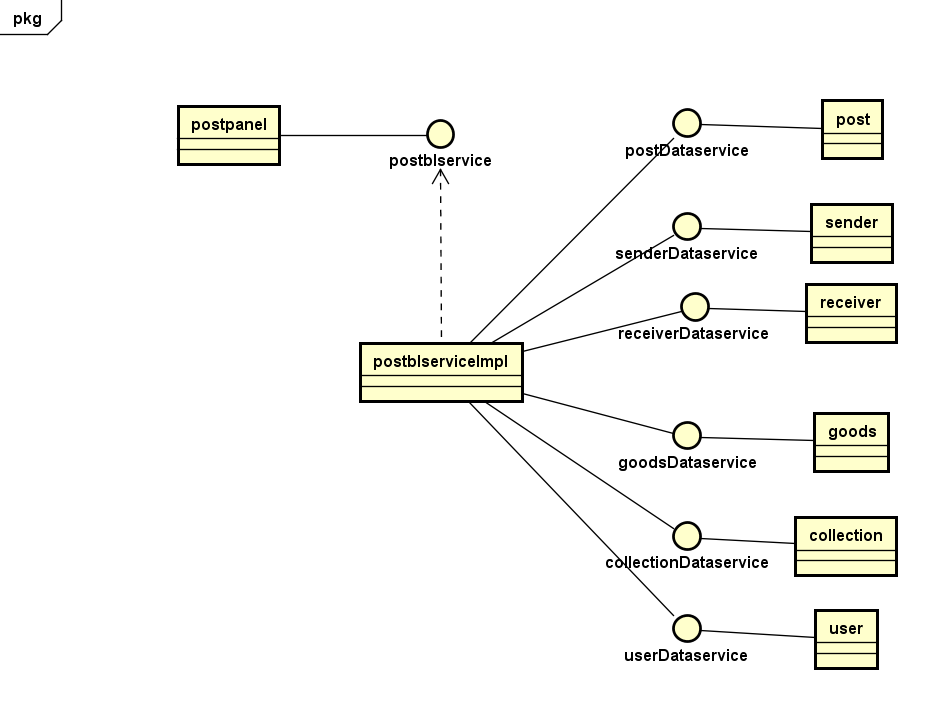
麻烦大家把自己之前写的部分计算下度量

书243页

比如寄件单这个，这是之前写的

### 寄件单详细设计描述



|  |  |
| --- | --- |
| 模块 | 职责 |
| postPanel | post的用户界面 |
| postblserviceImpl | 负责有关post的业务逻辑实现 |
| postblservice | post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post的controller类，而是封装到Panel中

**postPanel的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| PostPanel.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 |
| PostPanel.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post显示在表格中 |
| PostPanel.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，然后将所有的post显示在表格中 |
| 提供的服务（供接口） | | |
| PostPanel.clear | 语法 | private void clear() |
| 前置条件 | 用户点击清空按钮，或者post添加，修改，删除操作完成后 |
| 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 |
| PostPanel.save | 语法 | private void save() |
| 前置条件 | 用户点击保存按钮 |
| 后置条件 | 如果postId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 |
| PostPanel.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postblservice的add方法，然后clear清空 |
| PostPanel.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 |
| PostPanel.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post的id，调用postblservice的delete方法，然后clear清空 |
| PostPanel.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 |
| PostPanel.getSender | 语法 | private Sender getSender() |
| 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sender对象 |
| PostPanel.getReceiver | 语法 | private Receiver getReceiver() |
| 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receiver对象 |
| PostPanel.getGoods | 语法 | private Goods getGoods() |
| 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goods对象 |
| PostPanel.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 |
| postblservice | post业务逻辑接口 |
|  |  |

**postblserviceImpl的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| postblserviceImpl.getAll | 语法 | public Vector<Post> getAll() |
| 前置条件 | 需要查询所有的post时 |
| 后置条件 | 调用postDataService的getList方法，对获得的每一个post调用senderDataService，receiverDataservice，goodsDataservice，collectionDatas的getById方法，对获得的collection调用userDataservice的getById方法 |
| postblserviceImpl.getById | 语法 | public Post getById(int id) |
| 前置条件 | 需要根据id查询post时 |
| 后置条件 | 调用postDataService的getById方法，对获得的post调用senderDataService，receiverDataservice，goodsDataservice，collectionDatas的getById方法，对获得的collection调用userDataservice的getById方法 |
| postblserviceImpl.add | 语法 | public Post add(Post post) |
| 前置条件 | 需要新增一个post时 |
| 后置条件 | 调用senderDataService，receiverDataservice，goodsDataservice，collectionDatas的insert方法，  获得新增的其他实体类的主键id，设置post对应的外键的属性的值，调用postDataService的insert方法，获得新增的post的主键，根据该主键id查询post并返回这个新增的post |
| postblserviceImpl.update | 语法 | public Post update(Post post) |
| 前置条件 | 需要更新post时 |
| 后置条件 | 调用senderDataService，receiverDataservice，goodsDataservice，collectionDatas的update方法，  调用postDataService的update方法，获得返回的post的主键，根据该主键id查询post并返回这个post |
| postblserviceImpl.getCourier | 语法 | public Vector<User> getCourier() |
| 前置条件 | 需要获得post中collection的快递员user时 |
| 后置条件 | 调用userDataservice的getUserOfJob方法，获得返回的user集合，说明：该方法本应该是collectionblservice的方法，但postblservice也需要这个方法，写在这里可以节省代码 |
| postblserviceImpl.delete | 语法 | public boolean delete(int id) |
| 前置条件 | 需要删除一个post时 |
| 后置条件 | 调用postDataservice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 |
|  |  |
| senderDataservice.getList() | 获得所有的sender持久化对象 |
| senderDataservice.getById() | 根据id获得sender持久化对象 |
| senderDataservice.insert() | 增加一个sender持久化对象 |
| senderDataservice.update() | 更新一个sender持久化对象 |
| senderDataservice.delete() | 删除一个sender持久化对象 |
|  |  |
| receiverDataservice.getList() | 获得所有的receiver持久化对象 |
| receiverDataservice.getById() | 根据id获得receiver持久化对象 |
| receiverDataservice.insert() | 增加一个receiver持久化对象 |
| receiverDataservice.update() | 更新一个receiver持久化对象 |
| receiverDataservice.delete() | 删除一个receiver持久化对象 |
|  |  |
| goodsDataservice.getList() | 获得所有的goods持久化对象 |
| goodsDataservice.getById() | 根据id获得goods持久化对象 |
| goodsDataservice.insert() | 增加一个goods持久化对象 |
| goodsDataservice.update() | 更新一个goods持久化对象 |
| goodsDataservice.delete() | 删除一个goods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 |
|  |  |
| 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 |

那么**postpanel**这个类调用的只有postblservice的6个方法，和一个postblservice对象引用，

所以CBO为6，DAC为1，继承一个commonpanel，无子类，NOC为0，DIT为1

**postblservice**有postDataService 寄件单数据逻辑 senderDataService 寄件人数据逻辑

receiverDataService 收件人数据逻辑goodsDataService 货物数据逻辑userDataService 职员数据逻辑collectionDataService 收款单数据逻辑六个对象引用，以及他们的共6\*4=24个方法。

所以CBO为24，DAC为6，postblserviceImpl实现该接口，NOC为1，DIT为0

上次写的东西

https://github.com/xyf18362929560/SoftwareEngineering/tree/master/%E6%96%87%E6%A1%A3/11\_7\_%E7%BB%84%E5%91%98%E4%BD%9C%E4%B8%9A

<https://github.com/xyf18362929560/SoftwareEngineering.git>