# 模板

4.x.x xxxbl模块

（1）模块概述

xxxbl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。

xxxbl模块的职责及接口参见软件系统结构描述文档表9.

（2）整体结构

Businesslogicservice.BankManageblservice.BankManageblservice为展示层和业务层之间的接口。Dataservice.BankManagedataservice.BankManagedataservice为业务层和数据层之间的接口。BankController负责逻辑控制职责，BankManage负责业务逻辑职责。BankPO为银行账户的持久化数据对象。

图x xxxbl模块各个类设计

表x xxxbl模块各个类的职责

（3）模块内部类的接口规范

表x xxx的接口规范

（4）业务逻辑层的动态模型

图x XXXX的顺序图

图x xxx对象状态图

（5）业务逻辑层的设计原理

|  |  |
| --- | --- |
| 模块 | 职责 |
|  |  |
|  |  |

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
|  | 语法 |  |
| 前置条件 |  |
| 后置条件 |  |
|  | 语法 |  |
| 前置条件 |  |
| 后置条件 |  |

|  |  |
| --- | --- |
| 需要的服务（需接口） | |
|  |  |

4.x.x BankManagebl模块

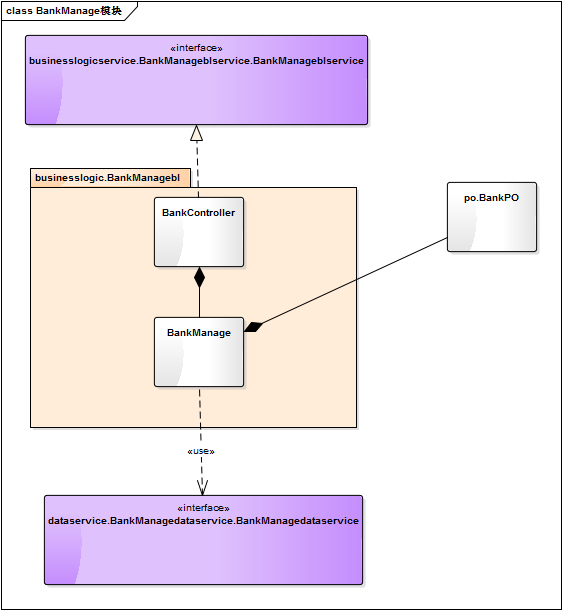
（1）模块概述

BankManagebl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。

BankManagebl模块的职责及接口参见软件系统结构描述文档表x.

（2）整体结构

Businesslogicservice.BankManageblservice.BankManageblservice为展示层和业务层之间的接口。Dataservice.BankManagedataservice.BankManagedataservice为业务层和数据层之间的接口。BankController负责逻辑控制职责，BankManage负责业务逻辑职责。BankPO为银行账户的持久化数据对象。



图x BankManagebl模块个各类的设计

表x BankManagebl模块各个类的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| BankManage | 银行账户管理的领域模型对象，拥有银行账户的属性以及对数据库的引用，辅助完成银行账户管理所需要的服务 |
| BankContr  ller | 负责实现银行账户管理所需要的服务 |

（3）模块内部类的接口规范

BankManage和BankController的接口规范如表x到表x所示。

表x BankController的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| BankController.add | 语法 | Public void add(BankVo vo) throws Exception |
| 前置条件 | 已创建一个BankManage对象，输入符合规则 |
| 后置条件 | 调用BankManage对象的add方法 |
| BankController.delete | 语法 | Public void delete(BankVo vo) throws Exception |
| 前置条件 | 已创建一个BankManage对象，输入符合规则 |
| 后置条件 | 调用BankManage对象的delete方法 |
| BankController.modify | 语法 | Public void modify(BankVo vo) throws Exception |
| 前置条件 | 已创建一个BankManage对象，输入符合规则 |
| 后置条件 | 调用BankManage的modify方法 |
| BankController.search | 语法 | Public BankVo[] search(String field) throws Exception |
| 前置条件 | 已创建一个BankManage对象，输入符合规则 |
| 后置条件 | 调用BankManage的search方法 |

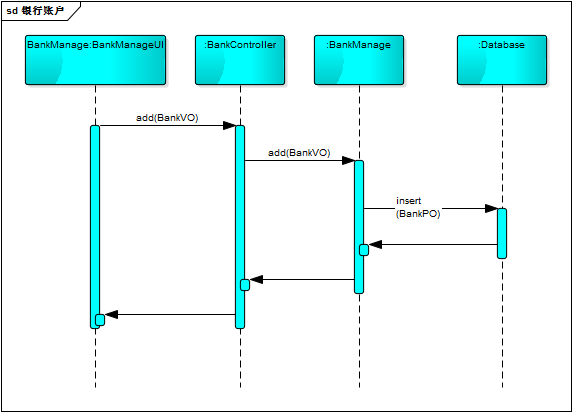
|  |  |
| --- | --- |
| 需要的服务（需接口） | |
| BankManage.add(BankVo vo) | 添加一个银行账户 |
| BankManage.delete(BankVo vo) | 删除一个银行账户 |
| BankManage.modify(BankVo vo) | 更新一个银行账户信息 |
| BankManage.search(String field) | 搜索符合条件的结果 |

表x BankManage的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| BankManage.add | 语法 | Public void add(BankVo vo) throws Exception |
| 前置条件 | 无 |
| 后置条件 | 新增一个银行账户，刷新列表 |
| BankManage.delete | 语法 | Public void delete(BankVo vo) throws Exception |
| 前置条件 | 已选中列表某账户 |
| 后置条件 | 删除一个银行账户 |
| BankManage.modify | 语法 | Public void modify(BankVo vo) throws Exception |
| 前置条件 | 已选中列表某账户 |
| 后置条件 | 更新一个银行账户的信息 |
| BankManage.search | 语法 | Public BankVo[] search(String field) throws Exception |
| 前置条件 | 无 |
| 后置条件 | UI列表中显示符合条件的结果 |

|  |  |
| --- | --- |
| 需要的服务（需接口） | |
| DatabaseFactory.getBankdatabase | 得到bank数据库的服务的引用 |
| BankManageDataService.find(String field) | 根据field查找持久化对象 |
| BankManageDataService.delete(BankPo po) | 删除单一持久化对象 |
| BankManageDataService.update(BankPo po) | 更新单一持久化对象 |
| BankManageDataService.insert(BankPo po) | 插入单一持久化对象 |
| BankManageDataService.init() | 初始化数据库 |
| BankManageDataService.finish() | 结束持久化数据库的使用 |
| BankManageDataService.finds() | 获得数据库所有数据 |
| BankManageDataService.setup(String ID) | 新建一个银行账户数据库 |
| BankManageDataService.getID(String account) | 返回指定帐目的数据库名称 |

（4）业务逻辑层的动态模型



图x 添加银行账户的顺序图

（5）业务逻辑层的设计原理

4.x.x FinancialReceiptbl模块

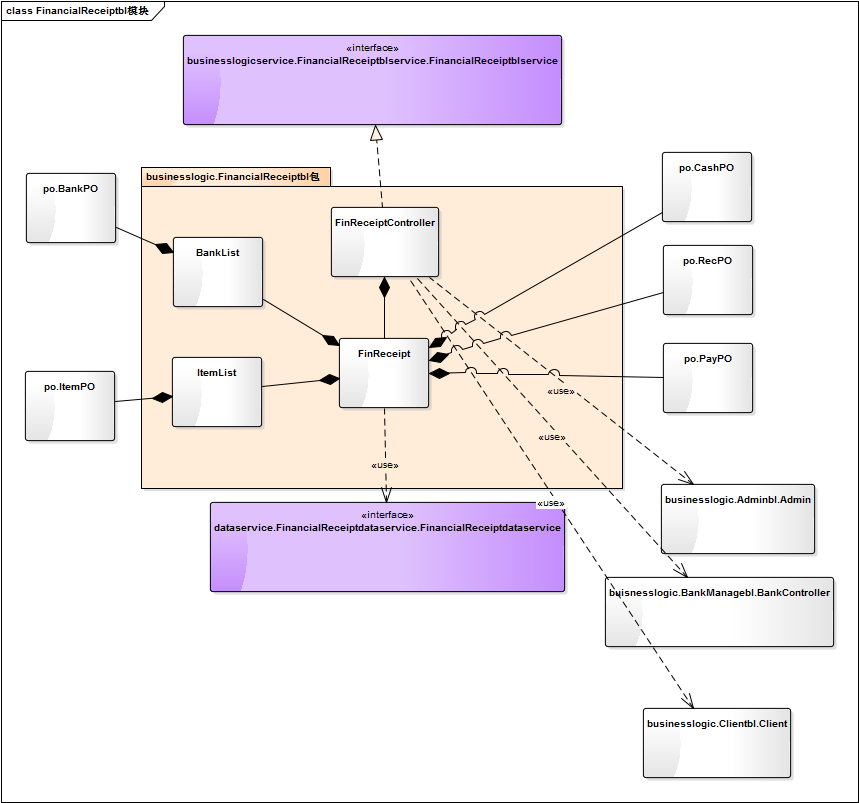
（1）模块概述

FinancialReceiptbl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。

FinancialReceiptbl模块的职责及接口参见软件系统结构描述文档表10.

（2）整体结构

Businesslogicservice.FinancialReceiptblservice.FinancialReceiptblservice为展示层和业务层之间的接口。Dataservice.FinancialReceiptdataservice.FinancialReceiptdataservice为业务层和数据层之间的接口。FinReceiptController负责逻辑控制职责，FinReceipt负责业务逻辑职责。BankList作为银行账户的容器类，BankPO为银行账户的持久化数据对象，这里作为转账的数据对象使用。ItemList作为条目的容器类，ItemPO为条目的持久化数据对象。CashPO,RecPO,PayPO分别为现金费用单，收款单，付款单的持久化数据对象。



图x FinancialReceiptbl模块各个类设计

表x FinancialReceiptbl模块各个类的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| FinReceiptControll  er | 负责实现制定财务单据所需要的服务 |
| FinReceipt | 制定财务单据的领域模型对象，拥有财务单据的属性以及对数据库的引用，辅助完成制定财务单据所需要的服务 |
| BankChangeList | 转账列表，拥有转账信息，辅助完成制定付款单，收款单 |
| ItemList | 条目清单，拥有条目信息，辅助完成制定现金费用单 |

（3）模块内部类的接口规范

表x FinReceiptCpmtroller的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| FinReceiptController.addBank | 语法 | Public double addBank(BankVO vo) |
| 前置条件 | 已创建FinReceipt对象，输入符合规则 |
| 后置条件 | 调用FinReceipt对象的addBank方法 |
| FinReceiptController.addItem | 语法 | Public double addItem(ItemVO vo) |
| 前置条件 | 已创建FinReceipt对象，输入符合规则 |
| 后置条件 | 调用FinReceipt对象的addItem方法 |
| FinReceiptController.deleteBank | 语法 | Public double deleteBank(BankVO vo) |
| 前置条件 | 已创建FinReceipt对象 |
| 后置条件 | 调用FinReceipt对象的deleteBank方法 |
| FinReceiptController.deleteItem | 语法 | Public double deleteItem(ItemVO vo) |
| 前置条件 | 已创建FinReceipt对象，输入符合规则 |
| 后置条件 | 调用FinReceipt对象的deleteItem方法 |
| FinReceiptController.makePayment | 语法 | Public Boolean makePayment(PayVo vo)  throws Exception |
| 前置条件 | 已创建FinReceipt对象，输入符合规则 |
| 后置条件 | 调用FinReceipt对象的makePayment方法 |
| FinReceiptController.makeReceive | 语法 | Public Boolean makeReceive(RecVo vo)  throws Exception |
| 前置条件 | 已创建FinReceipt对象，输入符合规则 |
| 后置条件 | 调用FinReceipt对象的makeReceive方法 |
| FinReceiptController.makeCash | 语法 | Public Boolean makeCash(CashVo vo)  throws Exception |
| 前置条件 | 已创建FinReceipt对象，输入符合规则 |
| 后置条件 | 调用FinReceipt对象的makeCash方法 |

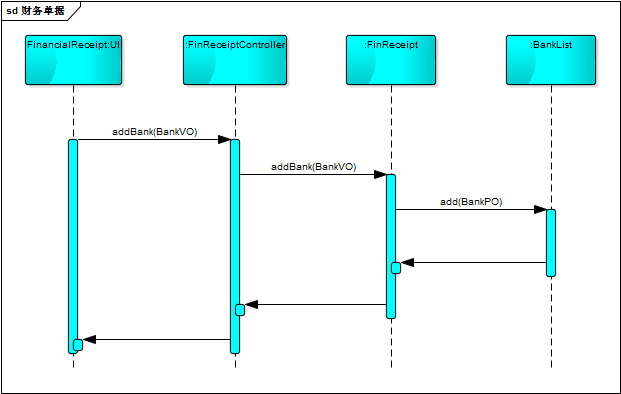
|  |  |
| --- | --- |
| 需要的服务（需接口） | |
| FinReceipt.addBank(BankVO vo) | 添加一条转账信息 |
| FinReceipt.addItem(ItemVO vo) | 添加一条条目信息 |
| FinReceipt.deleteBank(BankVO vo) | 删除一条转账信息 |
| FinReceipt.deleteItem(ItemVO vo) | 删除一条条目信息 |
| FinReceipt.makeCash(CashVO vo) | 生成现金费用单 |
| FinReceipt.makePayment(PayVo vo) | 生成付款单 |
| FinReceipt.makeReceive(RecVo vo) | 生成收款单 |
| BankController.finds() | 获得银行账户列表 |
| Client.queryClient() | get客户列表 |
| Admin.getCurrentUsers | get当前操作员 |

表x FinReceipt的接口规范

|  |  |  |
| --- | --- | --- |
| FinReceipt.addBank | 语法 | Public double addBank(BankVO vo) |
| 前置条件 | 无 |
| 后置条件 | 增加一条转账信息 |
| FinReceipt.addItem | 语法 | Public double addItem(ItemVO vo) |
| 前置条件 | 无 |
| 后置条件 | 增加一条条目信息 |
| FinReceipt.deleteBank | 语法 | Public double deleteBank(BankVO vo) |
| 前置条件 | 有可删除的转账信息 |
| 后置条件 | 删除一条转账信息 |
| FinReceipt.deleteItem | 语法 | Public double deleteItem(ItemVO vo) |
| 前置条件 | 有可删除的条目信息 |
| 后置条件 | 删除一条条目信息 |
| FinReceipt.makePayment | 语法 | Public Boolean makePayment(PayVo vo)  throws Exception |
| 前置条件 | 无 |
| 后置条件 | 生成付款单，提交给总经理 |
| FinReceipt.makeReceive | 语法 | Public Boolean makeReceive(RecVo vo)  throws Exception |
| 前置条件 | 无 |
| 后置条件 | 生成收款单，提交给总经理 |
| FinReceipt.makeCash | 语法 | Public Boolean makeCash(CashVo vo)  throws Exception |
| 前置条件 | 无 |
| 后置条件 | 生成现金费用单，提交给总经理 |

|  |  |
| --- | --- |
| 需要的接口（需接口） | |
| FinancialReceiptDataService.addPay(PayPo po) | 添加一个持久化对象PayPo |
| FinancialReceiptDataService.addRec(RecPo po) | 添加一个持久化对象RecPo |
| FinancialReceiptDataService.addCash(CashPo po) | 添加一个持久化对象CashPo |
| FinancialReceiptDataService.update(Receipt  Po[] receipt) | 更新数据库持久化对象信息 |

（4）业务逻辑层的动态模型



图x 向转账列表中添加银行账户的顺序图

（5）业务逻辑层的设计原理

4.x.x RunningTablebl模块

（1）模块概述

RunningTablebl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。

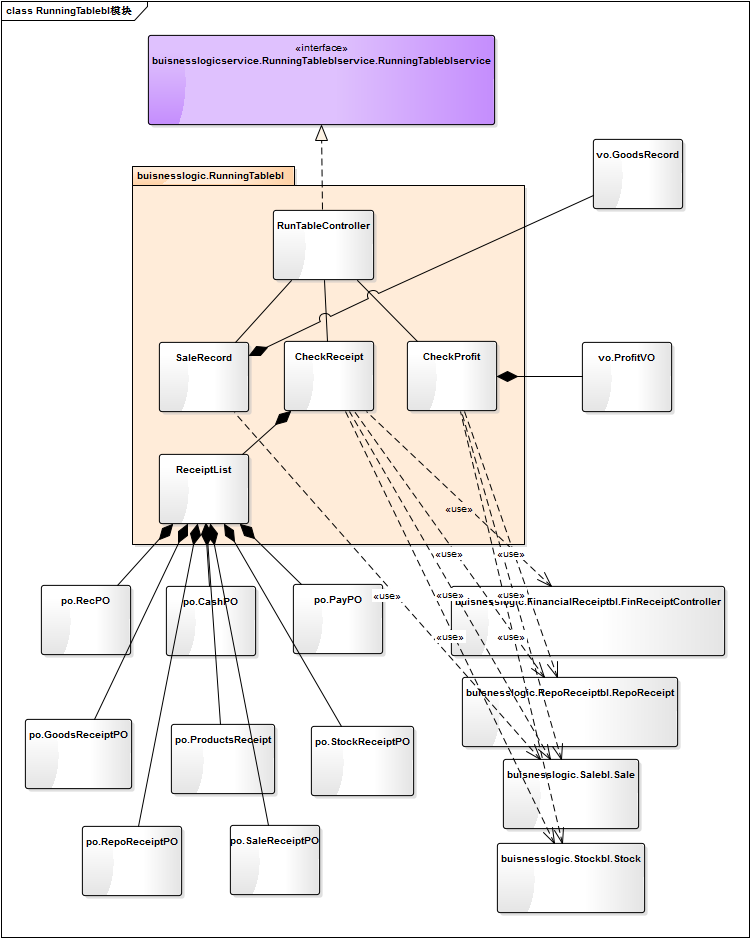
RunningTablebl模块的职责及接口参见软件系统结构描述文档表11.

（2）整体结构

Businesslogicservice.RunningTableblservice.RunningTableblservice为展示层和业务层之间的接口。RunTableController负责逻辑控制职责，业务逻辑职责由SaleRecord,

CheckReceipt,CheckProfit来共同实现。GoodsRecord为销售记录的临时对象，ProfitVO为利润的临时对象，ReceiptList为所有单据的容器类，负责处理单据显示的信息。RecPO,CashPO

,PayPO,GoodsRceiptPO,ProductsReceipt,StockReceiptPO,RepoReceiptPO,SaleReceiptPO为各种单据的持久化数据对象。



图x RunningTablebl模块各个类设计

表x RunningTablebl模块各个类的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| RunTableController | 负责实现查看经营表格的所需要的服务 |
| SaleRecord | 销售明细，可以对商品销售明细进行查询，辅助实现查看经营表格的功能 |
| CheckReceipt | 查看单据，可以对所有单据进行查询，并提供红冲功能，辅助实现查看经营表格的功能 |
| CheckProfit | 收支，可以计算一段时间内的收支状况，辅助时间查看经营表格的功能 |
| ReceiptList | 单据列表，可以统计单据信息，进行单据分类，控制不同单据显示的内容，辅助实现查看经营历程表的功能 |

（3）模块内部类的接口规范

表x RunTableController的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| RunTableController.getSaleTable | 语法 | Public GoodsRecord[] getSaleTable ( SaleConditionVo vo) throws Exception |
| 前置条件 | SaleRecord对象已被创建,条件符合规则 |
| 后置条件 | 调用SaleRecord对象的getSaleTable方法 |
| RunTableController.getReceipt | 语法 | Public ReceiptList getReceipt(ReceiptConditionVo vo) throws Exception |
| 前置条件 | CheckReceipt对象已被创建 |
| 后置条件 | 调用CheckReceipt对象的getReceipt方法 |
| RunTableController.getProfit | 语法 | Public profitVo getProfit(TimeConditionVo vo)  throws Exception |
| 前置条件 | CheckProfit对象已被创建，条件符合规则 |
| 后置条件 | 调用CheckProfit对象的getProft方法 |
| RunTableController.CreditNote | 语法 | Public void CreditNote(ReceiptVo vo)  throws Exception |
| 前置条件 | CheckReceipt对象已被创建 |
| 后置条件 | 调用CheckReceipt对象的CreditNote方法 |

|  |  |
| --- | --- |
| 需要的服务（需接口） | |
|  |  |

表x SaleRecord的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| SaleRecord.getSaleTable | 语法 | Public GoodsRecord[] getSaleTable()  throws Exception |
| 前置条件 | 无 |
| 后置条件 | 返回符合条件的销售情况 |

|  |  |
| --- | --- |
| 需要的服务（需接口） | |
| Sale.queryProduct(SaleConditionVo Vo) | 给定条件返回某一商品销售记录（条件：时间区间，商品名，客户，业务员，仓库） |

表x CheckReceipt的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| CheckReceipt.getReceipt | 语法 | Public ReceiptList getReceipt(ReceiptConditionVo vo) throws Exception |
| 前置条件 | 无 |
| 后置条件 | 返回符合条件的单据 |
| CheckReceipt.CreditNote | 语法 | Public void CreditNote(ReceiptVo vo)  throws Exception |
| 前置条件 | 该单据可建立红冲 |
| 后置条件 | 创建红冲单 |

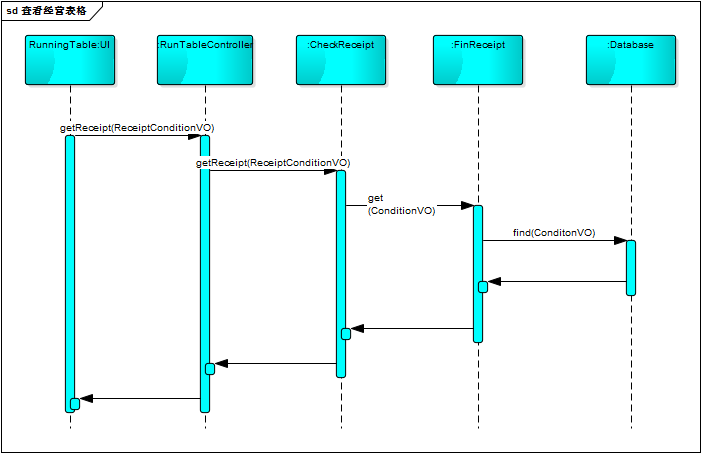
|  |  |
| --- | --- |
| 需要的服务（需接口） | |
| FinReceipt.get(ReceiptConditionVo vo) | 给定条件返回单据（条件：时间区间，客户，操作员） |
| Stock.queryReceipt(ReceiptFilter[] filters) | 给定条件返回单据（条件：时间区间，客户，操作员，仓库） |
| RepoReceipt.find(ReceiptConditionVo Vo) | 给定条件返回单据（条件：时间区间，客户，操作员，仓库） |
| Sale.queryReceipt(ReceiptFilter[] filters) | 给定条件返回单据（条件：时间区间，客户，操作员，仓库） |
| FinReceipt.creat(ReceiptVo vo) | 生成财务类单据 |
| Sale.makeReceipt(SaleReceiptVO receipt) | 生成销售单据 |
| Stock.makeReceipt(StockReceiptVo receipt) | 生成进货单据 |

表x CheckProfit的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| CheckProfit.getProfit | 语法 | Public profitVo getProfit(TimeConditionVo vo)  throws Exception |
| 前置条件 | 无 |
| 后置条件 | 返回收支情况 |
| CheckProfit.checkSaleAndStock | 语法 | Public void checkSaleAndStock()  throws Exception |
| 前置条件 | 无 |
| 后置条件 | 查询进货销售单，进行计算 |
| CheckProfit.checkOverFlow | 语法 | Public void checkOverFlow()  throws Exception |
| 前置条件 | 无 |
| 后置条件 | 查询报溢单，进行计算 |
| CheckProfit.checkGift | 语法 | Public void checkGift()  throws Exception |
| 前置条件 | 无 |
| 后置条件 | 查询赠送单，进行计算 |
| CheckProfit.checkBreak | 语法 | Public void checkBreak()  throws Exception |
| 前置条件 | 无 |
| 后置条件 | 查询报损单，进行计算 |

|  |  |
| --- | --- |
| 需要的服务（需接口） | |
| Stock.queryReceipt(ReceiptFilter[] filters) | 给定条件返回单据（条件：时间区间，客户，操作员，仓库） |
| RepoReceipt.find(ReceiptConditionVo Vo) | 给定条件返回单据（条件：时间区间，客户，操作员，仓库） |
| Sale.queryReceipt(ReceiptFilter[] filters) | 给定条件返回单据（条件：时间区间，客户，操作员，仓库） |

（4）业务逻辑层的动态模型



图x 查看财务单据的顺序图

（5）业务逻辑层的设计原理

4.x.x SetAccountbl模块

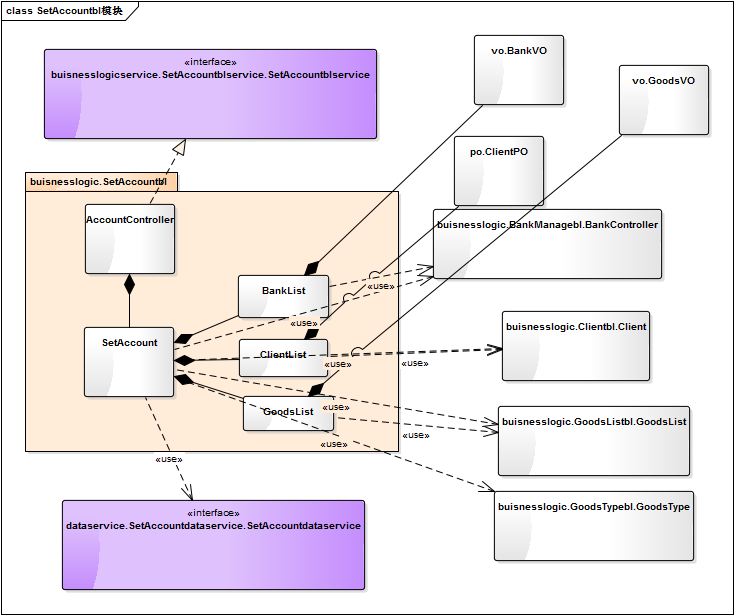
（1）模块概述

SetAccountbl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。

SetAccountbl模块的职责及接口参见软件系统结构描述文档表12.

（2）整体结构

Businesslogicservice.SetAccountblservice. SetAccountblservice为展示层和业务层之间的接口。Dataservice.SetAccountdataservice.SetAccountataservice为业务层和数据层之间的接口。AccountController负责逻辑控制职责，SetAccount负责业务逻辑职责。BankList为银行账户的容器类，BankVO为银行账户的数据对象。ClientList为客户的容器类，ClientPO为客户的数据对象,GoodSList为商品的容器类，GoodsVO为商品的数据对象



图x SetAccountbl模块各个类设计

表x SetAccountbl模块各个类的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| AccountController | 负责实现期初建账的服务 |
| SetAccount | 期初建账的对象，拥有对数据库的索引，客户列表，商品列表，银行账户列表，辅助实现期初建账的功能 |
| BankList | 银行账户列表，拥有银行账户信息，可检测前一个账目的的银行账户 |
| GoodsList | 商品列表，拥有商品信息，可检测前一个账目的商品 |
| ClientList | 客户列表，拥有客户信息，可检测前一个账目的客户 |

（3）模块内部类的接口规范

表x AccountController的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| AccountController.inherit | 语法 | Public void inherit(String name , String account Name ) |
| 前置条件 | 已创建SetAccount对象 |
| 后置条件 | 调用SetAccount对象的inherit方法 |
| AccountController.SetGoodsType | 语法 | Public void setGoodsType() throws Exception |
| 前置条件 | 已创建SetAccount对象 |
| 后置条件 | 调用SetAccount对象的setGoodsType方法 |
| AccountController.creatGoods | 语法 | Public void creatGoods(GoodsVo[] vo,String name) throws Exception |
| 前置条件 | 已创建SetAccount对象,分类设置完毕 |
| 后置条件 | 调用SetAccount对象的creatGoods方法 |
| AccountController.creatClient | 语法 | Public void creatClient(String name)  throws Exception |
| 前置条件 | 已创建SetAccount对象，商品添加完毕 |
| 后置条件 | 调用SetAccount对象的creatClient方法 |
| AccountController.CreatBank | 语法 | Public void CreatBank(String name)  throws Exception |
| 前置条件 | 已创建SetAccount对象，客户添加完毕 |
| 后置条件 | 调用SetAccount对象的CreatBank方法 |
| AccountController.setup | 语法 | Public void setup(String name) throws Exception |
| 前置条件 | 已创建SetAccount对象 |
| 后置条件 | 调用SetAccount对象的setup方法 |
| AccountController.checkGoodsType | 语法 | Public void checkGoodsType(String name) throws Exception |
| 前置条件 | 已创建SetAccount对象 |
| 后置条件 | 调用SetAccount对象的checkGoodsType方法 |
| AccountController.checkGoods | 语法 | Public GoodsVo[] checkGoods(String name)  throws Exception |
| 前置条件 | 已创建SetAccount对象 |
| 后置条件 | 调用SetAccount对象的checkGoods方法 |
| AccountController.checkClient | 语法 | Public ClientVo[] checkClient(String name)  throws Exception |
| 前置条件 | 已创建SetAccount对象 |
| 后置条件 | 调用SetAccount对象的checkClent方法 |
| AccountController.checkBank | 语法 | Public BankVo[] checkBank(String name)  throws Exception |
| 前置条件 | 已创建SetAccount对象 |
| 后置条件 | 调用SetAccount对象的checkBank方法 |
| AccountController.addGoods | 语法 | Public GoodsVo addGoods(GoodsVo vo)  throws Exception |
| 前置条件 | 已创建SetAccount对象 |
| 后置条件 | 调用SetAccount对象的addGoods方法 |
| AccountController.delGoods | 语法 | Public boolean delGoods(GoodsVo vo)  throws Exception |
| 前置条件 | 已创建SetAccount对象 |
| 后置条件 | 调用SetAccount对象的delGoods方法 |
| AccountController.addClient | 语法 | Public ClientVO addClient(ClientVO vo)  throws Exception |
| 前置条件 | 已创建SetAccount对象 |
| 后置条件 | 调用SetAccount对象的addClient方法 |
| AccountController.delClient | 语法 | Public boolean delClient(ClientVO vo)  throws Exception |
| 前置条件 | 已创建SetAccount对象 |
| 后置条件 | 调用SetAccount对象的delCLient方法 |
| AccountController.addBank | 语法 | Public BankVo addBank(BankVO vo)  throws Exception |
| 前置条件 | 已创建SetAccount对象 |
| 后置条件 | 调用SetAccount对象的addBank方法 |
| AccountController.delBank | 语法 | Public boolean delBank(BankVO vo)  throws Exception |
| 前置条件 | 已创建SetAccount对象 |
| 后置条件 | 调用SetAccount对象的delBank方法 |

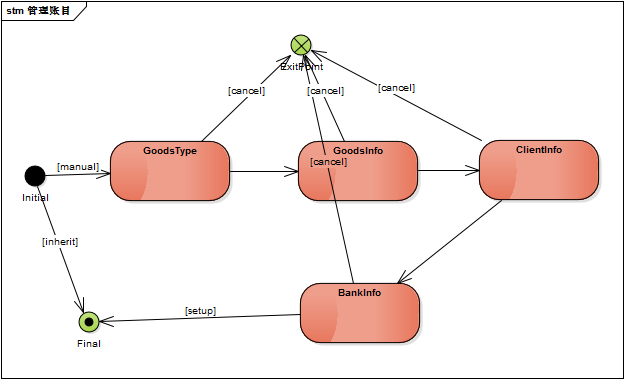
|  |  |
| --- | --- |
| 需要的服务（需接口） | |
| SetAccount.inherit | 继承该账目所有信息 |
| SetAccount.setGoodsType | 建立（或更新）新的商品类型数据库 |
| SetAccount.creatGoods | 建立（或更新）新的商品信息数据库 |
| SetAccount.creatClient | 建立(或更新)新的客户信息数据库 |
| SetAccount.creatBank | 建立（或更新）新的银行账户数据库 |
| SetAccount.setup | 保存期初数据，更新系统默认账目 |
| SetAccount.checkGoodsType | 显示期初商品分类 |
| SetAccount.checkGoods | 显示期初商品信息 |
| SetAccount.checkClient | 显示期初客户信息 |
| SetAccount.checkBank | 显示期初银行账户信息 |
| SetAccount.addGoods | 添加一个商品 |
| SetAccount.delGoods | 删除一个商品 |
| SetAccount.addClient | 添加一个客户 |
| SetAccount.delClient | 删除一个客户 |
| SetAccount.addBank | 添加一个银行账户 |
| SetAccount.delBank | 删除一个银行账户 |

表x SetAccount的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| SetAccount.inherit | 语法 | Public void inherit(String name , String account Name ) |
| 前置条件 | 系统已有过往账目 |
| 后置条件 | 继承该账目所有信息 |
| SetAccount.setGoodsTy  Pe | 语法 | Public void setGoodsType() throws Exception |
| 前置条件 | 无 |
| 后置条件 | 建立（或更新）新的商品类型数据库 |
| SetAccount.CreatGoods | 语法 | Public void CreatGoods(String name) throws Exception |
| 前置条件 | 已设置商品类型 |
| 后置条件 | 建立（或更新）新的商品信息数据库 |
| SetAccount.CreatClient | 语法 | Public void CreatClient(String name)  throws Exception |
| 前置条件 | 已添加商品信息完毕 |
| 后置条件 | 建立(或更新)新的客户信息数据库 |
| SetAccount.CreatBank | 语法 | Public void CreatBank(String name)  throws Exception |
| 前置条件 | 已添加客户信息完毕 |
| 后置条件 | 建立（或更新）新的银行账户数据库 |
| SetAccount.setUp | 语法 | Public void setUp(String name) throws Exception |
| 前置条件 | 新账目信息已填写完毕 |
| 后置条件 | 保存期初数据，更新系统默认账目 |
| SetAccount.checkGoodsType | 语法 | Public void checkGoodsType(String name) throws Exception |
| 前置条件 | 无 |
| 后置条件 | 显示期初商品分类 |
| SetAccount.checkGoods | 语法 | Public GoodsVo[] checkGoods(String name)  throws Exception |
| 前置条件 | 无 |
| 后置条件 | 显示期初商品信息 |
| SetAccount.checkClient | 语法 | Public ClientVo[] checkClient(String name)  throws Exception |
| 前置条件 | 无 |
| 后置条件 | 显示期初客户信息 |
| SetAccount.checkBank | 语法 | Public BankVo[] checkBank(String name)  throws Exception |
| 前置条件 | 无 |
| 后置条件 | 显示期初银行账户信息 |
| SetAccount.addGoods | 语法 | Public GoodsVo addGoods(GoodsVo vo)  throws Exception |
| 前置条件 | 无 |
| 后置条件 | 添加一个商品 |
| SetAccount.delGoods | 语法 | Public boolean delGoods(GoodsVo vo)  throws Exception |
| 前置条件 | 无 |
| 后置条件 | 删除一个商品 |
| SetAccount.addClient | 语法 | Public ClientVO addClient(ClientVO vo)  throws Exception |
| 前置条件 | 无 |
| 后置条件 | 添加一个客户 |
| SetAccount.delClient | 语法 | Public boolean delClient(ClientVO vo)  throws Exception |
| 前置条件 | 无 |
| 后置条件 | 删除一个客户 |
| SetAccount.addBank | 语法 | Public BankVo addBank(BankVO vo)  throws Exception |
| 前置条件 | 无 |
| 后置条件 | 添加一个银行账户 |
| SetAccount.delBank | 语法 | Public boolean delBank(BankVO vo)  throws Exception |
| 前置条件 | 无 |
| 后置条件 | 删除一个银行账户 |

|  |  |
| --- | --- |
| 需要的接口（需接口） | |
| DatabaseFactory.getAccountdatabase | 获得Account数据库服务的引用 |
| SetAccountDataService.SetUp(string[] name) | 保存所有期初数据库文件 |
| SetAccountDataService.Goods() | 查看期初所有商品信息 |
| SetAccountDataService.Bank() | 查看期初所有银行账户信息 |
| SetAccountDataService.Client() | 查看期初所有客户信息 |
| SetAccountDataService.GoodsType(  ) | 查看期初所有商品类别信息 |
| Client.makeTable(String account) | 传递账目名，新建客户数据库 |
| BankManage.setup(String  account ) | 传递账目名，新建银行账户数据库 |
| GoodsList.initialize(String account) | 传递账目名，新建商品分类数据库，商品信息数据库 |
| Client.queryTableName(String account) | 传递账目名，获得客户数据库名称 |
| BankManage.getData(String account) | 传递账目名，获得银行账户数据库名称 |
| GoodsList.getDatabase(String account) | 传递账目名，获得商品信息数据库名称 |

（4）业务逻辑层的动态模型



图x 期初建账的状态图

（5）业务逻辑层的设计原理