## 1.1人员机构管理模块设计文档

### 1.1.1说明

人员机构管理界面UserAndDepartmentManagePanel是总经理界面ManagerPanel的一部分，总经理界面通过JtabblePanel分为人员机构管理界面、审批单据界面、制定价格工资常量的界面。下面详细说明人员机构管理界面。

人员机构管理界面UserAndDepartmentManagePanel需要在构造方法里传入用户管理业务逻辑接口UserAndDepartmentManageblservice

UserblService定义了User人员的增删改查的方法。

其具体实现类UserAndDepartmentManageblserviceimpl需要在构造方法中传入多个数据接口

在设计的数据库中，一个User有两个外键分别于job、department对应。

业务实现需要3个数据接口

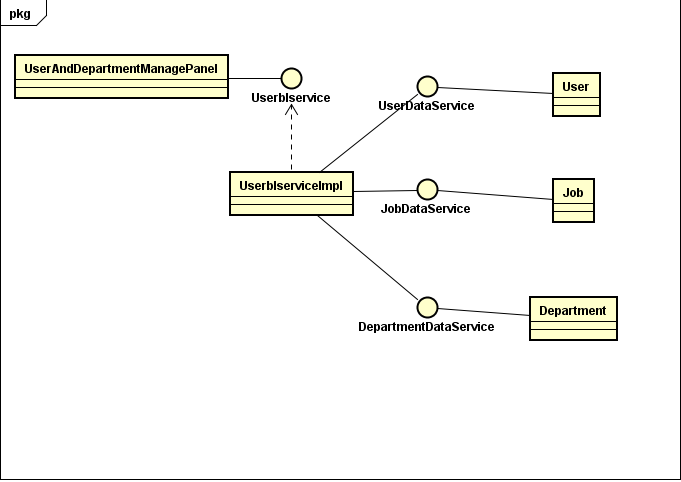
* UserDataService 用户数据逻辑
* JobDataService 工作数据逻辑
* DepartmentDataService 部门数据逻辑

以获得方法为例，从数据库获得一个User对象，需要通过UserDataService获得User对象（只包含其本身的属性，而不包含其他的对象），

再根据其外键的id获得User里包含的其他对象

若其他对象还包含着其他对象，需要继续封装

### 1.1.2人员机构管理详细设计描述



|  |  |
| --- | --- |
| 模块 | 职责 |
| UserAndDepartmentManagePanel | UserAndDepartmentManage的用户界面 |
| UserblserviceImpl | 负责有关User的业务逻辑实现 |
| Userblservice | User业务逻辑接口，定义需要的业务方法 |
| \*Dateservice | 各种实体类的数据接口，定义数据库操作方法 |

没有写关于UserAndDepartment的controller类，而是封装到Panel中

**UserAndDepartmentManagelPanel的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| UserAndDepartmentManagelPanel.view | 语法 | private void view() |
| 前置条件 | 用户点击表格的某一行 |
| 后置条件 | 根据选择的id调用查询方法，然后显示在下部的界面详细显示 |
| UserAndDepartmentManagelPanel.query | 语法 | private void query() |
| 前置条件 | 用户点击查询按钮 |
| 后置条件 | 根据查询文本框调用查询方法，然后将符合查询条件的User显示在表格中 |
| UserAndDepartmentManagelPanel.setViewData | 语法 | public void setViewData() |
| 前置条件 | 界面初始化 |
| 后置条件 | 调用查询方法，然后将结果的User显示在表格中 |
| UserAndDepartmentManagelPanel.clear | 语法 | private void clear() |
| 前置条件 | 用户点击取消按钮，或者User添加，修改，删除，查询操作完成后 |
| 后置条件 | 清空下面的文本框和按钮，重新获得数据，重新获得提示信息 |
| UserAndDepartmentManagelPanel.save | 语法 | private void save() |
| 前置条件 | 用户点击保存按钮 |
| 后置条件 | 如果UserId的文本框(隐藏)的值为空, 则是新增,调用add方法，否则为修改，调用update方法 |
| UserAndDepartmentManagelPanel.add | 语法 | private void add() |
| 前置条件 | 用户点击保存按钮后是新增操作，检查文本框内容是否存在以及合法 |
| 后置条件 | 调用Userblservice的add方法，然后clear清空 |
| UserAndDepartmentManagelPanel.update | 语法 | private void update() |
| 前置条件 | 用户点击保存按钮后是更新操作，检查文本框内容是否存在以及合法 |
| 后置条件 | 封装其他实体类的主键值和post的外键值，调用postblservice的update方法，然后clear清空 |
| UserAndDepartmentManagelPanel.delete | 语法 | private void delete() |
| 前置条件 | 用户点击删除按钮 |
| 后置条件 | 获得选择删除的User的id，调用Userblservice的delete方法，然后clear清空 |
| UserAndDepartmentManagelPanel.getUser | 语法 | private User getUser() |
| 前置条件 | add或update方法调用时 |
| 后置条件 | 根据文本框以及其他按钮封装User对象 |
| UserAndDepartmentManagelPanel.getJob | 语法 | private Job getJob() |
| 前置条件 | add或update方法调用时 |
| 后置条件 | 根据文本框以及其他按钮封装Job对象 |
| UserAndDepartmentManagelPanel.getDepartment | 语法 | private Department getDepartment() |
| 前置条件 | add或update方法调用时 |
| 后置条件 | 根据文本框以及其他按钮封装Department对象 |

|  |  |
| --- | --- |
| **需要的服务（需接口）** | |
| 服务名 | 服务 |
| Userblservice | User业务逻辑接口 |
|  |  |

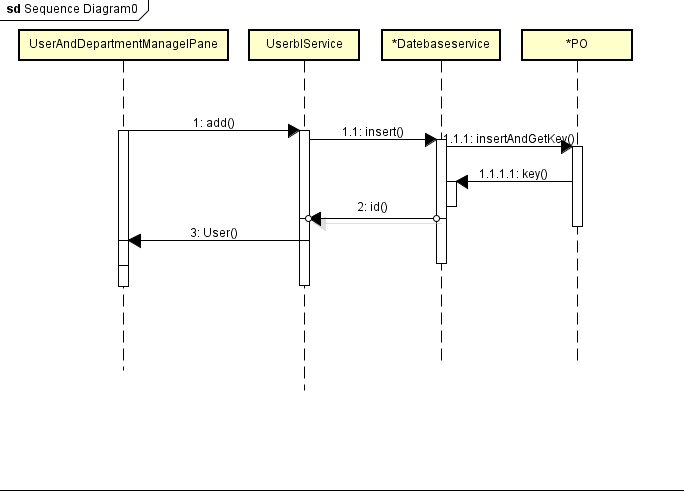
**UserblserviceImpl的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| UserblserviceImpl.getById | 语法 | public User getById(int id) |
| 前置条件 | 需要根据id查询post时 |
| 后置条件 | 调用User DataService的getById方法 |
| UserblserviceImpl.add | 语法 | public Post add(Post post) |
| 前置条件 | 需要新增一个User时 |
| 后置条件 | 调用User DataService的insert方法，  获得新增的其他实体类的主键id，设置User对应的外键的属性的值，调用User DataService的insert方法，获得新增User的主键，根据该主键id查询User并返回这个新增的User |
| UserblserviceImpl.update | 语法 | public User update(User user) |
| 前置条件 | 需要更新User时 |
| 后置条件 | 调用User DataService的update方法，  调用User DataService的update方法，获得返回的User的主键，根据该主键id查询User并返回这个User |
| UserblserviceImpl.delete | 语法 | public boolean delete(int id) |
| 前置条件 | 需要删除一个User时 |
| 后置条件 | 调用UserDataservice的delete方法 |

|  |  |
| --- | --- |
| **需要的服务（需接口）** | |
| 服务名 | 服务 |
| departmentDataservice.getList() | 获得所有的department持久化对象 |
| departmentDataservice.getById() | 根据id获得department持久化对象 |
| departmentDataservice.insert() | 增加一个department持久化对象 |
| departmentDataservice.update() | 更新一个department持久化对象 |
| departmentDataservice.delete() | 删除一个department持久化对象 |
|  |  |
| jobDataservice.getList() | 获得所有的job持久化对象 |
| jobDataservice.getById() | 根据id获得job持久化对象 |
| jobDataservice.insert() | 增加一个job持久化对象 |
| jobDataservice.update() | 更新一个job持久化对象 |
| jobDataservice.delete() | 删除一个job持久化对象 |
|  |  |
| userDataservice.getList() | 获得所有的user持久化对象 |
| userDataservice.getById() | 根据id获得user持久化对象 |
| userDataservice.insert() | 增加一个user持久化对象 |
| userDataservice.update() | 更新一个user持久化对象 |
| userDataservice.delete() | 删除一个user持久化对象 |

### 1.1.3人员机构管理动态模型

以新增User为例，当用户保存时User主键为空，则为新增操作，调用add方法，调用Userblservice的add方法，调用各个Dataservice的insert方法，通过数据库insert每个PO对象，返回其主键id。



### 1.1.4人员机构管理模块设计原理

利用分层的结构，和委托式的控制风格，将其业务逻辑独立出来，通过不同的数据管理对象管理数据，实现了高内聚低耦合的特点。

### 1.1.5度量分析

UserAndDepartmentManagePanel这个类调用的只有Userblservice的4个方法，和一个Userblservice对象引用，

所以CBO为4，DAC为1，继承一个commonpanel，无子类，NOC为0，DIT为1

Userblservice有UserDataService 用户数据逻辑 JobDataService 工作数据逻辑DepartmentDataService 部门数据逻辑3个对象引用，以及他们的共5\*3=15个方法。

所以CBO为15，DAC为3，UserblserviceImpl实现该接口，NOC为1，DIT为0

## 1.2审批单据模块设计文档

### 1.2.1说明

审批单据界面 DocumentApprovalPanel是总经理界面ManagerPanel的一部分，总经理界面通过JtabblePanel分为人员机构管理界面、审批单据界面、制定价格工资常量的界面。下面详细说明人员机构管理界面。

审批单据界面 DocumentApprovalPanel需要在构造方法里传入审批单据业务逻辑接口DocumentApprovalblservice

DocumentApprovalblService定义了查询和审批单据的方法。

其具体实现类DocumentApprovalblserviceimpl需要在构造方法中传入多个数据接口

业务实现需要8个数据接口

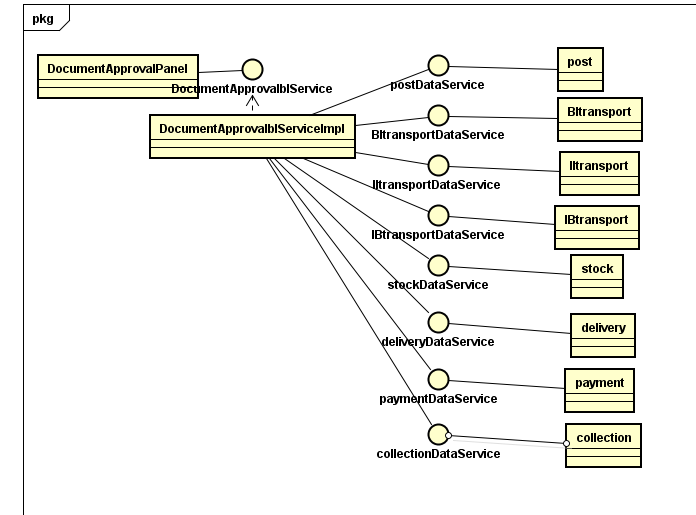
* postDataService 寄件单数据逻辑
* BItransportDataService BI运输表数据逻辑
* IItransportDataService II运输表数据逻辑
* IBtransportDataService IB运输表数据逻辑
* stockDataService 库存单数据逻辑
* deliveryDataService 派件单数据逻辑
* paymentDataService 付款单数据逻辑
* collectionDataService 收款单数据逻辑

以获得方法为例，从数据库获得一个post对象，需要通过postDataService获得post 对象（只包含其本身的属性，而不包含其他的对象），

再根据其外键的id获得post里包含的其他对象

若其他对象还包含着其他对象，需要继续封装

### 1.2.2审批单据详细设计描述



|  |  |
| --- | --- |
| 模块 | 职责 |
| DocumentApprovalPanel | DocumentApproval的用户界面 |
| DocumentApprovalblserviceImpl | 负责有关DocumentApproval的业务逻辑实现 |
| DocumentApprovalblservice | DocumentApproval业务逻辑接口，定义需要的业务方法 |
| \*Dateservice | 各种实体类的数据接口，定义数据库操作方法 |

没有写关于DocumentApproval的controller类，而是封装到Panel中

DocumentApprovalPane**的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| DocumentApprovalPanel.view | 语法 | private void view() |
| 前置条件 | 用户点击表格的某一行 |
| 后置条件 | 根据选择的id调用查询方法，然后显示在下部的界面详细显示 |
| DocumentApprovalPanel.query | 语法 | private void query() |
| 前置条件 | 用户点击查询按钮 |
| 后置条件 | 根据查询文本框调用查询方法，然后将符合查询条件的表单显示在表格中 |
| DocumentApprovalPanel.setViewData | 语法 | public void setViewData() |
| 前置条件 | 界面初始化 |
| 后置条件 | 调用查询方法，然后将结果的表单显示在表格中 |
| DocumentApprovalPanel.clear | 语法 | private void clear() |
| 前置条件 | 用户点击取消按钮，或者修改、审批操作完成后 |
| 后置条件 | 清空下面的文本框和按钮，重新获得数据，重新获得提示信息 |
| DocumentApprovalPanel.save | 语法 | private void save() |
| 前置条件 | 用户点击保存按钮 |
| 后置条件 | 如果id  的文本框(隐藏)的值为空, 则是新增,调用add方法，否则为修改，调用update方法 |
| DocumentApprovalPanel.add | 语法 | private void add() |
| 前置条件 | 用户点击保存按钮后是新增操作，检查文本框内容是否存在以及合法 |
| 后置条件 | 调用各个数据接口的add方法，然后clear清空 |
| DocumentApprovalPanel.update | 语法 | private void update() |
| 前置条件 | 用户点击保存按钮后是更新操作，检查文本框内容是否存在以及合法 |
| 后置条件 | 封装其他实体类的主键值和post的外键值，调用各个数据接口肚饿的update方法，然后clear清空 |
| DocumentApprovalPanel.delete | 语法 | private void delete() |
| 前置条件 | 用户点击删除按钮 |
| 后置条件 | 获得选择删除的User的id，调用各个数据接口的delete方法，然后clear清空 |
| DocumentApprovalPanel.getPost | 语法 | Private Post getPost () |
| 前置条件 | add或update方法调用时 |
| 后置条件 | 根据文本框以及其他按钮封装post对象 |
| DocumentApprovalPanel.getBItransport | 语法 | private BItransport getBItransport () |
| 前置条件 | add或update方法调用时 |
| 后置条件 | 根据文本框以及其他按钮封装BItransport对象 |
| DocumentApprovalPanel.getIItransport | 语法 | private IItransport getIItransport () |
| 前置条件 | add或update方法调用时 |
| 后置条件 | 根据文本框以及其他按钮封装IItransport对象 |
| DocumentApprovalPanel.geIBtransport | 语法 | private IBtransport getIBtransport () |
| 前置条件 | add或update方法调用时 |
| 后置条件 | 根据文本框以及其他按钮封装IBtransport对象 |
| DocumentApprovalPanel.getstock | 语法 | private stock getstock () |
| 前置条件 | add或update方法调用时 |
| 后置条件 | 根据文本框以及其他按钮封装stock对象 |
| DocumentApprovalPanel.getdelivery | 语法 | private delivery getdelivery () |
| 前置条件 | add或update方法调用时 |
| 后置条件 | 根据文本框以及其他按钮封装delivery对象 |
| DocumentApprovalPanel.getpayment | 语法 | private payment getpayment () |
| 前置条件 | add或update方法调用时 |
| 后置条件 | 根据文本框以及其他按钮封装payment对象 |
| DocumentApprovalPanel.getcollection | 语法 | private collection getcollection () |
| 前置条件 | add或update方法调用时 |
| 后置条件 | 根据文本框以及其他按钮封装collection对象 |

|  |  |
| --- | --- |
| 需要的服务（需接口） | |
| 服务名 | 服务 |
| DocumentApprovalblservice | User业务逻辑接口 |
|  |  |

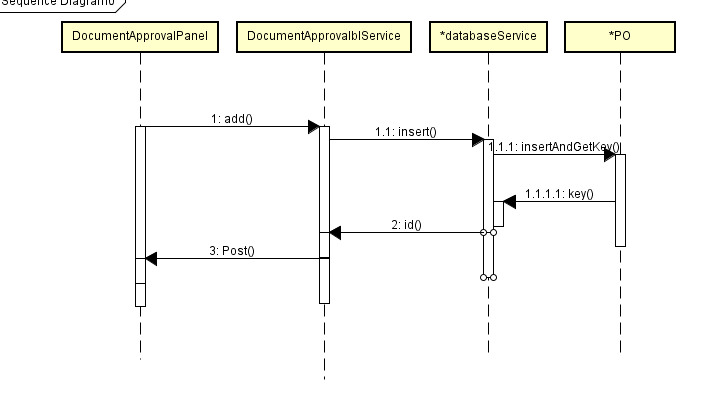
DocumentApprovalServiceImpl的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| DocumentApprovalServiceImpl.getById | 语法 | public E getById(int id) |
| 前置条件 | 需要根据id查询表单时 |
| 后置条件 | 调用各个数据接口的的getById方法， |
| DocumentApprovalServiceImpl.add | 语法 | public E add(E e) |
| 前置条件 | 需要新增一个表单时 |
| 后置条件 | 调用各个数据接口的insert方法，  获得新增的其他实体类的主键id，设置表单对应的外键的属性的值，调用各个数据接口的insert方法，获得新增的表单的主键，根据该主键id查询表单并返回这个新增的表单 |
| DocumentApprovalServiceImpl.update | 语法 | public Eupdate(E e) |
| 前置条件 | 需要更新表单时 |
| 后置条件 | 调用各个数据皆苦的update方法，  调用DocumentApprovalService的update方法，获得返回的比导弹的主键，根据该主键id查询表单 并返回这个表单 |
| DocumentApprovalServiceImpl.delete | 语法 | public boolean delete(int id) |
| 前置条件 | 需要删除一个表单时 |
| 后置条件 | 调用DocumentApprovalService的delete方法 |

|  |  |
| --- | --- |
| **需要的服务（需接口）** | |
| 服务名 | 服务 |
| postDataservice.getList() | 获得所有的post持久化对象 |
| postDataservice.getById() | 根据id获得post持久化对象 |
| postDataservice.insert() | 增加一个post持久化对象 |
| postDataservice.update() | 更新一个 post持久化对象 |
| postDataservice.delete() | 删除一个post持久化对象 |
|  |  |
| BItransportDataservice.getList() | 获得所有的BItransport持久化对象 |
| BItransportDataservice.getById() | 根据id获得BItransport持久化对象 |
| BItransportDataservice.insert() | 增加一个BItransport持久化对象 |
| BItransportDataservice.update() | 更新一个BItransport持久化对象 |
| BItransportDataservice.delete() | 删除一个BItransport持久化对象 |
|  |  |
| IItransportDataservice.getList() | 获得所有的IItransport持久化对象 |
| IItransportDataservice.getById() | 根据id获得IItransport持久化对象 |
| IItransportDataservice.insert() | 增加一个IItransport持久化对象 |
| IItransportDataservice.update() | 更新一个IItransport持久化对象 |
| IItransportDataservice.delete() | 删除一个IItransport持久化对象 |
|  |  |
| IBtransportDataservice.getList() | 获得所有的IBtransport持久化对象 |
| IBtransportDataservice.getById() | 根据id获得IBtransport持久化对象 |
| IBtransportDataservice.insert() | 增加一个IBtransport持久化对象 |
| IBtransportDataservice.update() | 更新一个IBtransport持久化对象 |
| IBtransportDataservice.delete() | 删除一个IBtransport持久化对象 |
|  |  |
| stockDataservice.getList() | 获得所有的stock持久化对象 |
| stockDataservice.getById() | 根据id获得stock持久化对象 |
| stockDataservice.insert() | 增加一个stock持久化对象 |
| stockDataservice.update() | 更新一个stock持久化对象 |
| stockDataservice.delete() | 删除一个stock持久化对象 |
|  |  |
| deliveryDataservice.getList() | 获得所有的delivery持久化对象 |
| deliveryDataservice.getById() | 根据id获得delivery持久化对象 |
| deliveryDataservice.insert() | 增加一个delivery持久化对象 |
| deliveryDataservice.update() | 更新一个delivery持久化对象 |
| deliveryDataservice.delete() | 删除一个delivery持久化对象 |
|  |  |
| paymentDataservice.getList() | 获得所有的payment持久化对象 |
| paymentDataservice.getById() | 根据id获得payment持久化对象 |
| paymentDataservice.insert() | 增加一个payment持久化对象 |
| paymentDataservice.update() | 更新一个payment持久化对象 |
| paymentDataservice.delete() | 删除一个payment持久化对象 |
|  |  |
| collectionDataservice.getList() | 获得所有的collection持久化对象 |
| collectionDataservice.getById() | 根据id获得collection持久化对象 |
| collectionDataservice.insert() | 增加一个collection持久化对象 |
| collectionDataservice.update() | 更新一个collection持久化对象 |
| collectionDataservice.delete() | 删除一个collection持久化对象 |

### 1.2.3审批单据动态模型

以新增post为例，当用户保存时post主键为空，则为新增操作，调用add方法，调用postblservice的add方法，调用各个Dataservice的insert方法，通过数据库insert每个PO对象，返回其主键id。



### 1.2.4审批单据模块设计原理

利用分层的结构，和委托式的控制风格，将其业务逻辑独立出来，通过不同的数据管理对象管理数据，实现了高内聚低耦合的特点。

### 1.2.5度量分析

DocumentApprovalPanel这个类调用的只有DocumentApprovalblservice的4个方法，和一个DocumentApprovalblservice对象引用，

所以CBO为4，DAC为1，继承一个commonpanel，无子类，NOC为0，DIT为1

Userblservice有postDataService 寄件单数据逻辑 BItransportDataService BI运输表数据逻辑IItransportDataService II运输表数据逻辑IBtransportDataService IB运输表数据逻辑stockDataService 库存单数据逻辑deliveryDataService 派件单数据逻辑paymentDataService 付款单数据逻辑collectionDataService 收款单数据逻辑8个对象引用，以及他们的共5\*8=40个方法。

所以CBO为40，DAC为8，DocumentApprovalServiceImpl实现该接口，NOC为1，DIT为0

## 1.3人员机构管理模块设计文档

### 1.3.1说明

制定价格工资常量的界面MakeConstantPanel是总经理界面ManagerPanel的一部分，总经理界面通过JtabblePanel分为人员机构管理界面、审批单据界面、制定价格工资常量的界面。下面详细说明人员机构管理界面。

制定价格工资常量的界面MakeConstantPanel需要在构造方法里传入用户管理业务逻辑接口MakeConstantblservice

MakeConstantblservice定义了基础价格常量的增删改查的方法。

其具体实现类MakeConstantblserviceimpl需要在构造方法中传入多个数据接口

业务实现需要1个数据接口

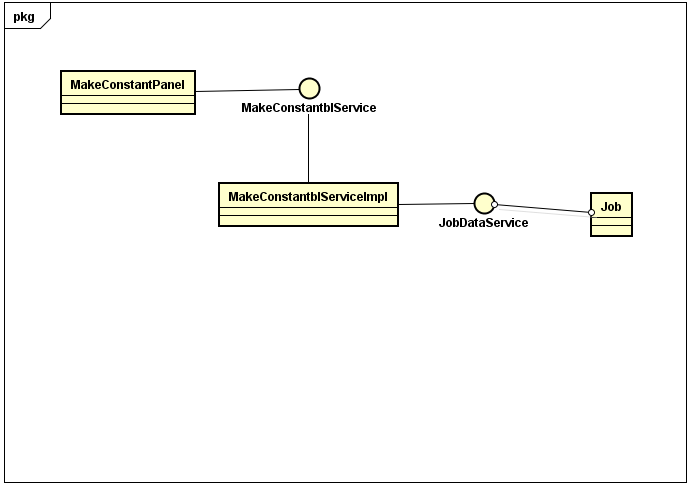
* JobDataService 工作数据逻辑

以获得方法为例，从数据库获得一个Job对象，需要通过JobDataService获得Job对象（只包含其本身的属性，而不包含其他的对象），

再根据其外键的id获得Job里包含的其他对象

若其他对象还包含着其他对象，需要继续封装

### 1.3.2人员机构管理详细设计描述



|  |  |
| --- | --- |
| 模块 | 职责 |
| MakeConstantPanel | MakeConstant的用户界面 |
| MakeConstantrblserviceImpl | 负责有关MakeConstant的业务逻辑实现 |
| MakeConstantblservice | MakeConstant业务逻辑接口，定义需要的业务方法 |
| \*Dateservice | 各种实体类的数据接口，定义数据库操作方法 |

没有写关于MakeConstant的controller类，而是封装到Panel中

**MakeConstantPanel的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| MakeConstantPanel.view | 语法 | private void view() |
| 前置条件 | 用户点击表格的某一行 |
| 后置条件 | 根据选择的id调用查询方法，然后显示在下部的界面详细显示 |
| MakeConstantPanel.query | 语法 | private void query() |
| 前置条件 | 用户点击查询按钮 |
| 后置条件 | 根据查询文本框调用查询方法，然后将符合查询条件的Job显示在表格中 |
| MakeConstantPanel.setViewData | 语法 | public void setViewData() |
| 前置条件 | 界面初始化 |
| 后置条件 | 调用查询方法，然后将结果的Job显示在表格中 |
| MakeConstantPanel.clear | 语法 | private void clear() |
| 前置条件 | 用户点击取消按钮，或者Jobr添加，修改，删除，查询操作完成后 |
| 后置条件 | 清空下面的文本框和按钮，重新获得数据，重新获得提示信息 |
| MakeConstantPanel.save | 语法 | private void save() |
| 前置条件 | 用户点击保存按钮 |
| 后置条件 | 如果JobId的文本框(隐藏)的值为空, 则是新增,调用add方法，否则为修改，调用update方法 |
| MakeConstantPanel.add | 语法 | private void add() |
| 前置条件 | 用户点击保存按钮后是新增操作，检查文本框内容是否存在以及合法 |
| 后置条件 | 调用MakeConstantblservice的add方法，然后clear清空 |
| MakeConstantPanel.update | 语法 | private void update() |
| 前置条件 | 用户点击保存按钮后是更新操作，检查文本框内容是否存在以及合法 |
| 后置条件 | 封装其他实体类的主键值和Job的外键值，调用MakeConstantblService的update方法，然后clear清空 |
| MakeConstantPanel.delete | 语法 | private void delete() |
| 前置条件 | 用户点击删除按钮 |
| 后置条件 | 获得选择删除的Job的id，调用MakeConstantblservice的delete方法，然后clear清空 |
| MakeConstantPanel.getJob | 语法 | private Job getJob() |
| 前置条件 | add或update方法调用时 |
| 后置条件 | 根据文本框以及其他按钮封装Job对象 |

|  |  |
| --- | --- |
| **需要的服务（需接口）** | |
| 服务名 | 服务 |
| MakeConstantblservice | User业务逻辑接口 |
|  |  |

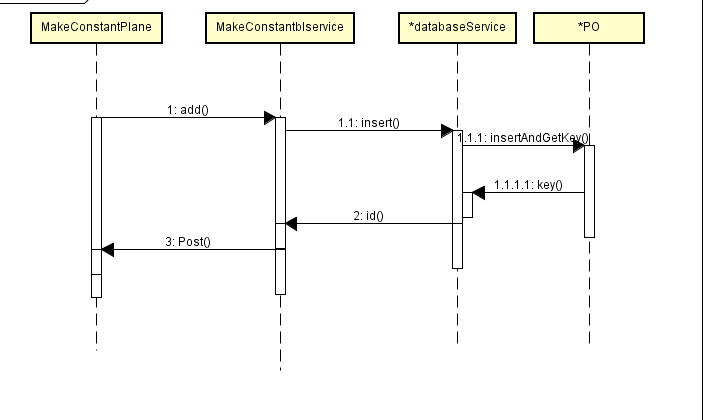
**MakeConstantblservicImpl的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| MakeConstantblservice Impl.getById | 语法 | public Job getById(int id) |
| 前置条件 | 需要根据id查询Job时 |
| 后置条件 | 调用JobDataService的getById方法 |
| MakeConstantblservice Impl.add | 语法 | public Post add(Post post) |
| 前置条件 | 需要新增一个Job时 |
| 后置条件 | 调用JobDataService的insert方法，  获得新增的其他实体类的主键id，设置Job对应的外键的属性的值，调用JobDataService的insert方法，获得新增Job的主键，根据该主键id查询Job并返回这个新增的Job |
| MakeConstantblservice Impl.update | 语法 | public Job update(Job job) |
| 前置条件 | 需要更新Job时 |
| 后置条件 | 调用JobDataService的update方法，  调用JobDataService的update方法，获得返回的Job的主键，根据该主键id查询Job并返回这个Job |
| MakeConstantblservice Impl.delete | 语法 | public boolean delete(int id) |
| 前置条件 | 需要删除一个Job时 |
| 后置条件 | 调用JobDataservice的delete方法 |

|  |  |
| --- | --- |
| **需要的服务（需接口）** | |
| 服务名 | 服务 |
| jobDataservice.getList() | 获得所有的job持久化对象 |
| jobDataservice.getById() | 根据id获得job持久化对象 |
| jobDataservice.insert() | 增加一个job持久化对象 |
| jobDataservice.update() | 更新一个job持久化对象 |
| jobDataservice.delete() | 删除一个job持久化对象 |

### 1.3.3人员机构管理动态模型

以新增Job为例，当用户保存时Job主键为空，则为新增操作，调用add方法，调用MakeConstantblservice的add方法，调用各个Jobservice的insert方法，通过数据库insert每个PO对象，返回其主键id。



### 1.3.4人员机构管理模块设计原理

利用分层的结构，和委托式的控制风格，将其业务逻辑独立出来，通过不同的数据管理对象管理数据，实现了高内聚低耦合的特点。

### 1.3.5度量分析

MakeConstantPanel这个类调用的只有MakeConstantblservice的4个方法，和一个MakeConstantblservice对象引用，

所以CBO为4，DAC为1，继承一个commonpanel，无子类，NOC为0，DIT为1

MakeConstantblservice有JobDataService 工作数据逻辑1个对象引用，以及他们的共5个方法。

所以CBO为5，DAC为1，MakeConstantrblserviceImpl实现该接口，NOC为1，DIT为0