4.2.1 AccountData模块

（1）模块概述

AccountData模块承担的需求参见需求规格说明文档功能需求及相关非功能需求

AccountData模块的职责及接口参见软件体系结构描述文档

（2）整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。Account模块对应的接口为展示层与业务逻辑层之间的businesslogicservice.accountblservice.AccountblService接口与业务逻辑层与数据层之间的dataservice.accountdataservice.AccountblService接口。为了隔离数据处理职责和逻辑控制职责，我们增加了AccountDataController，这样AccountDataController会将对账户的业务逻辑处理委托给Account对象。AccountPO是作为银行账户的持久化对象被添加到设计模型中去的。ReceiptPO是作为收款单的持久化对象被添加到设计模型中去的。PaymentPO是作为付款单的持久化对象被添加到设计模型中去的。而AccountManangement保存账户持久化对象，ReceiptManangement负责存储ReceiptPO持久化对象。PaymentManangement负责存储PaymentPO持久化对象。

AccountData模块的设计如图

AccountData模块各个类的职责如表

|  |  |
| --- | --- |
| 模块 | 职责 |
| AccountDataController | 负责实现账户数据管理所需要的服务 |
| AccountData | 账户数据管理信息，完成数据库存的管理与存储 |
| AccountManagement | 账户管理领域模型对象，存储账户信息 |
| ReceiptManagement | 收款单领域模型对象，存储收款单信息 |
| PaymentManagement | 付款单领域模型对象，存储付款单信息 |

（3）模块内部类的接口规范

AccountDataController接口规范：

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| AccountDataController.getAccount() | 语法 | Public AccountPO getAccount() |
| 前置条件 | 已创建一个AccountData领域对象 |
| 后置条件 | 调用AccountData领域的getAccount方法 |
| AccountDataController.addAccount(AccountPO po) | 语法 | Public void addAccount(AccountPO po) |
| 前置条件 | 已创建一个AccountData领域对象，输入符合规则 |
| 后置条件 | 调用AccountData领域的addAccount方法 |
| AccountDataController.delAccount(AccountPO po) | 语法 | Public void delAccount(AccountPO po) |
| 前置条件 | 已创建一个AccountData领域对象，输入符合规则，对象中应收应付为0 |
| 后置条件 | 调用AccountData领域的delAccount方法 |
| AccountDataController.updateAccount(AccountPO po1, AccountPO po2) | 语法 | Public void updateAccount(AccountPO po1, AccountPO po2) |
| 前置条件 | 已创建一个AccountData领域对象，输入符合规则 |
| 后置条件 | 调用AccountData领域的updateAccount方法 |
| AccountDataController.findAccount(String name) | 语法 | Public AccountPO findAccount(String name) |
| 前置条件 | 已创建一个AccountData领域对象，输入符合规则 |
| 后置条件 | 调用AccountData领域的findAccount方法 |
| AccountDataController.getAllAccount() | 语法 | Public AccountPO[] getAllAccount() |
| 前置条件 | 已创建一个AccountData领域对象，输入符合规则 |
| 后置条件 | 调用AccountData领域的getAllAccount方法 |
| AccountDataController.addReceipt(ReceiptPO po) | 语法 | Public void addReceipt(ReceiptPO po) |
| 前置条件 | 收款单得到总经理审批通过，已创建一个AccountData领域对象，输入符合规则 |
| 后置条件 | 调用AccountData领域的addReceipt方法 |
| AccountDataController.addPayment(PaymentPO po) | 语法 | Public void addPayment(PaymentPO po) |
| 前置条件 | 付款单得到总经理审批通过，已创建一个AccountData领域对象，输入符合规则 |
| 后置条件 | 调用AccountData领域的addPayment方法 |
| AccountDataController.getAllReceipt() | 语法 | Public ReceiptPO[] getAllReceipt() |
| 前置条件 | 已创建一个AccountData领域对象，输入符合规则 |
| 后置条件 | 调用AccountData领域的getAllReceipt方法 |
| AccountDataController.getAllPayment() | 语法 | Public PaymentPO[] getAllPayment() |
| 前置条件 | 已创建一个AccountData领域对象，输入符合规则 |
| 后置条件 | 调用AccountData领域的getAllPayment方法 |
| AccountDataController.clear | 语法 | Public void clear() |
| 前置条件 | 进行期初建账之前，已创建一个AccountData领域对象，输入符合规则 |
| 后置条件 | 调用AccountData领域的clear方法 |
| 需要的服务 | | |
| AccountData.getAccount() | 获得当前账户的信息 | |
| AccountData.addAccount(AccountPO po) | 添加账户 | |
| AccountData.delAccount(AccountPO po) | 删除账户 | |
| AccountData.updateAccount(AccountPO po1, AccountPO po2) | 更新账户 | |
| AccountData.findAccount(String name) | 根据关键词查找账户 | |
| AccountData.getAllAccount() | 获得所有的账户对象 | |
| AccountData.addReceipt(ReceiptPO po) | 添加收款单 | |
| AccountData.addPayment(PaymentPO po) | 添加付款单 | |
| AccountData.getAllReceipt() | 获得所有的收款单 | |
| AccountData.getAllPayment() | 获得所有的付款单 | |
| AccountData.clear | 清除所有数据 | |

AccountData接口规范：

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| AccountData.getAccount() | 语法 | Public AccountPO getAccount() |
| 前置条件 | 无 |
| 后置条件 | 返回当前账户AccountPO对象 |
| AccountData.addAccount(AccountPO po) | 语法 | Public void addAccount(AccountPO po) |
| 前置条件 | 数据库中不存在相同名称的AccountPO对象 |
| 后置条件 | 数据库添加AccountPO对象 |
| AccountData.delAccount(AccountPO po) | 语法 | Public void delAccount(AccountPO po) |
| 前置条件 | 数据库中存在相同名称的AccountPO对象，此对象中应付和应收为0 |
| 后置条件 | 数据库删除AccountPO对象 |
| AccountData.updateAccount(AccountPO po1, AccountPO po2) | 语法 | Public void updateAccount(AccountPO po1, AccountPO po2) |
| 前置条件 | 数据库中存在相同名称的AccountPO对象，输入的两个AccountPO对象的名称相同 |
| 后置条件 | 数据库更新AccountPO对象的信息 |
| AccountData.findAccount(String name) | 语法 | Public AccountPO findAccount(String name) |
| 前置条件 | 无 |
| 后置条件 | 根据名称查找并返回AccountPO对象 |
| AccountData.getAllAccount() | 语法 | Public AccountPO[] getAllAccount() |
| 前置条件 | 无 |
| 后置条件 | 返回一个包含所有AccountPO对象的数组 |
| AccountData.addReceipt(ReceiptPO po) | 语法 | Public void addReceipt(ReceiptPO po) |
| 前置条件 | 收款单得到总经理审批通过 |
| 后置条件 | 数据库添加ReceiptPO对象 |
| AccountData.addPayment(PaymentPO po) | 语法 | Public void addPayment(PaymentPO po) |
| 前置条件 | 付款单得到总经理审批通过 |
| 后置条件 | 数据库添加PaymentPO对象 |
| AccountData.getAllReceipt() | 语法 | Public ReceiptPO[] getAllReceipt() |
| 前置条件 | 无 |
| 后置条件 | 返回一个包含所有ReceiptPO对象的数组 |
| AccountData.getAllPayment() | 语法 | Public PaymentPO[] getAllPayment() |
| 前置条件 | 无 |
| 后置条件 | 返回一个包含所有PaymentPO对象的数组 |
| AccountData.clear | 语法 | Public void clear() |
| 前置条件 | 进行期初建账之前 |
| 后置条件 | 清除所有AccountPO，ReceiptPO，PaymentPO对象 |

（4）数据层的动态模型