诉开发人员,分析好后这样才可以进行程序设计这个步骤。如果不进行需求分析,开发人员没有一个明确的目标,这样所耗费的时间、精力、财力都是不值得的,很有可能这样设计出来的功能偏离了用户所需要的功能。所以,需求分析具有决策性、方向性,需求分析是必不可少的。

2.1 可行性分析

2.1.1 技术可行性

因为根据需求,该系统需要在不同平台不同地区运行电信 Unix 服务器出租管理系统,而且要有一个友好的界面操作和强大的功能,因此利用 Java 技术、Struts 2 框架、Hibernate 框架、Spring 框架、JSP 技术、AJAX 技术、JQuery 技术等。开发环境及开发工具如下所示:

Java 环境: jdk1.7.0_55。

Web 服务器: apache-tomcat-7.0.52 版本。

开发语言: java 语, Jsp 语言。

开发工具: MyEclipse Enterprise Workbench10.0.6 版本, 记事本。

数据库: ORACLR 11g。

所以该系统在技术上是可实现的。

该系统流程图 2.1, 该系统自上而下进行开发。该系统分为 8 个模块, 其中个人信息, 修改密码, 角色管理, 管理员管理是基本模块。资费、账务账号、业务账号、账单管理, 报表是电信 Unix 服务器出租管理系统的重点模块。

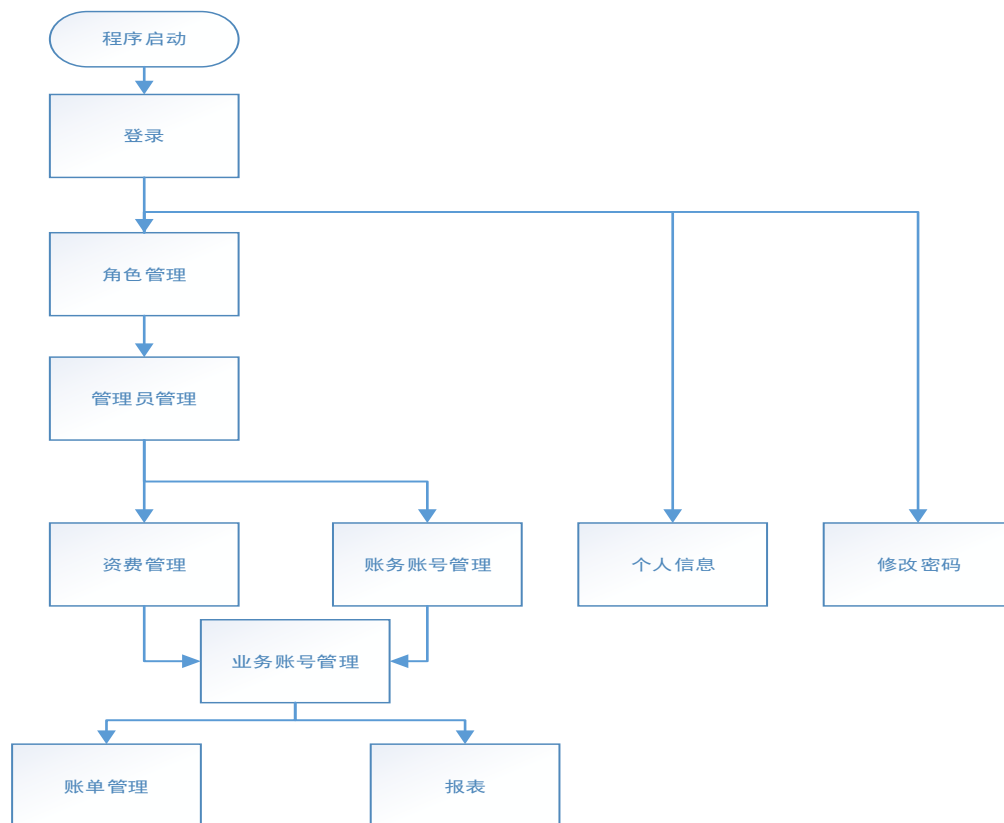


图 2.1 系统功能结构图

2.1.2 经济可行性

经济可行性是在评估待开发的软件生产的成本及效益，确保开发出来的软件最大利益化。开发成本包括开发人员的薪水、需要的硬件和公司的效益。一般来说，新开发的系统会比旧系统重用的成本要高很多。

电信 Unix 服务器出租管理系统开发所使用的技术、框架、软件、开发工具都是免费的，只需要投入少许的费用即可完成开发。系统投入使用后，维护简单方便，减少支出。

2.1.3 操作可行性

该系统具有友好的操作界面，功能一目了然。只需要把程序部署在服务器上即可通过专用网络或者互联网进行访问。只需要输入网址，登录即可使用，不需要对操作人员进行专业培训。因此，对于该系统是可操作的。

2.2 需求分析

2.2.1 系统开发的目标

电信 Unix 服务出租业务要具有实时和高效的运作，可以随时随地的登录账号管理，并且要有安全的权限机制，防止他人恶意破坏数据。该系统要有方便快捷的制定各种资费套餐，以应对特殊时刻特殊的套餐的办理，同时也要有管理 OS 账号的功能，即可对其进行开通、暂停、删除功能，也要有人工增加 OS 账号功能，还需要有方便管理和审查的账单、报表。

2.2.2 功能需求

该系统使用对象是管理员。

有以下功能：

1) 基础信息

登录：管理员使用用户名和密码登录系统。

主页：显示主页，并根据权限显示操作的功能模块。

修改密码：管理员修改密码。

查看个人信息：查看管理员的个人信息，如角色、姓名、电话、和创建时间等。

修改个人信息：修改管理员的个人信息中可修改的部分。如姓名和电话号码等。

2) 角色管理

一个系统因为有人员和级别的差异，就会导致角色拥有的权限有差异，这不仅能保证数据的安全，而且利于企业管理该平台，方便分配管理员的工作。

3) 管理员管理

相当于管理该系统的登入用户，该模块必须具备管理员查询、查看和修改，并且可以注册管理员，同时也要有重置密码功能，目的是为了了一些管理员忘记密码而不能登陆。

4) 资费管理

资费管理是该系统的重点和主要模块，资费套餐一旦开通是不能随意更改的。所以，如果一旦开通的资费就不能更改该资费，还要有按月租或者基本费用或者时长来排序以使用户选择更好的资费，该模块还要有查看详细资费的功能。

5) 账务账号管理

账务账号管理也是该系统的重点和主要模块，该模块必须以实名注册账务账号，而且还要能够灵活开通、修改、暂停、删除账务账号。因为有可能注册的人数多，所有要提供条件查询功能，可按一个条件也可按多个条件精确查询。同时也要有增加、查看账务账号功能，但是注册时应严格把控注册信息是否合法。

6) 业务账号管理

业务账号管理是该系统的主要模块，该模块和账务账号模块紧密关联，注册的用户必须要在账务账号模块存在的用户，否则不给予注册。该模块主要功能是用户在哪一个 Unix 服务器注册了账号，所使用的资费是什么，以方便结算和缴纳费用。该模块需要灵活管理，随时可以开通、暂停、删除、修改业务账号 也要根据条件（OS 账号、服务器 IP、身份证、状态）进行搜索。

7) 账单管理

账单管理是该系统的重点和主要模块，必须提供详细的账单，每个用户在不同的 Unix 服务器上产生的账单应该合并，以方便管理员统计费用，当然也需要提供一个详细的产生资费的详细条目。

8) 报表

报表需要有三种形式，一个是以客户时长产生报表，一个是以时长排行报表方便查看用户的总体情况，另外一个为资费使用率用来记录各个服务器资费注册的个数。

2.2.3 界面需求

界面是人机交互的媒介，一个好的界面在某种程度上决定了这个软件的用户量。许多软件在设计过程中很注重业务功能而忽略了界面设计，导致软件的使用价值降低。界面的设计原则是简易性、记忆负担最小化、安全性、人性化。

设计界面是按照图 2.2 作为主体框架进行设计和布局，有上往下分布：1 区域放 logo，2 区域为导航栏，3 区域为软件主体部分，4 区域为软件底部。主界面采用蓝色色调，字体为简体中文，颜色为黑色，大小为 12 像素。采用分页，方便浏览，表格选中行要突出，方便查找，界面交互方式是鼠标和键盘交互。

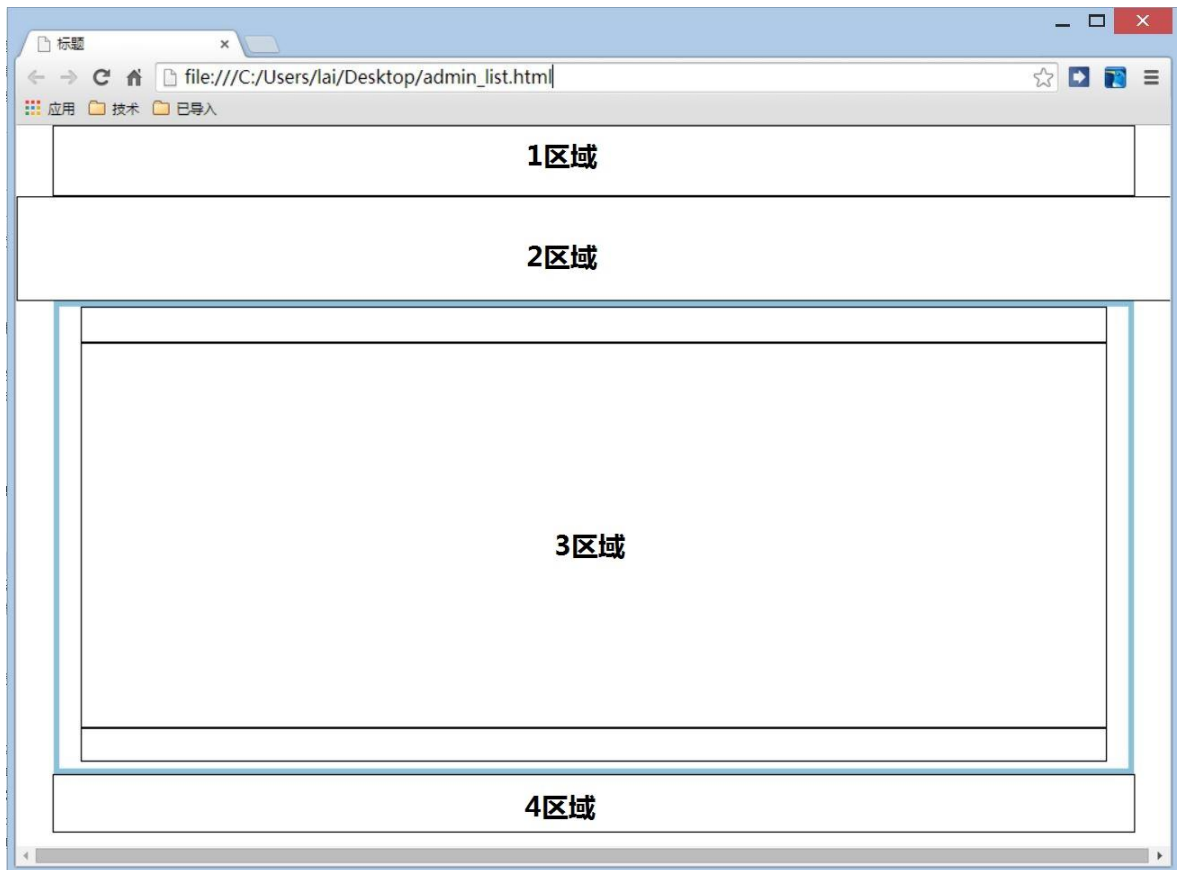


图 2.2 软件主体布局

3 系统设计

系统设计分为概要设计和详细设计。概要设计是把需求分解成模块，详细设计是把模块用代码实现。

3.1 概要设计

3.1.1 系统功能模块图

根据需求分析所提出的要求并分析做出了如下系统功能图，其中 NetCTOSS 是电信 Unix 服务器出租管理系统的简称。如图 3.1。

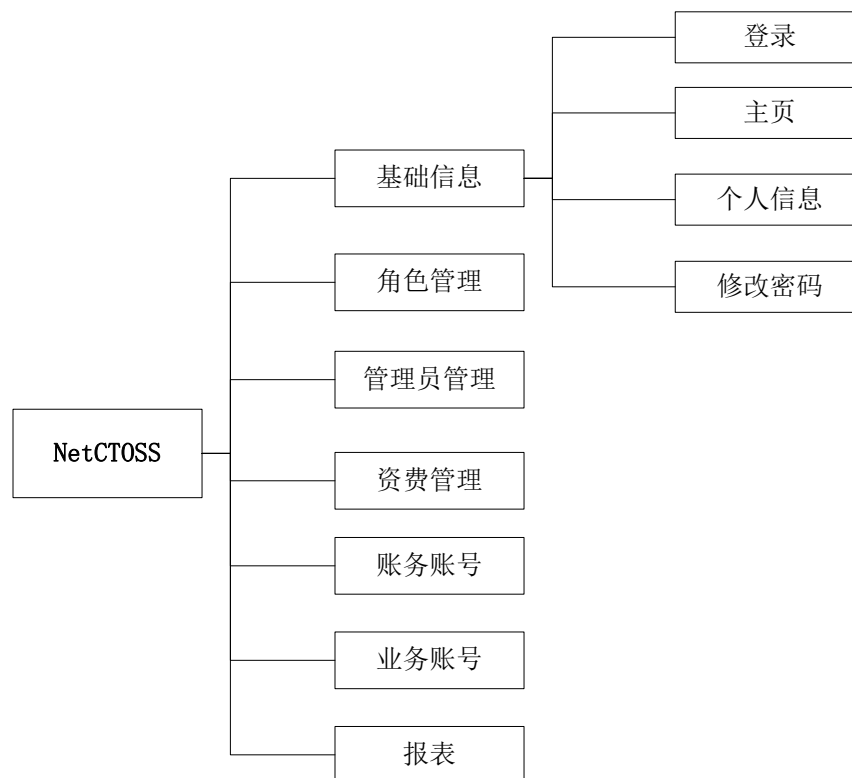


图 3.1 系统功能图

该系统分为 7 个模块，每个模块的详细内容如图 3.2，其中启动、删除资费就像我们的手机套餐启动和删除是一个道理，账务账号称为自然人，业务账号是自然人所办理的业务。账单是需要显示管理员浏览和查询。

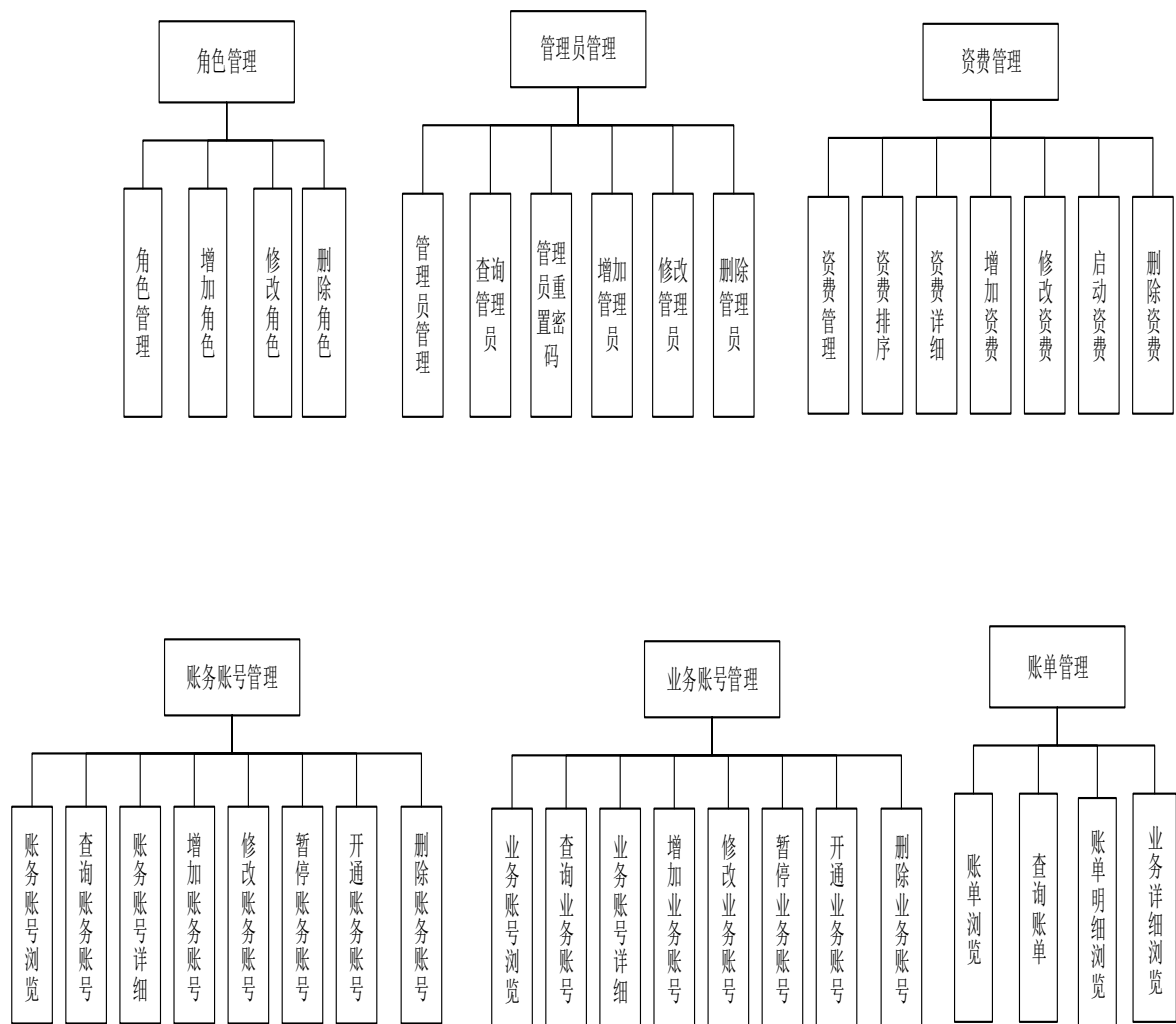


图 3.2 详细功能模块图

3.1.2 功能流程图

因为电信 Unix 服务器出租管理系统多处存在功能相似的地方，比如资费管理和账务账号、业务账号都存在增、删、改、查的功能，所以把它抽象出了四个基本流程。电信 Unix 服务器出租管理系统的安全机制是根据用户拥有的权限，然后在页面显示对应的模块。这样的机制可以方便的为以下三个流程减少了判断权限的操作，大大提高了程序的流畅性。

- 登入系统，首先要从页面中获取用户输入的用户名和密码及验证码，然后根据用户名从数据库中提取用户名和密码然后进行匹配，如果匹配成功，则根据角色 ID 获取对应的权限。流程图如图 3.3；

- 查询所有账务，查询所有账户是一个通用的功能。因为查询所有数据和搜索功能的代码复合了，所以查询所有账户时会判断搜索框有没有条件，如果有则进行条件查询。流程图如图 3.4；

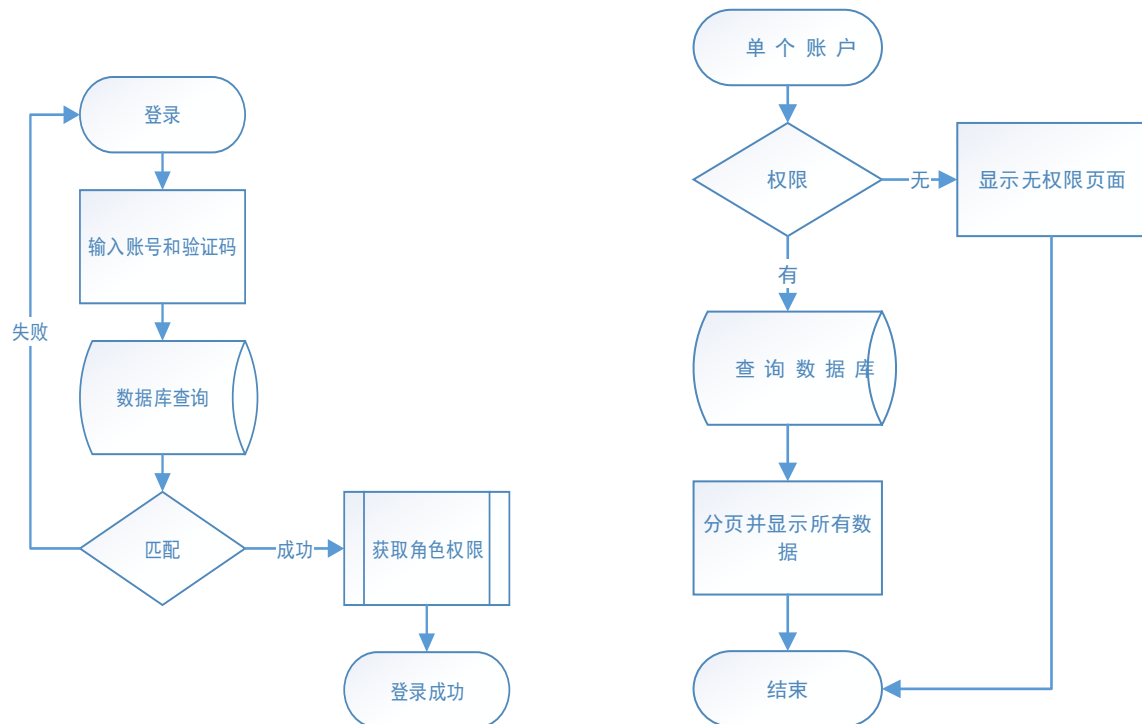


图 3.3 登录流程图

图 3.4 查询所有账号流程图

- 插入数据，一个通用的功能，插入数据之前会在页面进行判断输入的信息是否合法，在 java 代码里面也会进行判断，从而使插入数据库的数据能正常执行。这个流程图如图 3.5；
- 删除数据，根据选中的 id 进行删除。流程图如图 3.6；

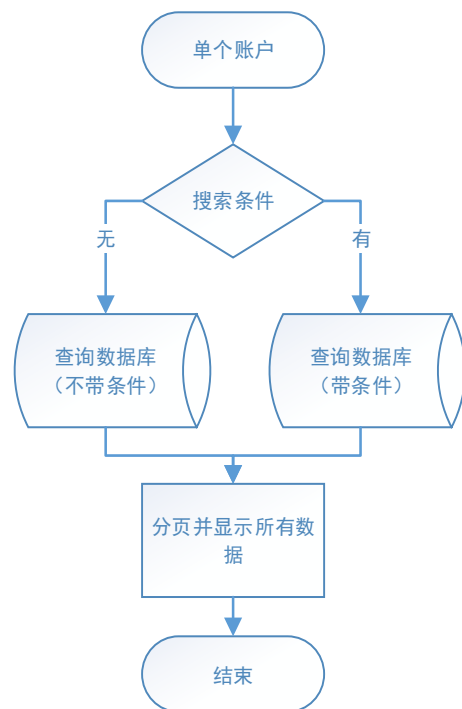


图 3.5 插入数据流程图

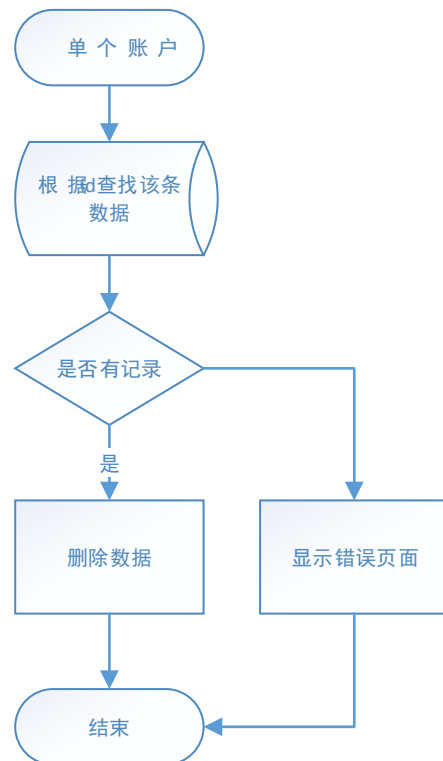


图 3.6 删除数据流程图

3.2 数据库设计

3.2.1 数据库逻辑设计

一个好的系统，设计数据库是关键，此系统与数据库紧密结合，数据库的逻辑设计决定了数据及其应用的整体性能，如果数据库逻辑设计的不好，则所有调优的办法提升性能不是很明显。

该系统的数据库遵循了数据库的三个范式，减少了数据库冗余，大大提高了数据库性能。

3.2.2 数据库物理设计

电信 Unix 服务器出租管理系统数据库设计的表如下：

ACCOUNT 账务信息表:

账务信息表主要用来记录账务账号的信息，用户在服务器注册账号的时候，会有一个账务账号，包括推荐人账号 ID，注册时间，登录系统账号，和一些个人信息。从账务角度看客户，即自然人，称为账务账号，如表 3.1。

表 3.1 账务信息表

字段名称	类型	备注	字段描述
ID	NUMBER(9)	PRIMARY KEY NOT NULL	账务账号 ID
RECOMMENDER_ID	NUMBER(9)	FOREIGN KEY	推荐人账务账号 ID
LOGIN_NAME	VARCHAR2(30)	UNIQUE NOT NULL	登录 NetCTOSS 系统的名称
STATUS	CHAR(1)	NOTNULLCHECK(0,1,2)	
CREATE_DATE	DATE	DEFAULT SYSDATE	创建日期
PAUSE_DATE	DATE	NULL	暂停日期（开通状态为空）
CLOSE_DATE	DATE	NULL	删除日期
REAL_NAME	VARCHAR2(20)	NOT NULL	客户姓名
IDCARD_NO	CHAR(18)	NOT NULL UNIQUE	身份证号码
BIRTHDATE	DATE	NULL	出生日期
GENDER	CHAR(1)	NOT NULL CHECK(0,1)	性别 0: 男 1: 女
OCCUPATION	VARCHAR2(50)	NULL	职业
TELEPHONE	VARCHAR2(15)	NOT NULL	联系电话
EMAIL	VARCHAR2(50)	NULL	电子邮箱
MAILADDRESS	VARCHAR2(50)	NULL	邮箱地址
ZIPCODE	CHAR(6)	NULL	邮编
QQ	VARCHAR2(15)	NULL	QQ
LAST_LOGIN_TIME	DATE	NULL	最后一次登录时间

SERVICE 业务账号表

业务账号表是存储业务信息的，用户在 Unix 服务器登录的时候需要一个业务账号，其中包括账务账号 ID（和业务账号关联）注册的 Unix 服务器 IP 地址、OS 账号、口令、开通状态、创建信息、资费编码（用来区分资费是否为包月或者是计费）。客户必须注册了账务账号之后才有权限注册业务账号。一个业务账号对应一个资费和—个 Unix 的 ip 地址和 OS 账号。如表 3.2。

表 3.2 业务账号表

字段名称	类型	备注	字段描述
ID	NUMBER(10)	PRIMARY KEY NOT NULL	业务账号 ID
ACCOUNT_ID	NUMBER(9)	FOREIGN KEY NOT NULL	账务账号 ID
UNIX_HOST	VARCHAR2(15)	和 OS_USERNAME 做联合 唯一键,NOT NULL	UNIX 服务器 IP 地址
OS_USERNAME	VARCHAR2(8)	和 UNIX_HOST 做联合唯一 键,NOT NULL	UNIX 服务器的 OS 账号
LOGIN_PASSWD	VARCHAR2(8)	NOT NULL	登录 UNIX 服务器的口令
STATUS	CHAR(1)	NOT NULL CHECK(0,1,2)	0:开通 1:暂停 2 删除
CREATE_DATE	DATE	DEFAULT SYSDATE	创建日期
PAUSE_DATE	DATE	NULL	暂停日期
CLOSE_DATE	DATE	NULL	删除日期
COST_ID	NUMBER(4)	FOREIGN KEY NOT NULL	资费编码

COST 资费表

资费表:针对 TELNET 服务的各种资费标准,包括资费名称、基本时长、月固定资费、开通状态、说明、创建信息。如表 3.3。

表 3.3 资费表

字段名称	类型	备注	字段描述
ID	NUMBER(4)	PRIMARY KEY NOT NULL	主键, 资费 ID
NAME	VARCHAR2(50)	NOT NULL	资费名称
BASE_DURATION	NUMBER(11)	NULL	包在线时长
BASE_COST	NUMBER(7,2)	NULL	月固定费, 可能有小数
UNIX_COST	NUMBER(7,4)	NULL	单位费用
STATUS	CHAR(1)	NOT NULL CHECK(0,1)	0: 开通 1: 暂停
DESCR	VARCHAR2(100)	NULL	对资费信息的说明
CREATEIME	DATE	DEFAULT SYSDATE	创建日期
STARTIME	DATE	NULL	启动日期

MONTH_DURATION 时长信息表

用来储存每个账号的月累计时长.如表 3.4。

表 3.4 时长信息表

字段名称	类型	备注	字段描述
SERVICE_ID	NUMBER		业务账号 ID
MONTH_ID	CHAR(6)		月份
SERVICE_DETAIL_ID	NUMBER(11)		业务详单 ID
SOFAR_DURATION	NUMBER(11)	NULL	基本时长

BILL 账单表

账单表示用来产生账单用的，客户（账务账号）每月的总费用信息。如表 3.5。

表 3.5 账单表

字段名称	类型	备注	字段描述
ID	NUMBER(11)	PRIMARY KEY NOT NULL	主键，账单 ID
ACCOUNT_ID	NUMBER(9)	FOREIGN KEY NOT NULL	账务账号 Id
BILL_MONTH	CHAR(6)	NOT NULL	账单月份,如 201301
COST	NUMBER(13,2)	NOT NULL	费用
PAYMENT_MODE	CHAR(1)	NULL CHECK(0,1,2,3)	0:现金 1:银行转账 2:邮局 汇款 3: 其他
PAY_STATE	CHAR(1)	DEFAULT 0 CHECK(0,1)	支付状态 0:未支付 1:已 支付，默认为 0
说明：支付方式和支付状态为用户预留功能。			

BILL_TIME 账单条目表

账单条目和账单表相关联，产生账单用。客户（账务账号）每月的详细费用信息，使用每个服务器对应的费用。如表 3.6。

表 3.6 账单条目表

字段名称	类型	备注	字段描述
TIME_ID	NUMBER(11)	PRIMARY KEY	主键，账单条目 ID
BILL_ID	NUMBER(11)	FOREIGN KEY NOT NULL	账单 ID
SERVICE_ID	NUMBER(10)	FOREIGN KEY NOT NULL	业务账号 ID
COST	NUMBER(13,2)	NULL	费用

ROLE_INFO 角色信息表

角色信息表用来储存角色名称。如表 3.7。

表 3.7 角色信息表

字段名称	类型	备注	字段描述
ID	NUMBER(11)	PRIMARY KEY NOT NULL	主键，角色名称
NAME	VARCHAR2(20)	NOT NULL	角色名称

ROLE_PRIVILEGE 角色权限表

角色权限表用来存储角色有哪些权限，权限用数字表示，可以方便标注角色有哪些权限。如表 3.8。

表 3.8 角色权限表

字段名称	类型	备注	字段描述
ROLE_ID	NUMBER(4)	PRIMARY KEY NOT NULL	角色 ID
PRIVILEGE_ID	NUMBER(4)	PRIMARY KEY NOT NULL	权限 ID

SERVICE_DETAIL 业务详单表

业务详单表，记录用户在 Unix 服务器上的登入/登出、ip 地址、进程号、OS 账号、费用等信息。如表 3.9。

表 3.9 业务详单表

字段名称	类型	备注	字段描述
ID	NUMBER(11)	PRIMARY KEY NOT NULL	主键，业务详单 ID
SERVICE_ID	NUMBER(10)	FOREIGN KEY NOT NULL	业务账号 ID
CLIENT_HOST	VARCHAR2(15)	NULL	OS 账号从该 IP 地址登录 Unix 服务器
OS_USERNAME	VARCHAR(8)	NULL	Unix 服务器上的 OS 账号
PID	NUMBER(11)	NULL	进程号
LOGIN_TIME	DATE	NULL	开始登录时间
LOGOUT_TIME	DATE	NULL	退出登录时间
DURATION	NUMBER(10,9)	NULL	时长
COST	NUMBER(20,6)	NULL	费用

E-R 图

E-R 是表示数据实体与实体之间的关系。其中 Account（账务信息表）它与自己组成一对多的关系，还与 Service（业务信息表）组成一对多的关系。Cost（资费表）与 Service 组成一对多的关系。Admin_Info（管理员表）与管理权限表组成一对多的关系。如图 3.7 所示：

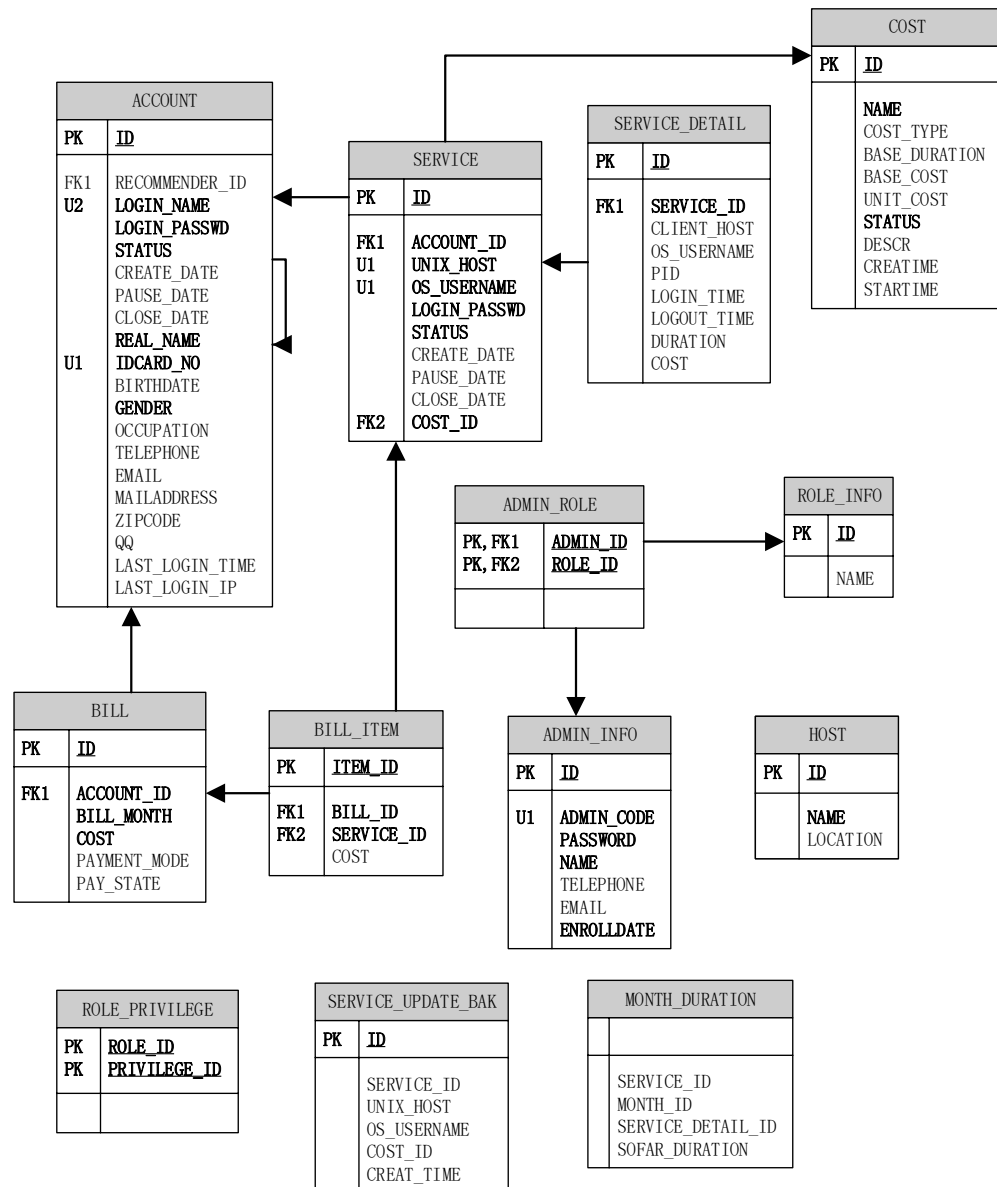


图 3.7 E-R 图

4 系统实现

电信 Unix 服务器出租管理系统采用分层设计，首先将设计出来的数据库模型化，在对象化实体化。然后写好各个功能接口，按接口写出具体功能实现，这就是编写 DAO 接口和实现。软件通过三个框架（Struts、Hibernate、Spring）通过配置将其组合为一个系统。

4.1 项目配置

4.1.1 使用到的包

NetCTOSS 项目中使用到的包和包的作用如下：



































- ▷  antlr.jar
- ▷  aopalliance-1.0.jar
- ▷  asm-attrs.jar
- ▷  asm.jar
- ▷  aspectjrt-1.1.1.jar
- ▷  aspectjweaver-1.5.2.jar
- ▷  c3p0-0.9.0.jar
- ▷  cglib-2.1.3.jar
- ▷  cglib-nodep-2.1_3.jar
- ▷  commons-beanutils.jar
- ▷  commons-collections-3.1.jar
- ▷  commons-dbcp-1.2.2.jar
- ▷  commons-digester-1.7.jar
- ▷  commons-fileupload-1.2.1.jar
- ▷  commons-logging.jar
- ▷  commons-pool-1.2.jar
- ▷  dom4j-1.6.1.jar
- ▷  ehcache-1.2.3.jar
- ▷  freemarker-2.3.15.jar
- ▷  hibernate-tools-2.1.3.jar
- ▷  hibernate3.jar
- ▷  jaxen-1.1-beta-6.jar
- ▷  jstl-1.2.jar
- ▷  jta.jar
- ▷  log4j-1.2.15.jar
- ▷  ognl-2.7.3.jar
- ▷  ojdbc15.jar
- ▷  quartz-1.6.0.jar
- ▷  spring-webmvc.jar
- ▷  spring.jar
- ▷  struts2-core-2.1.8.jar
- ▷  struts2-json-plugin-2.1.8.jar
- ▷  xwork-core-2.1.6.jar
- ▷  struts2-spring-plugin-2.0.11.1.jar

表 4.1 包的作用表

包名	作用	说明
antlr.jar	Antlr 支持	
asm.jar	操作 java 字节码	
asm-attrs.jar	ASM 字节码库	如 果 使 用 “cglib” 则 必要
aspectjweaver-1.5.2.jar	用于在 Spring 2.0 中集成 AspectJ LTW 织入器	
c3p0-0.9.0.jar	配置使用数据库连接池。	必须使用的 jar 包。
cglib-2.1.3.jar	实现 PO 字节码的动态生成	必需的 jar 包
commons-beanutils.jar	包含了一些 bean 工具类类，	是必须使用的 jar 包。
commons-collections-3.1.jar	Apache Commons 包中的一个，包含了一些 Apache 开发的集合类，功能比 java.util.* 强大。	必须使用的 jar 包。
commons-dbcp-1.2.2.jar	DBCP 数据库连接池	
commons-digester-1.7.jar	将 XML 文档转化为 Java 对象	
commons-fileupload-1.2.1.jar	实现 Jsp 的上传文件功能	
commons-logging.jar	实现日志功能	必须使用的 jar 包。
dom4j-1.6.1.jar	解析 XML 配置文件和 XML 映射元文件	
ehcache-1.2.3.jar	缓存数据库查询出来的对象	
freemarker-2.3.15.jar	WebWork 的核心包	必须使用的 jar 包。
hibernate3.jar	hibernate3 的核心类库	必须使用的 jar 包。
hibernate-tools-2.1.3.jar	Hibernate 的工具包	
jstl-1.2.jar	jstl 的核心标签库。	
log4j-1.2.15.jar	日志包，可以更加详细的记录日志。	必须使用的 jar 包。
ognl-2.7.3.jar	一个表达式语言	
quartz-1.6.0.jar	一个定时器	
spring.jar	Spring 的核心包	必须使用的 jar 包。

spring-webmvc.jar	这个 jar 文件包含 Spring MVC 框架相关的所有类。	必须使用的 jar 包。
struts2-core-2.1.8.jar	Struts 的核心包	必须使用的 jar 包。
struts2-spring-plugin-2.0.11.1.jar	用于 Struts 和 Spring 整合的一个插件	必须使用的 jar 包。
struts2-json-plugin-2.1.8.jar	用于传送字节的。	
xwork-core-2.1.6.jar	WebWork 的核心包	必须使用的 jar 包。

4.1.2 SSH 的配置

1 Web.xml 配置

Web.xml 文件是用配置：Listener，filter，Servlet 等。<context-param>是 web 容器参数，调用 spring 时在容器中寻找资源名称。<param-name>是参数名，<param-value>是参数值。Spring 提供了一个 OpenSessionInViewFilter 过滤器，它的主要功能是使每个过程绑定一个 Hibernate Session 完成事务，也可以在 web 进行延迟加载操作。配置中 ContextLoaderListener 作用是启动 web 容器时，自动装配 ApplicationContextd 的配置信息。

如下是 web.xml 的配置

```

<context-param>
  <param-name>contextConfigLocation</param-name>
  <param-value>classpath:applicationContext-*.xml</param-value>
</context-param>
<listener>
  <listener-class>org.springframework.web.util.Log4jConfigListener</listener-cl
ass>
</listener>
<listener>
  <listener-class>
    org.springframework.web.context.ContextLoaderListener
  </listener-class>
</listener>
<!-- 解决因 session 关闭而导致的延迟加载例外的问题 -->
<filter>
  <filter-name>openSessionInViewFilter</filter-name>
  <filter-class>
    org.springframework.orm.hibernate3.support.OpenSessionInViewFilter
  </filter-class>
</filter>
<filter>
  <filter-name>struts2filter</filter-name>
  <filter-class>
    org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter
  </filter-class>
</filter>
<filter-mapping>
  <filter-name>openSessionInViewFilter</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>

<filter-mapping>
  <filter-name>struts2filter</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
</web-app>

```

2 struts.xml 配置

在 SSH 框架起控制作用，其核心是 Controller，即 ActionServlet。而 ActionServlet 的核心就是 struts.xml。负责跳转页面，管理 Action。

3 Hibernate 配置

负责持久化层，完成数据库的映射为 Java 对象，提供数据库的各种操作。该配置是和数据库中的表一一对应。<generator>标签作用是表示主键的自增长作用，

<key>标识该表的主键，<one-to-many>标识一对多的关系。下面是 account.hbm.xml 配置

```
<hibernate-mapping>
  <class name="org.laiying.netctoss.pojo.Account" table="ACCOUNT"
  schema="LAI">
    <id name="id" type="java.lang.Integer">
      <column name="ID" precision="9" scale="0" />
      <generator class="sequence">
        <param name="sequence">ACCOUNT_SEQ</param>
      </generator>
    </id>
    <property name="recommenderId" type="java.lang.Integer">
      <column name="RECOMMENDER_ID" precision="9" scale="0" />
    </property>
    <property name="loginName" type="java.lang.String">
      <column name="LOGIN_NAME" length="30" not-null="true"
unique="true" />
    </property>
    .....
    <property name="lastLoginIp" type="java.lang.String">
      <column name="LAST_LOGIN_IP" length="15" />
    </property>
    <set name="bills" inverse="true">
      <key> <column name="ACCOUNT_ID" precision="9" scale="0" not-null="true"
/>
        </key>
      <one-to-many class="org.laiying.netctoss.pojo.Bill" />
    </set>
    <set name="services" inverse="true">
      <key><column name="ACCOUNT_ID" precision="9" scale="0" not-null="true"
/>
        </key>
      <one-to-many class="org.laiying.netctoss.pojo.Service" />
    </set>
  </class>
</hibernate-mapping>
```

4 Spring 配置

负责业务层管理，提供接口的调用和 DAO 的封装。实现 action 和 DAO 的注入。首先，spring 会建立一个数据库连接池，供程序对数据库进行操作。然后要想找到数据库的映射就要找到 Hibernate 持久化的数据库，找到.hbm.xml 文件，当要对某个对象进行操作的时候 spring 会自动找到对应的持久化文件。SessionFactory 负责初始化 Hibernate，<aop:aspect-autoproxy> 标签自动完成创建代理织入切面，<context:component-scan> 标签的作用是扫描包。

下面是 applicationContext-component.xml 配置文件。

```
<!-- 声明一个数据库连接池 -->
<bean id="myDataSource" destroy-method="close"
      class="org.apache.commons.dbcp.BasicDataSource">
  <property name="driverClassName"
    value="oracle.jdbc.driver.OracleDriver"></property>
  <property name="url"
    value="jdbc:oracle:thin:@localhost:1521:orcl"></property>
  <property name="username" value="lai"></property>
  <property name="password" value="0821"></property>
  <property name="initialSize" value="5"></property>
  <property name="maxActive" value="15"></property>
  <property name="maxIdle" value="8"></property>
</bean>
<!-- 声明 sessionFactory -->
<bean id="sessionFactory"
      class="org.springframework.orm.hibernate3.LocalSessionFactoryBean">
  <property name="dataSource" ref="myDataSource"></property>
  <property name="hibernateProperties">
    <props>
      <prop key="hibernate.dialect">org.hibernate.dialect.OracleDialect</prop>
      <prop key="hibernate.show_sql">true</prop>
      <prop key="hibernate.format_sql">true</prop></props></property>
  <property name="mappingResources">
    <list>
      <value>org/laiying/netctoss/pojo/Cost.hbm.xml</value>
      .....
      <value>org/laiying/netctoss/pojo/RolePrivilege.hbm.xml</value>
    </list>
  </property>
</bean>
<context:component-scan
base-package="org.laiying.netctoss"></context:component-scan>
  <aop:aspectj-autoproxy proxy-target-class="true" ></aop:aspectj-autoproxy>
</bean>
```

该配置的作用是调用存储过程和定时执行。<bean>定义的是调用 CreateBillJob，产生的对象放入 sessionFactory 里。<property name="cronExpression">是设置何时触发存储过程，每月的 1 号凌晨一点调用存储过程进行结算。

下面是 applicationContext-quartz.xml 配置信息

```
<!-- 定义调用存储过程的 Bean -->
<bean id="createBill"
      class="org.laiying.netctoss.util.CreateBillJob">
  <property name="sessionFactory" ref="sessionFactory">
  </property>
</bean>
<!-- 将 createBill 定义成任务 -->
<bean id="createBillTask"
      class="org.springframework.scheduling.quartz.MethodInvokingJobDetailFactory
Bean">
  <property name="targetObject" ref="createBill">
  </property>
  <property name="targetMethod" value="create">
  </property>
</bean>
<!-- 将任务指定触发时间和频率 -->
<bean id="createBillTrigger"
      class="org.springframework.scheduling.quartz.CronTriggerBean">
  <property name="jobDetail">
    <ref bean="createBillTask" />
  </property>
  <property name="cronExpression">
    <!-- 每月 1 日凌晨 0 点启动 -->
    <value>0 0 0 1 * ?</value>
  </property>
</bean>
```

下面的配置是进行事务声明和控制，产生 SessionFactory，<tx:advice>作用是事务控制。文件名为 applicationContext-transaction.xml。

```
<!-- 声明事务管理 -->
<bean id="txManager"
class="org.springframework.orm.hibernate3.HibernateTransactionManager">
  <property name="sessionFactory" ref="sessionFactory"></property>
</bean>
<!-- 开启事务管理的注解扫描 -->
<tx:annotation-drivenproxy-target-class="true"
transaction-manager="txManager"/>
<!-- 事务控制 -->
<tx:advice id="txAdvice"
Transaction-manager="txManager">
  <tx:attributes>
    <tx:method name="*" propagation="REQUIRED"/>
  </tx:attributes>
</tx:advice>
  <beanid="execptionBean" class="org.laiying.netctoss.util.LoggerException">
    </bean>
    <aop:config proxy-target-class="true">
      <aop:pointcut expression="within(org.laiying.netctoss.action..*)"
id="pointcut"/>
      <aop:aspect id="exceptionLogger" ref="execptionBean" order="4" >
        <aop:after-throwing method="execute" pointcut-ref="pointcut"
throwing="ex" />
      </aop:aspect>
    </aop:config>
  </beans>
```

4.2 实体设计

根据需求分析抽象出来的模型, 实体设计必须依照设计好的数据的各个表一一对应, 才能够良好的对数据库进行操作。

下面是项目中的重要表的部分实体:

- 1) Account 与数据库的账务信息表对应。

```
public class Account implements java.io.Serializable {  
    private Integer id;//账务账号 ID  
    private Integer recommenderId;//推荐人账务账号 ID  
    private String loginName;//登入系统的名称  
    private String loginPasswd;//登入系统的密码  
    private String status;//状态  
    private Date createDate;//创建日期  
    private Date pauseDate;//暂停日期  
    private Date closeDate;//删除日期  
    private String realName;//客户姓名  
    private String idcardNo;//身份证号码  
    private Date birthdate;//出生日期  
    private String gender;//性别  
    private String occupation;//职业  
    private String telephone;//联系电话  
    private String email;//电子邮件  
    private String mailaddress;//邮箱地址  
    private String zipcode;//邮编  
    private String qq;//QQ  
    private Date lastLoginTime;//最后一次登入时间  
    private String lastLoginIp;//最后一次登入 IP  
    private Set<Bill> bills = new HashSet<Bill>(0);  
    private Set<Service> services = new HashSet<Service>(0);
```

- 2) Service 与数据库的业务账号表对应。

```
public class Service implements java.io.Serializable {  
  
    private Integer id;//业务账号 ID  
    private Cost cost;//资费编码  
    private Account account;//账务账号 ID  
    private String unixHost;//UNIX 服务器地址  
    private String osUsername;//UNIX 服务器的 os 账号  
    private String loginPasswd;//登入 UNIX 服务器的密码  
    private String status;//状态  
    private Date createDate;//创建日期  
    private Date pauseDate;//暂停日期  
    private Date closeDate;//删除日期  
    private Set<ServiceDetail> serviceDetails = new HashSet<ServiceDetail>(0);  
    private Set<BillItem> billItems = new HashSet<BillItem>(0);
```

- 3) Bill 与数据库的账单表对应。

```
public class Bill implements java.io.Serializable {  
    private Integer id;//账单 ID  
    private Account account;//帐外帐号 ID  
    private String billMonth;//账单月份  
    private Double cost;//费用  
    private String paymentMode;//支付类型  
    private String payState;//支付状态  
    private Set<BillItem> billItems = new HashSet<BillItem>(0);
```

4) BillItem 与数据库的账单条目表对应。

```
public class BillItem implements java.io.Serializable {  
    private Integer itemId;//账单条目 ID  
    private Service service;//账单 ID  
    private Bill bill;//业务账号 ID  
    private Double cost;//费用
```

5) Cost 与数据库的资费表对应。

```
public class Cost implements java.io.Serializable {  
    private Integer id;//资费 ID  
    private String name;//资费名称  
    private Integer costType;//资费类别  
    private Integer baseDuration;//包在线时长  
    private Double baseCost;//月固定费  
    private Double unitCost;//单位费用  
    private String status;//开通状态  
    private String descr;//对资费信息的说明  
    private Date createTime;//创建日期  
    private Date startTime;//启动日期
```

4.3 出现的问题及解决方案

错误提示	原因	解决办法
java.lang.UnsupportedClassVersionError: Bad version number in .class file	因为装的jdk和编译器版本不一致导致的错：（当时用的时myeclipse 6.6）	更换高版本的myeclipse
org.springframework.dao.InvalidDataAccessResourceUsageException: Could not execute JDBC batch update;	数据库和 Hibernate 没有完全映射到（表明，列名写错之类的）	更正 Hibernate 映射
Servlet.service() for servlet jsp threw exception: Page-encoding specified in XML prolog (UTF-8) is different from that specified in page directive (utf-8)	因为代码没有规范化，UTF-8 和 utf-8 有着明显的区别导致出错	注重代码的规范，尽量避免出错
Caused by: java.lang.ClassNotFoundException: org.apache.commons.dbcp.BasicDataSource	缺少 quartz-1.6.0.jar	加入 quartz-1.6.0.jar
The type javax.servlet.http.HttpServletRequest cannot be resolved. It is indirectly referenced from required .class files	没有导入 Java EE 5 包导致 application.	
unable to load configuration	包重复了导致配置无法加载	
org.springframework.beans.factory.BeanCreationException: Error creating bean with name 'sessionFactory'	缺少 struts2-spring-plugin-2.0.11.1.jar 导致的	导入 struts2-spring-plugin-2.0.11.1.jar

4.4 功能实现

4.4.1 登录功能

在页面用户输入的信息，在页面通过 JQuery 技术获得输入账号，密码和验证码，Struts 会根据页面请求在 Struts.xml 上找到对应的 LoginAction。同时 Hibernate 会持久化数据和 Spring 会根据名字找到 LoginAction 注入 AdminInfoDAO 和 RoleDAO。其中为了更好得显示界面，把输入的验证码做了一个标志 2，用户名和密码做了一个标志 1 如果验证码写错了则根据标志提示验证码错误，如果用户名或密码输错了则会提示用

用户名或密码错误。如果输入的信息完全无误，则会向数据库查找该用户的操作权限，以便显示相应权限的用户界面。代码如下所示：

```
public String execute(){
    if (adminCode == null){
        return "login"; }
    AdminInfo admin = null;
    String realCode = (String)session.get("code");
    if(!realCode.equalsIgnoreCase(code)){
        request.put("loginFlag", 2);
        return "login";}
    try {
        admin = adminInfoDAO.findByAdminCode(adminCode);
        if (admin == null){
            request.put("loginFlag", 1);
            return "login";
        }else{
            if(!admin.getPassword().equals(password)){
                request.put("loginFlag", 1);
                return "login"; } }
        Set<Integer> privilegeIds = new HashSet<Integer>();
        for (Role role:admin.getRoles()){
            role = roleDAO.findById(role.getId());
            privilegeIds.addAll(role.getPrivilegeIds()); }
        Set<String> urls = new HashSet<String>();
        for (Integer id:privilegeIds) {
            urls.addAll(PrivilegeReader.getModulUrlsById(id));}
        session.put("urls", urls);
        session.put("admin", admin);
    } catch (DAOException e) {
        e.printStackTrace();
        return "error"; }
    return "success"; }
```

查询用户信息 当 execute 方法执行到 findByAdminCode 方法时，根据 hql 语句向表 adminCode 查询出用户的详细信息的关键代码：

```
public AdminInfo findByAdminCode(String adminCode) throws DAOException {
    String hql = "from AdminInfo where adminCode=?";
    List list = getHibernateTemplate().find(hql, adminCode);
    if (!list.isEmpty()){
        return (AdminInfo) list.get(0); }
    return null;
}
```

界面效果图如图 4.1。



图 4.1 登录界面

4.4.2 角色管理

4.4.2.1 增加角色功能

用户点击角色管理模块的增加按钮时,请求会根据 struts.xml 指定 struts-role.xml,然后根据 Action 名字 role_add 找到对用的 roleAction,然后找到对应的请求 add,关键代码如下:

```
public void insert(Role role) throws DAOException {  
    getHibernateTemplate().save(role);  
    for (Integer privilegeId:role.getPrivilegeIds()){  
        RolePrivilege rp = new RolePrivilege(new RolePrivilegeId(role.getId(),  
privilegeId));  
        getHibernateTemplate().save(rp);  
    }  
}
```

界面截图如图 4.2。

图 4.2 增加角色界面

4.4.2.2 修改角色功能

点击修改按钮时，会把当前角色的 id 传给请求 Role_mode，根据 id 查询数据库并显示在修改页面上，用户只需填写要修改的内容，点击保存时，js 首先会检查输入的内容是否合法，然后更新数据库。关键代码如下：

```
public void update(final Role role) throws DAOException {
    getHibernateTemplate().execute(new HibernateCallback() {
        public Object doInHibernate(Session session) throws
        HibernateException, SQLException {
            Query = session.createQuery("delete from RolePrivilege where
            id.roleId=?");
            query.setInteger(0, role.getId());
            query.executeUpdate();
            Role r = (Role) session.get(Role.class, role.getId());
            r.setName(role.getName());
            session.update(r);
            for (Integer privilegeId:role.getPrivilegeIds()){
                RolePrivilege rp = new RolePrivilege(new
                RolePrivilegeId(r.getId(), privilegeId));
                session.save(rp);
            }
            return null;
        }
    });
}
```

界面截图如图 4.3。

图 4.3 修改角色界面

4.4.3 管理员

4.4.3.1 搜索管理员功能

搜索功能的原理是，当用户点击搜索按钮时，会根据用户输入的条件转变成执行 hql 语句时增加条件查询，此代码和列出所有管理员的代码进行了重合，只要在查询

全部管理员之前进行 if 条件判断，如果有条件就在数据库语句加上查询条件。这样的代码复用很好的解决了代码冗余问题，是代码变的更加简洁。

```
public String execute(){
    privileges = new ArrayList<Privilege>();
    Privilege p = new Privilege();
    p.setId(0);
    p.setModuleName("全部");
    privileges.add(p);
    privileges.addAll(PrivilegeReader.getModules());
    try {
        roleNames = new ArrayList<String>();
        shortRoleNames = new ArrayList<String>();
        if (roleName != null && !roleName.equals("")){
            admins = adminInfoDAO.findRole(roleName, page, pageSize);
            totalPages = adminInfoDAO.totalPagesRole(roleName, pageSize);
        } else if (privilegeId != 0) {
            List<Integer> roleIds =
            roleDAO.getRoleIdByPrivilegeId(privilegeId);
            admins = adminInfoDAO.findRole(roleIds, page, pageSize);
            totalPages = adminInfoDAO.totalPagesRole(roleIds, pageSize);
        } else {
            admins = adminInfoDAO.findPage(page, pageSize);
            totalPages = adminInfoDAO.totalPages(pageSize);
        }
        for(AdminInfo admin:admins){
            StringBuilder str = new StringBuilder();
            StringBuilder str1 = new StringBuilder();
            int i = 0;
            for (Role role:admin.getRoles()){
                if(i > 0){
                    str.append(",");
                } else if (i == 0){
                    str1.append(role.getName());
                    if (admin.getRoles().size() > 1){
                        str1.append("...");
                    }
                }
                str.append(role.getName());
                i++;
            }
            roleNames.add(str.toString());
            shortRoleNames.add(str1.toString());
        }
    } catch (DAOException e) {
        e.printStackTrace();
        return "success";
    }
    return "success";
}
```

界面截图如图 4.4 所示。



图 4.4 搜索界面

4.4.3.2 密码重置

如果管理员密码忘记了，一时难以记起，密码重置就是一个很好的工具。点击重置后密码和用户名一致。代码如下：

```
public void resetPwd(final String[] adminCodes) throws DAOException {
    if (adminCodes != null && adminCodes.length > 0){
        getHibernateTemplate().execute(new HibernateCallback() {
            public Object doInHibernate(Session session) throws
            HibernateException,
                SQLException {
                Query = session.createQuery("update ADMIN_INFO set
                PASSWORD=? where ADMIN_CODE=?");
                for (String adminCode:adminCodes){
                    query.setString(0, adminCode);
                    query.setString(1, adminCode);
                    query.executeUpdate();
                }
                return null;
            }
        });
    }
}
```

4.4.4 资费管理

4.4.4.1 排序功能

排序功能其实是数据库的排序，当选中某个按钮时就根据哪一个列排序，由大到小排序。关键代码如下：

```

@SuppressWarnings("unchecked")
public List<Cost> findPage(final Integer page, final Integer pageSize, final
Integer sortNo) throws DAOException {
    List<Cost> list = getHibernateTemplate().executeFind(new
HibernateCallback() {
        public Object doInHibernate(Session session) throws
HibernateException,
        SQLException {
            String hql = "from Cost";
            if (sortNo / 10 == 1){
                hql = hql + " order by baseCost ";
            } else if (sortNo / 10 == 2) {
                hql = hql + " order by unitCost ";
            } else if (sortNo / 10 == 3) {
                hql = hql + " order by baseDuration ";
            }
            if (sortNo % 10 == 1){
                hql = hql + "asc";
            } else if (sortNo % 10 == 2){
                hql = hql + "desc";
            }
            Query = session.createQuery(hql);
            Integer begin = (page - 1) * pageSize;
            query.setFirstResult(begin);
            query.setMaxResults(pageSize);
            List<Cost> list = query.list();
            return list;
        }
    });
    return list;
}

```

软件截图如图 4.5。



图 4.5 资费排序界面

4.4.5 业务账号

4.4.5.1 开通和暂停功能

开通或者暂停在数据库设计的时候用 0 和 1 表示，“0”是开通，“1”是暂停，当开通状态转为暂停状态时，改变数据库的 status 字段为“1”同时也改变 service 表的 status 字段为“1”，以便做到暂停的账务账号的同时也暂停业务账号的状态。关键代码如下：

```

public void setStatus(Integer id, String status) throws DAOException {
    Account a = (Account) getHibernateTemplate().get(Account.class, id);
    a.setStatus(status);
    if ("0".equals(status)) {
        a.setCloseDate(null);
        a.setPauseDate(null);
    } else if ("1".equals(status)) {
        a.setPauseDate(new Date(System.currentTimeMillis()));
    } else if ("2".equals(status)) {
        a.setCloseDate(new Date(System.currentTimeMillis()));
    }
    getHibernateTemplate().update(a);
}

```

```

public void setServiceStatusByAccountId(Integer accountId, String status)
    throws DAOException {
    Account a = (Account) getHibernateTemplate().get(Account.class,
accountId);
    for (Service s:a.getServices()) {
        s.setStatus(status);
        if ("0".equals(status)) {
            s.setCloseDate(null);
            s.setPauseDate(null);
        } else if ("1".equals(status)) {
            s.setPauseDate(new Date(System.currentTimeMillis()));
        } else if ("2".equals(status)) {
            s.setCloseDate(new Date(System.currentTimeMillis()));
        }
        getHibernateTemplate().save(s);
    }
}

```

界面截图如图 4.6 所示。

OS 账号:	<input type="text"/>	服务器 IP:	<input type="text"/>	身份证:	<input type="text"/>	状态:	全部	<input type="button" value="搜索"/>	<input type="button" value="增加"/>
业务ID	账务账号ID	身份证	姓名	OS 账号	状态	服务器 IP	资费		
2008	1010	330682196903190613	guojing	guojing	开通	192.168.0.20	包月	暂停	修改 删除
2001	1010	330682196903190613	guojing	guojing	开通	192.168.0.26	5.9元套餐	暂停	修改 删除
2004	1011	330902197108270429	huangrong	huangr	暂停	192.168.0.23	包月	开通	修改 删除
2003	1011	330902197108270429	huangrong	huangr	暂停	192.168.0.20	8.5元套餐	开通	修改 删除

图 4.6 账务账号列表界面

4.4.6 账单管理

4.4.6.1 生成账单功能

生成账单是用储存过程实现的，当数据挖掘系统从 Unix 服务器的日志文件读取出来存到数据库之前，会触发一个触发器来计算费用，如果用户所使用的是包月套餐则按包月套餐进行计费，不计时长，如果是按时收费套餐。则根据使用时长和基本费

用相乘进行计费，把 Unix 日志获取的数据存入数据库之后，在每个月的月末凌晨 0 点进行结算，系统通过 Spring 配置文件调用存储过程计算账单和账单条目。然后显示在页面上。界面截图如下：

下图是不同的人每月的账单。

账单ID	姓名	身份证	账务账号	费用	月份	支付方式	支付状态	
242	guojing	330682196903190613	xl18z60	25.9	2014年05月			明细
246	luwushuang	320211199307310346	lpj90	10.51	2014年05月			明细
243	huangrong	330902197108270429	dgbf70	34.4	2014年05月			明细
205	huangrong	330902197108270429	dgbf70	34.4	2014年05月			明细
204	guojing	330682196903190613	xl18z60	25.9	2014年05月			明细

图 4.7 账单界面

下图是单个人一个月产生的费用。

账务账号：xl18z60		身份证：330682196903190613	姓名：guojing	计费时间：2014年05月	总费用：25.9	返回	
账单明细ID	OS 账号	服务器 IP	账务账号ID	时长	费用	资费	
181	guojing	192.168.0.20	1010	4小时48分钟34秒	20.0	包月	详单
182	guojing	192.168.0.26	1010	0小时2分钟36秒	5.9	5.9元套餐	详单

图 4.8 账单条目界面

下图是一项资费的明细。

账务账号： xl18z60

OS 账号： guojing

服务器 IP： 192.168.0.20

计费时间： 2014年05月

费用： 20.0

返回

客户登录 IP	登入时刻	登出时刻	时长（秒）	费用	资费
192.168.1.34	2014-05-20 00:00:00.0	2014-05-20 00:00:00.0	6068	0.0	包月
192.168.25.10	2014-05-20 00:00:00.0	2014-05-20 00:00:00.0	11	0.0	包月
192.168.25.10	2014-05-20 00:00:00.0	2014-05-20 00:00:00.0	11	0.0	包月
192.168.25.10	2014-05-20 00:00:00.0	2014-05-20 00:00:00.0	6	0.0	包月
192.168.25.10	2014-05-20 00:00:00.0	2014-05-20 00:00:00.0	11	0.0	包月
192.168.25.10	2014-05-20 00:00:00.0	2014-05-20 00:00:00.0	7	0.0	包月
192.168.25.10	2014-05-26 00:00:00.0	2014-05-26 00:00:00.0	3	0.0	包月
192.168.25.10	2014-05-26 00:00:00.0	2014-05-26 00:00:00.0	3	0.0	包月
192.168.25.10	2014-05-20 00:00:00.0	2014-05-20 00:00:00.0	11	0.0	包月
192.168.25.10	2014-05-20 00:00:00.0	2014-05-20 00:00:00.0	9	0.0	包月

上一页

12

下一页

图 4.9 详细账单界面

4.4.7 报表功能

点击报表功能时，选择一种报表的时候页面会传一个 tag 标记（客户使用时长用 tag1 表示，时长排行用 tag2 表示，资费使用率用 tag3 表示）这样就可以明显区分报

表。客户使用时长是联合业务详单表（service_detail）、业务账号表（service）、账务信息表（accout）查询出账务账号、客户名称、身份证号码、电话月份、累计时长。时长排行是也是联合查询上述表并用组函数 max 和 sum 计算每台 Unix 服务器上的时长排行。资费使用率是根据 Unix 的 IP 查询统计拥有的资费数量。查询核心代码如下：

```

if ("tag1".equals(tag)) {
    reportSumDurations =
reportDAO.findSumDurationPage(page, pageSize);
    totalPages = reportDAO.totalPagesSumDuration(pageSize);

    months = new ArrayList<String>();
    durations = new ArrayList<String>();
    for (ReportSumDuration rsd: reportSumDurations) {
        months.add(rsd.getMonth().substring(0, 4) + "年" +
rsd.getMonth().substring(4) + "月");
        int n = rsd.getDuration();
        durations.add((n / 3600) + "小时" + (n / 60 % 60) + "分
钟" + (n % 60) + "秒");
    }

} else if ("tag2".equals(tag)) {
    List<String> hosts = reportDAO.findAllHost();
    List<ReportSumDuration> list = new
ArrayList<ReportSumDuration>();
    for (String host:hosts) {
        list.addAll(reportDAO.findThreeDuration(host));
    }
    if (list.size() % pageSize == 0) {
        totalPages = list.size() / pageSize;
    } else {
        totalPages = list.size() / pageSize + 1;
    }
    if ((page * pageSize) < list.size()) {
        reportSumDurations = list.subList((page - 1) * pageSize ,
page * pageSize);
    } else {
        reportSumDurations = list.subList((page - 1) * pageSize,
list.size());
    }
    durations = new ArrayList<String>();
    for (ReportSumDuration rsd: reportSumDurations) {
        int n = rsd.getDuration();
        durations.add((n / 3600) + "小时" + (n / 60 % 60) + "分
钟" + (n % 60) + "秒");
    }
} else if ("tag3".equals(tag)) {
    reportCostUseds = reportDAO.findHostPage(page, pageSize);
    totalPages = reportDAO.totalPagesHost(pageSize);
}

```


界面截图如图 4.10。

客户使用时长

时长排行榜

资费使用率

账号 ID	账务帐号	客户名称	身份证号码	电话	月份	累积时长
1010	xl18z60	guojing	330682196903190613	13338924567	2013年04月	44小时17分钟0秒
1010	xl18z60	guojing	330682196903190613	13338924567	2013年05月	88小时34分钟0秒
1010	xl18z60	guojing	330682196903190613	13338924567	2013年08月	1小时41分钟8秒
1010	xl18z60	guojing	330682196903190613	13338924567	2014年05月	4小时51分钟10秒
1011	dgbf70	huangrong	330902197108270429	13637811357	2013年04月	45小时17分钟0秒
1011	dgbf70	huangrong	330902197108270429	13637811357	2013年08月	1小时0分钟0秒
1011	dgbf70	huangrong	330902197108270429	13637811357	2014年05月	1小时5分钟39秒
1019	ljxj90	luwushuang	320211199307310346	13186454984	2013年04月	2小时0分钟0秒
1019	ljxj90	luwushuang	320211199307310346	13186454984	2013年08月	8小时8分钟1秒
1019	ljxj90	luwushuang	320211199307310346	13186454984	2014年05月	145小时25分钟24秒

首页 上一页 1 2 下一页 末页

图 4.10 报表界面

4.4.8 基础功能

4.4.8.1 验证码功能

登录操作的时候会产生一个随机的验证码。运用 java 的 awt 组件进行画图，根据数字画出相应的图片，就产生了输入图片里的数字就能准确判断验证码。鼠标点击图片就更换的原理是在图片名号后面加上随机数字。确保每点击一次就能更换图片。核心代码如下：

```

public class ImageUtils {
    private static String str = "1234567890QWERTYUIOPASDFGHJKLZXCVBNM";
    private static char[] chars = str.toCharArray();
    private static int WIDTH = 140;
    private static int HEIGHT = 40;
    private static int SIZE = 4;
    private static int LINES = 6;
    private static int FONT_SIZE = 40;
    public static Map<String, BufferedImage> getImage() {
        BufferedImage image = new BufferedImage(WIDTH, HEIGHT,
            BufferedImage.TYPE_INT_RGB);
        Graphics g = image.createGraphics();
        g.setColor(Color.GRAY);
        g.fillRect(0, 0, WIDTH, HEIGHT);
        StringBuffer bf = new StringBuffer();
        Random r = new Random();
        for (int i = 0; i < SIZE; i++) {
            char c = chars[r.nextInt(chars.length)];
            g.setColor(getColor());
            g.setFont(new Font(null, Font.BOLD, FONT_SIZE));
            g.drawString("" + c, i * WIDTH / SIZE, HEIGHT / 10 * 9);
            bf.append(c);
        }
        for (int i = 0; i < LINES; i++) {
            g.setColor(getColor());
            g.drawLine(r.nextInt(WIDTH), r.nextInt(HEIGHT), r.nextInt(WIDTH),
                r.nextInt(HEIGHT));
        }
        Map<String, BufferedImage> map = new HashMap<String,
        BufferedImage>();
        map.put(bf.toString(), image);
        return map;
    }
    private static Color getColor() {
        Random r = new Random();
        return new Color(r.nextInt(255), r.nextInt(255), r.nextInt(255));
    }
    public static InputStream imageToStream(BufferedImage image) throws
    Exception{
        ByteArrayOutputStream bos = new ByteArrayOutputStream();
        JPEGImageEncoder encoder = JPEGCodec.createJPEGEncoder(bos);
        encoder.encode(image);
        byte[] bytes = bos.toByteArray();
        InputStream is = new ByteArrayInputStream(bytes);
        return is;
    }
}

```

5 测试

测试是为了尽可能多的发现软件的缺陷，让程序更加健壮。使上线后能够稳定的运行。一个好的测试方案是在开发后期软件提升性能的办法。

5.1 测试原则及测试方法

5.1.1 测试的任务

一个新开发的系统避免不了 BUG 和功能上的缺陷，所以在上线之前需要一个测试，模拟真实的环境来运行系统，目的是为了检验系统是否存在缺陷或者未知的错误，以保证上线时能稳定的运行。

5.1.2 测试方案

测试有两种方法：黑盒测试和白盒测试。黑盒测试又称为功能测试，测试者不需要掌握程序的代码、内部结构和编程语言的专门之知识。测试人员只需知道该系统是做什么的，有什么功能，然后根据功能做操作来验证是否得出预期的结果。白盒测试又叫结构测试，以编程语言的角度来设计测试案例，测试者输入验证数据流在程序流动路径，并确定适当的输出，类似测试电路中的节点。

电信 Unix 服务器出租管理系统开发过程进行的测试步骤如下：

1) 模块测试：也叫单元测试，一般由程序员测试。单元测试是为了确保开发的模块可以正常运行，为系统集成后能稳定运行作保证。

2) 集成测试：也叫综合测试、组装测试、联合测试。软件各个模块开发好后，将所有模块集成到一个系统，因为可能会因为接口或者函数重名或者某些原因会导致集成后有未知的 BUG。

3) 验收测试：这一步是验证软件的有效性。目的是验证是否符合客户的需求。如果功能和性能与用户要求一致，软件是可以通过的。这个阶段发现的问题往往和需求分析阶段的差距有关。

4) 平行运行：所谓平行运行就是同时运行，新开发的系统和旧系统进行比较，看结果差异。目的有：

➤ 在新系统直接部署在旧系统上有很大的风险。

- 用户需要适应新系统。
- 新系统需要经过用户的测试才能发现更隐藏的 bug。

5.2 软件测试用例

5.2.1 登录测试

(1) 登录测试

测试用描述	操作过程	预期结果	用例类别
输入正确的用户名和密码但输入错误的验证码	登录时输入和图片验证码不一样的字母	提示验证码错误并留在登录页面	功能点

(2) 账务账号测试

测试用描述	操作过程	预期结果	用例类别
在搜索框填写一个正确的身份证号码或者是姓名或者登录名	点击搜索	会有符合条件的账务账号显示	功能点
点击下一页或者上一页	点击下一页或者上一页按钮	会翻页显示结果，如果到了最后一页，则下一页按钮不能用，如果是第一页，则上一页按钮不能用	功能点
选择一个账号 ID，暂停该账号	点击暂停按钮	点击立马变成开通的按钮，查看业务所属账号看是否为暂停状态	功能点
在增加页面输入也提示不符的内容从	点击增加按钮并输入内容	如果输入的信息符合提示信息不会显示错误框，如果不符合则会显示错误提示框且不能提交	功能点

6 结论

经过这次论文的撰写，通过软件工程的思路，对电信 Unix 服务器出租管理系统的整个开发做了详细的阐述。同时遵循结构程序设计，按照当前最流行的软件开发的流程进行开发。

该系统是基于 java 三大框架包括 Struts 框架、Hibernate 框架、Spring 框架整合，并且采用 AJAX 和 JQuery 等交互技术结合，使得页面展现更加人性化。同时阐述了 J2EE 的相关技术，ORACLE 技术，CSS 美化技术。

该电信 Unix 服务出租系统有一个完善的管理平台，包含了角色管理和管理员管理，资费管理，账务账号和业务账号管理，账单管理，报表功能。该系统完成了需求分析的所有功能。关于数据库的设计，遵循了三个范式，而且还采用数据库计费，大大提高了系统的性能。

该系统虽然完成了所有功能，但还是觉得有些方面还有欠缺，比如账单并没有实现打印功能，也没有黑名单的功能，加上现在还有没有真是的环境来提取数据，只能模拟用户登入登出，使得数据有点不太合常理。不可否认，该系统存在的缺陷，由于是一个人开发的系统并且需要依赖外部系统才能实现真正的功能。

参考文献

- [1]. Simache、Kaaniche. Availability Assessment of SunOS/Solaris UNIX Systems based on Syslogd and wtmpx log files: a case study. 11th Pacific Rim International Symposium on Dependable Computing. 2005
- [2]. 华文华. 国内电信计费系统现状分析. 人民邮电. 2002
- [3]. 李连祥、刘晓亮. 电信计费的内涵与外延. 中国计费网. 2004
- [4]. 卢丹. ORACLE 数据库性能的监控和调优技术. 企业文化: 中. 2012
- [5]. 杨少敏、王红敏. Oracle 11g 数据库应用简明教程. 清华大学出版社. 2010
- [6]. 张海藩. 软件工程导论 (第五版). 清华大学出版社. 2008
- [7]. 布鲁斯埃克尔. Thinking in java (第四版). 机械工业出版社. 2007
- [8]. Bob bryla、kevin loney、刘伟琴 (译). Oracle Database 11g DBA 手册. 清华大学出版社. 2009
- [9]. 孙卫琴. 精通 Struts: 基于 MVC 的 Java Web 设计与开发. 电子工业出版社. 2004
- [10]. 孙卫琴. 精通 Hibernate: Java 对象持久化技术详解 (第 2 版). 电子工业出版社. 2010
- [11]. 计文柯. Spring 技术内幕 深入解析 Spring 架构与设计原理. 机械工业出版社. 2010
- [12]. 何丽. 精通 DIV+CSS 网页样式与布局. 清华大学出版社. 2011
- [13]. Bear Bibeault、Yehuda Katz. jQuery 实战 (第 2 版). 人民邮电出版社. 2012
- [14]. 弗兰纳根 著、淘宝前端团队 译. JavaScript 权威指南 (第 6 版). 机械工业出版社. 2012
- [15]. Guarana UI: A JQuery Based UI Library for Nokia WRT. Forum Nokia. 2010
- [16]. Price, D (Price, D); Tucker, A (Tucker, A). Solaris zones: Operating system support for consolidating commercial workloads. 18th Large Installation System Administration Conference (LISA 18). 2004
- [17]. Futagawa. Integrating network services of windows and UNIX for single sign-on. International Conference on Cyberworlds (CW 2004). 2004

.

致谢

经过这次毕业设计及论文的撰写,通过软件工程的思路,设计及实现了电信 UNIX 服务器出租业务管理系统,这次的毕业设计给了我很多宝贵经验,并了解了大型软件开发的流程及实现,最重要的是在我的软件开发生涯中开了一个好头,让我觉得开发一个软件并没有想象中的难。我会坚定不移的走软件开发这条路。在这里感谢杨国为老师为我指导,无论从技术还是理论上的迷惑都一一帮我解答,他认真,细致,一丝不苟的风格,不断的引导我,才使得撰写论文顺利进行,是我学习的榜样。

在撰写论文过程中,遇到了很多困难,但始终要坚信问题总是可以解决的,只要有毅力,查资料,问老师,最终把全部问题解决了,所有要坚信,只要坚持不懈,有毅力,问题便会不攻自破。

最后,我还要感谢大学四年教过我的老师,您们精心培育我们,日日夜夜为我们操劳,感谢你们,让我四年后的今天有了质的飞跃,可以为社会做出贡献了。

感谢信息工程学院,感谢母校四年来对我的精心培养。