ESM系统架构设计

**编号**：**Zpark-ESM-HLD**

**版本 1.0**

|  |  |  |  |
| --- | --- | --- | --- |
| 作者： |  | 日期： |  |
| **Email：** |  | **组号：** |  |
| 审批： |  | 日期： |  |

**变更记录**

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 版本 | 变更说明 | 作者 |
|  | 1.0 | 创建 |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 1. 引言

## 文档目的

本文档定义软件系统的架构以及全局性的设计元素、算法。

本文档将作为详细设计、数据库设计的依据。

## 读者

软件设计工程师、软件编码工程师、软件测试工程师。

# 技术选择

采用J2EE，即客户端页面为JSP，服务器端业务逻辑采用Java语言实现，实体类为JavaBean，Web服务器采用tomcat9.0。

# 逻辑分层结构

表示层

控制层

业务层

数据库

数据访问层

图1 系统架构模型示意图

表示层：采用Java Web开发技术。

控制层：采用SpringMVC的Controller组件（或者J2EE中的Servlet组件）。

业务层：封装业务逻辑（业务JavaBean）。

数据访问层：负责访问数据库，处理事务（数据访问JavaBean）。

数据库存储层：MySQL数据库。

# 设计元素的映射

1、每个页面映射到一个JSP页面。

2、每个控制类映射到两个类，一个是表现控制层的Controller/Servlet；另一个是业务层的业务接口(实现类)。业务类的接口与上层的Controller/Servlet通信。 Controller/Servlet位于表现控制层，业务接口及其实现类位于业务逻辑层。

3、每个实体类映射到两个类，一个是仅包含属性的类，成为实体PO，其符合JavaBean规范，也叫“值对象”；另一个是DAO类，包含操作方法的类，专门用于访问数据库(也可增加DAO的接口)，DAO位于数据访问层。

4、可以增加另一个实体JavaBean称为“VO”，是Controller用的。

根据以上映射原则，将鲁棒性分析中的分析类映射为架构中的各个类，如下表所示：

注：灰色底部分为鲁棒性分析中的分析类表

| 模块名 | 分析类名称 | 分析类类型 | 映射到架构上的  设计类名称 | 设计类类型 |
| --- | --- | --- | --- | --- |
| 组织结构管理模块 | ESM-ZZJG-1 | 边界类 | post\_query.jsp | JSP页面 |
| ESM-ZZJG-2 | 边界类 | post\_add.jsp | JSP页面 |
| ESM-ZZJG-3 | 边界类 | post\_relationship\_add.jsp | JSP页面 |
| ESM-ZZJG-4 | 边界类 | post\_relationship\_add\_updown.jsp | JSP页面 |
| ESM-ZZJG-5 | 边界类 | post\_relationship\_delete.jsp | JSP页面 |
| ESM-ZZJG-6 | 边界类 | post\_relationship\_delete\_updown.jsp | JSP页面 |
| ESM-ZZJG-7 | 边界类 | post\_relationship\_query.jsp | JSP页面 |
| ESM-ZZJG-8 | 边界类 | post\_relationship\_query\_down.jsp | JSP页面 |
| ESM-ZZJG-9 | 边界类 | post\_relationship\_query\_up.jsp | JSP页面 |
| 主页面 | 边界类 | index.jsp | JSP页面 |
| 异常页面 | 边界类 | exception.jsp | JSP页面 |
| newPositionControl | 控制类 | NewPositionController.java | Controller |
| <<interface>> ZzjgService.java中的方法：NewPosition() | **业务接口：ZzjgService** |
| deletePositionControl | 控制类 | DeletePositionController.java | Controller |
| <<interface>> ZzjgService.java中的方法：DeletePosition() | **ZzjgService** |
| updatePositionControl | 控制类 | UpdatePositionController.java | Controller |
| <<interface>> ZzjgService.java中的方法：UpdatePosition() | **ZzjgService** |
| queryPositionControl | 控制类 | QueryPositionController.java | Controller |
| <<interface>> ZzjgService.java中的方法：QueryPosition() | **ZzjgService** |
| addUpDownControl | 控制类 | AddUpDownController.java | Controller |
| <<interface>> ZzjgService.java中的方法：AddUpDown() | **ZzjgService** |
| deleteUpDownControl | 控制类 | DeleteUpDownController.java | Controller |
| <<interface>> ZzjgService.java中的方法：DeleteUpDown() | **ZzjgService** |
| queryUpDownControl | 控制类 | QueryUpDownController.java | Controller |
| <<interface>> ZzjgService.java中的方法：QueryUpDown() | **ZzjgService** |
|  |  | ZzjgServiceImp.java | 业务接口的实现类 |
| Position | 实体类 | ZzjgDao.java | DAO组件 |
| Position.java | PO值对象 |
|  |  | PositionVO.java | VO值对象 |
|  |  | Connection.java | 连接数据库 |
|  |  | 其它类 | 公共类；  样式文件；  JS文件；  配置文件；等等 |

# 包结构设计

规划包的结构可以按层绘画或按模块规划：

以层为考虑，每层键一个包，包内建不同组件的子包。

以模块为考虑，每模块建一个包，包内建各层的子包。

本系统是层规划，每层建一些包：

（1）页面包：存放模块的页面；可配合css包、js包等，分别存放页面样式文件和JavaScript代码文件。

（2）Controller包：存放模块的所有Controller，每个用例一个Controller。

（3）业务逻辑包：存放模块的业务接口和业务实现类。

（4）数据访问包：存放DAO接口和DAO实现类。

（5）公共组件包：存放一些公共组件。

（6）JavaBean包：VO包和PO包；VO包中的JavaBean为Controller所用，PO包中的JavaBean为DAO所用。

## “组织结构管理”模块的包设计

1、页面包名称：“jsp.zzjg”；

2、css包名称：“css.zzjg”；

3、js包名称；“js.zzjg”；

4、Controller包名称：“ Controller.zzjg”，其中有本模块的所有Controller类；

5、VO包名称：“VO.zzjg”，其中有类“PositionVO”；

6、业务逻辑包名称：“ service.zzjg”，其中又有接口“ZzjgService”和实现类“ZzjgServiceImp”

7、数据访问包：“ dao.zzjg” ，其中有DAO类“zzjgDao”

8、PO包名称：“PO.zzjg”，其中有PO类“Position”；

9、JDBC包名称：“JDBC”，其中有链接数据库类“Connection”

10、公共包名称：“util.zzjg”。

## 总包图



Controller.zzjg

### 页面层

  

### 页面控制层：

 

Controller

### 业务逻辑层：

 

### 数据访问层：

  

### 脚本语言

在JSP中采用JavaScript脚本语言，验证表单数据的完整性，有效性。

如“新建岗位”用例中的岗位信息是否完整，是否符合数据字典和规则。

验证用户“修改密码”用例的两次新密码输入是否相等。

其它一些需要在页面进行的处理逻辑。

# 调用关系图——组织结构管理调用关系图



其它模块：略