中广有线枣庄分公司绩效考核系统

研究成果报告

**一、本项目研究的主要内容**

项目进行的主要内容分为需求分析，系统模块设计，数据库设计，后台逻辑代码编写，前台页面代码编写，数据导入脚本代码编写，模拟打分测试脚本代码编写。

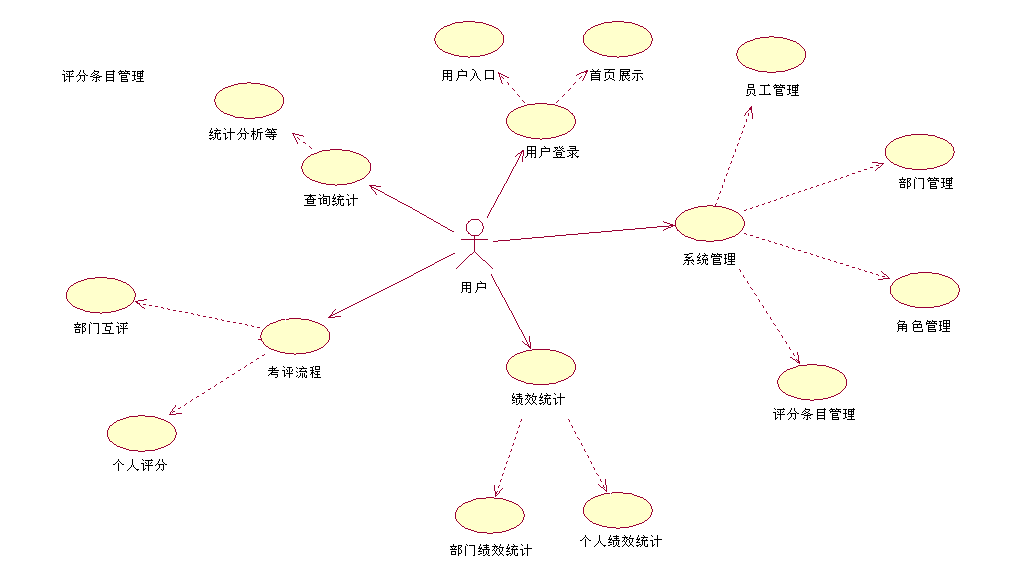
1. 需求分析

所谓"需求分析"，是指对要解决的问题进行详细的分析，弄清楚问题的要求，包括需要输入什么数据，要得到什么结果，最后应输出什么。可以说，在软件工程当中的“需求分析”就是确定要计算机“做什么”，要达到什么样的效果。可以说需求分析是做系统之前必做的。

本系统要解决的主要问题为中广有线枣庄分公司的部门以及员工绩效考评问题。系统需要实现的主要需求有：针对部门、员工的打分（其中部门打分需要实现部门间的互评），根据打分数据计算部门员工绩效得分，部门负责人上传周例会、月例会以及学习培训材料，短信、通知、公告发布功能。以上需求全部基于B/S架构，由浏览器访问服务器实现。

1. 系统模块设计

根据需求，将系统模块设计如下：



各子模块功能的描述：

**用户登录**

用户可以通过用户入口登录系统，根据登录系统人员的权限不同，登录后所能操作的功能也不相同。登录后进入首页展示，根据登录用户的角色，显示不同的首页信息，提供不同的功能跳转。

1. 用户入口

用户可以通过用户入口进行用户验证从而登录系统，登录时要实现异步提交。只有用户名和密码都正确的时候可以进入首页展示。如果不正确可以在入口处做出提示。

1. 首页展示以及角色判断

用户根据不同的部门职位和对软件的需求被划分为不同的角色，这一模块便是根据不同用户角色向用户展示不同的内容。例如对于普通用户来说，只展示绩效考评流程；对于领导登录后可以查看权限范围内的所有考评方案的信息列表，并可以查看相应方案的考评结果总览。

**系统管理**

用户登录之后通过系统管理模块可以进行员工管理、部门管理、角色管理、评分条目管理的功能。对于不同的用户会有不同的权限，只有具有权限的用户才可以实现相应权限所能操作的功能。目前版本所有管理权限都交由系统管理员。

1. 员工管理

在员工管理模块，用户可以对用户信息进行增、删、改、查和角色的分配。对于用户的增、删、改以及角色的分配，必须具有一定的权限。如果没有管理员权限不可以进行这些操作。

1. 部门管理

在部门管理模块，用户可以进行部门的增、删、改、查，负责人的任命和修改，以及上下级部门关系的建立以及修改。对于部门的增删改需要具有权限。

1. 角色管理

在角色管理模块可以对角色进行增、删、改、查以及角色的分配。增删改以及分配都必须具有权限，否则不能进行

1. 评分条目管理

在角色管理模块可以对评分所针对的条目进行增、删、改、查。其中条目分为部门互评的条目以及个人打分的条目。增删查操作都必须具有权限，否则不能进行。

1. 职位管理

在职位管理模块可以对员工职位进行增、删、改、查。

**考评流程**

本模块针对不同的考评打分类型划分为部门互评和个人打分两个模块。不同的用户可以根据自己的角色权限在本模块中完成自己的考评打分流程。

1. 部门互评

部门互评模块针对部门平时的业务工作联系实现部门之间以及上下级部门之间的互评打分以及对本次打分的修改。

1. 个人打分

在这一项中可以设置投票的依据项目。可以对项目进行增、删、改、查，通过投票项的设置来制定投票的依据标准。

**绩效统计**

本模块针对系统管理员，对已经完成的部门以及个人打分进行统计汇总，并计算出当每个部门以及个人的具体绩效得分。共分为部门绩效统计和个人绩效统计两部分。

1. 部门绩效统计

本功能根据部门互评所得数据，对相应部门所获取的打分数据以相同条目合并然后根据相应算法算出条目得分，并且根据其他得分要素统计算出部门所得出的实际绩效得分。

1. 个人绩效统计

本功能根据个人打分所得出的具体数据，以及所在部门得分和其他其他要素得分以相应的算法为基准计算得出每个员工的具体绩效得分。

**查询统计**

本模块可以根据具体需求对相应打分数据进行查询统计。

**例会以及学习培训材料上传和查看**

1. 周例会上传

本功能实现部门负责人每周定时上传本部门的周例会。

1. 月例会上传

本功能实现部门负责人每月定时上传本部门的月例会。

1. 学习培训材料上传

本功能实现部门负责人每月定时上传本部门进行的学习培训的相关材料。

1. 相应例会、学习培训材料的查看

本功能实现对相关部门的相关周例会、月例会、学习培训材料的查看。

**邮件**

本模块实现员工之间基于本系统的邮件通讯功能。员工所发送的邮件内容包括文本以及附件。发送人的选取包括单人发送，部门发送，员工类别发送。

1. 写邮件

本功能实现编写邮件、发送邮件的功能。

1. 收件箱

本功能实现员工查看自己已经接收的邮件。

1. 未读邮件

本功能实现员工查看自己的未读邮件。

1. 已发邮件

本功能实现员工查看自己已经发送的邮件。

1. 数据库设计

在目前的企业信息系统中，数据库还是最佳的数据存储方式。本系统采用领域模型驱动的方式和自顶向下的思路进行数据库设计，分析系统业务，根据职责定义对象。对象符合封装的特性，确保与职责相关的数据项被定义在一个对象之内，这些数据项能够完整描述该职责，不会出现职责描述缺失。并且一个对象有且只有一项职责，如果一个对象要负责两个或两个以上的职责，应进行分拆。在映射后得出的数据库表结构中，应再根据第四范式进行进一步修改，确保不存在多值依赖。

1. 后台逻辑代码编写

本系统后台逻辑代码以java语言编写，版本为java1.6。运用旧版本的java可以保证系统运行稳定。Jsp框架采用了struts2，以保证前端页面和后台逻辑代码的链接稳定。

在后台类的设计上本系统遵循低耦合高内聚原则，尽量将系统后台逻辑模块化，提高了模块复用性。

1. 前台页面代码编写

前台页面采用html/css/javascript技术编写，在开发过程中运用了部分现有第三方库，降低了开发成本，缩短了开发周期。

页面动态化采用了jsp技术，与后台采用的struts2框架搭配使用，提高了系统的稳定性和可维护性。

1. 数据库导入脚本编写

对于商用的信息管理系统，初期的基础数据都极其重要并且会耗费大量工作量。系统提供的数据录入和编辑接口往往受限于浏览器输入困难，造成了业务人员输入基础数据的困难。为了解决这些问题，本系统开发了初期基础数据导入脚本。初期基础数据的导入采用excel表通过脚本直接导入数据库的方式。Excel文件遵循一定的规则编写，方便非技术人员批量修改和创建数据。

1. 模拟打分测试脚本代码编写

系统开发基本完成后就需要大量的数据测试。由于公司人员众多，进行一次全员打分需要消耗大量的人力和时间。为了解决这一问题，本系统开发了模拟打分测试脚本。本脚本以python语言为基础，批量模拟员工打分和部门打分，方便程序开发完成初期快速测试、debug和打分数据的获取，提高了测试效率。

**二、关键技术及创新点**

* 1. 部分数据库表结构设计思路

**员工表**

|  |
| --- |
| create table staff (  staff\_id int(11) not null auto\_increment,  staff\_name varchar(20) not null,  staff\_no varchar(20) not null, -- gonghao  staff\_sex varchar(10),  staff\_password varchar(20) not null,  department\_id int(11) not null,  post\_id int(11) not null,  role\_id int(11) not null,  class\_id text, -- string ;;  primary key (staff\_id),  foreign key(role\_id) references role(role\_id),  foreign key(post\_id) references post(post\_id),  foreign key(department\_id) references department(department\_id)  ) default charset=utf8; |

其中：

字符编码采取utf8，和后台代码以及前端页面统一，尽量避免编码不符产生的乱码问题。

唯一主键staff\_id来标识staff，并且用到了sql的自增功能。

每一个staff的post（职位）、role（角色）、department（部门）都分别外键链接到相应的表，增强了数据库安全性。

**部门条目表**

|  |
| --- |
| create table department\_score\_item(  score\_item\_id int(11) not null auto\_increment,  score\_item\_name varchar(200),  score\_description varchar(200),  score\_value float,  score\_reference varchar(200),  score\_calculation\_method text,  department\_id int(11),  score\_category\_id int(11),  score\_creation\_date date,  score\_in\_use int(1),  primary key(score\_item\_id),  foreign key(department\_id) references department(department\_id),  foreign key(score\_category\_id) references score\_category(score\_category\_id)  )default charset=utf8; |

部门条目为部门互评打分服务，每一个条目关联一个打分部门和一个被打分部门。但在本系统中，为了方便条目的管理，条目关联一个部门id，此部门为此条目的被打分部门。

**部门条目对照表**

|  |
| --- |
| create table department\_score\_relation(  department\_score\_relation\_id int(11) not null auto\_increment,  score\_item\_id int(11),  department\_id int(11),  primary key(department\_score\_relation\_id),  foreign key(score\_item\_id) references department\_score\_item(score\_item\_id),  foreign key(department\_id) references department(department\_id)  )default charset=utf8; |

打分部门与条目的关联由部门条目对照表关联完成。此对照表中的部门id为关联条目的打分部门。由此可以看出，部门打分时只需根据对照表找出需要打分的条目，也就是打分部门只针对条目打分。需要访问被打分部门时再根据条目表的所属部门id获取相应的部门信息。

**部门打分记录表**

|  |
| --- |
| create table department\_score\_record(  department\_score\_record\_id int(11) not null auto\_increment,  department\_from\_id int(11),  department\_to\_id int(11),  score\_item\_id int(11),  score float,  score\_comment text,  score\_date date,  which\_year int(11),  which\_month int(11),  modify\_times int(11),  owner int(11), -- staffId  primary key(department\_score\_record\_id),  foreign key(department\_from\_id) references department(department\_id),  foreign key(department\_to\_id) references department(department\_id),  foreign key(score\_item\_id) references department\_score\_item(score\_item\_id),  foreign key(owner) references staff(staff\_id)  )default charset=utf8; |

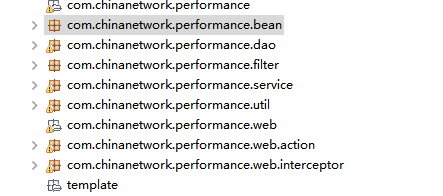
部门打分表记录部门打分的详细信息。每一条打分记录信息记录一个部门针对一个条目的打分信息。

**部门得分表**

|  |
| --- |
| create table department\_score(  department\_total\_score\_id int(11) not null auto\_increment,  department\_id int(11),  total\_score float,  score\_date date,  which\_year int(11),  which\_month int(11),  annotation text,  owner int(11), -- staffId  primary key(department\_total\_score\_id),  foreign key(department\_id) references department(department\_id),  foreign key(owner) references staff(staff\_id)  )default charset=utf8; |

部门打分全部完成后，管理员需要进行算分操作。算分操作主要执行将部门打分表中的详细打分数据汇总生成部门得分，插入部门得分表。部门得分表记录了。

* 1. 后台代码目录结构



其中：

Bean目录存放数据库实体类。

Dao目录存放数据库操作类。

Filter目录存放系统过滤器类。

Service目录存放逻辑代码类。

Util目录存放系统工具类。

Action目录存放后台与前台交互的action类。

* 1. 算分核心逻辑

算分逻辑如下：

1、总经理班子成员（满分100分）

（1）总经理：由中广总公司直接考核

（2）副总经理（满分100分）：即为主任绩效考核成绩（分管2个（含）以上部门的，取所分管部门主任总成绩的平均值）

2、主任（满分100分）：

公司整体运营指标考核成绩（占权重50%）+部门绩效考核成绩（占权重50%）。

3、副主任（满分100分）：

公司整体运营指标考核成绩（占权重50%）+（部门成绩×50%+个人绩效成绩×50%）（占权重50%）。

4、员工（满分100分）：

公司整体运营指标考核成绩（占权重50%）+（部门成绩×30%+个人绩效成绩×70%）（占权重50%）。

5、月度绩效奖金核算公式

每月实际发放绩效奖金=该岗位绩效考核奖金基数×该岗位套算后的月度考核成绩

* 1. 算分核心算法代码

部门算分核心sql代码：

|  |
| --- |
| String sql = "select department\_to\_id, sum(caonima) as score from (select department\_to\_id, score\_item\_id, avg(score) as caonima from department\_score\_record where which\_year = ? and which\_month = ? group by score\_item\_id) as temp  group by department\_to\_id;"; |

个人算分核心算法：

|  |
| --- |
| /\*\*  \* 市公司各部门副主任和员工，以及各区营业部的各个科室副科长和科员的绩效考评的计算公式  \*  \* **@param** staff  \* 被打分的员工  \* **@param** staffScoreMap  \* 员工各项评分细则的得分Map，以打分者的职位不同分类  \* **@param** year  \* 绩效考评年份  \* **@param** month  \* 绩效考评月份  \* **@param** fStaffA  \* 公式参数，A打分员工的职位名称  \* **@param** fStaffB  \* 公式参数，B打分员工的职位名称  \* **@param** coefficientA  \* 公式参数，A打分员工对应的权重  \* **@param** coefficientB  \* 公式参数，B打分员工对应的权重  \* **@param** staffScoreCoefficient  \* 公式参数，员工评分得分所占的权重  \* **@param** depScoreCoefficient  \* 公式参数，员工所属的部门得分所占的权重  \* **@return** totalScore 返回根据公式计算出的分数  \* **@throws** Exception  \*/  **public** **float** calculateStaff(Staff staff, Map<Staff, Float> staffScoreMap,  **int** year, **int** month, String fStaffA, String fStaffB,  **float** coefficientA, **float** coefficientB,  **float** staffScoreCoefficient, **float** depScoreCoefficient)  **throws** Exception {  **float** totalScore = 0;  Iterator iterator = staffScoreMap.keySet().iterator();  List<Float> scoreListA = **new** ArrayList<Float>();  List<Float> scoreListB = **new** ArrayList<Float>();  **while** (iterator.hasNext()) {  Staff staffKey = (Staff) iterator.next();  System.out.println("打分员工名称：" + staffKey.getStaffName());  System.out.println("所打分数：" + staffScoreMap.get(staffKey));  **if** (staffKey.getPost().getPostName().equals(fStaffA)) {  scoreListA.add(staffScoreMap.get(staffKey));  // totalScore=totalScore+(float)(staffScoreMap.get(staffKey)\*coefficientA);  } **else** **if** (staffKey.getPost().getPostName().equals(fStaffB)) {  // totalScore=totalScore+(float)(staffScoreMap.get(staffKey)\*coefficientB);  scoreListB.add(staffScoreMap.get(staffKey));  }  }  **float** totalA = 0;  **float** totalB = 0;  **for** (**int** i = 0; i < scoreListA.size(); i++)  totalA += scoreListA.get(i);  **for** (**int** i = 0; i < scoreListB.size(); i++)  totalB += scoreListB.get(i);  totalA = totalA / scoreListA.size();  totalB = totalB / scoreListB.size();  totalScore = coefficientA \* totalA + coefficientB \* totalB;  System.out.println(fStaffA + "打分为：" + totalA);  System.out.println(fStaffB + "打分为：" + totalB);  System.out.println("员工 " + staff.getStaffName() + " 的个人得分为: "  + coefficientA + "\*" + totalA + " + " + coefficientB + "\*"  + totalB + " = " + totalScore);  DepartmentScore departmentScore = calculateScoreDao  .getScoreOfDepartmentBelongStaff(staff.getDepartment()  .getDepartmentId(), year, month);  System.out.println("所属部门："  + departmentScore.getDepartment().getDepartmentName());  System.out.println("所属部门分数：" + departmentScore.getScore());  **float** fuck = totalScore;  totalScore = totalScore \* staffScoreCoefficient  + departmentScore.getScore() \* depScoreCoefficient;  System.out.println("员工 " + staff.getStaffName() + " 部门加权得分为: " + fuck  + "\*" + staffScoreCoefficient + " + "  + departmentScore.getScore() + "\*" + depScoreCoefficient  + " = " + totalScore);  **return** totalScore;  } |

* 1. 基础数据导入脚本中，部门条目的关联部门在excel文件中写为”分管副总、关联部门[3, 10-19]”。方括号中数字为部门编号，单独数字代表单独部门，10-19代表部门编号范围。数字编号解析代码具体如下:

|  |
| --- |
| #-\*-coding:utf-8-\*-  import re  str = '分管副总、关联部门[3, 10-19]'  #/'fuck you' --> 'fuckyou'  def trim(str):  return ''.join(str.split(' '))  # '1-4' --> [1, 2, 3, 4]  def getNumbers(str):  #print 'geting number %s'%str  start, end = str.split('-')  return range(int(start), int(end)+1)  def parse(str):  patten = re.compile(r'\[[,\d-]\*\]')  numbersets = patten.findall(trim(str))  ret = []  for numberset in numbersets:  #print '-------------------begin------------------------------'  #print 'numberset:',numberset  numbers = numberset[1:-1].split(',')  #print 'numbers:',numbers  for num in numbers:  #print 'num:',num  if num.isdigit():  ret.append(int(num))  #print 'single number added : %s' % num  elif num.find('-') != -1:  ret.extend(getNumbers(num))  #print '-------------------end------------------------------'  #print  return ret  #print parse(str) |

**三、取得的成就**

本系统现已正式上线运行，应用于本公司的每月绩效考评和例会以及学习培训资料上传，并进行了3次系统更新升级和服务器维护，已经成为了本公司的日常常用软件。