快递物流信息系统

详细设计文档

编写人：陶子涵

谭琼

徐文杰

徐家逸

编写时间：2015.11.04

更新历史 3

1. 引言 4

1.1编制目的 4

1.2词汇表 4

1.3参考资料 4

2. 产品概述 4

3. 体系结构概述 4

4. 结构视角 5

4.1业务逻辑层的分解 5

4.1.1 corporationbl模块 5

4.1.1.1模块概述 5

4.1.1.2整体结构 5

**4.1.1.3模块内部类的接口规范** 6

**4.1.1.4业务逻辑层的动态模型** 12

**4.1.1.5业务逻辑的设计原理** 16

4.1.2 cosntbl模块 16

4.1.2.1模块概述 16

4.1.2.2整体结构 17

4.1.2.3模块内部类的接口规范 18

4.1.2.4业务逻辑层的动态模型 20

4.1.2.5业务逻辑的设计原理 22

4.1.3 financebl模块 23

4.1.3.1模块概述 23

4.1.3.2整体结构 23

4.1.3.3模块内部类的接口规范 25

4.1.3.4业务逻辑层的动态模型 44

4.1.3.5业务逻辑的设计原理 55

4.1.4 userbl模块 56

4.1.4.1模块概述 56

4.1.4.2整体结构 56

4.1.4.3模块内部类的接口规范 57

4.1.4.4业务逻辑层的动态模型 61

4.1.4.5业务逻辑的设计原理 62

4.1.5 inventorybl模块 62

4.1.5.1模块概述 62

4.1.5.2整体结构 62

4.1.5.3模块内部类的接口规范 63

4.1.5.4业务逻辑层的动态模型 72

4.1.5.5业务逻辑的设计原理 72

4.1.6 transitbl模块 72

4.1.6.1模块概述 72

4.1.6.2整体结构 72

4.1.6.3模块内部类的接口规范 75

4.1.6.4业务逻辑层的动态模型 105

4.1.6.5业务逻辑的设计原理 119

4.1.7 formapprovalbl模块 120

**4.1.7.1模块概述** 120

**4.1.7.2整体结构** 120

**4.1.7.3模块内部类的接口规范** 122

**4.1.7.4业务逻辑层的动态模型** 128

**4.1.7.5业务逻辑的设计原理** 129

4.2界面层的分解 129

4.2.1.导航设计 129

4.2.1.1快递员 129

4.2.1.2营业厅业务员 133

4.2.1.3中转中心业务员 140

4.2.1.4中转中心仓库管理人员 144

4.2.1.5财务人员 151

4.2.1.6总经理 161

4.2.1.7管理员 168

4.2.2.界面设计 171

4.2.2.1快递员 171

4.2.2.2营业厅业务员 172

4.2.2.3中转中心业务员 173

4.2.2.4中转中心仓库管理人员 174

4.2.2.5财务人员 174

4.2.2.6管理员 175

4.2.2.7参与者主界面 176

4.2.3.界面原型化 177

4.2.3.1快递员 177

4.2.3.2营业厅业务员 181

4.2.3.3中转中心业务员 194

4.2.3.4中转中心仓库管理人员 198

4.2.3.5财务人员 205

4.2.3.6总经理 217

4.2.3.7管理员 225

4.2.3.8登录 230

# 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| 更新时间 | 更新者 | 更新原因 | 版本号 |
|  |  |  |  |

# 1. 引言

## 1.1编制目的

本报告详细完成对快递物流系统的详细设计，达到指导后续软件构造的目的，同时实现和测试人员及用户的沟通。

本报告面向开发人员、测试人员及最终用户而编写，是了解系统的导航。

## 1.2词汇表

|  |  |  |
| --- | --- | --- |
| **词汇名称** | **词汇含义** | **备注** |
| \_ui | 表示某展示层 |  |
| \_bl | 表示某逻辑层 |  |
| \_data | 表示某数据层 |  |

## 1.3参考资料

1. IEEE Std 1016-1998 《IEEE Recommended Practice for Software Design Descriptions》

2.《快递物流系统体系结构设计文档v1.7》

3.丁二玉，刘钦.计算与软件工程（卷二）[M]机械工业出版社，2012：134—182

# 2. 产品概述

《参考快递物流系统用例文档v1.0》和《快递物流系统软件需求规格说明文档》中对产品的概括描述。

# 3. 体系结构概述

参考《快递物流系统体系结构设计文档1.7》中对体系结构设计的概述。

# 4. 结构视角

## 4.1业务逻辑层的分解

### 4.1.1 corporationbl模块

#### 4.1.1.1模块概述

Corporationbl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。

Corporationbl模块的职责及接口参见软件体系结构描述文档v1.8表5.3.2-5

#### 4.1.1.2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。比如展示层和业务逻辑层之间，我们添加businesslogicservice.corporationblservice.CarInfocorporationblService接口和businesslogicservice.corporationblservice.DriverInfocorporationblService接口。业务逻辑层和数据层之间添加dataservice.coporationdataservice.CarInfocorporationdataService接口和dataservice.corporationdataService.DriverInfodataService接口。为了隔离业务逻辑职责，我们增加了CarInfoController和DriverInfoController，这样CarInfoController和DriverInfoController会将对公司信息（包括车辆信息和司机信息）的业务逻辑处理委托给CarInfoCorporation对象和DriverInfoCorporation对象。DriverInfo和CarInfo是作为公司信息的车辆信息和司机信息的持久化对象被添加到设计模型中去的。

Corpoartionbl模块的设计如图4.1.1-1所示。

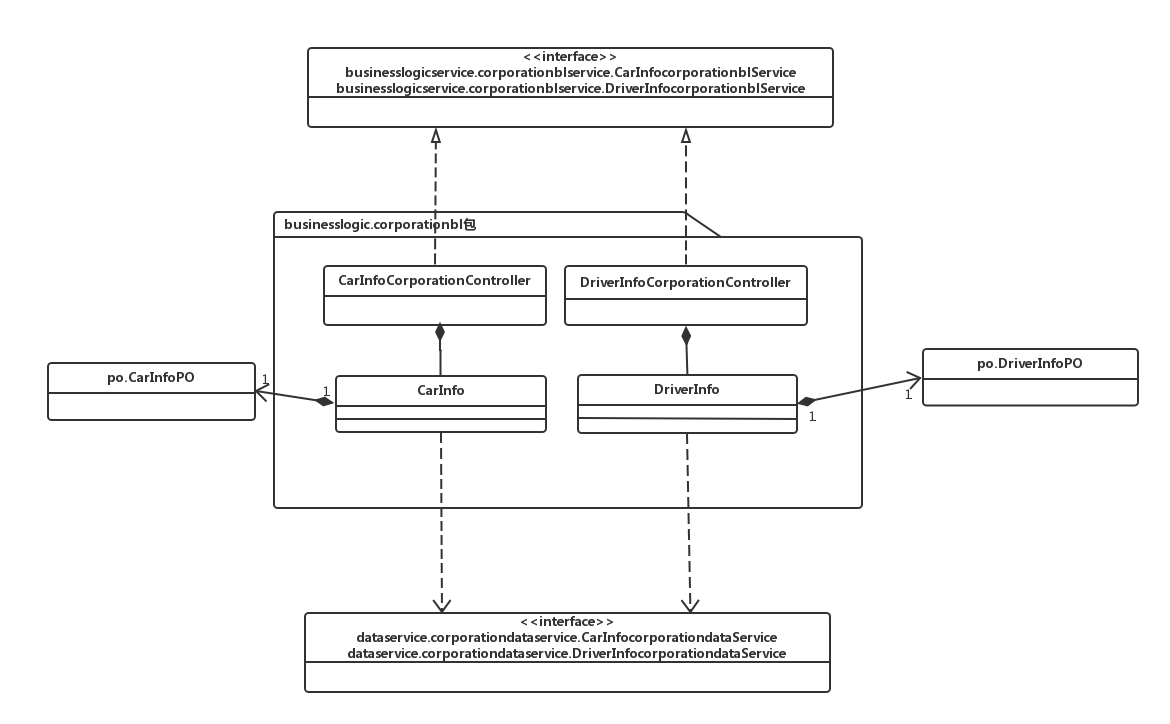


图4.1.1-1 coporationbl模块各个类的设计

Corporationbl模块各个类的职责如表4.1.1-2所示。

**表4.1.1-1 Corporationbl模块各个类的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| CarInfoController | 负责实现维护车辆信息 |
| CarInfo | 维护公司信息的领域模型对象，可以帮助完成车辆信息维护所需要的服务 |
| DriverInfoController | 负责实现维护司机信息 |
| DriverInfo | 维护司机信息的领域模型，可以帮助完成司机信息维护界面所需要的服务 |

#### **4.1.1.3模块内部类的接口规范**

**表4.1.1-2 CarInfoController的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| CarInfoController.create | | 语法 | public CarInfoVO create () |
| 前置条件 | 已创建一个CarInfo领域对象 |
| 后置条件 | 调用CarInfo领域对象的create方法 |
| CarInfoController.add | | 语法 | public boolean add (CarInfoVO vo) |
| 前置条件 | 已创建一个CarInfo领域对象 |
| 后置条件 | 调用CarInfo领域对象的CarInfoAdd方法 |
| CarInfoController.delete | | 语法 | public boolean delete () |
| 前置条件 | 已创建一个CarInfo领域对象 |
| 后置条件 | 调用CarInfo领域对象的delete方法 |
| CarInfoController.find | | 语法 | public CarInfoVO find (String carNumber) |
| 前置条件 | 已创建一个CarInfo领域对象，并且输入符合输入规则 |
| 后置条件 | 调用CarInfo领域对象的find方法 |
| CarInfoController.update | | 语法 | public CarInfoVO update () |
| 前置条件 | 已创建一个CarInfo领域对象 |
| 后置条件 | 调用CarInfo领域对象的update方法 |
| CarInfoController.cancel | | 语法 | public boolean cancel() |
| 前置条件 | 已创建一个CarInfo领域对象 |
| 后置条件 | 调用CarInfo领域对象的addInfo方法 |
| CarInfoController.addInfo | | 语法 | public boolean addInfo (String carNumber,  String plateNumber,  String engineNUmber,  String chassisNumber,  String buyTime,  String activeTime) |
| 前置条件 | 已创建一个CarInfo领域对象 |
| 后置条件 | 调用CarInfo领域对象的addInfo方法 |
| **需要的服务（需接口）** | | | |
| 服务名 | 服务 | | |
| CarInfo.create() | 选择增加车辆信息 | | |
| CarInfo.add(CarInfoVO vo) | 确认增加车辆信息 | | |
| CarInfo.delete(CarInfoVO vo) | 删除车辆信息 | | |
| CarInfo.find(String carNumber) | 查找车辆信息 | | |
| CarInfo.update() | 更新车辆信息 | | |
| CarInfo.cancel() | 取消添加车辆信息任务 | | |
| CarInfo.addInfo() | 添加车辆信息 | | |

**表4.1.1-3 DriverInfoController的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| DriverInfoController. create | | 语法 | public DriverInfoVO create () |
| 前置条件 | 已创建一个DriverInfo领域对象 |
| 后置条件 | 调用DriverInfo领域对象的create方法 |
| DriverInfoController. add | | 语法 | public boolean ddd (DriverInfoVO vo) |
| 前置条件 | 已创建一个DriverInfo领域对象 |
| 后置条件 | 调用DriverInfo领域对象的ddd方法 |
| DriverInfoController. delete | | 语法 | public boolean delete () |
| 前置条件 | 已创建一个DriverInfo领域对象 |
| 后置条件 | 调用DriverInfo领域对象的delete方法 |
| DriverInfoController. find | | 语法 | public DriverInfoVO find (String driverNumber) |
| 前置条件 | 已创建一个DriverInfo领域对象，并且输入符合输入规则 |
| 后置条件 | 调用DriverInfo领域对象的find方法 |
| DriverInfoController. update | | 语法 | public DriverInfoVO update () |
| 前置条件 | 已创建一个DriverInfo领域对象 |
| 后置条件 | 调用DriverInfo领域对象的update方法 |
| DriverInfoController. cancel | | 语法 | public booolean cancel () |
| 前置条件 | 已创建一个DriverInfo领域对象 |
| 后置条件 | 调用DriverInfo领域对象的cancel方法 |
| DriverInfoController. addInfo | | 语法 | public DriverInfoVO DriverInfoAddInfo () |
| 前置条件 | 已创建一个DriverInfo领域对象 |
| 后置条件 | 调用DriverInfo领域对象的addInfo方法 |
| **需要的服务（需接口）** | | | |
| 服务名 | 服务 | | |
| DriverInfo.create() | 选择增加司机信息 | | |
| DriverInfo.add(CarInfoVO vo) | 确认增加司机信息 | | |
| DriverInfo.delete(CarInfoVO vo) | 删除司机信息 | | |
| DriverInfo.find(String carNumber) | 查找司机信息 | | |
| DriverInfo.update() | 更新司机信息 | | |
| DriverInfo.cancel() | 取消更新司机信息 | | |
| DriverInfo.addnfo() | 添加司机信息 | | |

**表4.1.1-4 CarInfo的接口规范**

|  |
| --- |
| **提供的服务（供接口）** |

|  |  |  |
| --- | --- | --- |
| CarInfo.create | 语法 | public CarInfoVO create() |
| 前置条件 | 无 |
| 后置条件 | 增加一个CarInfoVO对象 |
| CarInfo.add | 语法 | public boolean add(CarInfoVO vo) |
| 前置条件 | 不存在相同的VO对象 |
| 后置条件 | 无 |
| CarInfo.delete | 语法 | public boolean delete() |
| 前置条件 | 存在相同的VO对象 |
| 后置条件 | 删除一个CarInfoVO对象 |
| CarInfo.find | 语法 | public CarInfoVO find(String plateNumber) |
| 前置条件 | 输入正确的车牌号 |
| 后置条件 | 返回一个CarInfoVO对象 |
| CarInfo.update | 语法 | public CarInfoVO update() |
| 前置条件 |  |
| 后置条件 | 返回一个CarInfoVO |
| CarInfo.cancel | 语法 | public boolean cancel() |
| 前置条件 | 处于编辑状态 |
| 后置条件 | 取消车辆信息添加任务 |
| CarInfo.addInfo | 语法 | public boolean addInfo(String carNumber;  String plateNumber，  String engineNUmber，  String chassisNumber，  String buyTime，  String activeTime) |
| 前置条件 | 添加正确的车辆信息 |
| 后置条件 | 无 |

|  |  |
| --- | --- |
| **需要的服务（需接口）** | |
| DatabaseFactory.getCoporationDatabase | 得到coporation数据库的服务的引用 |
| CorporationDataService. carInfoCorparationdata.delete(carInfoPO po) | 删除单一持久化对象 |
| CorporationDataService. carInfoCorparationdata.insert(carInfoPO po) | 插入单一持久化对象 |
| CorporationDataService. carInfoCorparationdata.update(carInfoPO po) | 更新单一持久化对象 |
| CorporationDataService. carInfoCorparationdata.find(long carNumber) | 查找单一持久化对象 |

**表4.1.1-5 DriverInfo的接口规范**

|  |
| --- |
| **提供的服务（供接口）** |

|  |  |  |
| --- | --- | --- |
| DriverInfo.create | 语法 | public DriverInfoVO create() |
| 前置条件 | 无 |
| 后置条件 | 增加一个DriverInfoVO对象 |
| DriverInfo.add | 语法 | public boolean add(DriverInfoVO vo) |
| 前置条件 | 不存在相同的VO对象 |
| 后置条件 | 无 |
| DriverInfo.delete | 语法 | public boolean delete(DriverInfoVO vo) |
| 前置条件 | 存在相同的VO对象 |
| 后置条件 | 删除一个DriverInfoVO对象 |
| DriverInfo.find | 语法 | public DriverInfoVO find(long driverNumber) |
| 前置条件 | 输入正确的银行账户名 |
| 后置条件 | 返回一个DriverInfoVO对象 |
| DriverInfo.update | 语法 | public DriverInfoVO update() |
| 前置条件 | 无 |
| 后置条件 | 返回一个DriverInfoVO |
| DriverInfo.cancel | 语法 | public boolean cancel() |
| 前置条件 | 处于编辑状态 |
| 后置条件 | 取消车辆信息添加任务 |
| DriverInfo.addInfo | 语法 | public boolean addInfo(**long** driverNumber，  String name，  String birthday，  String idcard，  long cellphone，  String licensedate，  SexList sex) |
| 前置条件 | 输入正确的司机信息 |
| 后置条件 | 无 |

|  |  |
| --- | --- |
| **需要的服务（需接口）** | |
| DatabaseFactory.getCoporationDatabase | 得到coporation数据库的服务的引用 |
| CorporationDataService. dariverInfoCorparationdata.find(long driverNumber) | 查找单一持久化对象 |
| CorporationDataService. dariverInfoCorparationdata.insert(DriverInfoPO po) | 插入单一持久化对象 |
| CorporationDataService. dariverInfoCorparationdata.delete(DriverInfoPO po) | 删除单一持久化对象 |
| CorporationDataService. dariverInfoCorparationdata.update(DriverInfoPO po) | 更新单一持久化对象 |

#### **4.1.1.4业务逻辑层的动态模型**

1）DriverInfo

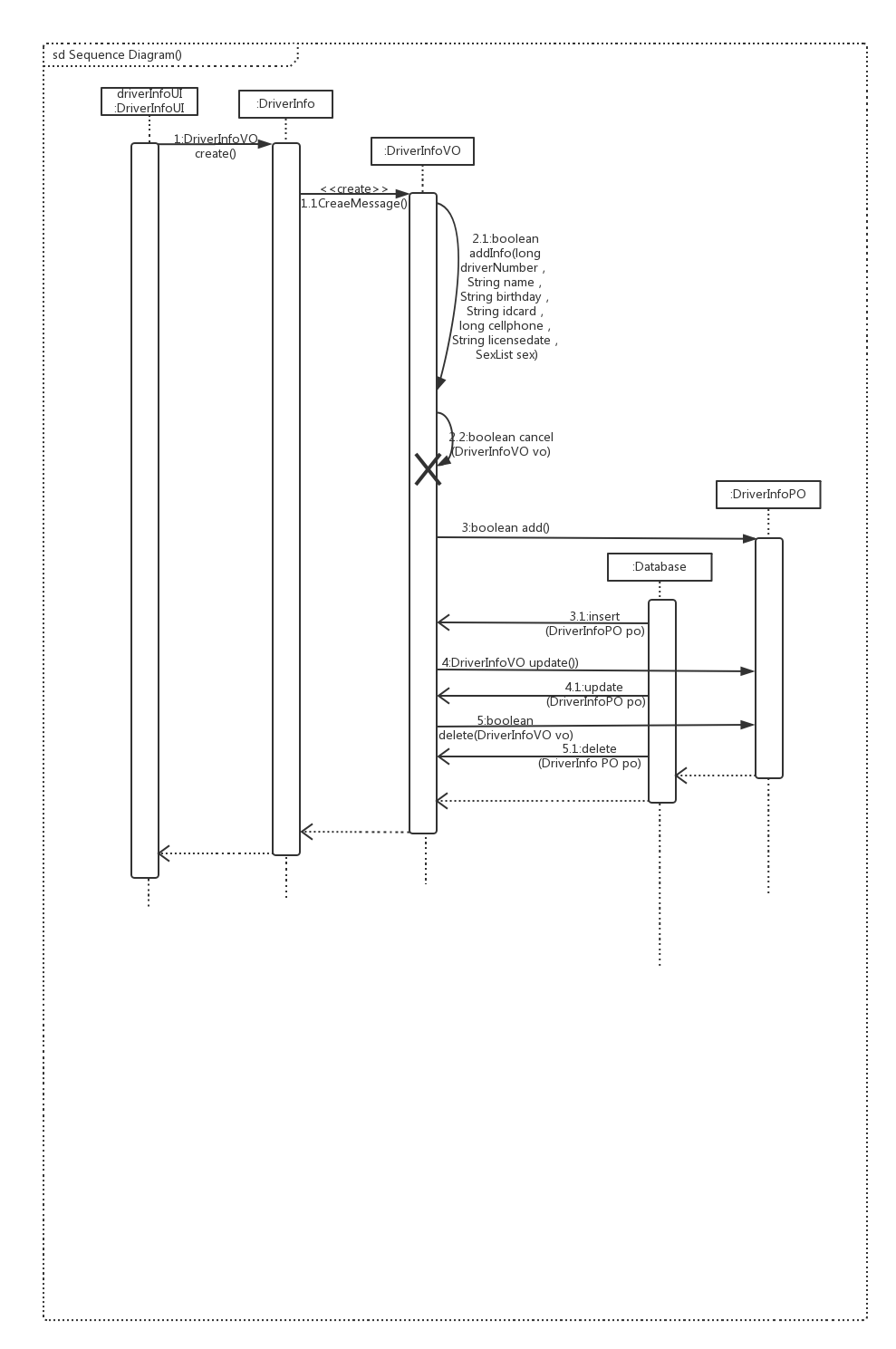


图4.1.1.4-1 DriverInfo\_Add and Delete

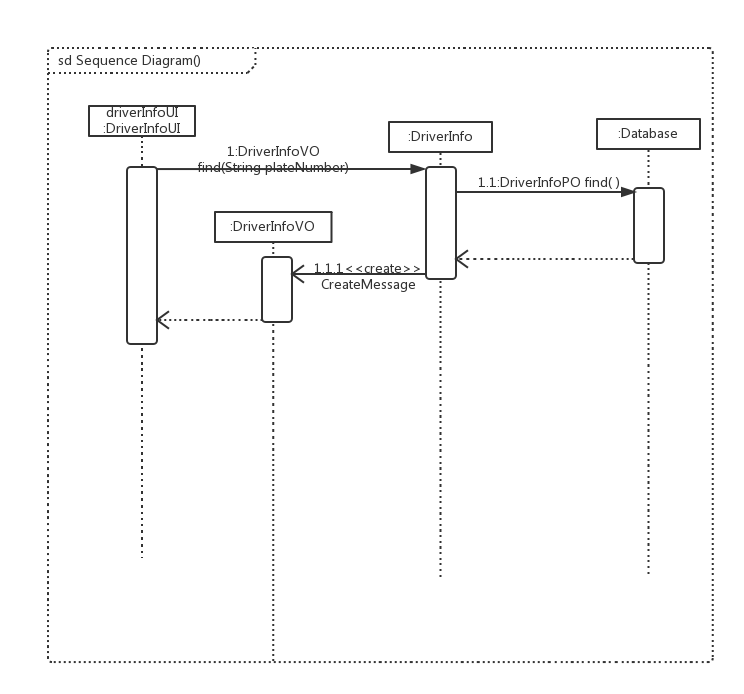


图4.1.1.4-2DriverInfo\_Find

2)CarInfo

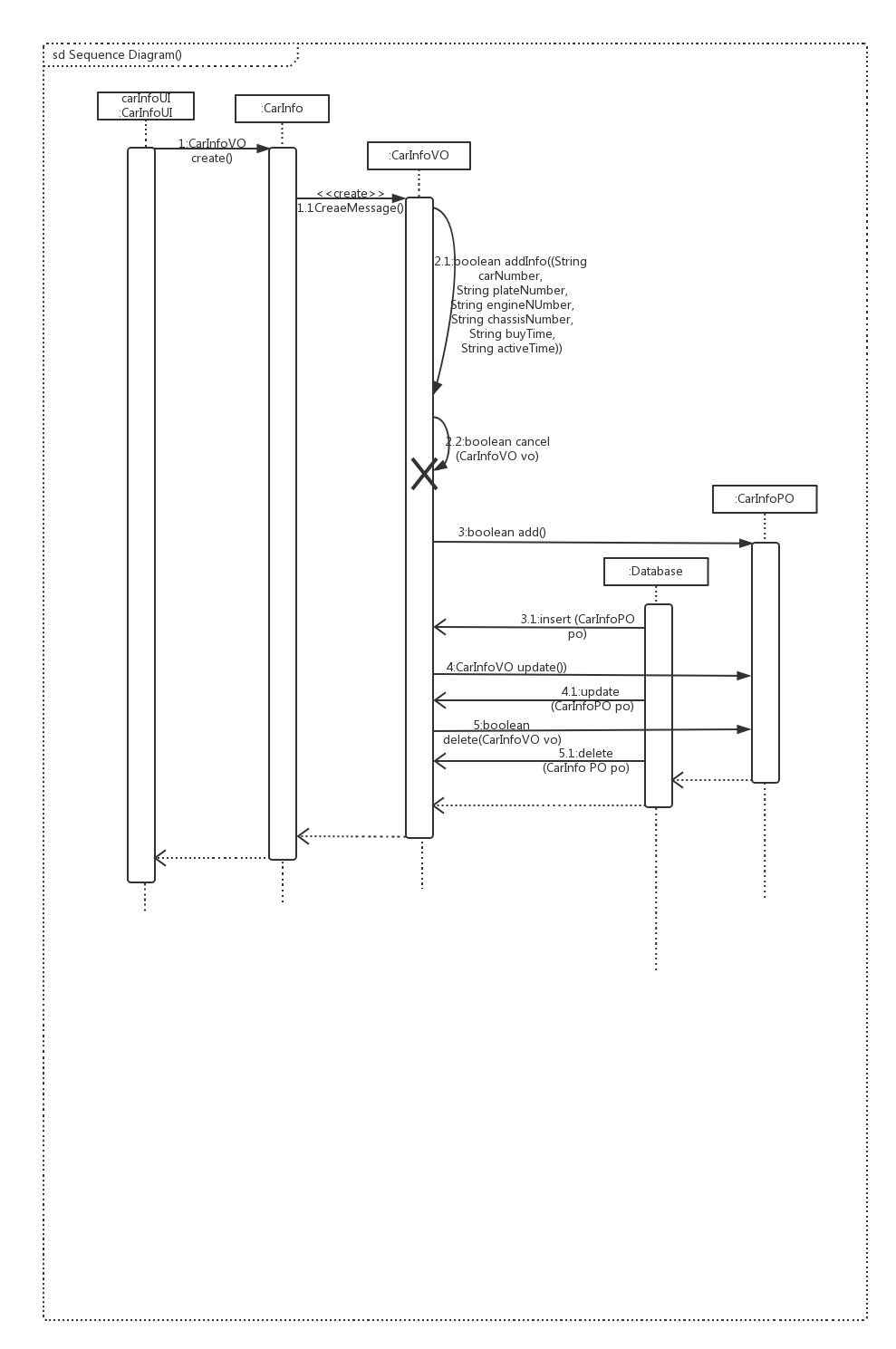


图4.1.1.1-4 CarInfo\_Add and Delete

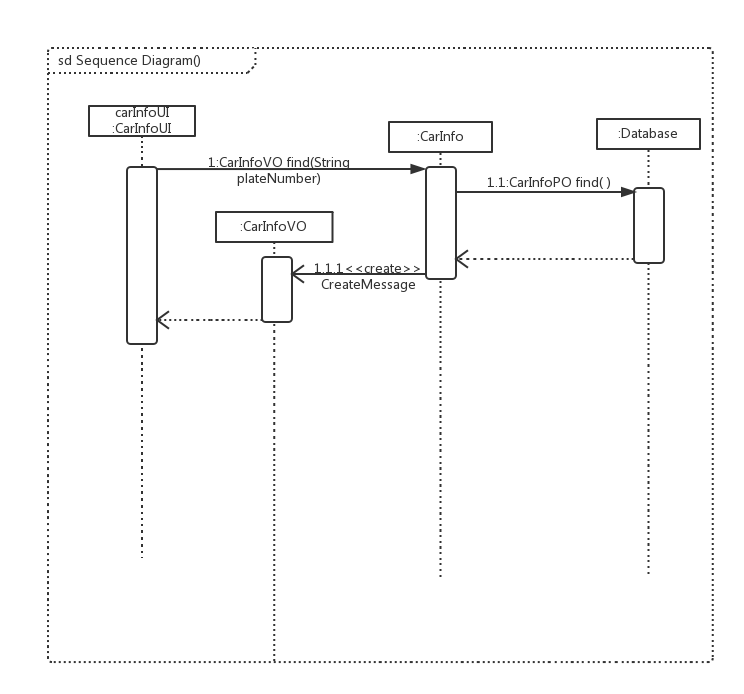


图4.1.1.4.4 CarInfo\_Find

#### **4.1.1.5业务逻辑的设计原理**

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的对象。

### 4.1.2 cosntbl模块

#### 4.1.2.1模块概述

Constbl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。

Constbl模块的职责及接口参见软件体系结构描述文档v1.8表5.3.2-3

#### 4.1.2.2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。比如展示层和业务逻辑层之间，我们添加businesslogicservice.corporationblservice.CarInfocorporationblService接口和businesslogicservice.corporationblservice.DriverInfocorporationblService接口。业务逻辑层和数据层之间添加dataservice.coporationdataservice.CarInfocorporationdataService接口和dataservice.corporationdataService.DriverInfodataService接口。为了隔离业务逻辑职责，我们增加了CarInfoController和DriverInfoController，这样CarInfoController和DriverInfoController会将对公司信息（包括车辆信息和司机信息）的业务逻辑处理委托给CarInfoCorporation对象和DriverInfoCorporation对象。DriverInfo和CarInfo是作为公司信息的车辆信息和司机信息的持久化对象被添加到设计模型中去的。

Corpoartionbl模块的设计如图4.1.2-2所示。

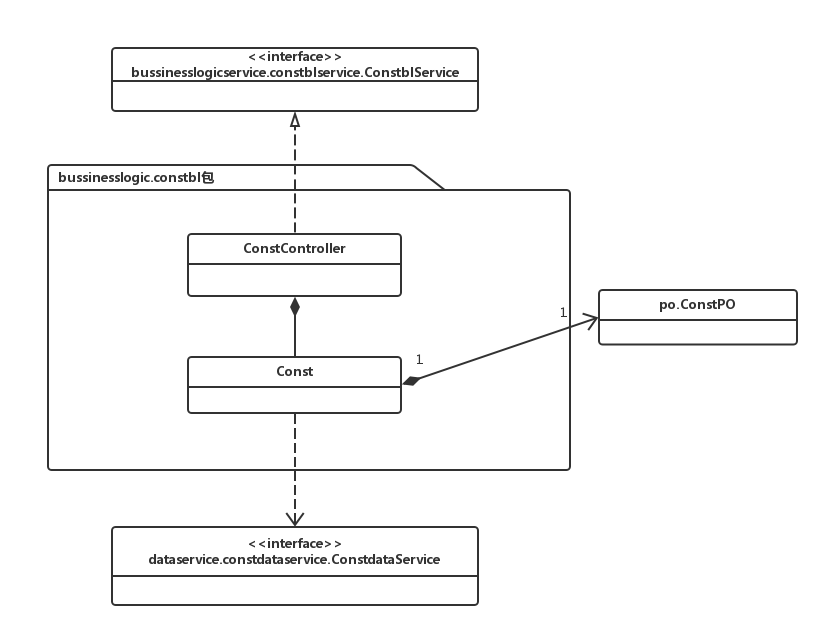


图4.1.2-1 constbl模块各个类的设计

Constbl模块各个类的职责如表4.1.2-2所示。

**表4.1.2-1 constbl模块各个类的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| ConstController | 负责制定距离价格常量和薪资标准 |
| Const | 维护公司常量信息的领域模型对象，可以帮助完成制定距离价格常量和薪资标准界面所需要的服务 |

#### 4.1.2.3模块内部类的接口规范

ConstController和Const的接口规范如表4.1.2-2和表4.1.2-3所示。

**表4.1.2-2 ConstController的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| ConstController.find | 语法 | public ConstVO find() | |
| 前置条件 | 已创建一个Const领域对象 | |
| 后置条件 | 调用Const领域对象的find方法 | |
| ConstController.update | 语法 | public ConstVO update() | |
| 前置条件 | 已创建一个Cosnt领域对象 | |
| 后置条件 | 调用Const领域对象的update方法 | |
| ConstController.setSalary | 语法 | public boolean setSalary(double basesalary,double receiveExpressPrize,double sendExpressPrize, double salaryEachInCity, double salaryEachOutCity) | |
| 前置条件 | 已创建一个Const领域对象 | |
| 后置条件 | 调用Const领域对象的setSalary方法 | |
| ConstController.setPrice | 语法 | public boolean setPrice(double distancePrice) | |
| 前置条件 | 已创建一个Const领域对象 | |
| 后置条件 | 调用Const领域对象的serPrice方法 | |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| Cons.find() | 查找常量信息 | |
| Const.update() | 更新常量信息 | |
| Const. setSalary(double basesalary,double receiveExpressPrize, double sendExpressPrize, double salaryEachInCity, double salaryEachOutCity) | 设置薪资标准 | |
| const.setPrice(double distancePrice) | 设置价格常量 | |

**表4.1.2-3 Const的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| Cons.find | 语法 | public ConstVO find() |
| 前置条件 | 无 |
| 后置条件 | 返回一个ConstVO |
| Const.update | 语法 | public ConstVO update() |
| 前置条件 | 无 |
| 后置条件 | 返回一个ConstVO |
| const.setSalary | 语法 | public boolean setSalary(double basesalary,double receiveExpressPrize, double sendExpressPrize, double salaryEachInCity, double salaryEachOutCity) |
| 前置条件 | 启动制定薪资标准任务 |
| 后置条件 | 在制定薪资标准任务中增加薪资调整信息 |
| const.setPrice | 语法 | public boolean setPrice(double distancePrice) |
| 前置条件 | 启动制定价格常量任务 |
| 后置条件 | 在制定价格常量任务中增加价格信息 |
| **需要的服务（需接口）** | | |
| DatabaseFactory.getConstDatabase | 得到Const数据库的服务的引用 | |
| ConstDataService.ConstInfoConstdata.update(constPO po) | 更新单一持久化constPO对象 | |
| ConstDataService. ConstInfoConstdata.find() | 查找单一持久化constPO对象 | |

#### 4.1.2.4业务逻辑层的动态模型

1）const

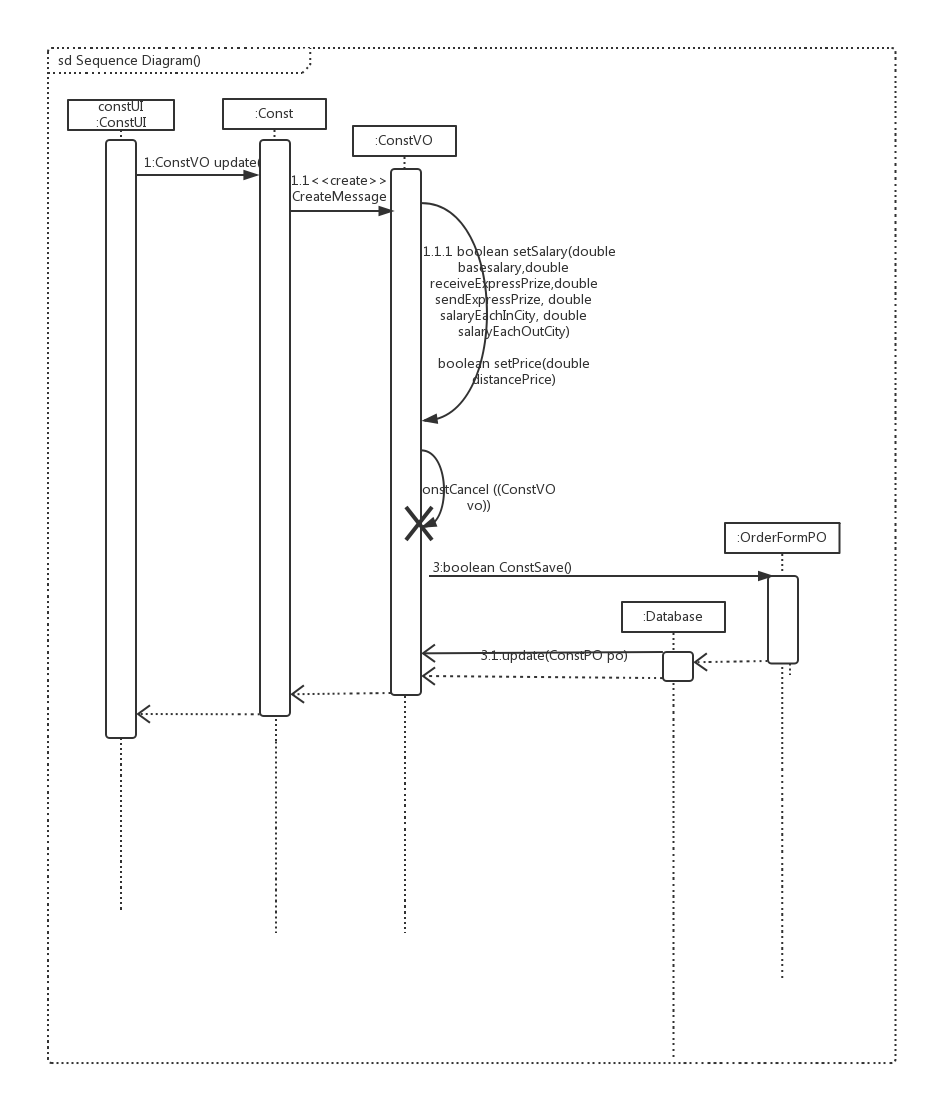


图4.1.2.4-1Const\_update

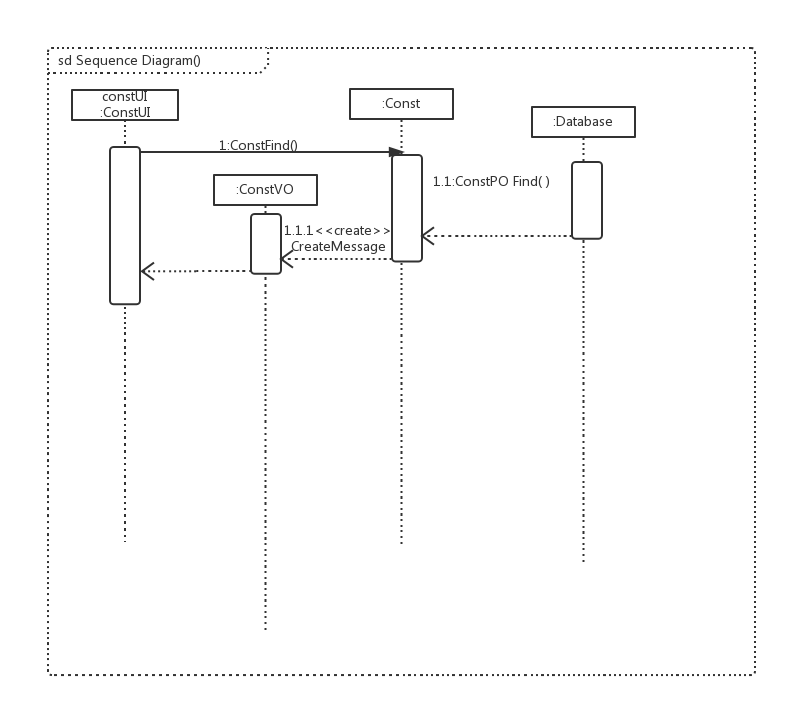


图4.1.2.4-1Const\_find

#### 4.1.2.5业务逻辑的设计原理

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的对象。

### 4.1.3 financebl模块

#### 4.1.3.1模块概述

Financebl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。

Financebl模块的职责及接口参见软件体系结构描述文档v1.8表5.3.2-4

#### 4.1.3.2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。比如展示层和业务逻辑层之间，我们添加businesslogicservice.financeblservice.AccountInfoFinanceblService

businesslogicservice.financeblservice.PaymentFormFinanceblService businesslogicservice.financeblservice.ReceiptFormFinanceblService businesslogicservice.financeblservice.ReceiptGatherFinanceblService businesslogicservice.financeblservice.CostPayChartFinanceblService

businesslogicservice.financeblservice.BeginningAccountFinanceblService businesslogicservice.financeblservice.ProfitChartFinanceblService接口。业务逻辑层和数据层之间添加dataservice.financedataservice.AccountInfodataService dataservice.financedataService.BeginningAccountdataService dataservice.financedataService.CostPayChartdataService dataservice.financedataService.PaymentFormdataService dataservice.financedataService.ReceiptFormdataService接口。为了隔离业务逻辑职责，我们增加了AccountInfoController PaymentFormController ReceiptFormController ReceiptGatherController CostPayChartController BeginningAccountController ProfitChartController这些将对财务信息的业务逻辑处理委托给这些Controller所对应的对象。AccountInfo PaymentForm ReceiptForm ReceiptGather BeginningAccount 和 CostPayChart是作为财务信息的持久化对象被添加到设计模型中去的。

Financebl模块的设计如图4.1.3-1所示。

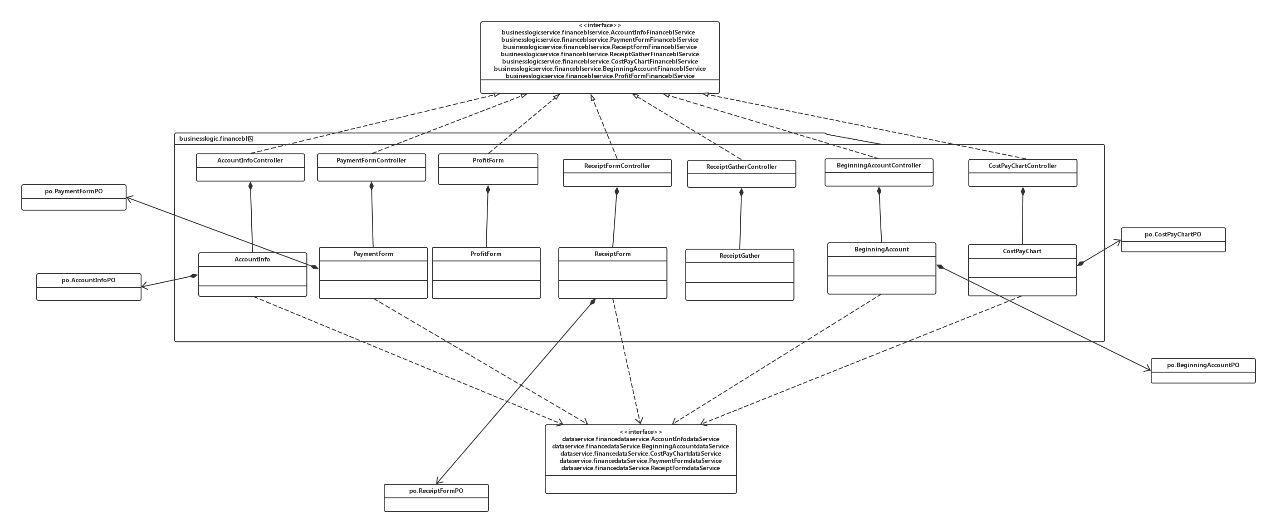


图4.1.3-1 financebl模块各个类的设计

Financebl模块各个类的职责如表4.1.4-2所示。

**表4.1.3-2 financebl模块各个类的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| AccountInfoController | 负责实现对应银行账号信息所需要的服务 |
| AccountInfo | 银行账号的领域模块对象，拥有账号名，密码，余额，可以帮助完成增删改查账户 |
| PaymentFormController | 负责实现对应付款单信息所需要的服务 |
| PaymentForm | 付款单的领域模块对象，拥有付款单的信息，可以帮助完成生产付款单界面操作 |
| ReceiptFormController | 负责实现生成收款单信息所需要的服务 |
| ReceiptForm | 收款单的领域模块对象，拥有时间日期金额收款人等的信息，可以帮助完成生成收款单界面操作 |
| ReceiptGatherController | 负责实现收款单汇总信息所需要的服务 |
| ReceiptGather | 收款单汇总的领域模块对象，拥有收款单信息，日期信息，营业厅信息，可以帮助完成收款单汇总界面操作 |
| BeginningAccountController | 负责实现期初建账所需要的服务 |
| BeginningAccount | 期初建账的领域模块对象，拥有用户信息车辆信息库存信息和账目信息，可以帮助完成期初建账界面操作 |
| CostPayChartController | 负责实现成本收益表信息所需要的服务 |
| CostPayChart | 成本收益表的领域模块对象，拥有总收入，总支出和总利润三个数据，可以帮助完成成本收益表所需界面 |
| ProfitChartController | 负责实现经营情况表信息所需要的服务 |
| ProfitChart | 经营情况表的领域模块对象，拥有日期，地区和收款单付款单的信息，可以帮助完成经营情况表所需界面 |

#### 4.1.3.3模块内部类的接口规范

接口规范如表4.1.3-3所示

**表4.1.3-3a AccountInfoController的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| AccountInfoController.find | | 语法 | public AccountInfoVO find(long bankAccount) |
| 前置条件 | 已创建一个AccountInfo领域对象，并且输入符合输入规则 |
| 后置条件 | 调用AccountInfo领域对象的find方法 |
| **需要的服务（需接口）** | | | |
| 服务名 | 服务 | | |
| AccountInfo.find(long bankAccount) | 选择要查看的账户信息 | | |

**表4.1.3-3b AccountInfo的接口规范**

|  |
| --- |
| **提供的服务（供接口）** |

|  |  |  |
| --- | --- | --- |
| AccountInfo.find | 语法 | public AccountInfoVO find(long bankAccount) |
| 前置条件 | 输入正确的银行账户名 |
| 后置条件 | 返回一个AccountInfoVO对象 |

|  |  |
| --- | --- |
| **需要的服务（需接口）** | |
| DatabaseFactory.getFinanceDatabase | 得到Finance数据库的服务的引用 |
| FinanceDataService. accountInfoCorparationdata.find (long bankAccount) | 查看单一持久化对象 |

**表4.1.4-3c PaymentFormController的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| PaymentFormController.PaymentFormCancel | | 语法 | public void PaymentFormCancel (PaymentFormVO vo) |
| 前置条件 | 已创建一个PaymentForm领域对象， |
| 后置条件 | 调用PaymentForm领域对象的PaymentFormCancel方法 |
| PaymentFormController.PaymentFormSave | | 语法 | public boolean PaymentFormSave (PaymentFormVO vo) |
| 前置条件 | 已创建一个PaymentForm领域对象， |
| 后置条件 | 调用PaymentForm领域对象的PaymentFormSave方法 |
| PaymentFormController.PaymentFormSubmit | | 语法 | public boolean PaymentFormSubmit (PaymentFormVO vo) |
| 前置条件 | 已创建一个PaymentForm领域对象， |
| 后置条件 | 调用PaymentForm领域对象的PaymentFormSubmit方法 |
| PaymentFormController.PaymentFormWithdraw | | 语法 | public boolean PaymentFormWithdraw (PaymentFormVO vo) |
| 前置条件 | 已创建一个PaymentForm领域对象， |
| 后置条件 | 调用PaymentForm领域对象的PaymentFormWithdraw方法 |
| PaymentFormController.PaymentFormCreate | | 语法 | public PaymentFormVO PaymentGatherCreate () |
| 前置条件 | 已创建一个PaymentForm领域对象， |
| 后置条件 | 调用PaymentForm领域对象的PaymentFormCreate方法 |
| PaymentFormController.PaymentFormAddRent | | 语法 | public boolean PaymentFormAddFreight (double freight,long id) |
| 前置条件 | 已创建一个PaymentForm领域对象， |
| 后置条件 | 调用PaymentForm领域对象的PaymentFormAddRent方法 |
| PaymentFormController.PaymentFormAddFreight | | 语法 | public boolean PaymentFormAddFreight (double freight,long id) |
| 前置条件 | 已创建一个PaymentForm领域对象， |
| 后置条件 | 调用PaymentForm领域对象的PaymentFormAddFreight方法 |
| PaymentFormController.PaymentFormAddSalary | | 语法 | public boolean PaymentFormAddSalary (double salary,double bonus,int month) |
| 前置条件 | 已创建一个PaymentForm领域对象， |
| 后置条件 | 调用PaymentForm领域对象的PaymentFormAddSalary方法 |
| **需要的服务（需接口）** | | | |
| 服务名 | 服务 | | |
| PaymentForm. PaymentFormCancel (PaymentFormVO vo) | 删除选中付款单 | | |
| PaymentForm.save(PaymentFormVO vo) | 保存付款单 | | |
| PaymentForm.submit(PaymentFormVO vo) | 提交付款单 | | |
| PaymentForm.withdraw(PaymentFormVO vo) | 撤回付款单 | | |
| PaymentForm.create() | 新建付款单 | | |
| PaymentForm.PaymentFormAddSalary(double freight,long id) | 添加工资信息 | | |
| PaymentForm.PaymentFormAddFreight (double freight,long id) | 添加运费信息 | | |
| PaymentForm.PaymentFormAddRent (double rent,int year) | 添加租金信息 | | |

**表4.1.3-3d PaymentForm的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| PaymentForm.PaymentFormCancel | 语法 | public void PaymentFormcancel (PaymentFormVO vo) |
| 前置条件 | 单子为编辑状态 |
| 后置条件 | 删除一个PaymentFormVO |
| PaymentForm.PaymentFormSave | 语法 | public boolean PaymentFormsave (PaymentFormVO vo) |
| 前置条件 | 单子为编辑状态 |
| 后置条件 | 将单子改为草稿状态 |
| PaymentForm.PaymentFormSubmit | 语法 | public boolean PaymentFormsubmit (PaymentFormVO vo) |
| 前置条件 | 单子为草稿状态 |
| 后置条件 | 将单子改为提交状态 |
| PaymentForm. PaymentFormWithdraw | 语法 | public boolean PaymentFormwithdraw (PaymentFormVO vo) |
| 前置条件 | 单子为提交状态 |
| 后置条件 | 将单子改为草稿状态 |
| PaymentForm.PaymentFormCreate | 语法 | public PaymentFormVO PaymentFormCreate () |
| 前置条件 | 无 |
| 后置条件 | 增加一个PaymentFormVO，并将单子改为编辑状态 |
| PaymentFormController.PaymentFormAddRent | 语法 | public boolean PaymentFormAddRent (double rent,int year) |
| 前置条件 | 启动一个新建付款单任务 |
| 后置条件 | 在一个新建付款单任务中增加租金信息 |
| PaymentFormController.PaymentFormAddFreight | 语法 | public boolean PaymentFormAddFreight (double freight,long id) |
| 前置条件 | 启动一个新建付款单任务 |
| 后置条件 | 在一个新建付款单任务中增加运费信息 |
| PaymentFormController.PaymentFormAddSalary | 语法 | public boolean PaymentFormAddSalary (double salary,double bonus,int month) |
| 前置条件 | 启动一个新建付款单任务 |
| 后置条件 | 在一个新建付款单任务中增加工资信息 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| DatabaseFactory.getFinanceDatabase | 得到Finance数据库的服务的引用 | |
| FinanceDataService. paymentFormFinancedata.find(long NO) | 查找单一持久化对象 | |
| FinanceDataService. paymentFormFinancedata.insert(paymentFormPO po) | 插入单一持久化对象 | |
| FinanceDataService. paymentFormFinancedata.delete(paymentFormPO po) | 删除单一持久化对象 | |
| FinanceDataService. paymentFormFinancedata.update(paymentFormPO po) | 更新单一持久化对象 | |

**表4.1.3-3e ReceiptFormController的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| ReceiptFormController.RecieptFormCancel | | 语法 | public void RecieptFormcancel (RecieptFormVO vo) |
| 前置条件 | 已创建一个ReceiptForm领域对象， |
| 后置条件 | 调用ReceiptForm领域对象的RecieptFormCancel方法 |
| ReceiptFormController.RecieptFormSave | | 语法 | public boolean RecieptFormsave (RecieptFormVO vo) |
| 前置条件 | 已创建一个ReceiptForm领域对象， |
| 后置条件 | 调用ReceiptForm领域对象的RecieptFormSave方法 |
| ReceiptFormController.RecieptFormSubmit | | 语法 | public boolean RecieptFormsubmit (RecieptFormVO vo) |
| 前置条件 | 已创建一个ReceiptForm领域对象， |
| 后置条件 | 调用ReceiptForm领域对象的RecieptFormSubmit方法 |
| ReceiptFormController.RecieptFormWithdraw | | 语法 | public boolean RecieptFormwithdraw (RecieptFormVO vo) |
| 前置条件 | 已创建一个ReceiptForm领域对象， |
| 后置条件 | 调用ReceiptForm领域对象的RecieptFormWithdraw方法 |
| ReceiptFormController.RecieptFormCreate | | 语法 | public RecieptFormVO RecieptFormcreate () |
| 前置条件 | 已创建一个ReceiptForm领域对象， |
| 后置条件 | 调用ReceiptForm领域对象的RecieptFormCreate方法 |
| ReceiptFormController.ReceiptFormAddInfo | | 语法 | public boolean ReceiptFormAddInfo (String date,double money,String express,long id) |
| 前置条件 | 已创建一个ReceiptForm领域对象， |
| 后置条件 | 调用ReceiptForm领域对象的ReceiptFormAddInfo方法 |
| **需要的服务（需接口）** | | | |
| 服务名 | 服务 | | |
| RecieptForm.RecieptFormCancel (RecieptFormVO vo) | 删除选中收款单 | | |
| RecieptForm.RecieptFormSave (RecieptFormVO vo) | 保存收款单 | | |
| RecieptForm.RecieptFormSubmit (RecieptFormVO vo) | 提交收款单 | | |
| RecieptForm.RecieptFormWithdraw (RecieptFormVO vo) | 撤回收款单 | | |
| RecieptForm.RecieptFormCreate () | 创建收款单 | | |
| ReceiptForm.ReceiptFormAddInfo(String date,double money,String express,long id) | 在收款单中添加详细信息 | | |

**表4.1.3-3f ReceiptForm的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| ReceiptForm.RecieptFormCancel | | 语法 | public void RecieptFormcancel (RecieptFormVO vo) |
| 前置条件 | 单子为编辑状态 |
| 后置条件 | 删除一个RecieptFormVO |
| ReceiptForm.RecieptFormSave | | 语法 | public boolean RecieptFormsave (RecieptFormVO vo) |
| 前置条件 | 单子为编辑状态 |
| 后置条件 | 将单子改为草稿状态 |
| ReceiptForm.RecieptFormSubmit | | 语法 | public boolean RecieptFormsubmit (RecieptFormVO vo) |
| 前置条件 | 单子为草稿状态 |
| 后置条件 | 将单子改为提交状态 |
| ReceiptForm.RecieptFormWithdraw | | 语法 | public boolean RecieptFormwithdraw (RecieptFormVO vo) |
| 前置条件 | 单子为提交状态， |
| 后置条件 | 将单子改为草稿状态 |
| ReceiptForm.RecieptFormCreate | | 语法 | public RecieptFormVO RecieptFormcreate () |
| 前置条件 | 无 |
| 后置条件 | 增加一个RecieptFormVO，并将单子改为编辑状态 |
| ReceiptForm.ReceiptFormAddInfo | | 语法 | public boolean ReceiptFormAddInfo (String date,double money,String express,long id) |
| 前置条件 | 启动一个新建收款单任务 |
| 后置条件 | 在一个新建的收款单任务中增加收款信息 |
| **需要的服务（需接口）** | | | |
| 服务名 | 服务 | | |
| DatabaseFactory.getFinanceDatabase | 得到Finance数据库的服务的引用 | | |
| FinanceDataService. receiptFormFinancedata.find(long NO) | 查找单一持久化对象 | | |
| FinanceDataService. receiptFormFinancedata.insert(receiptFormPO po) | 插入单一持久化对象 | | |
| FinanceDataService. receiptFormFinancedata.delete(receiptFormPO po) | 删除单一持久化对象 | | |
| FinanceDataService. FinanceDataService. receiptFormFinancedata.update(receiptFormPO po) | 更新单一持久化对象 | | |

**表4.1.3-3g ReceiptGatherController的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| ReceiptGatherController.RecieptGatherCreate | 语法 | public RecieptGatherVO RecieptGatherCreate () |
| 前置条件 | 已创建一个ReceiptForm领域对象， |
| 后置条件 | 调用ReceiptForm领域对象的RecieptGatherCreate方法 |
| ReceiptGatherController.RecieptGatherCancel | 语法 | public void RecieptGatherCancel (RecieptGatherVO vo) |
| 前置条件 | 已创建一个ReceiptForm领域对象， |
| 后置条件 | 调用ReceiptForm领域对象的RecieptGatherCancel方法 |
| ReceiptGatherController.RecieptGatherSave | 语法 | public boolean RecieptGatherSave (RecieptGatherVO vo) |
| 前置条件 | 已创建一个ReceiptForm领域对象， |
| 后置条件 | 调用ReceiptForm领域对象的RecieptGatherSave方法 |
| ReceiptGatherController.RecieptGatherSubmit | 语法 | public boolean RecieptGatherSubmit (RecieptGatherVO vo) |
| 前置条件 | 已创建一个ReceiptForm领域对象， |
| 后置条件 | 调用ReceiptForm领域对象的RecieptGatherSubmit方法 |
| ReceiptGatherController.RecieptGatherWithdraw | 语法 | public boolean RecieptGatherWithdraw (RecieptGatherVO vo) |
| 前置条件 | 已创建一个ReceiptForm领域对象， |
| 后置条件 | 调用ReceiptForm领域对象的RecieptGatherWithdraw方法 |
| ReceiptGatherController.ReceiptGatherFormAddInfo | 语法 | public boolean ReceiptGatherAddInfo (ArrayList< ReceiptFormVO> ) |
| 前置条件 | 已创建一个ReceiptForm领域对象， |
| 后置条件 | 调用ReceiptForm领域对象的ReceiptGatherFormAddInfo方法 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| RecieptGather.RecieptGatherCancel (RecieptGatherVO vo) | 删除选中收款单 | |
| RecieptGather.RecieptGatherSave (RecieptGatherVO vo) | 保存收款单 | |
| RecieptGather.RecieptGatherSubmit (RecieptFormVO vo) | 提交收款单 | |
| RecieptGather.RecieptGatherWithdraw (RecieptFormVO vo) | 撤回收款单 | |
| RecieptGather.RecieptGatherCreate () | 创建收款单 | |
| RecieptGather.ReceiptGatherFormAddInfo(ArrayList< ReceiptFormVO> ) |  | |

**表4.1.3-3h ReceiptGather的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| ReceiptGather.RecieptGatherCreate | | 语法 | public RecieptGatherVO RecieptGatherCreate () |
| 前置条件 | 无 |
| 后置条件 | 增加一个RecieptGatherVO，并将单子改为编辑状态 |
| ReceiptGather.RecieptGatherCancel | | 语法 | public void RecieptGatherCancel (RecieptGatherVO vo) |
| 前置条件 | 单子为编辑状态 |
| 后置条件 | 删除一个RecieptGatherVO |
| ReceiptGather.RecieptGatherSave | | 语法 | public boolean RecieptGatherSave (RecieptGatherVO vo) |
| 前置条件 | 单子为编辑状态 |
| 后置条件 | 将单子改为草稿状态 |
| ReceiptGather.RecieptGatherSubmit | | 语法 | public boolean RecieptGatherSubmit (RecieptGatherVO vo) |
| 前置条件 | 单子为草稿状态 |
| 后置条件 | 将单子改为提交状态 |
| ReceiptGather.RecieptGatherWithdraw | | 语法 | public boolean RecieptGatherWithdraw (RecieptGatherVO vo) |
| 前置条件 | 单子为提交状态 |
| 后置条件 | 将单子改为草稿状态 |
| ReceiptGather.ReceiptGatherFormAddInfo | | 语法 | public boolean ReceiptGatherAddInfo (ArrayList< ReceiptFormVO> ) |
| 前置条件 | 启动一个新建收款单汇总报表任务 |
| 后置条件 | 在一个新建的收款单汇总报表任务中增加收款单信息 |
| **需要的服务（需接口）** | | | |
| 服务名 | 服务 | | |
| 无 | 无 | | |

**表4.1.3-3i BeginningAccountController的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| BeginningAccountController.init | | 语法 | public boolean init (int year, ArrayList< UserPO >,ArraList< carInfoPO > ,ArrayList< DriverInfoPO > ,ArrayList< AccountPO >) |
| 前置条件 | 已创建一个BeginningAccount领域对象， |
| 后置条件 | 调用BeginningAccount领域对象的init方法 |
| **需要的服务（需接口）** | | | |
| 服务名 | 服务 | | |
| BeginningAccount.init | 期初建账 | | |

**表4.1.3-3j BeginningAccount的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| BeginningAccount.init | | 语法 | public boolean init (int year, ArrayList< UserPO >,ArraList< carInfoPO > ,ArrayList< DriverInfoPO > ,ArrayList< AccountPO >) |
| 前置条件 | 要求期初建账 |
| 后置条件 | 生成新账目，并保存一份初始账目 |
| **需要的服务（需接口）** | | | |
| 服务名 | 服务 | | |
| DatabaseFactory.getFinanceDatabase | 得到Finance数据库的服务的引用 | | |

**表4.1.3-3k** CostPayChartController**的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| CostPayChartController.CostPayChartAddInfo | 语法 | public boolean costPayChartAddInfo (receiptFormPO po,paymentFormPO po) |
| 前置条件 | 已创建一个CostPayForm领域对象， |
| 后置条件 | 调用CostPayForm领域对象的CostPayChartAddInfo方法 |
| CostPayChartController.CostPayChartAdd | 语法 | public boolean costPayChartCreat(CostPayChartVO vo) |
| 前置条件 | 已创建一个CostPayForm领域对象， |
| 后置条件 | 调用CostPayForm领域对象的CostPayChartAdd  方法 |
| CostPayChartController.CostPayChartCheck | 语法 | public CostPayChartVO costPayChartCreat(long NO) |
| 前置条件 | 已创建一CostPayForm领域对象， |
| 后置条件 | 调用CostPayForm领域对象的CostPayChartCheck方法 |
| CostPayChartController.CostPayChartCreate | 语法 | public CostPayChartVO costPayChartCreate() |
| 前置条件 | 已创建一个CostPayForm领域对象， |
| 后置条件 | 调用CostPayForm领域对象的CostPayChartCreate方法 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| CostPayChartController.CostPayChartAddInfo(receiptFormPO po,paymentFormPO po) | 加表中的详细信息 | |
| CostPayChartController.CostPayChartAdd(CostPayChartVO vo) | 加表 | |
| CostPayChartController.CostPayChartCheck(long NO) | 查看成本收益表 | |
| CostPayChartController.CostPayChartCreate() | 创建成本收益表 | |

**表4.1.3-3l**CostPatChart**的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| CostPayChart.CostPayChartCreate | 语法 | public CostPayChartVO CostPayChartCreate() |
| 前置条件 | 无 |
| 后置条件 | 增加一个VO |
| CostPayChart.CostPayChartAdd | 语法 | public boolean CostPayChartAdd (CostPayChartVO vo) |
| 前置条件 | 不存在相同NO的vo |
| 后置条件 | 无 |
| CostPayChart.CostPayChartCheck | 语法 | public CostPayChartVO CostPayChartCheck (long NO) |
| 前置条件 | 无 |
| 后置条件 | 返回一个VO |
| CostPayChart.CostPayChartAddInfo | 语法 | public boolean CostPayChartAddInfo (receiptFormPO po,paymentFormPO po) |
| 前置条件 | 启动一个新建成本收益表任务 |
| 后置条件 | 在一个新建成本收益表任务中增加收入、支出、利润信息 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| FinanceDatabl. costPayChartFinancedata.find(long NO) | 查找单一持久化对象 | |
| FinanceDataService. costPayChartFinancedata.insert(costPayChartPO po) | 插入单一持久化对象 | |
| FinanceDataService. costPayChartFinancedata.delete(costPayChartPO po) | 删除单一持久化对象 | |
| FinanceDataService. costPayChartFinancedata.update(costPayChartPO po) | 更新单一持久化对象 | |
| DatabaseFactory.getFinanceDatabase | 得到Finance数据库的服务的引用 | |

**表4.1.3-3m** ProfitChartController**的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| ProfitChartController.ProfitChartCreate | | 语法 | public CostPayChartVO CostPayChartCreate() |
| 前置条件 | 无 |
| 后置条件 | 增加一个VO |
| ProfitChartController.ProfitChartAdd | | 语法 | public boolean CostPayChartAdd (CostPayChartVO vo) |
| 前置条件 | 不存在相同NO的vo |
| 后置条件 | 无 |
| ProfitChartController.ProfitChartCheck | | 语法 | public CostPayChartVO CostPayChartCheck (long NO) |
| 前置条件 | 无 |
| 后置条件 | 返回一个VO |
| **需要的服务（需接口）** | | | |
| 服务名 | 服务 | | |
| ProfitChart.ProfitChartCreate() | 创建一个表 | | |
| ProfitChart.ProfitChartAdd(ProfitVO vo) | 插入详细内容 | | |
| ProfitChart.ProfitChartCheck(long NO) | 查看表单 | | |

**表4.1.3-3n** ProfitChart**的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| ProfitChart.ProfitChartCreate | | 语法 | public ProfitChartVO ProfitChartCreate () |
| 前置条件 | 无 |
| 后置条件 | 增加一个VO |
| ProfitChart.ProfitChartAdd | | 语法 | public boolean ProfitChartAdd (ProfitVO vo) |
| 前置条件 | 不存在相同NO的vo |
| 后置条件 | 无 |
| ProfitChart.ProfitChartCheck | | 语法 | public ProfitChartVO void ProfitChartCheck (long NO) |
| 前置条件 | 输入正确的NO |
| 后置条件 | 返回一个vo |
| **需要的服务（需接口）** | | | |
| 服务名 | 服务 | | |
| DatabaseFactory.getFinanceDatabase | 得到Finance数据库的服务的引用 | | |
| FinanceDataService. paymentFormFinancedata.find(long NO) | 查找单一持久化对象 | | |
| FinanceDataService. receiptFormFinancedata.find(long NO) | 查找单一持久化对象 | | |

#### 4.1.3.4业务逻辑层的动态模型

1) AccountInfo

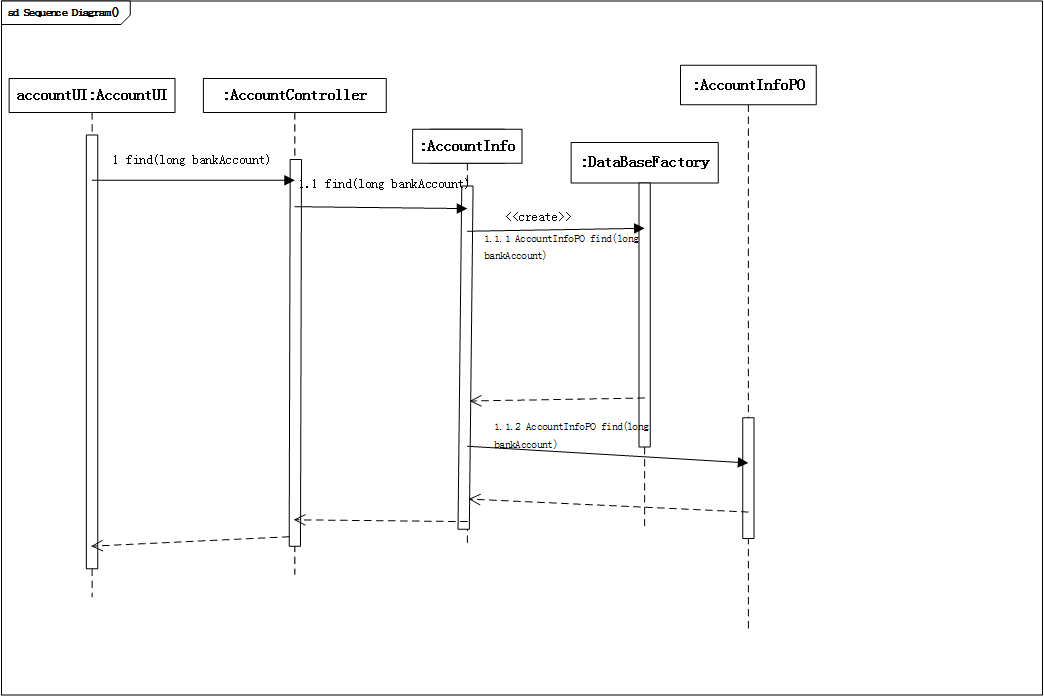


图4.1.3.4-1 AccountInfo \_find

2) BegninningAccount

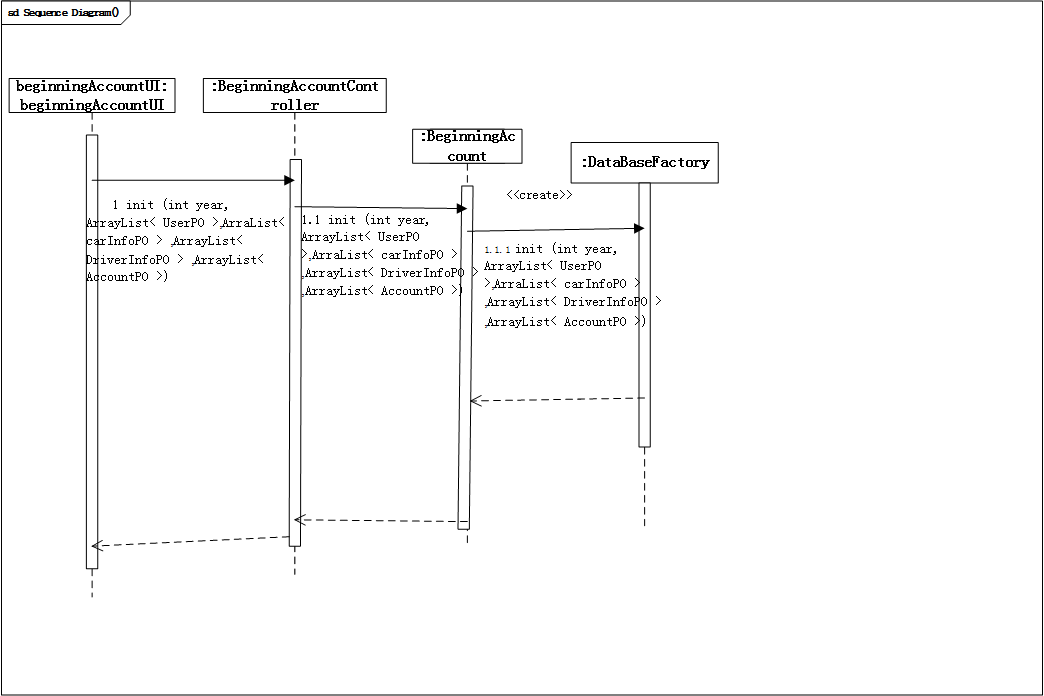


图4.1.3.4-2 BegninningAccount\_init

3) CostPayChart

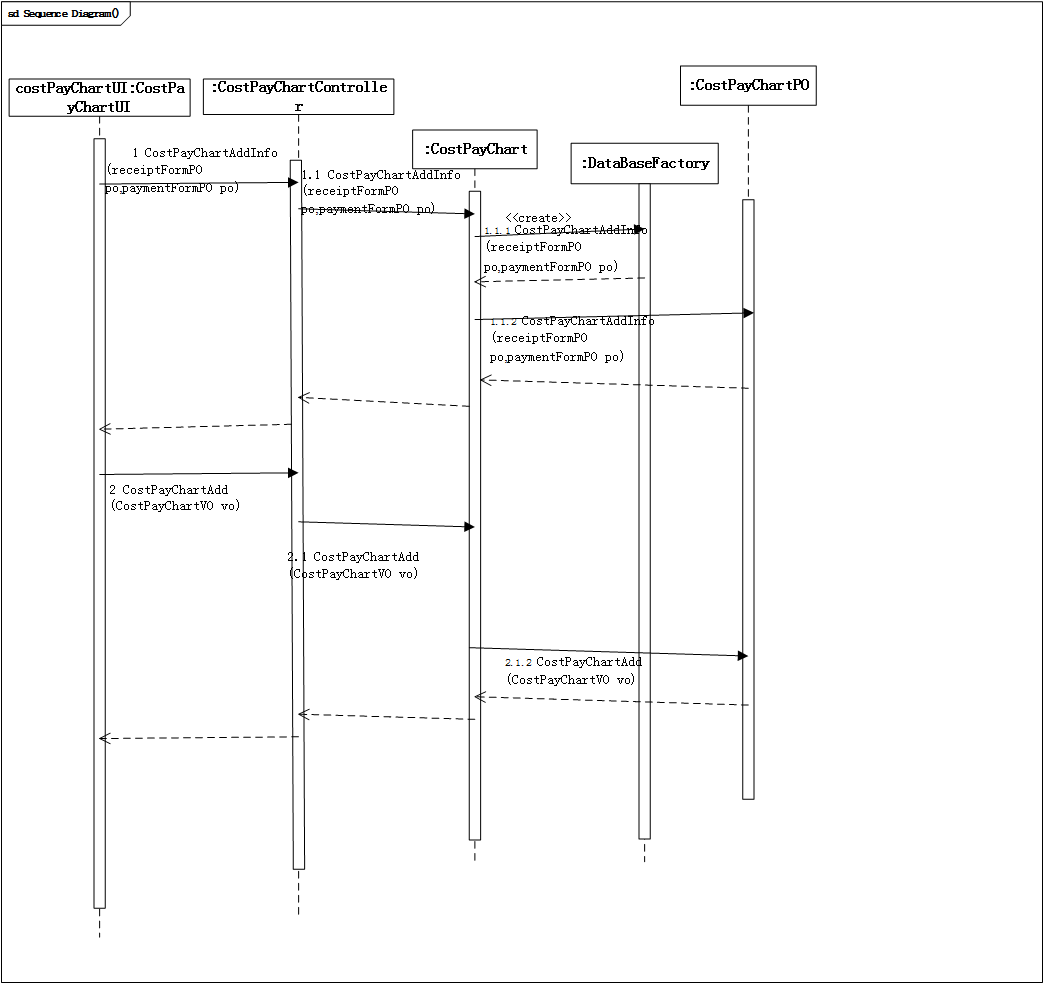


图4.1.3.4-3CostPayChart\_Add

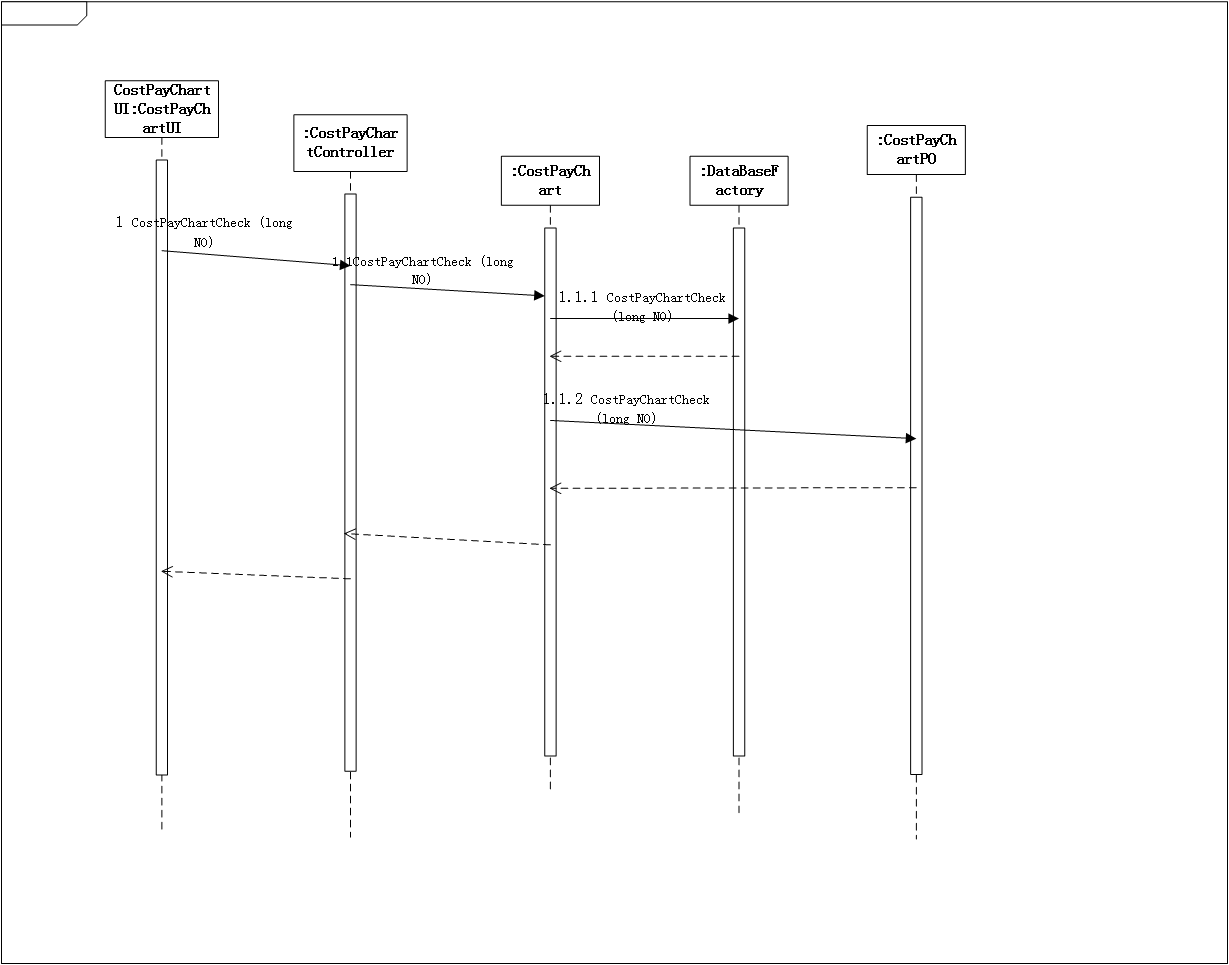


图4.1.3.4-4 CostPayChart\_Check

4) PaymentForm

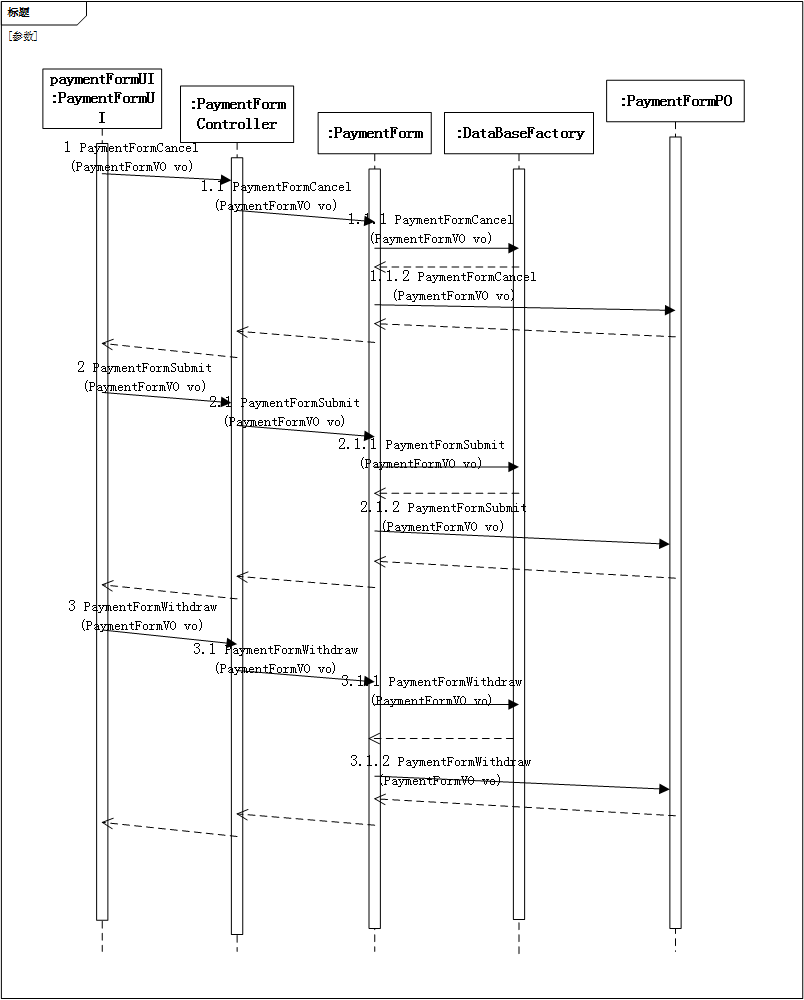


图4.1.3.4-5 PaymentForm\_Option

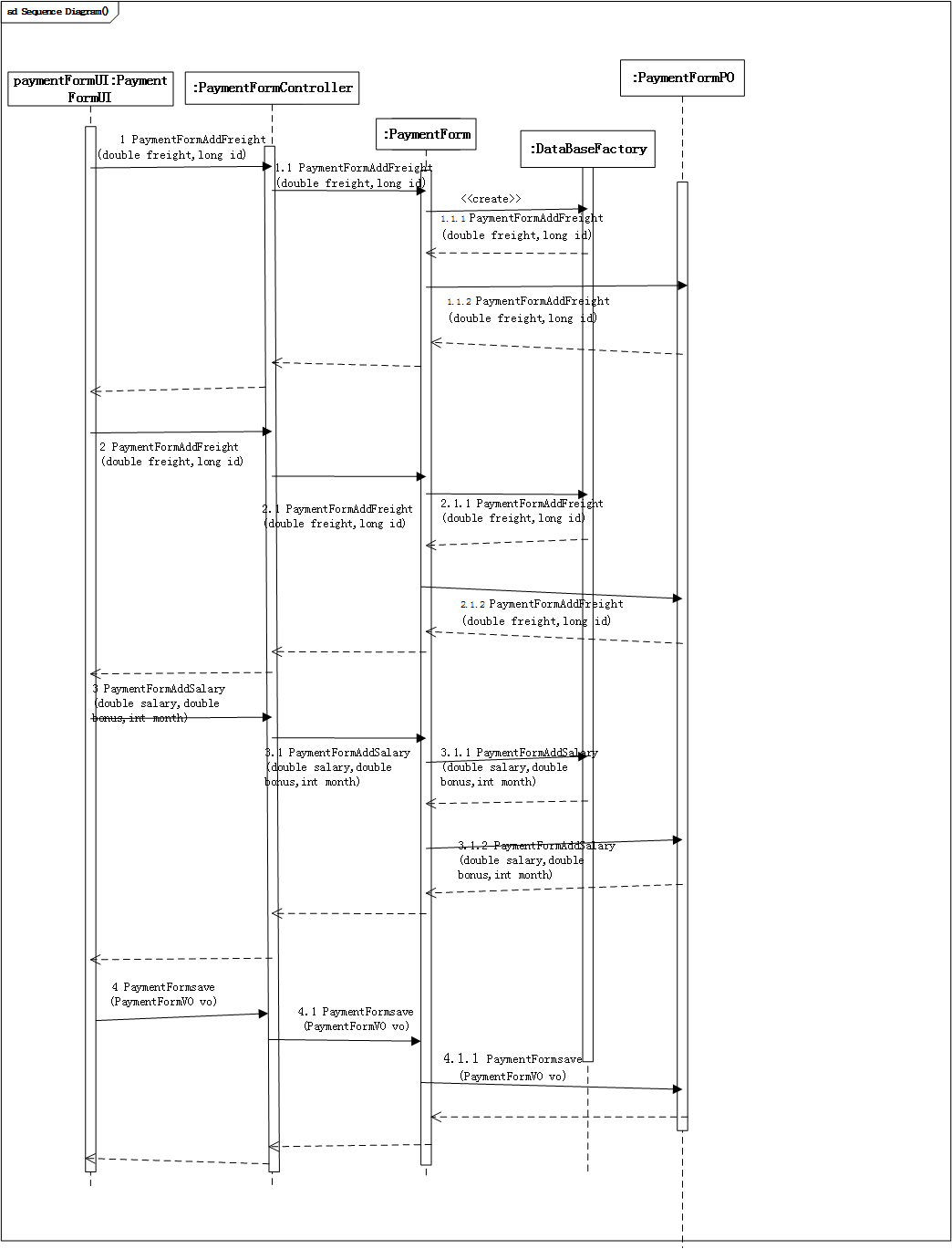


图4.1.3.4-6PaymentForm\_PaymentFormAdd

5) ProfitChart

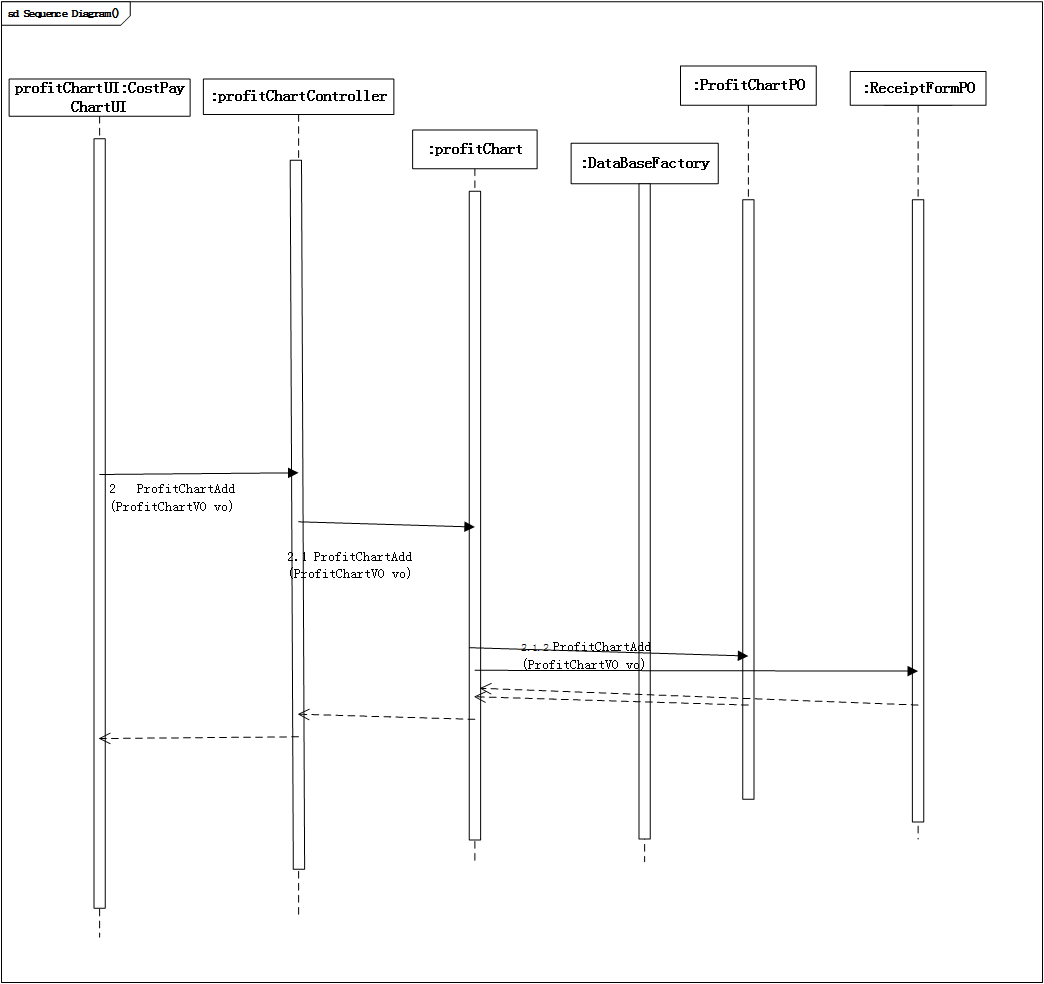


图4.1.3.4-7ProfitChart \_Add

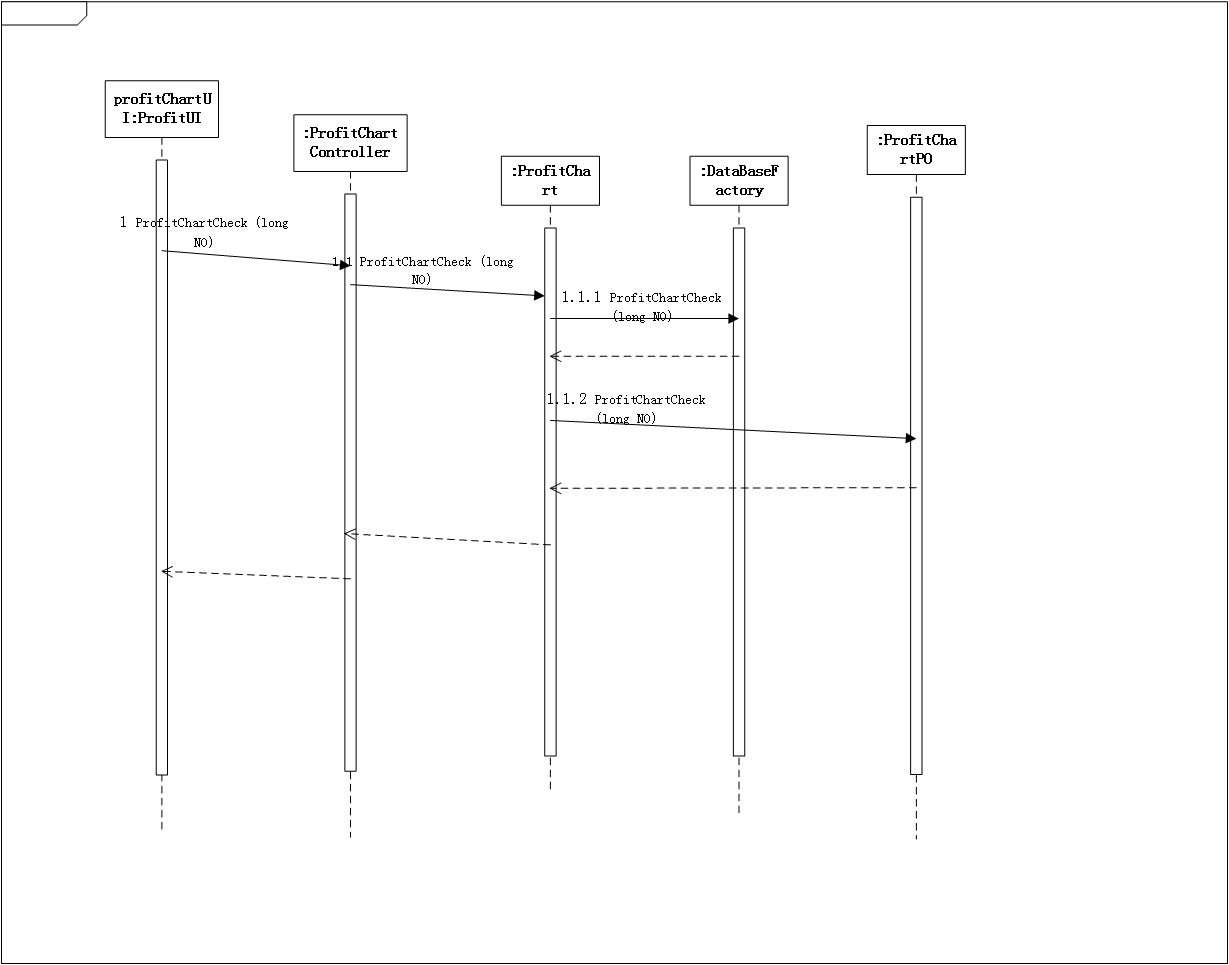


图4.1.3.4-8 ProfitChart\_Check

6) ReceiptForm

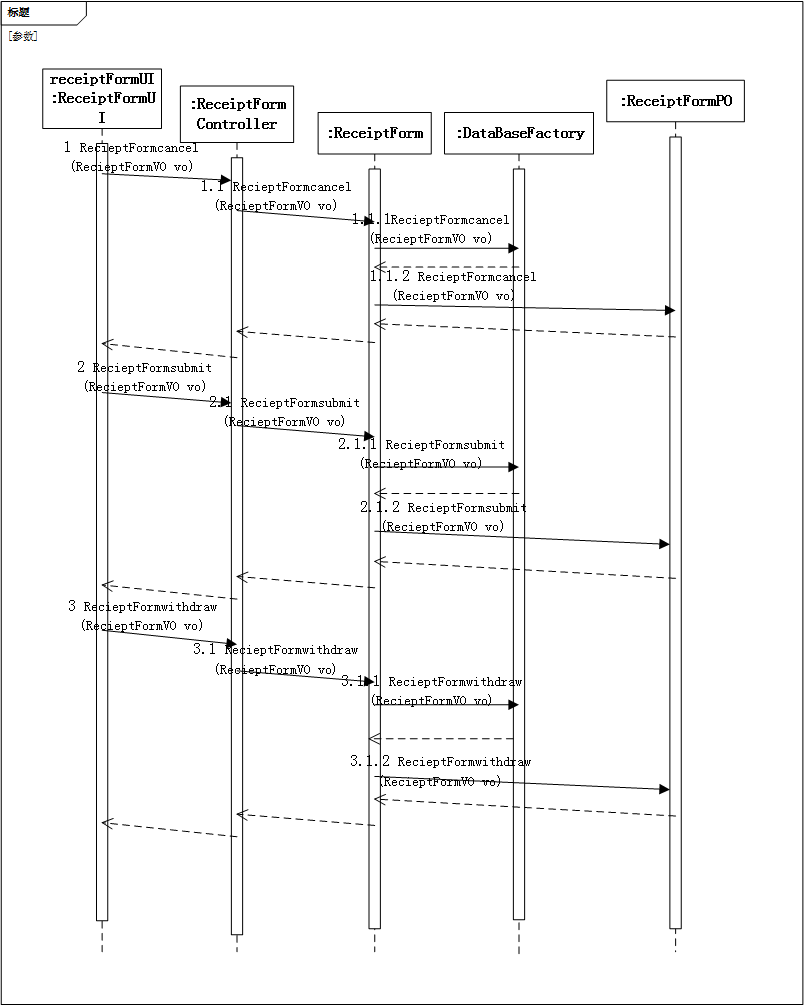


图4.1.3.4-9 ReceiptForm\_Option

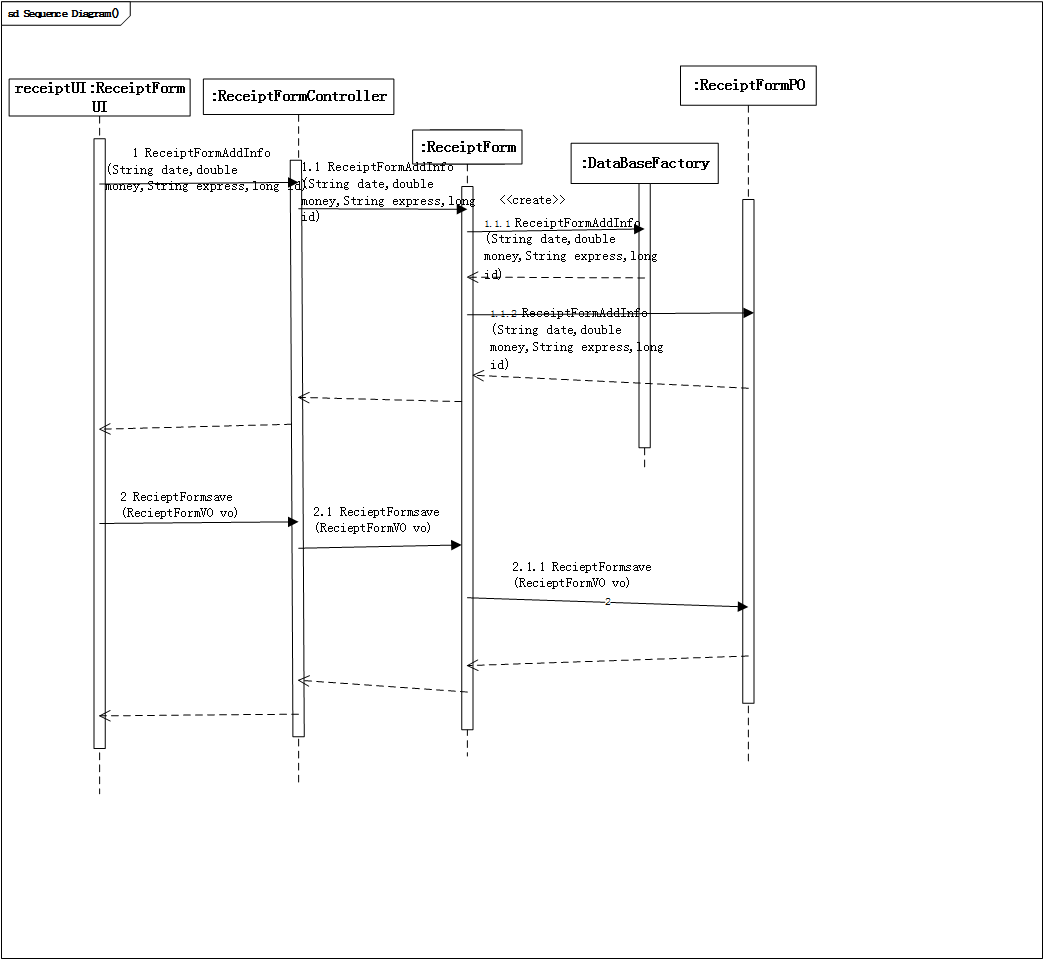


图4.1.3.4-10 ReceiptForm\_ReceiptFormAdd

7) ReceiptGather

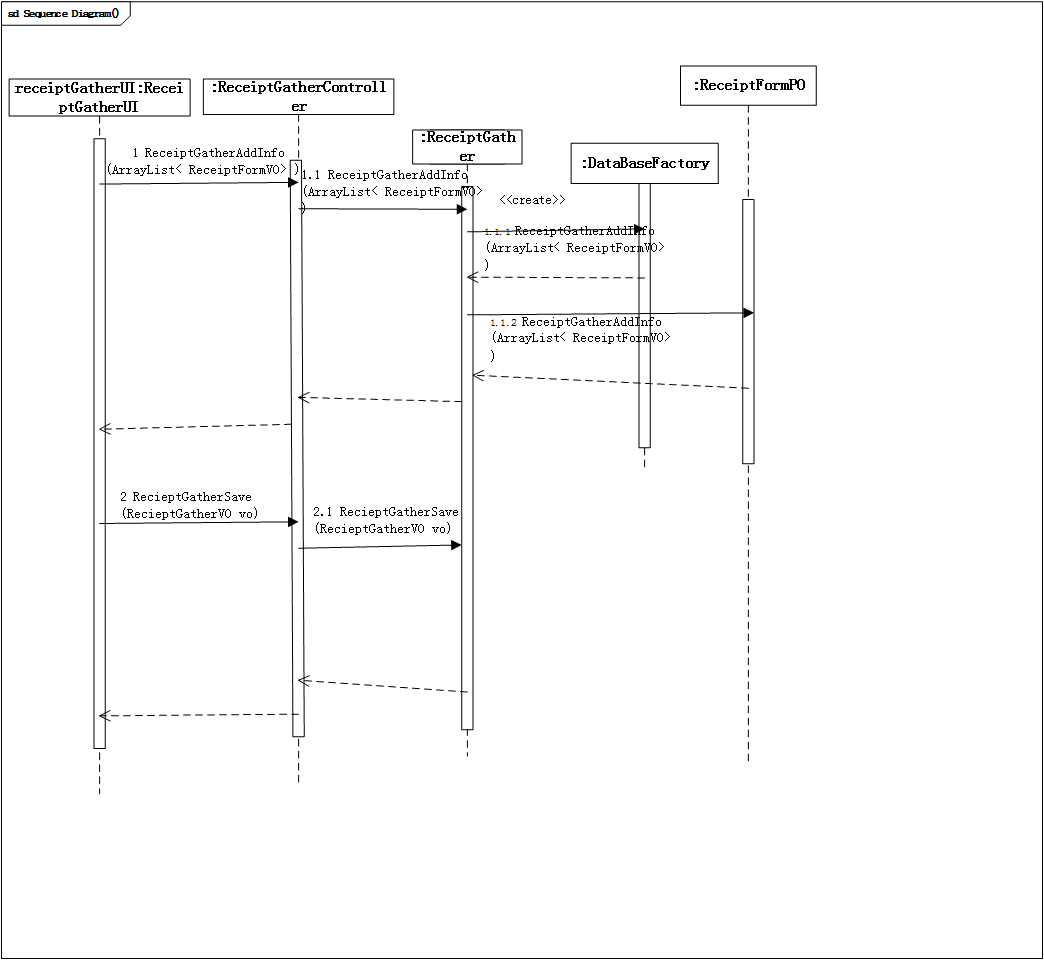


图4.1.3.4-11 ReceiptGather\_Add

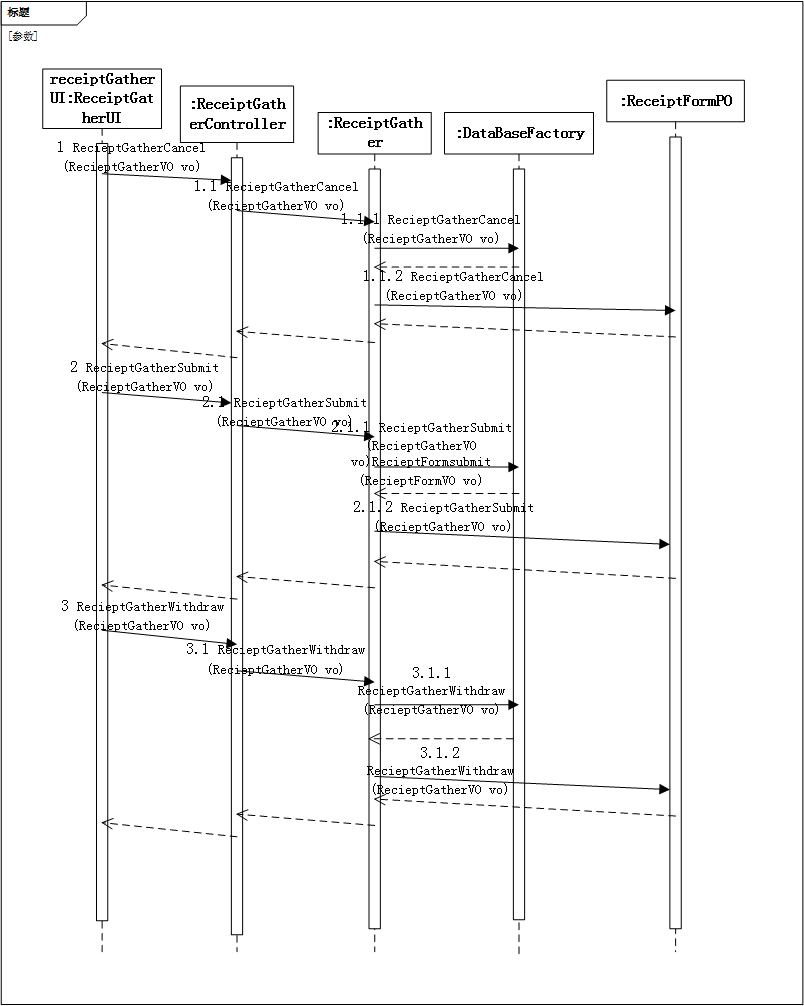


图4.1.3.4-12 ReceiptGather\_Option

#### 4.1.3.5业务逻辑的设计原理

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的对象。

### 4.1.4 userbl模块

#### 4.1.4.1模块概述

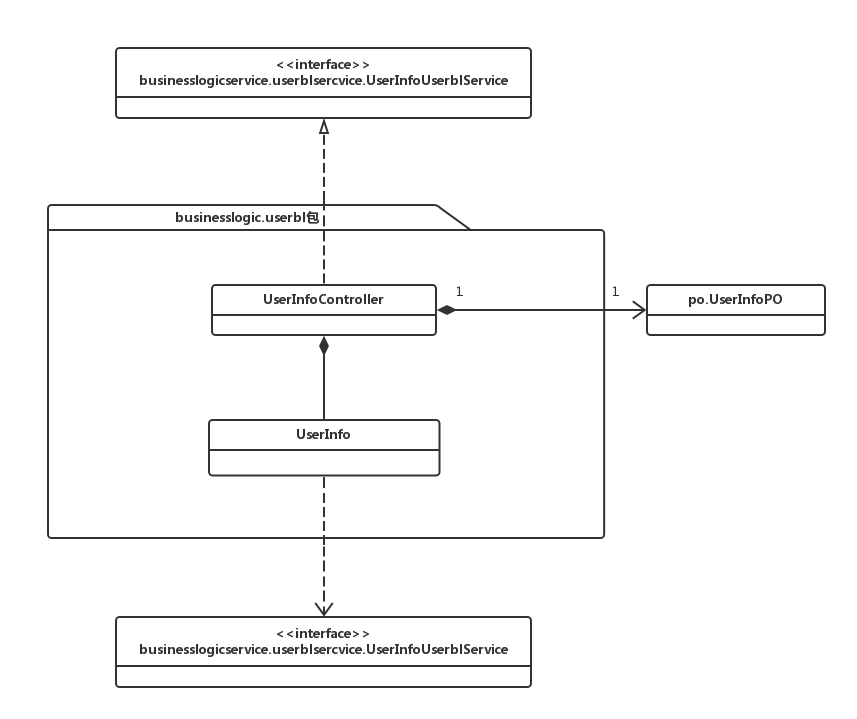
Userbl模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

Userbl模块的职责及接口参见软件体系结构描述文档v1.8表5.3.2-6

#### 4.1.4.2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。比如展示层和业务逻辑层之间，我们添加businesslogicservice.userblsercvice.UserInfoUserblService接口。业务逻辑层和数据层之间添加dataservice.userdataservice.UserdataService接口。为了隔离业务逻辑职责，我们增加了UserInfoController，这样UserInfoController会将对用户的业务逻辑处理委托给User对象。UserInfo是作为用户信息的持久化对象被添加到设计模型中去的。

userbl模块的设计如图4.1.4-1所示。



**图4.1.4-1 userbl模块各个类的设计**

Userbl模块各个类的的职责如表4.1.4-1所示

**表4.1.4-1 userbl模块各个类的职责**

|  |  |
| --- | --- |
| 模块 | 职责 |
| LoginController | 负责实现对应于登陆界面所需要的服务 |
| UserInfoController | 负责实现用户信息管理界面所需要的服务 |
| UserInfo | 系统用户的领域模型对象，拥有用户数据的用户名、密码和职业 |

#### 4.1.4.3模块内部类的接口规范

**表4.1.4-2 UserInfoController的接口规范**

|  |
| --- |
| **提供的服务（供接口）** |

|  |  |  |
| --- | --- | --- |
| UserInfoController.create | 语法 | public UserInfoVO create() |
| 前置条件 | 已创建一个UserInfo领域对象 |
| 后置条件 | 调用UserInfo领域对象的create方法 |
| Userbl.UserInfoUserbl.add | 语法 | public boolean add(UserInfoVO vo) |
| 前置条件 | 已创建一个UserInfo领域对象，并且输入符合输入规则 |
| 后置条件 | 调用UserInfo领域对象的add方法 |
| Userbl.UserInfoUserbl.delete | 语法 | public boolean delete(UserInfoVO vo) |
| 前置条件 | 已创建一个UserInfo领域对象 |
| 后置条件 | 调用UserInfo领域对象的delete方法 |
| Userbl.UserInfoUserbl.find | 语法 | public UserInfoVO find(long userAccount) |
| 前置条件 | 已创建一个UserInfo领域对象，并且输入符合输入规则 |
| 后置条件 | 调用UserInfo领域对象的find方法 |
| Userbl.UserInfoUserbl.login | 语法 | public boolean login(long userAccount, String password) |
| 前置条件 | 已创建一个UserInfo领域对象，并且输入符合输入规则 |
| 后置条件 | 调用UserInfo领域对象的login方法 |
| Userbl.UserInfoUserbl.update | 语法 | public UserInfoVO update() |
| 前置条件 | 已创建一个UserInfo领域对象 |
| 后置条件 | 调用UserInfo领域对象的update方法 |

|  |  |
| --- | --- |
| **需要的服务（需接口）** | |
| UserInfo.create() | 加入一个VO对象 |
| UserInfo.add(UserInfoVO vo) | 加入一个用户对象 |
| UserInfo.delete(UserInfoVO vo) | 删除一个用户对象 |
| UserInfo.find(long userAccount) | 查找一个用户对象 |
| UserInfo.login(long userAccount, String password) | 登录相应的用户界面 |
| UserInfo.update() | 更新用户信息 |

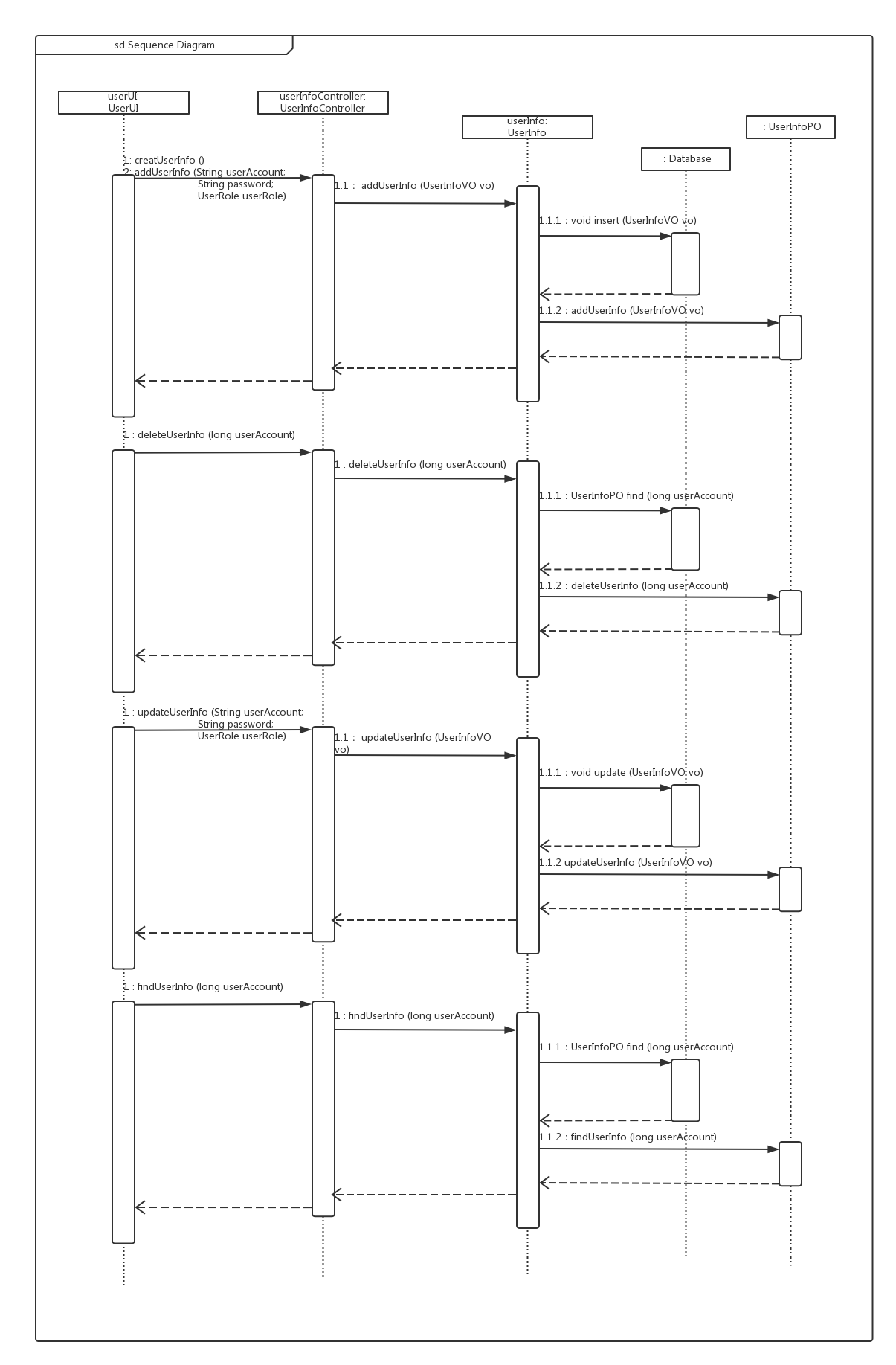
**表4.1.4-3 UserInfo的接口规范**

|  |
| --- |
| **提供的服务（供接口）** |

|  |  |  |
| --- | --- | --- |
| UserInfo.create | 语法 | public UserInfoVO create() |
| 前置条件 | 无 |
| 后置条件 | 增加一个UserInfoVO对象 |
| UserInfo.add | 语法 | public boolean add(UserInfoVO vo) |
| 前置条件 | 不存在相同的VO对象 |
| 后置条件 | 无 |
| UserInfo.delete | 语法 | public boolean delete(UserInfoVO vo) |
| 前置条件 | 存在相同的VO对象 |
| 后置条件 | 删除一个UserInfoVO对象 |
| UserInfo.find | 语法 | public UserInfoVO find(long userAccount) |
| 前置条件 | 输入用户名 |
| 后置条件 | 返回一个UserInfoVO对象 |
| UserInfo.login | 语法 | public boolean login(long userAccount, String password) |
| 前置条件 | password符合输入规则 |
| 后置条件 | 查找是否存在相应的User，根据输入password返回相应的值 |
| UserInfo.update | 语法 | public UserInfoVO update() |
| 前置条件 | 无 |
| 后置条件 | 返回一个UserInfoVO |

|  |  |  |
| --- | --- | --- |
| **需要的服务（需接口）** | | |
| DatabaseFactory.getUserDatabase | 得到user数据库的服务的引用 | |
| userDataService. UserInfodataService.insert(UserInfoPO po) | 插入单一持久化UserInfoPO对象 |
| TransitDataService. UserInfodataService.delete(UserInfoPO po) | 删除单一持久化UserInfoPO对象 |
| TransitDataService. UserInfodataService.update(UserInfoPO po) | 更新单一持久化UserInfoPO对象 |
| TransitDataService. UserInfodataService.find(long userAccount) | 根据userAccount查找单一持久化UserInfoPO对象 |

#### 4.1.4.4业务逻辑层的动态模型



#### 4.1.4.5业务逻辑的设计原理

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的对象。

### 4.1.5 inventorybl模块

#### 4.1.5.1模块概述

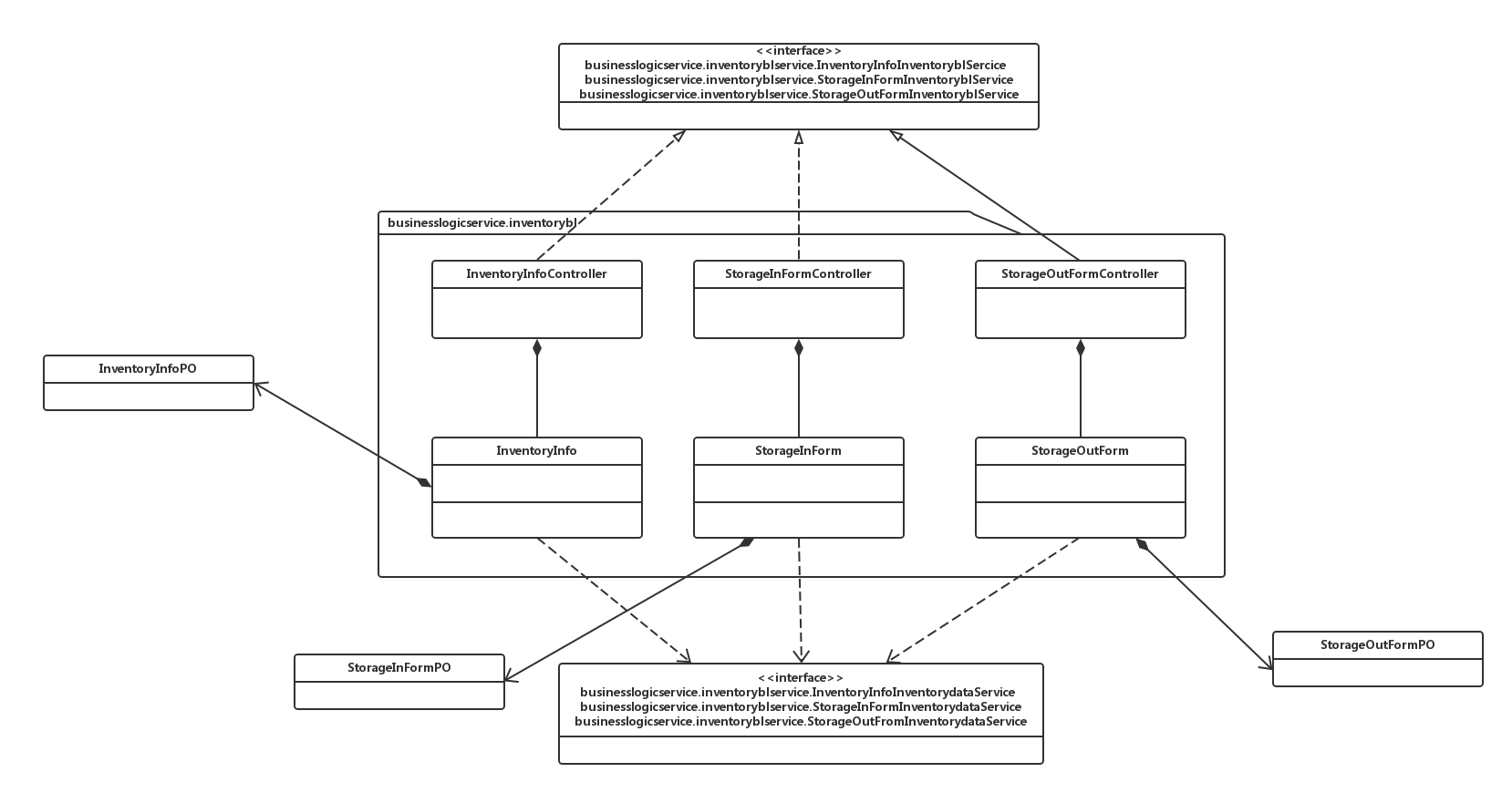
Inventorybl模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

Inventorybl模块的职责及接口参见软件体系结构描述文档v1.8表5.3.2-2

#### 4.1.5.2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。比如展示层和业务逻辑层之间，我们添加businesslogicservice.inventoryblservice.InventoryInfoInventoryblSercice接口、businesslogicservice.inventoryblservice.StorageInFormInventoryblService接口、businesslogicservice.inventoryblservice.StorageOutFormInventoryblService接口。业务逻辑层和数据层之间添加dataservice.inventorydataservice.InventoryInfoInventorydataService接口、dataservice.inventorydataservice.StorageInFormInventorydataService接口、dataservice.inventorydataservice.StorageOutFromInventorydataService接口。为了隔离业务逻辑职责，我们增加了InventoryInfoController、StorageInFormController、StorageOutFormController，这样InventoryInfoController、StorageInFormController、StorageOutFormController会将对用户的业务逻辑处理委托给InventoryInfo、StorageInForm、StorageOutForm对象。InventoryInfoPO、StorageInFormPO、StorageOutFormPO是作为用户信息的持久化对象被添加到设计模型中去的。

Inventorybl模块的设计如图4.1.2-1所示。

****

Inventorybl模块各个类的职责如表4.1.5-1所示

**表4.1.5-1 Inventorybl模块各个类的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| InventoryInfoController | 负责实现仓库管理界面所需要的服务 |
| StorageInFormController | 负责实现入库界面所需要的服务 |
| StorageOutFormController | 负责实现出库界面所需要的服务 |
| InventoryInfo | 仓库的领域模型对象，拥有仓库信息，可以帮助完成仓库管理界面所需要的服务 |
| StorageInForm | 仓库的领域模型对象，拥有入库单信息，可以帮助完成入库界面所需要的服务 |
| StorageOutForm | 仓库的领域模型对象，拥有出库单信息，可以帮助完成出库界面所需要的服务 |

#### 4.1.5.3模块内部类的接口规范

**表4.1.5-2 inventoryblController模块的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| inventoryblController.InvetoryInfoInventorybl. invetoryInfoSetWarningLimit | 语法 | public boolean invetoryInfoSetWarningLimit (double limit) |
| 前置条件 | 已创建一个Inventory领域对象 |
| 后置条件 | 调用inventory领域对象的invetoryInfoSetWarningLimit 方法 |
| inventoryblController. InvetoryInfoInventorybl.inventoryAlarm | 语法 | public Zone inventoryAlarm(InventoryVO vo) |
| 前置条件 | 已创建一个Inventory领域对象 |
| 后置条件 | 调用inventory领域对象的inventoryAlarm方法 |
| inventoryblController. InvetoryInfoInventorybl. invetoryAdjustment | 语法 | public boolean invetoryAdjustment (Zone zone) |
| 前置条件 | 已创建一个Inventory领域对象 |
| 后置条件 | 调用inventory领域对象的invetoryAdjustment方法 |
| inventoryblController. InvetoryInfoInventorybl.inventoryCheck | 语法 | public InventoryInfoVO storageCheck(String startTime, String endTime) |
| 前置条件 | 已创建一个Inventory领域对象 |
| 后置条件 | 调用inventory领域对象的inventoryCheck方法 |
| inventoryblController. InvetoryInfoInventorybl.inventoryCounting | 语法 | public inventoryInfoVO inventoryCounting(Zone zone) |
| 前置条件 | 已创建一个Inventory领域对象 |
| 后置条件 | 调用inventory领域对象的inventoryCounting方法 |

|  |  |  |
| --- | --- | --- |
| inventoryblController.StorageInFormInventorybl.create | 语法 | public StorageInFormVO StorageInFormcreate () |
| 前置条件 | 已创建一个Inventory领域对象 |
| 后置条件 | 调用inventory领域对象的StorageInFormInventorybl.create方法 |
| inventoryblController.StorageInFormInventorybl.cancel | 语法 | public void StorageInFormcancel (StorageInFormVO vo) |
| 前置条件 | 已创建一个Inventory领域对象 |
| 后置条件 | 调用inventory领域对象的StorageInFormInventorybl.cancel方法 |
| inventoryblController.StorageInFormInventorybl.save | 语法 | public boolean StorageInFormsave (StorageInFormVO vo) |
| 前置条件 | 已创建一个Inventory领域对象 |
| 后置条件 | 调用inventory领域对象的StorageInFormInventorybl.save方法 |
| inventoryblController.StorageInFormInventorybl.submit | 语法 | public boolean StorageInFormsubmit (StorageInFormVO vo) |
| 前置条件 | 已创建一个Inventory领域对象 |
| 后置条件 | 调用inventory领域对象的StorageInFormInventorybl.submit方法 |
| inventoryblController.StorageInFormInventorybl.withdraw | 语法 | public boolean StorageInFormwithdraw (StorageInFormVO vo) |
| 前置条件 | 已创建一个Inventory领域对象 |
| 后置条件 | 调用inventory领域对象的StorageInFormInventorybl.withdraw方法 |
| inventoryblController.StorageInFormInventorybl.storageInFormAddInfo | 语法 | public boolean storageInFormAddInfo (String date, String destination, Zone zone,int line,int shelf,int tag,long ID) |
| 前置条件 | 已创建一个Inventory领域对象 |
| 后置条件 | 调用inventory领域对象的storageInFormAddInfo方法 |
| inventoryblController.StorageOutFormInventorybl.create | 语法 | public void StorageOutFormcreate (StorageOutFormVO vo) |
| 前置条件 | 已创建一个Inventory领域对象 |
| 后置条件 | 调用inventory领域对象的StorageOutFormInventorybl.create方法 |
| inventoryblController.StorageOutFormInventorybl.cancel | 语法 | public void StorageOutFormcancel (StorageOutFormVO vo) |
| 前置条件 | 已创建一个Inventory领域对象 |
| 后置条件 | 调用inventory领域对象的StorageOutFormInventorybl.cancel方法 |
| inventoryblController.StorageOutFormInventorybl.save | 语法 | public boolean StorageOutFormsave (StorageOutFormVO vo) |
| 前置条件 | 已创建一个Inventory领域对象 |
| 后置条件 | 调用inventory领域对象的StorageOutFormInventorybl.save方法 |
| inventoryblController.StorageOutFormInventorybl.submit | 语法 | public boolean StorageOutFormsubmit (StorageOutFormVO vo) |
| 前置条件 | 已创建一个Inventory领域对象 |
| 后置条件 | 调用inventory领域对象的StorageOutFormInventorybl.submit方法 |
| inventoryblController.StorageOutFormInventorybl.withdraw | 语法 | public boolean StorageOutFormwithdraw (StorageOutFormVO vo) |
| 前置条件 | 已创建一个Inventory领域对象 |
| 后置条件 | 调用inventory领域对象的StorageOutFormInventorybl.withdraw方法 |

|  |  |  |  |
| --- | --- | --- | --- |
| inventoryblController.StorageOutFormInventorybl.storageOutFormAddInfo | 语法 | | public boolean storageOutFormAddInfo (String date, String destination, Transport transport, String relayNumber,String motorNumber) |
| 前置条件 | | 已创建一个Inventory领域对象 |
| 后置条件 | | 调用inventory领域对象的storageOutFormAddInfo方法 |
| **需要的服务（需接口）** | | | |
| inventorybl.InvetoryInfoInventorybl. invetoryInfoSetWarningLimit | | 加入一个库存警戒值对象 | |
| inventorybl. InvetoryInfoInventorybl.inventoryAlarm | | 加入一个超过警戒值的分区 | |
| inventorybl. InvetoryInfoInventorybl. invetoryAdjustment | | 调整库存分区 | |
| inventorybl. InvetoryInfoInventorybl.inventoryCheck | | 查看库存 | |
| inventorybl. InvetoryInfoInventorybl.inventoryCounting | | 库存盘点 | |
| inventorybl.StorageInFormInventorybl.create | | 创建一个入库单对象 | |
| inventorybl.StorageInFormInventorybl.cancel | | 删除一个入库单对象 | |
| inventorybl.StorageInFormInventorybl.save | | 保存一个入库单对象 | |
| inventorybl.StorageInFormInventorybl.submit | | 提交一个入库单对象 | |
| inventorybl.StorageInFormInventorybl.withdraw | | 撤回一个入库单对象 | |
| inventorybl.StorageInFormInventorybl.storageInFormAddInfo | | 加入一个入库单对象 | |
| inventorybl.StorageOutFormInventorybl.create | | 创建一个出库单对象 | |
| inventorybl.StorageOutFormInventorybl.cancel | | 删除一个入库单对象 | |
| inventorybl.StorageOutFormInventorybl.save | | 保存一个入库单对象 | |
| inventorybl.StorageOutFormInventorybl.submit | | 提交一个入库单对象 | |
| inventorybl.StorageOutFormInventorybl.withdraw | | 撤回一个入库单对象 | |
| inventorybl.StorageOutFormInventorybl.storageOutFormAddInfo | | 加入一个入库单对象 | |

**表4.1.5-3 inventorybl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| inventorybl.InvetoryInfoInventorybl. invetoryInfoSetWarningLimit | 语法 | public boolean invetoryInfoSetWarningLimit (double limit) |
| 前置条件 | 启动一个设置库存警戒值的任务 |
| 后置条件 | 在一个设置库存警戒值的任务中增加警戒值 |
| inventorybl. InvetoryInfoInventorybl.inventoryAlarm | 语法 | public Zone inventoryAlarm(InventoryVO vo) |
| 前置条件 | 启动一个库存报警的任务 |
| 后置条件 | 关闭一个库存报警任务 |
| inventorybl. InvetoryInfoInventorybl. invetoryAdjustment | 语法 | public boolean invetoryAdjustment (Zone zone) |
| 前置条件 | 库存报警了 |
| 后置条件 | 机动区改为库存报警的区 |
| inventorybl. InvetoryInfoInventorybl.inventoryCheck | 语法 | public InventoryInfoVO storageCheck(String startTime, String endTime) |
| 前置条件 | 启动一个查看库存任务 |
| 后置条件 | 关闭一个库存查看任务 |
| inventorybl. InvetoryInfoInventorybl.inventoryCounting | 语法 | public inventoryInfoVO inventoryCounting(Zone zone) |
| 前置条件 | 要求库存盘点 |
| 后置条件 | 可以导出Excel |

|  |  |  |
| --- | --- | --- |
| inventorybl.StorageInFormInventorybl.create | 语法 | public StorageInFormVO StorageInFormcreate () |
| 前置条件 | 无 |
| 后置条件 | 增加一个StorageInFormVO，并将单子改为编辑状态 |
| inventorybl.StorageInFormInventorybl.cancel | 语法 | public void StorageInFormcancel (StorageInFormVO vo) |
| 前置条件 | 单子为编辑状态 |
| 后置条件 | 删除一个StorageInFormVO |
| inventorybl.StorageInFormInventorybl.save | 语法 | public boolean StorageInFormsave (StorageInFormVO vo) |
| 前置条件 | 单子为编辑状态 |
| 后置条件 | 将单子改为草稿状态 |
| inventorybl.StorageInFormInventorybl.submit | 语法 | public boolean StorageInFormsubmit (StorageInFormVO vo) |
| 前置条件 | 单子为草稿状态 |
| 后置条件 | 将单子改为提交状态 |
| inventorybl.StorageInFormInventorybl.withdraw | 语法 | public boolean StorageInFormwithdraw (StorageInFormVO vo) |
| 前置条件 | 单子为提交状态 |
| 后置条件 | 将单子改为草稿状态 |
| inventorybl.StorageInFormInventorybl.storageInFormAddInfo | 语法 | public boolean storageInFormAddInfo (String date, String destination, Zone zone,int line,int shelf,int tag,long ID) |
| 前置条件 | 启动一个新建入库单的任务 |
| 后置条件 | 在一个新建入库单的任务中增加入库信息 |
| inventorybl.StorageOutFormInventorybl.create | 语法 | public StorageOutFormVO StorageOutFormcreate () |
| 前置条件 | 无 |
| 后置条件 | 增加一个StorageOutFormVO，并将单子改为编辑状态 |
| inventorybl.StorageOutFormInventorybl.cancel | 语法 | public void StorageOutFormcancel (StorageOutFormVO vo) |
| 前置条件 | 单子为编辑状态 |
| 后置条件 | 删除一个StorageOutFormVO |
| inventorybl.StorageOutFormInventorybl.save | 语法 | public boolean StorageOutFormsave (StorageOutFormVO vo) |
| 前置条件 | 单子为编辑状态 |
| 后置条件 | 将单子改为草稿状态 |
| inventorybl.StorageOutFormInventorybl.submit | 语法 | public boolean StorageOutFormsubmit (StorageOutFormVO vo) |
| 前置条件 | 单子为草稿状态 |
| 后置条件 | 将单子改为提交状态 |
| inventorybl.StorageOutFormInventorybl.withdraw | 语法 | public boolean StorageOutFormwithdraw (StorageOutFormVO vo) |
| 前置条件 | 单子为提交状态 |
| 后置条件 | 将单子改为草稿状态 |

|  |  |  |
| --- | --- | --- |
| inventorybl.StorageOutFormInventorybl.storageOutFormAddInfo | 语法 | public boolean storageOutFormAddInfo (String date, String destination, Transport transport, String relayNumber,String motorNumber) |
| 前置条件 | 启动一个新建出库单的任务 |
| 后置条件 | 在一个新建出库单的任务中增加出库信息 |
| **需要的服务（需接口）** | | |
| DatabaseFactory.getInventoryDatabase | 得到inventor数据库的服务的引用 | |
| inventoryDataService. inventoryInfoInventorydata.delete(inventoryInfoPO po) | 删除单一持久化inventoryInfoPO对象 | |
| inventoryDataService. inventoryInfoInventorydata.add(inventoryInfoPO po) | 增加单一持久化inventoryInfoPO对象 | |
| inventoryDataService. inventoryInfoInventorydata.update(inventoryInfoPO po) | 更新单一持久化inventoryInfoPO对象 | |
| inventoryDataService. inventoryInfoInventorydata.find(long NO) | 查找单一持久化inventoryInfoPO对象 | |
| inventoryDataService. inventoryInfodata. StorageOutFormInventorydata.insert(StorageOutFormPO po) | 插入单一持久化StorageOutFormPO对象 | |
| inventoryDataService. StorageOutFormInventorydata.delete(StorageOutFormPO po) | 删除单一持久化StorageOutFormPO对象 | |
| inventoryDataService. inventoryInfodata. StorageOutFormInventorydata.update(inventoryInfoPO po) | 更新单一持久化inventoryInfoPO对象 | |
| inventoryDataService. StorageOutFormInventorydata.find(long NO) | 查找单一持久化StorageOutFormPO对象 | |
| inventoryDataService. StorageInFormInventorydata.insert(StorageInFormPO po) | 插入单一持久化StorageInFormPO对象 | |
| inventoryDataService. StorageInFormInventorydata.delete(StorageInFormPO po) | 删除单一持久化StorageInFormPO对象 | |
| inventoryDataService. StorageInFormInventorydata.update(StorageInFormPO po) | 更新单一持久化StorageInFormPO对象 | |
| inventoryDataService. StorageInFormInventorydata.find(long NO) | 根据单据编号查找单一持久化StorageInFormPO对象 | |

#### 4.1.5.4业务逻辑层的动态模型

#### 4.1.5.5业务逻辑的设计原理

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的对象。

### 4.1.6 transitbl模块

#### 4.1.6.1模块概述

Transitbl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。  
Transitbl模块的职责及接口参见软件体系结构描述文档v1.8表5.3.2-1

#### 4.1.6.2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。比如展示层和业务逻辑层之间，我们添加bussinesslogicservice.transitblservice.CarOfficeFormTransitblService接口、

bussinesslogicservice.transitblservice.CarInputFormTransitblService接口、

bussinesslogicservice.transitblservice.DeliveryFormTransitblService接口、

bussinesslogicservice.transitblservice.OfficeArrivalFormTransitblService接口、

bussinesslogicservice.transitblservice.OrderFormTransitblService接口、

bussinesslogicservice.transitblservice.StoreArrivalFormTransitblService接口、

bussinesslogicservice.transitblservice.TransferFormTransitblService接口。业务逻辑层和数据层之间添加dataservice.transitdataservice.CarinputFormTransitdataService接口、

dataservice.transitdataservice.CarOfficeFormTransitdataService接口、

dataservice.transitdataservice.DeliveryFormTransitdataService接口、

dataservice.transitdataservice.OfficeArrivalFormTransitdataService接口、

dataservice.transitdataservice.OrderFormTransitdataService接口、

dataservice.transitdataservice.StoreArrivalFormTransitdataService接口、

dataservice.transitdataservice.TransferFormTransitdataService接口、。为了隔离业务逻辑职责，我们增加了OrderFormController、OfficeArrivalFormController、CarOfficeFormController、DeliveryFormController、StoreArrivalFormController、TransferFormController、CarInputFormControlelr，这样OrderFormController会将对寄件单的业务逻辑处理委托给OrderForm对象，OfficeArrivalFormController会将对营业厅到达单的业务逻辑处理委托给OfficeArrivalFormController对象，CarOfficeFormController会将对营业厅装车单的业务逻辑处理委托给CarOfficeFormController对象，DeliveryFormController会将对派件单的业务逻辑处理委托给DeliveryForm对象，StoreArrivalFormController会将对中转中心到达单的业务逻辑处理委托给StoreArrivalForm对象，TransferFormController会将对中转单的业务逻辑处理委托给TransferForm对象，CarInputFormControlelr会将对中转中心装车单的业务逻辑处理委托给CarInputForm对象。OrderFormPO、OfficeArrivalFormPO、CarOfficeFormPO、DeliveryFormPO、StoreArrivalFormPO、TransferFormPO、CarInputFormPO是作为公司信息的车辆信息和司机信息的持久化对象被添加到设计模型中去的。

Transitbl模块的设计如图4.1.6-1所示。



图4.1.6-1 Transitbl模块各个类的设计

Transitbl模块各个类的职责如表4.1.3-2所示。

**表4.1.6-1 Transitbl模块各个类的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| OrderFormController | 负责实现寄件单的处理 |
| OrderForm | 维护寄件单的领域模型对象，可以帮助完成寄件单处理界面所需要的服务 |
| OfficeArrivaFormController | 负责实现营业厅到达单的处理 |
| OfficeArrivalForm | 维护营业厅到达单的领域模型对象，可以帮助完成营业厅到达单处理界面所需要的服务 |
| CarOfficeFormController | 负责实现营业厅装车单的处理 |
| CarOfficeForm | 维护营业厅装车单的领域模型对象，可以帮助完成营业厅装车单处理界面所需要的服务 |
| DeliveryFormController | 负责实现派件单的处理 |
| DeliveryForm | 维护派件单的领域模型对象，可以帮助完成派件单处理界面所需要的服务 |
| StoreArrivalFormController | 负责实现寄件单的处理 |
| StoreArrivalForm | 维护寄件单的领域模型对象，可以帮助完成寄件单处理界面所需要的服务 |
| TransferFormController | 负责实现寄件单的处理 |
| TransferForm | 维护寄件单的领域模型对象，可以帮助完成寄件单处理界面所需要的服务 |
| CarInputFormController | 负责实现寄件单的处理 |
| CarInputForm | 维护寄件单的领域模型对象，可以帮助完成寄件单处理界面所需要的服务 |

#### 4.1.6.3模块内部类的接口规范

**表4.1.6-2 OrderFormController的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| OrderFormController.  OrderFormcreate | 语法 | | public OrderFormVO OrderFormCreate () |
| 前置条件 | | 已创建一个OrderForm领域对象 |
| 后置条件 | | 调用OrderForm领域对象的OrderFormcreate方法 |
| OrderFormController.  OrderFormcancel | 语法 | | public void OrderFormCancel (OrderFormVO vo) |
| 前置条件 | | 已创建一个OrderForm领域对象 |
| 后置条件 | | 调用OrderForm领域对象的OrderFormcancel方法 |
| OrderFormController.  OrderFormsave | 语法 | | public boolean OrderFormSave (OrderFormVO vo) |
| 前置条件 | | 已创建一个OrderForm领域对象 |
| 后置条件 | | 调用OrderForm领域对象的OrderFormsave方法 |
| OrderFormController.  OrderFormsubmit | 语法 | | public boolean OrderFormSubmit (OrderFormVO vo) |
| 前置条件 | | 已创建一个OrderForm领域对象 |
| 后置条件 | | 调用OrderForm领域对象的OrderFormsubmit方法 |
| OrderFormController.  OrderFormwithdraw | 语法 | | public boolean OrderFormWithdraw (OrderFormVO vo) |
| 前置条件 | | 已创建一个OrderForm领域对象 |
| 后置条件 | | 调用OrderForm领域对象的OrderFormwithdraw方法 |
| OrderFormController.  addOrderer | 语法 | | public Boolean addOrderer(String name, String address, String company, String tel, String mobiletel) |
| 前置条件 | | 已创建一个OrderForm领域对象 |
| 后置条件 | | 调用OrderForm领域对象的addOrderer方法 |
| OrderFormController.  addReceiver | 语法 | | public Boolean addReciever(String name, String address, String company, String tel, String mobiletel) |
| 前置条件 | | 已创建一个OrderForm领域对象 |
| 后置条件 | | 调用OrderForm领域对象的addReceiver方法 |
| OrderFormController.  addGoodsInfo | 语法 | | public boolean addGoodsInfo (int number, double weight, double size, String name) |
| 前置条件 | | 已创建一个OrderForm领域对象 |
| 后置条件 | | 调用OrderForm领域对象的addGoodsInfo方法 |
| OrderFormController.  chooseType | 语法 | | public boolean chooseType(ExpressType type) |
| 前置条件 | | 已创建一个OrderForm领域对象 |
| 后置条件 | | 调用OrderForm领域对象的chooseType方法 |
| OrderFormController.  addPackingCharges | 语法 | | public boolean addPackingCharges(double packingCharges) |
| 前置条件 | | 已创建一个OrderForm领域对象 |
| 后置条件 | | 调用OrdeForm领域对象的addPakingCharges方法 |
| OrderFormController.  calculateCost | 语法 | | public boolean addCalculateCost(double packingCharges) |
| 前置条件 | | 已创建一个OrderForm领域对象 |
| 后置条件 | | 调用OrderForm领域对象的addCalculateCost方法 |
| OrderFormController.  addID | 语法 | | public boolean addID(long ID) |
| 前置条件 | | 已创建一个OrdeForm领域对象 |
| 后置条件 | | 调用OrderForm领域对象的addID方法 |
| OrderFormController.  addRealReceiverandDate | 语法 | | public boolean addID(long ID) |
| 前置条件 | | 已创建一个OrderForm领域对象 |
| 后置条件 | | 调用OrderForm领域对象的addRealReceiverandDate方法 |
| **需要的服务（需接口）** | | | |
| 服务名 | | 服务 | |
| OrderForm.  OrderFormcreate () | | 选择创建寄件单 | |
| OrderForm.  OrderFormcancel (OrderFormVO vo) | | 选择取消寄件单的编辑 | |
| OrderForm.  OrderFormsave (OrderFormVO vo) | | 保存寄件单 | |
| OrderForm. OrderFormsubmit (OrderFormVO vo) | | 提交寄件单 | |
| OrderForm. OrderFormwithdraw (OrderFormVO vo) | | 撤销寄件单 | |
| OrderForm.  addOrderer(String name,String address,String company,String tel,String mobiletel) | | 添加寄件人信息 | |
| OrderForm.  addReceiver (String name,String address,String company,String tel,String mobiletel) | | 添加收件人信息 | |
| OrderForm.  addGoodInfo(int number,double weight,double size,String name) | | 添加快件信息 | |
| OrderForm. chooseType(ExpressType type) | | 添加快递类型 | |
| OrderForm. addPackingCharges(double packingCharges) | | 添加包装费 | |
| OrderForm. addCalculateCost(double packingCharges) | | 计算费用合计 | |
| OrderForm. addID(long ID) | | 添加托运订单条形码号 | |
| OrderForm. addRealReceiveInfo (String name, String date) | | 添加收件人姓名和收件时间 | |

**表4.1.6-3 OfficeArrivalFormController的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| OfficeArrivalFormController. OfficeArrivalCreate | 语法 | public OrderFormVO OfficeArrivalCreate () |
| 前置条件 | 已创建一个OfficeArrivalForm领域对象 |
| 后置条件 | 调用OfficeArrivalForm领域对象的OfficeArrivalCreate方法 |
| OfficeArrivalFormController. OfficeArrivalCancel | 语法 | public void OfficeArrivalCancel (OfficeArrivalFormVO vo) |
| 前置条件 | 已创建一个OfficeArrivalForm领域对象 |
| 后置条件 | 调用OfficeArrivalForm领域对象的OfficeArrivalCancel方法 |
| OfficeArrivalFormController. OfficeArrivalSave | 语法 | public boolean OfficeArrivalSave (OfficeArrivalFormVO vo) |
| 前置条件 | 已创建一个OfficeArrivalForm领域对象 |
| 后置条件 | 调用OfficeArrivalForm领域对象的OfficeArrivalSave方法 |
| OfficeArrivalFormController. OfficeArrivalSubmit | 语法 | public boolean OfficeArrivalSubmit (OfficeArrivalFormVO vo) |
| 前置条件 | 已创建一个OfficeArrivalForm领域对象 |
| 后置条件 | 调用OfficeArrivalForm领域对象的OfficeArrivalSubmit方法 |
| OfficeArrivalFormController. OfficeArrivalWithdraw | 语法 | public boolean OfficeArrivalWithdraw (OfficeArrivalFormVO vo) |
| 前置条件 | 已创建一个OfficeArrivalForm领域对象 |
| 后置条件 | 调用OfficeArrivalForm领域对象的OfficeArrivalWithdraw方法 |
| OfficeArrivalFormController.  addOfficeArrivalInfo | 语法 | public boolean addOfficeArrivalInfo (long relayformnumber,String startingpoint, State state) |
| 前置条件 | 已创建一个OfficeArrivalForm领域对象 |
| 后置条件 | 调用OfficeArrivalForm领域对象的addOfficeArrivalInfo方法 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| OfficeArrivalForm. OfficeArrivalCreate () | 选择创建营业厅到达单 | |
| OfficeArrivalForm. OfficeArrivalCancel (OfficeArrivalFormVO vo) | 选择取消营业厅到达单的编辑 | |
| OfficeArrivalForm. OfficeArrivalSave (OfficeArrivalFormVO vo) | 保存营业厅到达单 | |
| OfficeArrivalForm. OfficeArrivalSubmit (OfficeArrivalFormVO vo) | 提交营业厅到达单 | |
| OfficeArrivalForm. OfficeArrivalWithdraw (OfficeArrivalFormVO vo) | 撤销营业厅到达单 | |
| OfficeArrivalForm.  addOfficeArrivalInfo | 添加营业厅到达单信息 | |

**表4.1.6-4 CarOfficeFormController的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| CarOfficeFormController.  CarOfficeFormcreate | 语法 | public CarOfficeFormVO  CarOfficeFormcreate () |
| 前置条件 | 已创建一个CarOfficeForm领域对象 |
| 后置条件 | 调用CarOfficeForm领域对象的CarOfficeFormCreate方法 |
| CarOfficeFormController.  CarOfficeFormCancel | 语法 | public void CarOfficeFormCancel (CarOfficeFormVO vo) |
| 前置条件 | 已创建一个CarOfficeForm领域对象 |
| 后置条件 | 调用CarOfficeForm领域对象的OrderFormCancel方法 |
| CarOfficeFormController.  CarOfficeFormSave | 语法 | public boolean CarOfficeFormSave (CarOfficeFormVO vo) |
| 前置条件 | 已创建一个CarOfficeForm领域对象 |
| 后置条件 | 调用CarOfficeForm领域对象的CarOfficeFormSave方法 |
| CarOfficeFormController.  CarOfficeFormSubmit | 语法 | public boolean CarOfficeFormSubmit (CarOfficeFormVO vo) |
| 前置条件 | 已创建一个CarOfficeForm领域对象 |
| 后置条件 | 调用CarOfficeForm领域对象的CarOfficeFormsubmit方法 |
| CarOfficeFormController.  CarOfficeFormWithdraw | 语法 | public boolean CarOfficeFormWithdraw (CarOfficeFormVO vo) |
| 前置条件 | 已创建一个CarOfficeForm领域对象 |
| 后置条件 | 调用CarOfficeForm领域对象的CarOfficeFormwithdraw方法 |
| CarOfficeController.  addCarOfficeInfo | 语法 | public boolean addCarOfficeInfo(String motorNumber, String destination, String carNumber, String loadingMember, String supercargo, ArrayList<Long> allIDs) |
| 前置条件 | 已创建一个CarOfficeForm领域对象 |
| 后置条件 | 调用Car Office Form领域对象的add CarOfficeFormInfo方法 |
| CarOfficeFormController.  calculateFreight | 语法 | public double OfficeCalculateFreight (String destination) |
| 前置条件 | 已创建一个CarOfficeForm领域对象 |
| 后置条件 | 调用CarOfficeForm领域对象的OfficeCalculateFreight方法 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| CarOfficeForm. CarOfficeFormcreate () | 选择创建营业厅装车单 | |
| OfficeArrivalForm. CarOfficeForm (CarOfficeFormVO vo) | 选择取消营业厅装车单的编辑 | |
| CarOfficeForm. CarOfficeFormsave (CarOfficeFormVO vo) | 保存营业厅装车单 | |
| CarOfficeForm. CarOfficeFormsubmit (CarOfficeFormVO vo) | 提交营业厅装车单 | |
| CarOfficeForm. CarOfficeFormwithdraw (CarOfficeFormVO vo) | 撤销营业厅装车单 | |
| CarOfficeForm. addCarOfficeInfo(String motorNumber, String destination, String carNumber, String loadingMember, String supercargo, ArrayList<Long> allIDs) | 添加营业厅到装车信息 | |
| CarOfficeForm. OfficeCalculateFreight (String destination) | 计算运费 | |

**表4.1.6-5 DeliveryFormController的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| DeliveryFormController. DeliveryFormCreate | 语法 | public DeliveryFormVO DeliveryFormCreate () |
| 前置条件 | 已创建一个DeliveryForm领域对象 |
| 后置条件 | 调用DeliveryForm领域对象的DeliveryFormCreate方法 |
| DeliveryFormController. DeliveryFormCancel | 语法 | public voidDeliveryFormCancel (DeliveryFormVO vo) |
| 前置条件 | 已创建一个DeliveryForm领域对象 |
| 后置条件 | 调用DeliveryForm领域对象的DeliveryFormCancel方法 |
| DeliveryFormController. DeliveryFormSave | 语法 | public boolean DeliveryFormSave (DeliveryFormVO vo) |
| 前置条件 | 已创建一个DeliveryForm领域对象 |
| 后置条件 | 调用DeliveryForm领域对象的DeliveryFormSave方法 |
| DeliveryFormController. DeliveryFormSubmit | 语法 | public boolean DeliveryFormSubmit (DeliveryFormVO vo) |
| 前置条件 | 已创建一个DeliveryForm领域对象 |
| 后置条件 | 调用DeliveryForm领域对象的DeliveryFormSubmit方法 |
| DeliveryFormController. DeliveryFormWithdraw | 语法 | public boolean DeliveryFormWithdraw (DeliveryFormVO vo) |
| 前置条件 | 已创建一个DeliveryForm领域对象 |
| 后置条件 | 调用DeliveryForm领域对象的DeliveryFormWithdraw方法 |
| DeliveryFormController.  addDeliveryFormInfo | 语法 | public boolean addDeliveryFormInfo (String arrivaeDate, long ID, String deliveryMan) |
| 前置条件 | 已创建一个DeliveryForm领域对象 |
| 后置条件 | 调用DeliveryForm领域对象的addDeliveryFormInfo方法 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| DeliveryForm. DeliveryFormCreate () | 选择创建派件单 | |
| DeliveryForm. DeliveryFormCancel (DeliveryFormVO vo) | 选择取消派件单的编辑 | |
| DeliveryForm. DeliveryFormSave (DeliveryFormVO vo) | 保存派件单 | |
| DeliveryForm. DeliveryFormSubmit (DeliveryFormVO vo) | 提交派件单 | |
| DeliveryForm. DeliveryFormWithdraw (DeliveryFormVO vo) | 撤销派件单 | |
| DeliveryForm. addDeliveryFormInfo(String arrivaeDate, long ID, String deliveryMan) | 添加派件单信息 | |

**表4.1.6-6 StoreArrivalFormController的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| StoreArrivalFormController. StoreArrivalFormCreate | 语法 | public StoreArrivalFormVO StoreArrivalFormCreate () |
| 前置条件 | 已创建一个StoreArrivalForm领域对象 |
| 后置条件 | 调用StoreArrivalForm领域对象的StoreArrivalFormCreate方法 |
| StoreArrivalFormController. StoreArrivalFormCancel | 语法 | public void StoreArrivalFormCancel (StoreArrivalFormVO vo) |
| 前置条件 | 已创建一个StoreArrivalForm领域对象 |
| 后置条件 | 调用StoreArrivalForm领域对象的StoreArrivalFormCancel方法 |
| StoreArrivalFormController. StoreArrivalFormSave | 语法 | public boolean StoreArrivalFormSave (StoreArrivalFormVO vo) |
| 前置条件 | 已创建一个StoreArrivalForm领域对象 |
| 后置条件 | 调用StoreArrivalForm领域对象的StoreArrivalFormSave方法 |
| StoreArrivalFormController. StoreArrivalFormSubmit | 语法 | public boolean StoreArrivalFormSubmit (StoreArrivalFormVO vo) |
| 前置条件 | 已创建一个StoreArrivalForm领域对象 |
| 后置条件 | 调用StoreArrivalForm领域对象的StoreArrivalFormSubmit方法 |
| StoreArrivalFormController. StoreArrivalFormWithdraw | 语法 | public boolean StoreArrivalFormWithdraw (StoreArrivalFormVO vo) |
| 前置条件 | 已创建一个StoreArrivalForm领域对象 |
| 后置条件 | 调用StoreArrivalForm领域对象的StoreArrivalFormWithdraw方法 |
| StoreArrivalFormController.  addStoreArrivalInfo | 语法 | public boolean addStoreArrivalInfo (long relayformnumber ,String startingpoint,State state ) |
| 前置条件 | 已创建一个StoreArrivalForm领域对象 |
| 后置条件 | 调用StoreArrivalForm领域对象的addStoreArrivalInfo方法 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| StoreArrivalForm. StoreArrivalFormCreate () | 选择创建中转中心到达单 | |
| StoreArrivalForm. StoreArrivalFormCancel (StoreArrivalFormVO vo) | 选择取消中转中心到达单的编辑 | |
| StoreArrivalForm. StoreArrivalFormSave (StoreArrivalFormVO vo) | 保存中转中心到达单 | |
| StoreArrivalForm. StoreArrivalFormSubmit (StoreArrivalFormVO vo) | 提交中转中心到达单 | |
| StoreArrivalForm. StoreArrivalFormWithdraw (StoreArrivalFormVO vo) | 撤销中转中心到达单 | |
| StoreArrivalForm.  addStoreArrivalInfo (long relayformnumber ,String startingpoint,State state ) | 添加中转中心到达单信息 | |

**表4.1.6-7 TransferFormController的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| TransferFormController. TransferFormCreate | 语法 | public TransferFormVO TransferFormCreate () |
| 前置条件 | 已创建一个TransferForm领域对象 |
| 后置条件 | 调用TransferForm领域对象的TransferFormCreate方法 |
| TransferFormController. TransferFormCancel | 语法 | public void TransferFormCancel (TransferFormVO vo) |
| 前置条件 | 已创建一个TransferForm领域对象 |
| 后置条件 | 调用TransferForm领域对象的TransferFormCancel方法 |
| TransferFormController. TransferFormSave | 语法 | public boolean TransferFormSave (TransferFormVO vo) |
| 前置条件 | 已创建一个TransferForm领域对象 |
| 后置条件 | 调用TransferForm领域对象的TransferFormSave方法 |
| TransferFormController. TransferFormSubmit | 语法 | public boolean TransferFormSubmit (TransferFormVO vo) |
| 前置条件 | 已创建一个TransferForm领域对象 |
| 后置条件 | 调用TransferForm领域对象的TransferFormSubmit方法 |
| TransferFormController. TransferFormWithdraw | 语法 | public boolean TransferFormWithdraw (TransferFormVO vo) |
| 前置条件 | 已创建一个TransferForm领域对象 |
| 后置条件 | 调用TransferForm领域对象的TransferFormWithdraw方法 |
| TransferFormController.  addTransferFormInfo | 语法 | public boolean addTransferFormInfo (Transport transport,String loadingMember,String relayNumber,String date) |
| 前置条件 | 已创建一个TransferForm领域对象 |
| 后置条件 | 调用TransferForm领域对象的addTransferFormInfo方法 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| TransferForm. TransferFormCreate () | 选择创建中转单 | |
| TransferForm. TransferFormCancel (TransferFormVO vo) | 选择取消中转单的编辑 | |
| TransferForm. TransferFormSave (TransferFormVO vo) | 保存中转单 | |
| TransferForm. TransferFormSubmit (TransferFormVO vo) | 提交中转单 | |
| TransferForm. TransferFormWithdraw (TransferFormVO vo) | 撤销中转单 | |

**表4.1.6-8 CarInputFormController的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| CarInputFormController. CarInputFormCreate | 语法 | public CarInputFormVO CarInputFormCreate () |
| 前置条件 | 已创建一个CarInputForm领域对象 |
| 后置条件 | 调用CarInputForm领域对象的CarInputFormCreate方法 |
| CarInputFormController. CarInputFormCancel | 语法 | public void CarInputFormCancel (CarInputFormVO vo) |
| 前置条件 | 已创建一个CarInputForm领域对象 |
| 后置条件 | 调用CarInputForm领域对象的CarInputFormCancel方法 |
| CarInputFormController. CarInputFormSave | 语法 | public boolean CarInputFormSave (TCarInputFormFormVO vo) |
| 前置条件 | 已创建一个CarInputForm领域对象 |
| 后置条件 | 调用CarInputForm领域对象的CarInputFormSave方法 |
| CarInputFormController. CarInputFormSubmit | 语法 | public boolean CarInputFormSubmit (CarInputFormVO vo) |
| 前置条件 | 已创建一个CarInputForm领域对象 |
| 后置条件 | 调用CarInputForm领域对象的CarInputFormSubmit方法 |
| CarInputFormController. CarInputFormWithdraw | 语法 | public boolean CarInputFormFWithdraw (CarInputFormVO vo) |
| 前置条件 | 已创建一个CarInputForm领域对象 |
| 后置条件 | 调用CarInputForm领域对象的CarInputFormWithdraw方法 |
| CarInputFormController.addInfo | 语法 | public boolean addCarInputFormInfo (String motorNumber, String destination, String carNumber, String loadingMember,String supercargo,String date, ArrayList<Long> allIDs) |
| 前置条件 | 已创建一个CarInputForm领域对象 |
| 后置条件 | 调用CarInputForm领域对象的addInfo方法 |
| CarInputFormController.calculateFreight | 语法 | public double RelayCaculateFreight(String destination) |
| 前置条件 | 已创建一个CarInputForm领域对象 |
| 后置条件 | 调用CarInputForm领域对象的calculateFreight方法 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| CarInputForm. CarInputFormCreate () | 选择创建中转中心装车单 | |
| CarInputForm. CarInputFormCancel (CarInputFormVO vo) | 选择取消中转中心装车的编辑 | |
| CarInputForm. CarInputFormSave (CarInputFormVO vo) | 保存中转中心装车单 | |
| CarInputForm. CarInputFormFormSubmit (CarInputFormVO vo) | 提交中转中心装车单 | |
| CarInputForm. CarInputFormWithdraw (CarInputFormVO vo) | 撤销中转中心装车单 | |

**表4.1.6-9OrderForm的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| OrderForm.  OrderFormCreate | 语法 | | public OrderFormVO OrderFormCreate () |
| 前置条件 | | 无 |
| 后置条件 | | 增加一个OrderFormVO，并将单子状态改为编辑状态 |
| OrderForm.  OrderFormCancel | 语法 | | public void OrderFormCancel (OrderFormVO vo) |
| 前置条件 | | 单子为编辑状态 |
| 后置条件 | | 删除一个OrderFormVO |
| OrderForm.  OrderFormSave | 语法 | | public boolean OrderFormSave (OrderFormVO vo) |
| 前置条件 | | 单子为编辑状态 |
| 后置条件 | | 将单子改为草稿状态 |
| OrderForm.  OrderFormSubmit | 语法 | | public boolean OrderFormSubmit (OrderFormVO vo) |
| 前置条件 | | 单子为草稿状态 |
| 后置条件 | | 将单子改为提交状态 |
| OrderForm.  OrderFormWithdraw | 语法 | | public boolean OrderFormWithdraw (OrderFormVO vo) |
| 前置条件 | | 单子为提交状态 |
| 后置条件 | | 将单子改为草稿状态 |
| OrderForm.  addOrderer | 语法 | | public boolean addOrderer(String name,String address,String company,String tel,String mobiletel) |
| 前置条件 | | 启动一个订单处理任务 |
| 后置条件 | | 在一个订单处理任务中增加寄件人信息 |
| OrderForm.  addReceiver | 语法 | | public boolean addReceiver(String name,String address,String company,String tel,String mobiletel) |
| 前置条件 | | 启动一个订单处理任务 |
| 后置条件 | | 在一个订单处理任务中增加收件人信息 |
| OrderForm.  addGoodsInfo | 语法 | | public boolean addGoodsInfo(int number,double weight,double size,String name) |
| 前置条件 | | 启动一个订单处理任务 |
| 后置条件 | | 在一个订单处理任务中增加托运货物信息 |
| OrderForm.  chooseType | 语法 | | public boolean chooseType(ExpressType type) |
| 前置条件 | | 启动一个订单处理任务 |
| 后置条件 | | 在一个订单处理任务中增加快递种类信息 |
| OrderForm.  addPackingCharges | 语法 | | public boolean addPackingCharges(double packingCharges) |
| 前置条件 | | 启动一个订单处理任务 |
| 后置条件 | | 在一个订单处理任务中增加包装费信息 |
| OrderForm.  caculateCost | 语法 | | public boolean addCalculateCost(double packingCharges) |
| 前置条件 | | 已添加包装费 |
| 后置条件 | | 在一个订单处理任务中自动计算费用合计：运费加包装费 |
| OrderForm.  addID | 语法 | | public boolean addID(long ID) |
| 前置条件 | | 启动订单处理任务 |
| 后置条件 | | 在订单处理任务中增加订单号 |
| OrderForm.  addRealReceiverandDate | 语法 | | public boolean addRealReceiveInfo (String name, String date) |
| 前置条件 | | 已经根据订单号找到相应订单，启动一个收件信息输入任务 |
| 后置条件 | | 在一个收件信息输入任务中增加收件姓名和收件时间 |
| **需要的服务（需接口）** | | | |
| DatabaseFactory.getTransitDatabase | | 得到Transit数据库的服务的引用 | |
| TransitDataService. OrderFormdata.insert OrderForm PO po) | | 插入单一持久化OrderFormPO对象 | |
| TransitDataService. OrderFormdata.delete(OrderForm PO po) | | 删除单一持久化OrderFormPO对象 | |
| TransitDataService. OrderFormdata.update(OrderForm PO po) | | 更新单一持久化OrderFormPO对象 | |
| TransitDataService. OrderFormdata.find(long NO) | | 根据编号查找单一持久化OrderFormPO对象 | |
| constblservice.ConstInfoblService.find() | | 查找常量信息 | |

**表4.1.6-10 OfficeArrivalForm的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| OfficeArrivalForm. OfficeArrivalFormCreate | 语法 | | public OfficeArrivalFormVO OfficeArrivalFormCreate () |
| 前置条件 | | 无 |
| 后置条件 | | 增加一个OfficeArrivalFormVO，并将单子状态改为编辑状态 |
| OfficeArrivalForm. OfficeArrivalFormCancel | 语法 | | public void OfficeArrivalFormCancel (OfficeArrivalFormVO vo) |
| 前置条件 | | 单子为编辑状态 |
| 后置条件 | | 删除一个OfficeArrivalFormVO |
| OfficeArrivalForm. OfficeArrivalFormSave | 语法 | | public boolean OfficeArrivalFormSave (OfficeArrivalFormVO vo) |
| 前置条件 | | 单子为编辑状态 |
| 后置条件 | | 将单子改为草稿状态 |
| OfficeArrivalForm. OfficeArrivalFormSubmit | 语法 | | public boolean OfficeArrivalFormSubmit (OfficeArrivalFormVO vo) |
| 前置条件 | | 单子为草稿状态 |
| 后置条件 | | 将单子改为提交状态 |
| OfficeArrivalForm. OfficeArrivalFormWithdraw | 语法 | | public boolean OfficeArrivalFormWithdraw (OfficeArrivalFormVO vo) |
| 前置条件 | | 单子为提交状态 |
| 后置条件 | | 将单子改为草稿状态 |
| **需要的服务（需接口）** | | | |
| DatabaseFactory.getTransitDatabase | | 得到Transit数据库的服务的引用 | |
| TransitDataService. OfficeArrivalFormdata.insert(OfficeFormPO po) | | 插入单一持久化OfficeArrivalFormPO对象 | |
| TransitDataService. OfficeArrivalFormdata.delete(OfficeFormPO po) | | 删除单一持久化OfficeArrivalFormPO对象 | |
| TransitDataService. OfficeArrivalFormdata.update(OfficeFormPO po) | | 更新单一持久化OfficeArrivalFormPO对象 | |
| TransitDataService. OfficeArrivalFormdata.find(long NO) | | 根据编号查找单一持久化OfficeArrivalFormPO对象 | |

**表4.1.6-11 CarOfficeForm的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| CarOfficeForm.  CarOfficeFormCreate | 语法 | | public CarOfficeFormVO CarOfficeFormCreate () |
| 前置条件 | | 无 |
| 后置条件 | | 增加一个CarOfficeFormVO，并将单子状态改为编辑状态 |
| CarOfficeForm.  CarOfficeFormCancel | 语法 | | public void CarOfficeForm Cancel (CarOfficeFormVO vo) |
| 前置条件 | | 单子为编辑状态 |
| 后置条件 | | 删除一个CarOfficeFormVO |
| CarOfficeForm.  CarOfficeFormSave | 语法 | | public boolean CarOfficeFormSave (CarOfficeFormVO vo) |
| 前置条件 | | 单子为编辑状态 |
| 后置条件 | | 将单子改为草稿状态 |
| CarOfficeForm.  CarOfficeFormSubmit | 语法 | | public boolean CarOfficeFormSubmit (CarOfficeFormVO vo) |
| 前置条件 | | 单子为草稿状态 |
| 后置条件 | | 将单子改为提交状态 |
| CarOfficeForm. CarOfficeFormWithdraw | 语法 | | public boolean CarOfficeFormWithdraw (CarOfficeFormVO vo) |
| 前置条件 | | 单子为提交状态 |
| 后置条件 | | 将单子改为草稿状态 |
| **需要的服务（需接口）** | | | |
| DatabaseFactory.  getTransitDatabase | | 得到Transit数据库的服务的引用 | |
| TransitDataService. CarOfficeFormdata.insert(CarOfficeFormPO po) | | 插入单一持久化CarOfficeFormPO对象 | |
| TransitDataService. CarOfficeFormdata.delete(CarOfficeFormPO po) | | 删除单一持久化CarOfficeFormPO对象 | |
| TransitDataService. CarOfficeFormdata.update(CarOfficeFormPO po) | | 更新单一持久化CarOfficeFormPO对象 | |
| TransitDataService.  CarOfficeFormdata.find(long NO) | | 根据编号查找单一持久化CarOfficeFormPO对象 | |

**表4.1.6-12 DeliveryForm的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| DeliveryForm.  DeliveryFormCreate | 语法 | | public DeliveryFormVO DeliveryFormCreate () |
| 前置条件 | | 无 |
| 后置条件 | | 增加一个DeliveryFormVO，并将单子状态改为编辑状态 |
| DeliveryForm.  DeliveryFormCancel | 语法 | | public void DeliveryFormCancel (DeliveryFormVO vo) |
| 前置条件 | | 单子为编辑状态 |
| 后置条件 | | 删除一个DeliveryFormVO |
| DeliveryForm.  DeliveryFormSave | 语法 | | public boolean DeliveryFormSave (DeliveryFormVO vo) |
| 前置条件 | | 单子为编辑状态 |
| 后置条件 | | 将单子改为草稿状态 |
| DeliveryForm.  DeliveryFormSubmit | 语法 | | public boolean DeliveryFormSubmit (DeliveryFormVO vo) |
| 前置条件 | | 单子为草稿状态 |
| 后置条件 | | 将单子改为提交状态 |
| DeliveryForm.  DeliveryFormWithdraw | 语法 | | public boolean DeliveryFormWithdraw (DeliveryFormVO vo) |
| 前置条件 | | 单子为提交状态 |
| 后置条件 | | 将单子改为草稿状态 |
| DeliveryForm.  addDeliveryFormInfo | 语法 | | public boolean addDeliveryFormInfo (String arrivaeDate, long ID, String deliveryMan) |
| 前置条件 | | 启动一个新建派件单任务 |
| 后置条件 | | 在一个新建派件单任务中增加派件信息 |
| **需要的服务（需接口）** | | | |
| DatabaseFactory.  getTransitDatabase | | 得到Transit数据库的服务的引用 | |
| TransitDataService. DeliveryFormdata.insert(DeliveryFormPO po) | | 插入单一持久化DeliveryFormPO对象 | |
| TransitDataService. DeliveryFormdata.delete(DeliveryFormPO po) | | 删除单一持久化DeliveryFormPO对象 | |
| TransitDataService. DeliveryFormdata.update(DeliveryFormPO po) | | 更新单一持久化DeliveryFormPO对象 | |
| TransitDataService. DeliveryFormdata.find(long NO) | | 根据编号查找单一持久化DeliveryFormPO对象 | |

**表4.1.6 - StoreArrivalForm的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| StoreArrivalForm. StoreArrivalFormCreate | 语法 | | public StoreArrivalForm VO StoreArrivalForm Create () |
| 前置条件 | | 无 |
| 后置条件 | | 增加一个StoreArrivalForm VO，并将单子状态改为编辑状态 |
| StoreArrivalForm. StoreArrivalFormCancel | 语法 | | public void StoreArrivalForm Cancel (StoreArrivalFormVO vo) |
| 前置条件 | | 单子为编辑状态 |
| 后置条件 | | 删除一个StoreArrivalFormVO |
| StoreArrivalForm. StoreArrivalFormSave | 语法 | | public boolean StoreArrivalFormSave (StoreArrivalFormVO vo) |
| 前置条件 | | 单子为编辑状态 |
| 后置条件 | | 将单子改为草稿状态 |
| StoreArrivalForm. StoreArrivalFormSubmit | 语法 | | public boolean StoreArrivalFormSubmit (StoreArrivalFormVO vo) |
| 前置条件 | | 单子为草稿状态 |
| 后置条件 | | 将单子改为提交状态 |
| StoreArrivalForm. StoreArrivalFormWithdraw | 语法 | | public boolean StoreArrivalFormWithdraw (StoreArrivalFormVO vo) |
| 前置条件 | | 单子为提交状态 |
| 后置条件 | | 将单子改为草稿状态 |
| StoreArrivalForm.  addStoreArrivalInfo | 语法 | | public boolean addStoreArrivalInfo (long relayformnumber ,String startingpoint,State state ) |
| 前置条件 | | 启动一个新建中转中心到达单任务 |
| 后置条件 | | 在一个新建中转中心到达单任务中增加到达信息 |
| **需要的服务（需接口）** | | | |
| DatabaseFactory.getTransitDatabase | | 得到Transit数据库的服务的引用 | |
| TransitDataService. StoreArrivalFormdata.insert(StoreArrivalFormPO po) | | 插入单一持久化StoreArrivalFormPO对象 | |
| TransitDataService. StoreArrivalFormdata.delete(StoreArrivalFormPO po) | | 删除单一持久化StoreArrivalFormPO对象 | |
| TransitDataService. StoreArrivalFormdata.update(StoreArrivalFormPO po) | | 更新单一持久化StoreArrivalFormPO对象 | |
| TransitDataService. StoreArrivalFormdata.find(long NO) | | 根据编号查找单一持久化StoreArrivalFormPO对象 | |

**表4.1.6-14 TransferForm的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| TransferForm.  TransferFormCreate | 语法 | | public TransferFormVO TransferFormCreate () |
| 前置条件 | | 无 |
| 后置条件 | | 增加一个TransferFormVO，并将单子状态改为编辑状态 |
| TransferForm.  TransferFormCancel | 语法 | | public void TransferFormCancel (TransferFormVO vo) |
| 前置条件 | | 单子为编辑状态 |
| 后置条件 | | 删除一个TransferFormVO |
| TransferForm.  TransferFormSave | 语法 | | public boolean TransferFormSave (TransferFormVO vo) |
| 前置条件 | | 单子为编辑状态 |
| 后置条件 | | 将单子改为草稿状态 |
| TransferForm.  TransferFormSubmit | 语法 | | public boolean TransferFormSubmit TransferFormVO vo) |
| 前置条件 | | 单子为草稿状态 |
| 后置条件 | | 将单子改为提交状态 |
| TransferForm.  TransferFormWithdraw | 语法 | | public boolean TransferFormWithdraw (TransferFormVO vo) |
| 前置条件 | | 单子为提交状态 |
| 后置条件 | | 将单子改为草稿状态 |
| TransferForm.  addTransferFormInfo | 语法 | | public boolean addTransferFormInfo (Transport transport,String loadingMember,String relayNumber,String date) |
| 前置条件 | | 启动一个新建中转单任务 |
| 后置条件 | | 在一个中转单任务中增加中转信息 |
| **需要的服务（需接口）** | | | |
| DatabaseFactory.  getTransitDatabase | | 得到Transit数据库的服务的引用 | |
| TransitDataService. TransferFormdata.insert(TransferFormPO po) | | 插入单一持久化TransferFormPO对象 | |
| TransitDataService. TransferFormdata.delete(TransferFormPO po) | | 删除单一持久化TransferFormPO对象 | |
| TransitDataService. TransferFormdata.update(TransferFormPO po) | | 更新单一持久化TransferFormPO对象 | |
| TransitDataService. TransferFormdata.find(long NO) | | 根据编号查找单一持久化TransferFormPO对象 | |
| corporationblservice.CarInfocorporationblSerivce.find(String plateNumber) | | 根据车牌号查找CarInfo对象 | |

**表4.1.6-15 CarInputForm的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| CarInputForm.  CarInputFormCreate | 语法 | public CarInputFormVO CarInputFormCreate () |
| 前置条件 | 无 |
| 后置条件 | 增加一个CarInputFormVO，并将单子状态改为编辑状态 |
| CarInputForm.  CarInputFormCancel | 语法 | public void CarInputForm Cancel (CarInputFormVO vo) |
| 前置条件 | 单子为编辑状态 |
| 后置条件 | 删除一个CarInputFormVO |
| CarInputForm.  CarInputFormSave | 语法 | public boolean CarInputFormSave (CarInputFormVO vo) |
| 前置条件 | 单子为编辑状态 |
| 后置条件 | 将单子改为草稿状态 |
| CarInputForm.  CarInputFormSubmit | 语法 | public boolean CarInputFormSubmit CarInputFormVO vo) |
| 前置条件 | 单子为草稿状态 |
| 后置条件 | 将单子改为提交状态 |
| CarInputForm.  CarInputFormWithdraw | 语法 | public boolean CarInputFormWithdraw (CarInputFormVO vo) |
| 前置条件 | 单子为提交状态 |
| 后置条件 | 将单子改为草稿状态 |
| carInputForm.  addCarInputFormInfo | 语法 | public boolean addCarInputFormInfo (String motorNumber, String destination, String carNumber, String loadingMember,String supercargo,String date, ArrayList<Long> allIDs) |
| 前置条件 | 启动一个新建中转中心装车单任务 |
| 后置条件 | 在一个中转中心装车单任务中增加装车信息 |
| carInputForm.  RelayCaculateFreight | 语法 | public double RelayCaculateFreight(String destination) |
| 前置条件 | 已添加到达地信息 |
| 后置条件 | 自动根据出发地和到达地计算运费 |
| **需要的服务（需接口）** | | |
| DatabaseFactory.getTransitDatabase | | 得到Transit数据库的服务的引用 |
| TransitDataService. CarInputFormdata.insert(CarInputFormPO po) | | 插入单一持久化CarInputFormPO对象 |
| TransitDataService. CarInputFormdata.delete(CarInputFormPO po) | | 删除单一持久化CarInputFormPO对象 |
| TransitDataService. CarInputFormdata.update(CarInputFormPO po) | | 更新单一持久化CarInputFormPO对象 |
| TransitDataService. CarInputFormdata.find(long NO) | | 根据编号查找单一持久化CarInputFormPO对象 |
| corporationblservice.CarInfocorporationblSerivce.find(String plateNumber) | | 根据车牌号查找CarInfo对象 |

#### 4.1.6.4业务逻辑层的动态模型

1）OrderForm



图 4.1.6.4-1 OrderForm\_Create,Save,Cancel,Submit,update and Withdraw



图4.1.6.4-2 OrderForm\_Find

2)OfficeArrivalForm

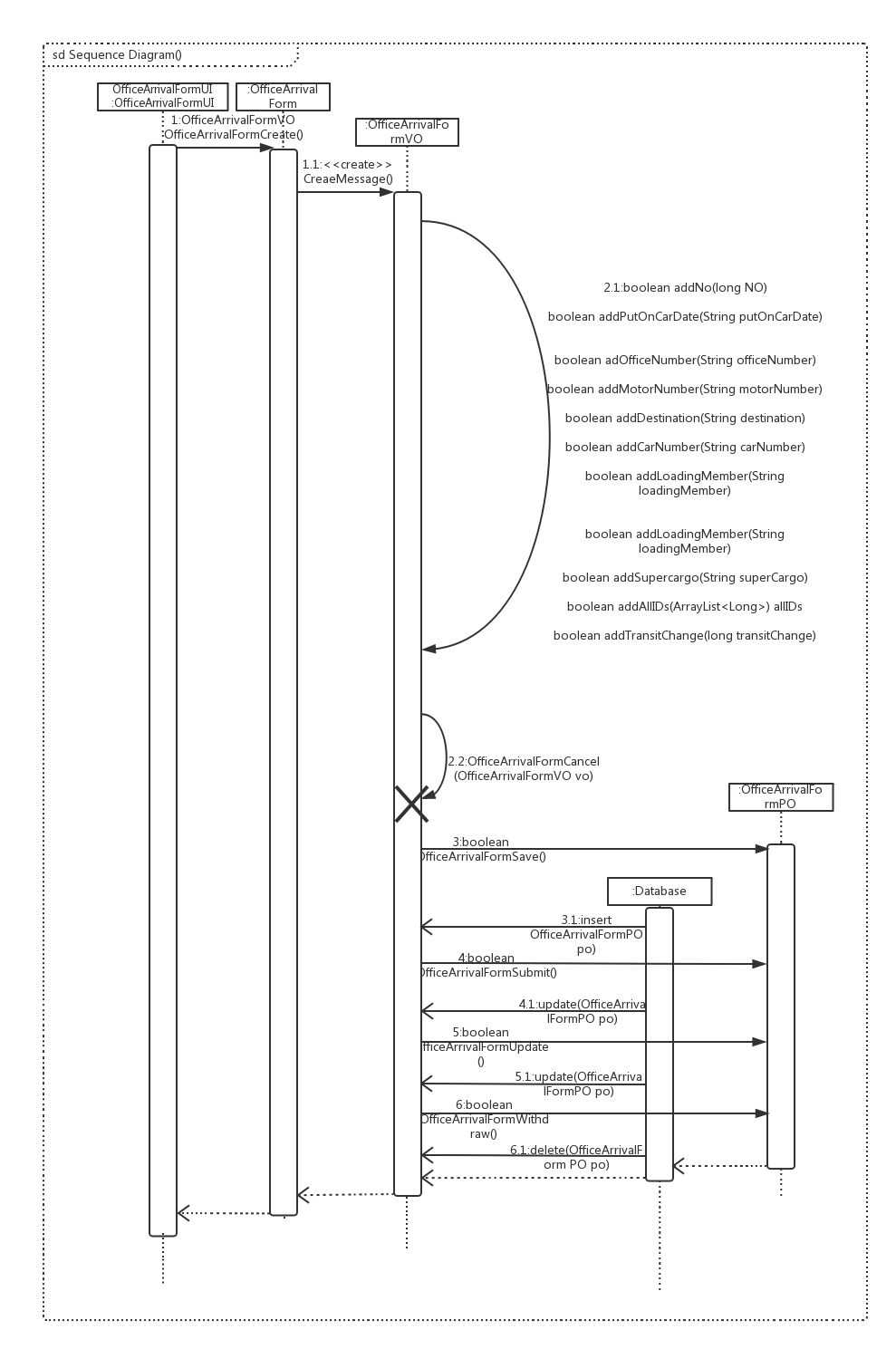


图 4.1.6.4-3 OfficeArrivalForm \_Create,Save,Cancel,Submit,update and Withdraw

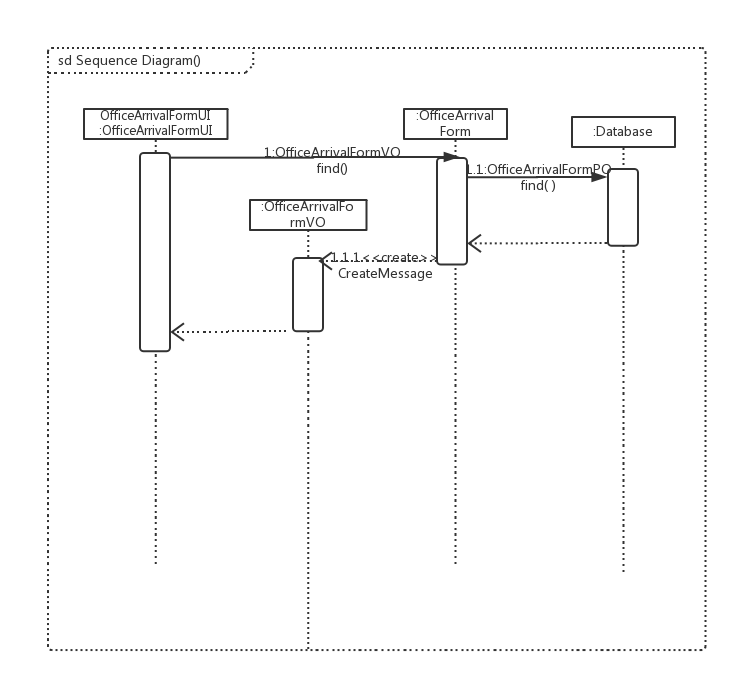


图 4.1.6.4-4 OfficeArrivalForm \_Find

3)CarOfficeForm

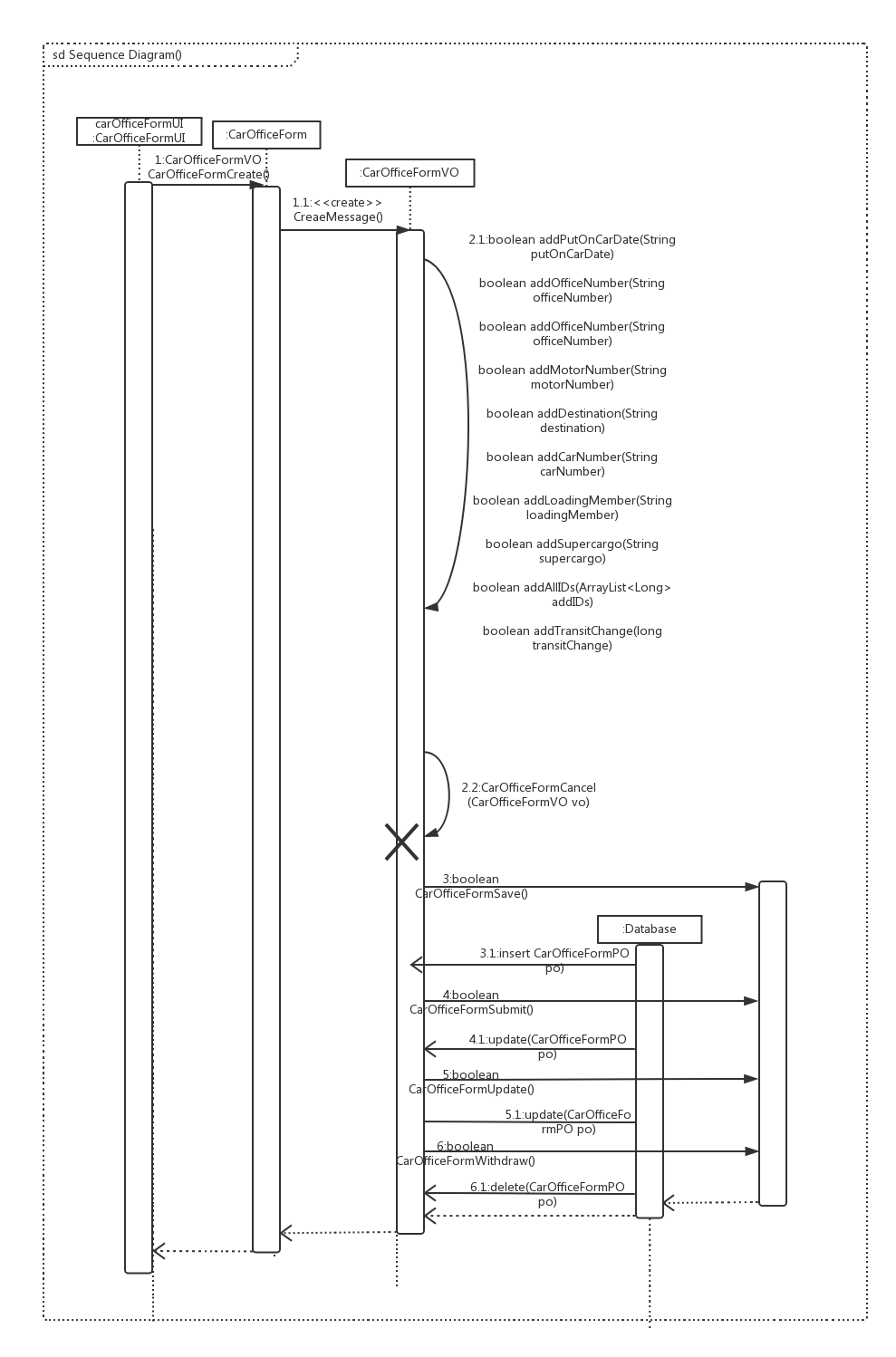


图4.1.6.4-5 CarOfficeForm \_Create,Save,

Cancel,Submit,update and Withdraw

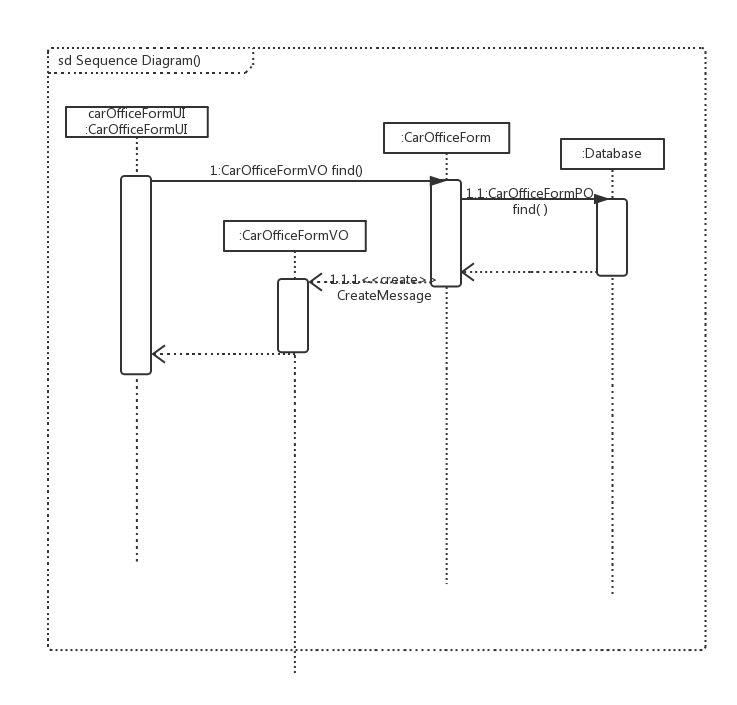


图 4.1.6.4-6 CarOfficeForm \_Find

4)DeliveryFrom

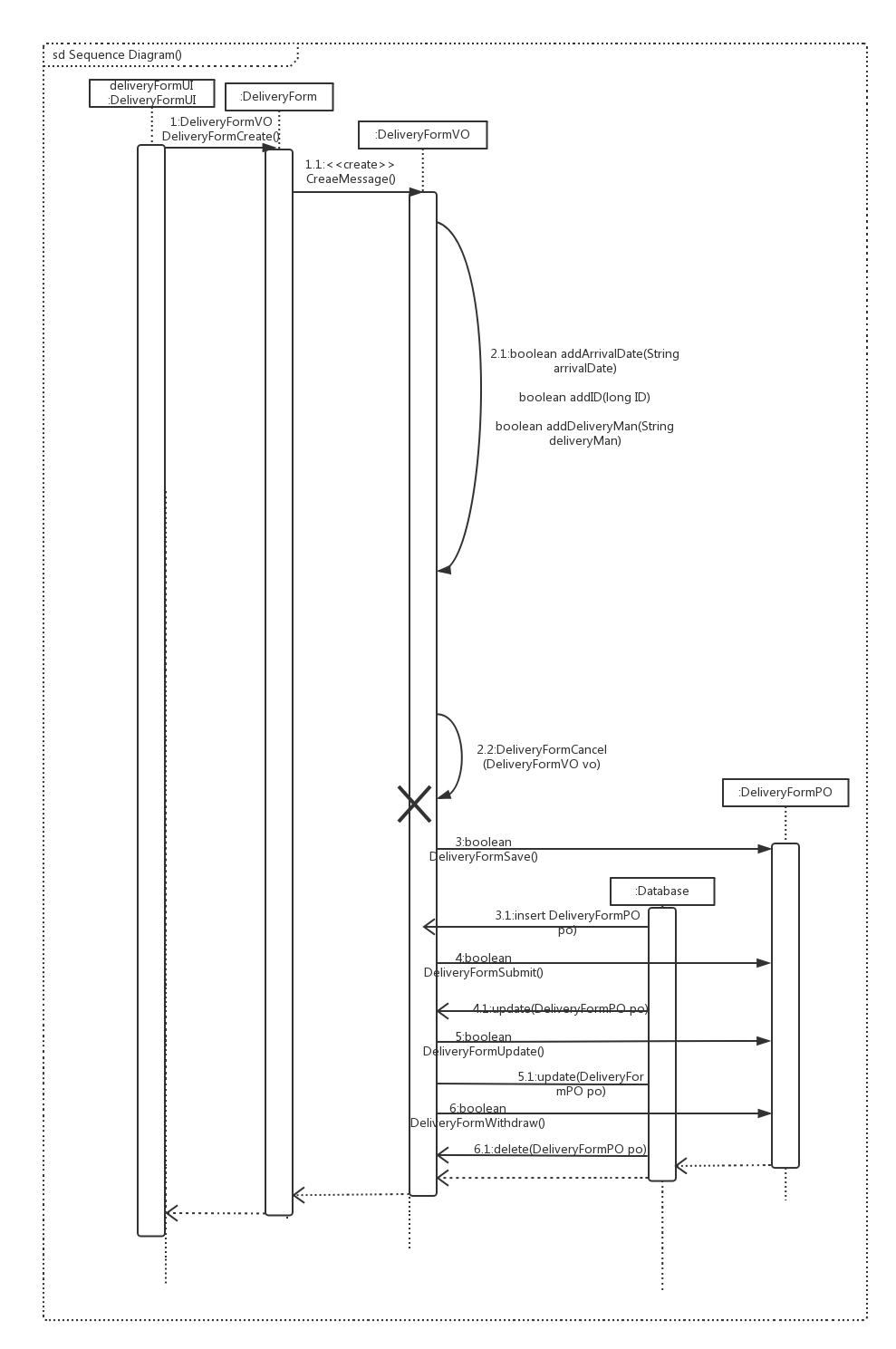


图4.1.6.4-7 DeliveryFrom \_Create,Save,

Cancel,Submit,update and Withdraw

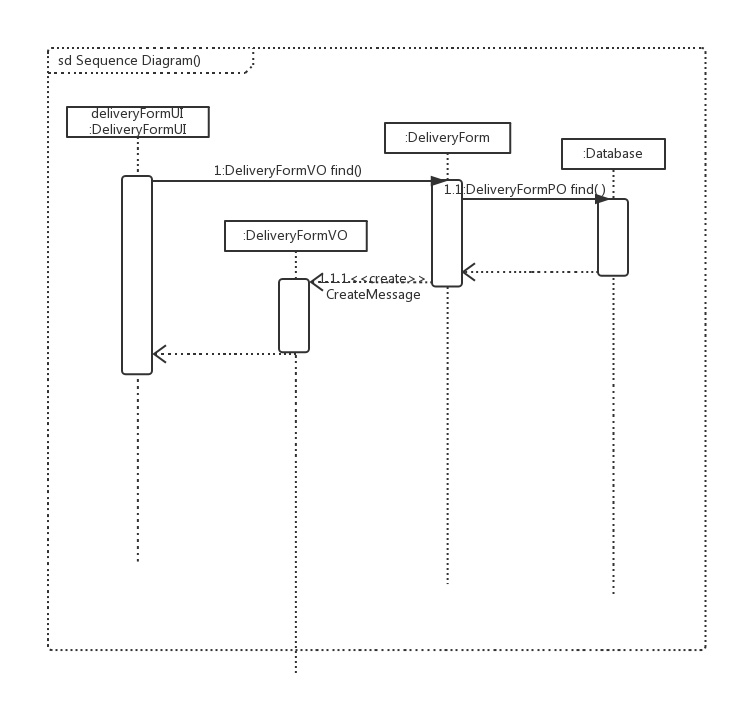


图 4.1.6.4-8 DeliveryFrom \_Find

5)StoreArrivalForm

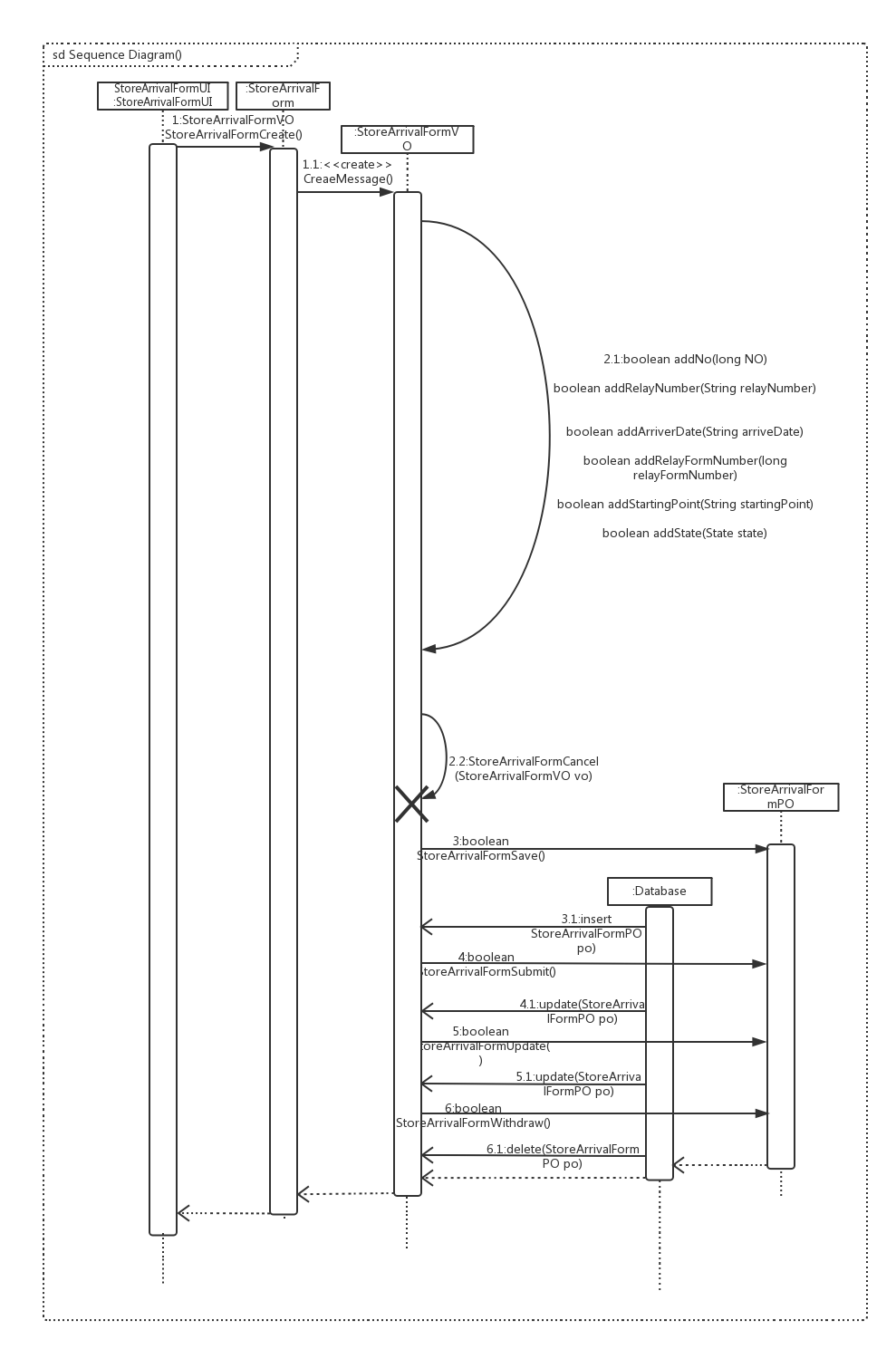


图4.1.6.4-9 StoreArrivalForm \_Create,Save,

Cancel,Submit,update and Withdraw

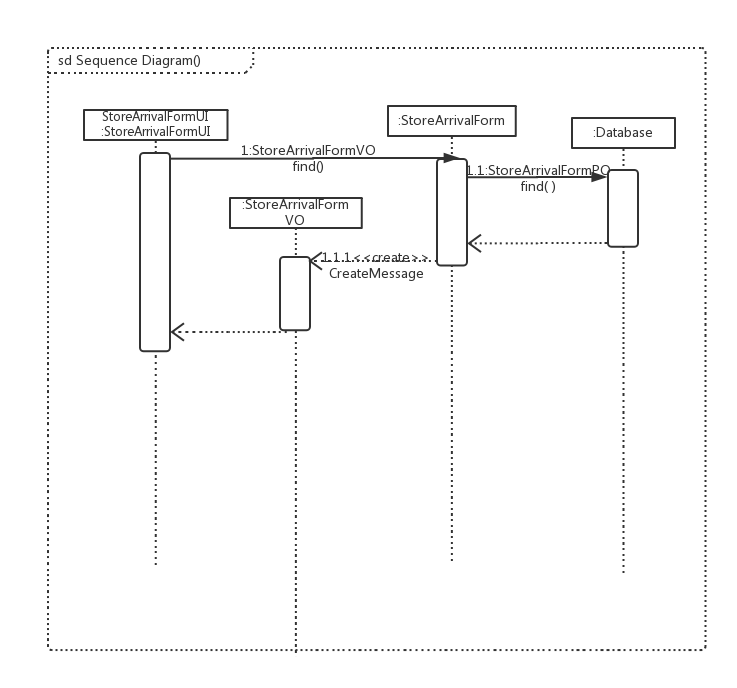


图 4.1.6.4-10 StoreArrivalForm \_Find

6)TransferForm

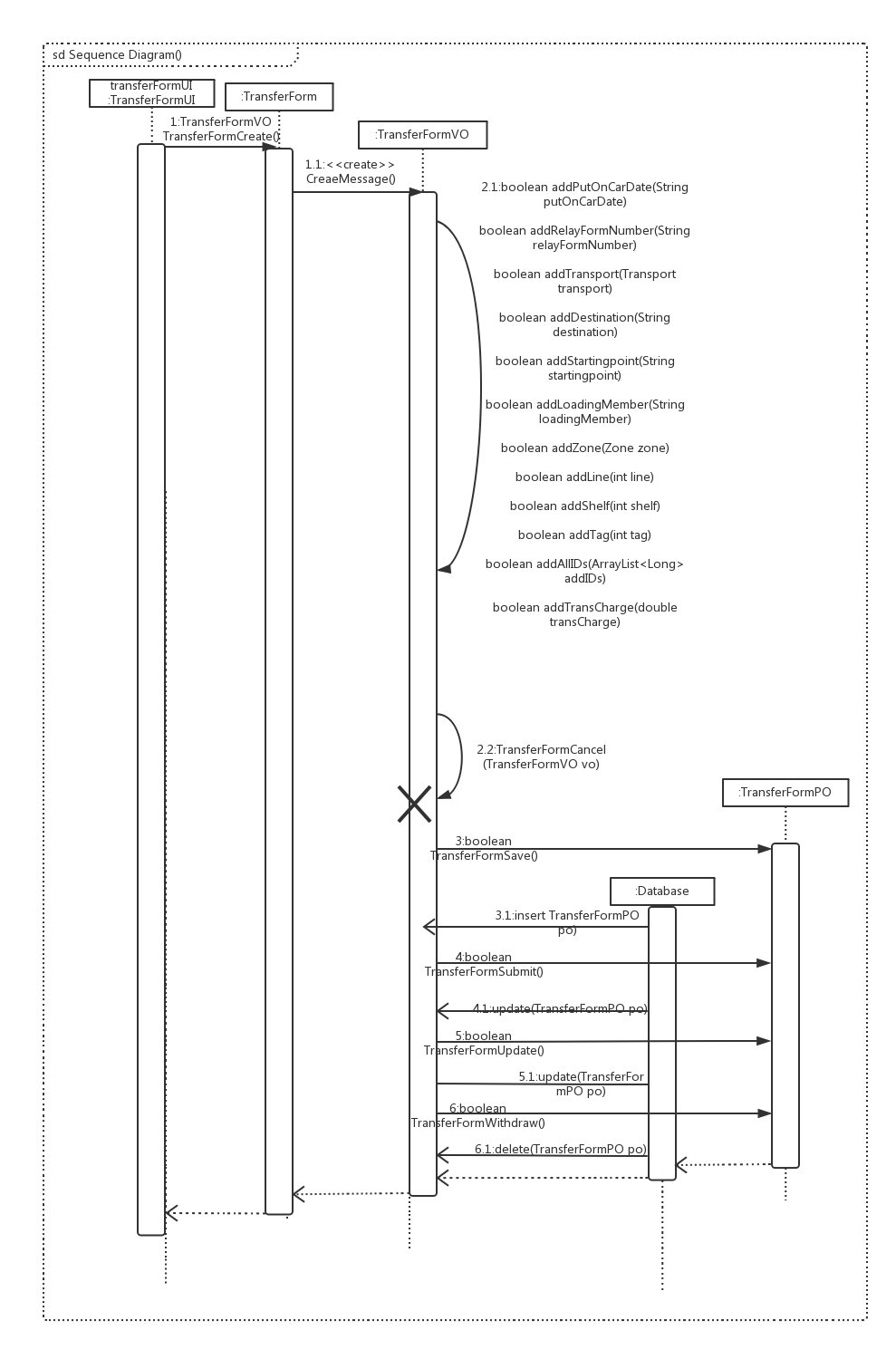


图4.1.6.4-11 TransferForm \_Create,Save,

Cancel,Submit,update and Withdraw

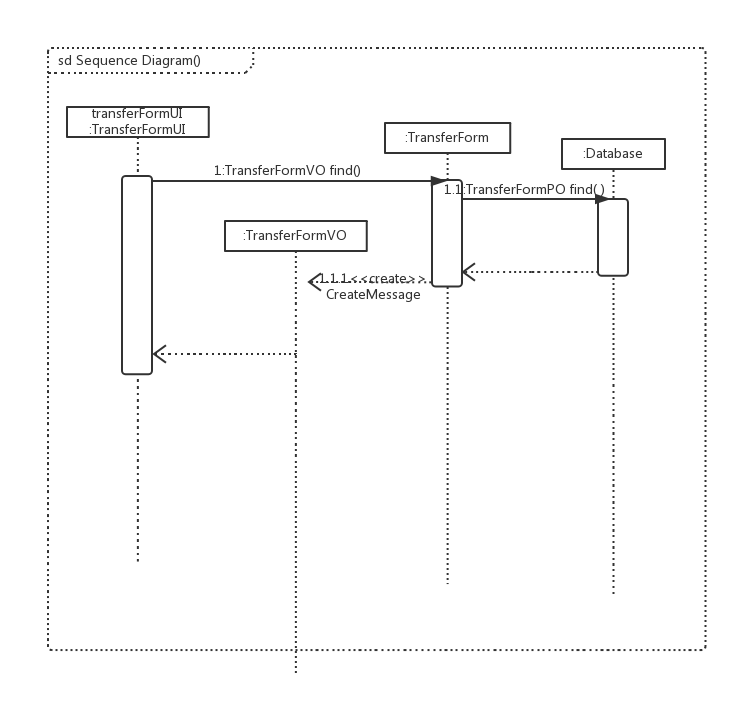


图 4.1.6.4-12 TransferForm \_Find

7)CarInputForm

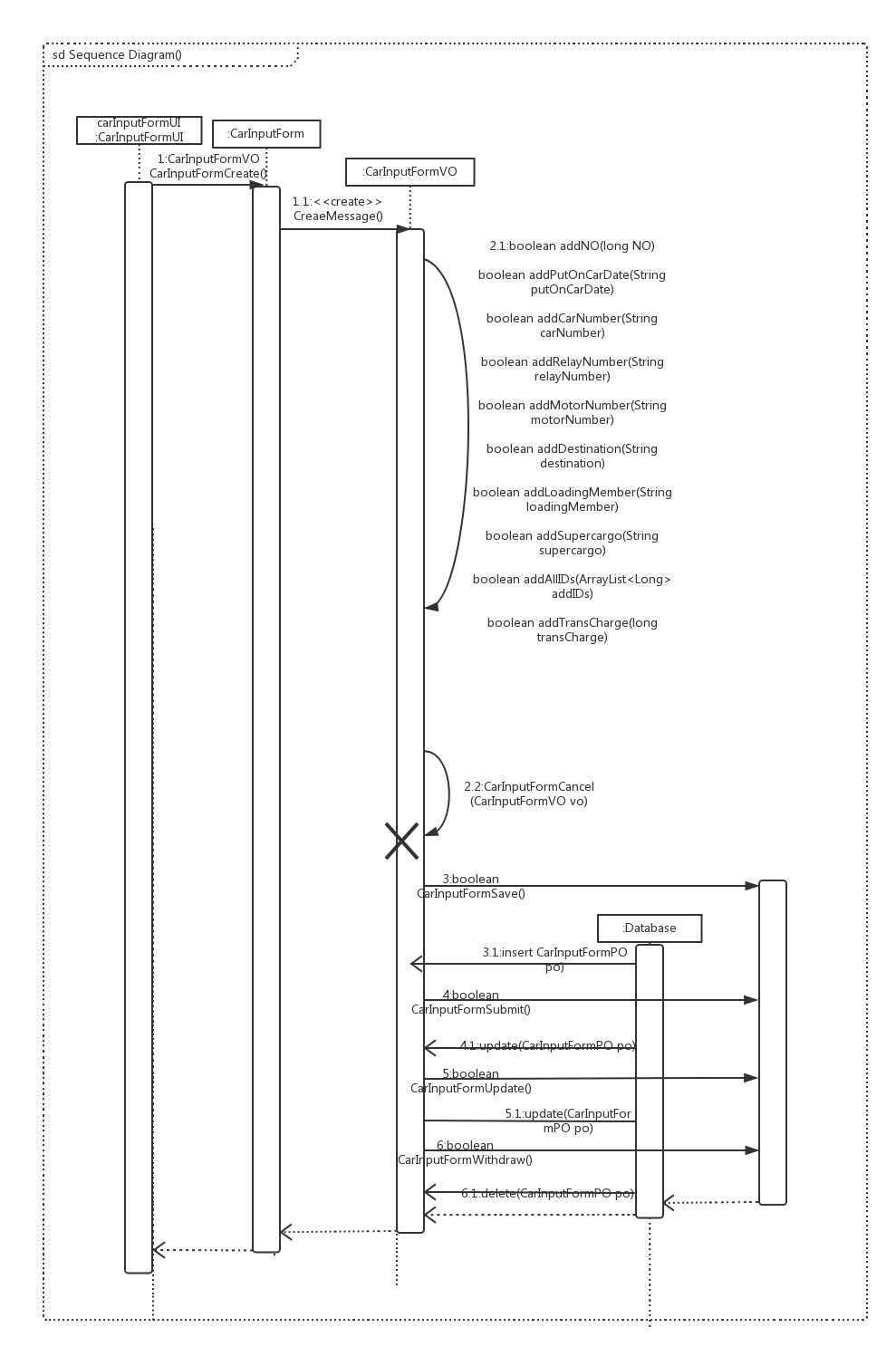


图4.1.6.4-13 CarInputForm \_Create,Save,

Cancel,Submit,update and Withdraw

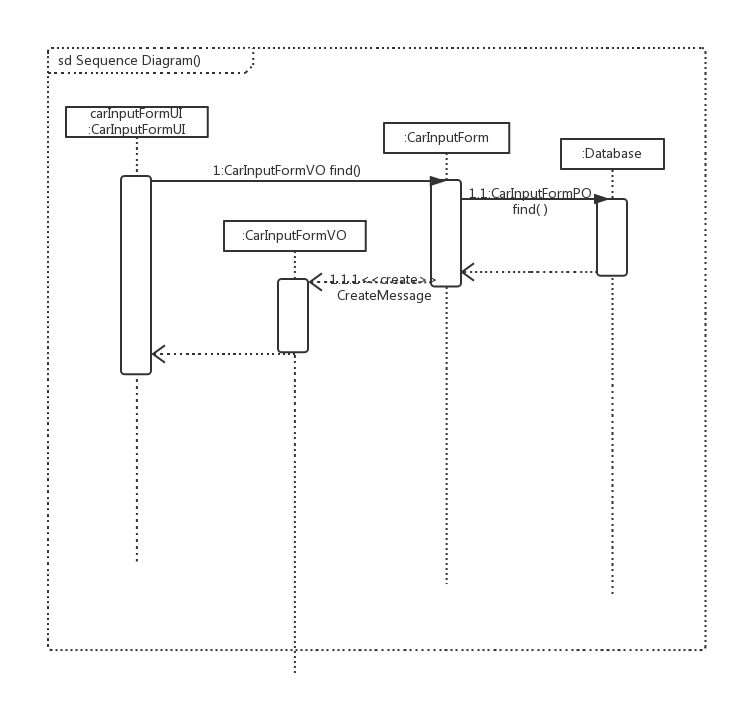


图 4.1.6.4-13 CarInputForm \_Find

#### 4.1.6.5业务逻辑的设计原理

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的对象。

### 4.1.7 formapprovalbl模块

#### **4.1.7.1模块概述**

 Formapprovalbl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。  
 Formapprovalbl模块的职责及接口参见软件体系结构描述文档v1.8表5.3.2-7

#### **4.1.7.2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。比如展示层和业务逻辑层之间，我们添加businesslogicservice.formapprovalblservice.CarInputFormFormapprovalblService接口、businesslogicservice.formapprovalblservice.CarOfficeFormFormapprovalblService接口、

businesslogicservice.formapprovalblservice.DeliveryFormFormapprovalblService接口、

businesslogicservice.formapprovalblservice.OfficeArrivalFormFormapprovalblService接口、

businesslogicservice.formapprovalblservice.OrderFormFormapprovalblService接口、

businesslogicservice.formapprovalblservice.PaymentFormFormapprovalblService接口、

businesslogicservice.formapprovalblservice.ReceiptFormFormapprovalblService接口、

businesslogicservice.formapprovalblservice.StorageInFormFormapprovalblService接口、

businesslogicservice.formapprovalblservice.StorageOutFormFormapprovalblService接口、

businesslogicservice.formapprovalblservice.StoreArrivalFormFormapprovalblService接口、

businesslogicservice.formapprovalblservice.TransferFormFormapprovalblService接口。业务逻辑层和数据层之间添加dataservice.transitdataservice.CarInputFormTransitdataService接口、dataservice.transitdataservice.CarOfficeTransitdataService接口、dataservice.transitdataservice.DeliveryFormTransitdataService接口、dataservice.transitdataservice.OfficeArrivalFormTransitdataService接口、dataservice.transitdataservice.OrderFormTransitdataService接口、dataservice.financedataservice.PaymentFormdataService接口、dataservice.financedataservice.ReceiptFormdataService接口、dataservice.inventorydataservice.StorageInFormInventorydataService接口、dataservice.inventorydataservice.StorageOutFromInventorydataService接口、dataservice.transitdataservice.StoreArrivalFormTransitdataService接口、dataservice.transitdataservice.TransferFormdTransitataService接口。为了隔离业务逻辑职责，我们增加了FormapprovalController，这样FormapprovalController会将对用户的业务逻辑处理委托给Formapproval对象。CarInputFormPO、CarOfficeFormPO、DeliveryFormPO、OfficeArrivalFormPO、OrderFormPO、PaymentFormPO、ReceiptFormPO、StorageInFormPO、StorageOutFormPO、StoreArrivalFormPO、TransferFormPO是作为用户信息的持久化对象被添加到设计模型中去的。

formapprovalbl模块的设计如图4.1.7-1所示。

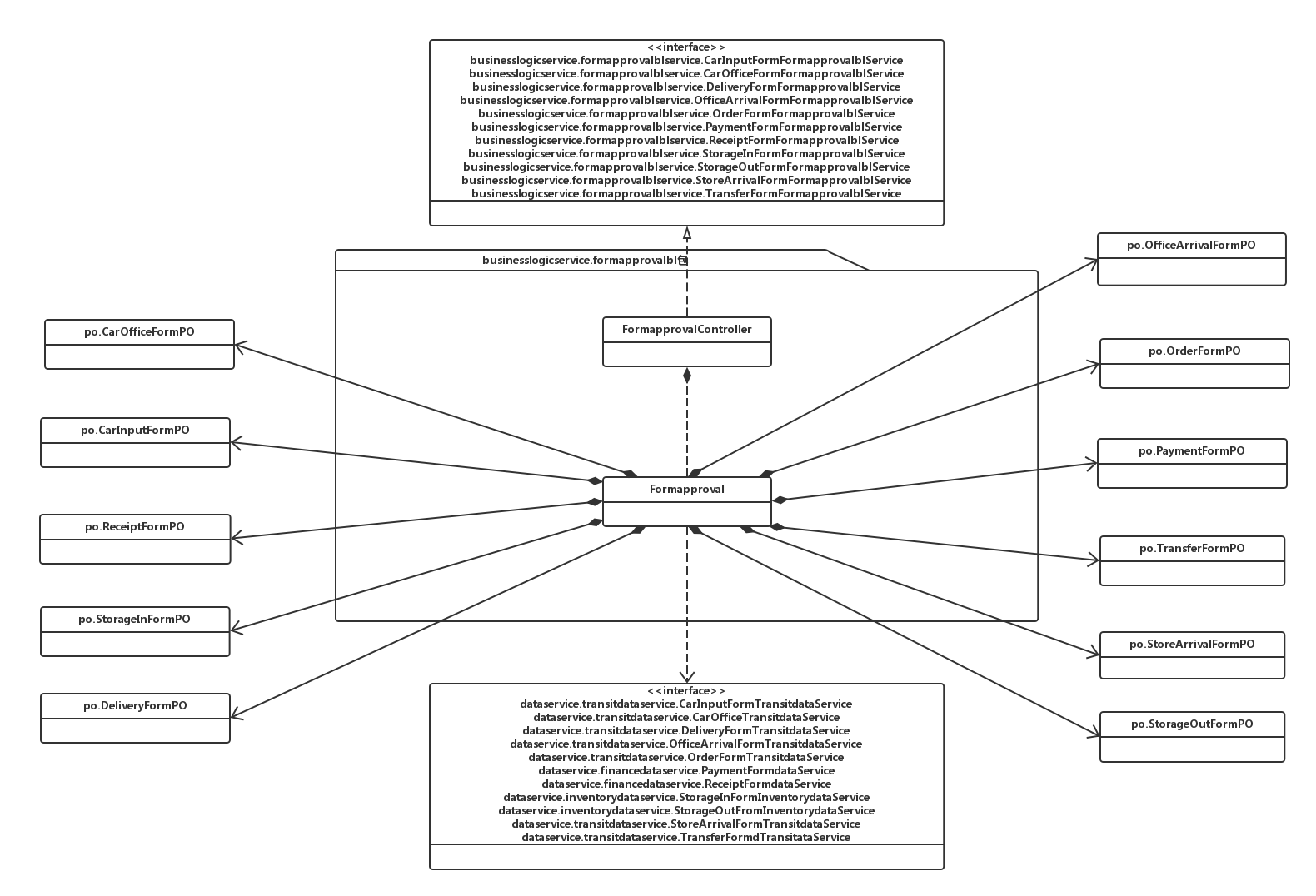


图4.1.7-1 formapprovalbl模块各个类的设计

formapprovalbl模块各个类的职责如表4.1.7-1所示

**表4.1.7-1 formapprovalbl模块各个类的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| FormapprovalController | 负责实现审批界面锁需要的服务 |
| Formapproval | 审批的领域模型对象，拥有各种提交状态的单据信息，可帮助完成审批界面所需要的服务 |

#### **4.1.7.3模块内部类的接口规范**

FormapprovalController和Formapproval的接口规范如表4.1.7-2和表4.1.7-3所示

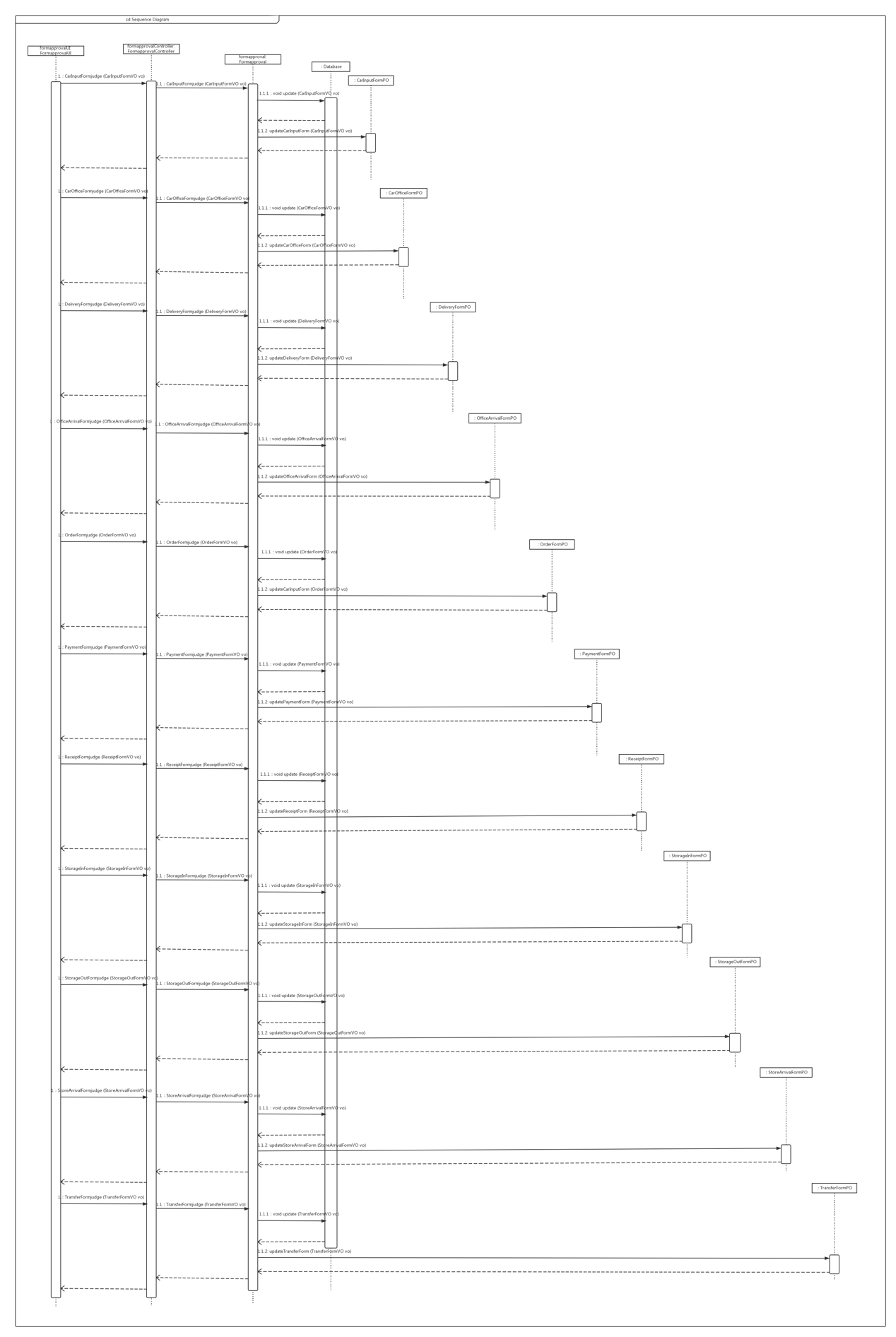
**表4.1.7-2 formapproval模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| FormapprovalblController.OderFormbl.judge | 语法 | | Public boolean judge (oderFormVO vo) |
| 前置条件 | | 已创建一个Formapproval领域对象 |
| 后置条件 | | 调用Formapproval领域对象的OderFormbl.judge方法 |
| FormapprovalblController.OfficeArrivalFormbl.judge | 语法 | | Public boolean judge (officeArrivalFormVO vo) |
| 前置条件 | | 已创建一个Formapproval领域对象 |
| 后置条件 | | 调用Formapproval领域对象的OfficeArrivalFormbl.judge方法 |
| FormapprovalblController.CarOfficeFormbl.judge | 语法 | | Public boolean judge (carOfficeFormVO vo) |
| 前置条件 | | 已创建一个Formapproval领域对象 |
| 后置条件 | | 调用Formapproval领域对象的CarOfficeFormbl.judge方法 |
| FormapprovalblController.DeliveryFormbl.judge | 语法 | | Public boolean judge (deliveryFormVO vo) |
| 前置条件 | | 已创建一个Formapproval领域对象 |
| 后置条件 | | 调用Formapproval领域对象的DeliveryFormbl.judge方法 |
| FormapprovalblController.StoreArrivalFormbl.judge | 语法 | | Public boolean judge (storeArrivalFormVO vo) |
| 前置条件 | | 已创建一个Formapproval领域对象 |
| 后置条件 | | 调用Formapproval领域对象的StoreArrivalFormbl.judge方法 |
| FormapprovalblController.TransferFormbl.judge | 语法 | | Public boolean judge (transferFormVO vo) |
| 前置条件 | | 已创建一个Formapproval领域对象 |
| 后置条件 | | 调用Formapproval领域对象的TransferFormbl.judge方法 |
| FormapprovalblController.CarInputFormbl.judge | 语法 | | Public boolean judge (carInputFormVO vo) |
| 前置条件 | | 已创建一个Formapproval领域对象 |
| 后置条件 | | 调用Formapproval领域对象的CarInputFormbl.judge方法 |
| FormapprovalblController.StorageOutForm.judge | 语法 | | Public boolean judge (storageOutFormVO vo) |
| 前置条件 | | 已创建一个Formapproval领域对象 |
| 后置条件 | | 调用Formapproval领域对象的StorageOutForm.judge方法 |
| FormapprovalblController.StorageInFormbl.judge | 语法 | | Public boolean judge (storageInFormVO vo) |
| 前置条件 | | 已创建一个Formapproval领域对象 |
| 后置条件 | | 调用Formapproval领域对象的StorageInFormbl.judge方法 |
| FormapprovalblController.PaymentFormbl.judge | 语法 | | Public boolean judge (paymentFormVO vo) |
| 前置条件 | | 已创建一个Formapproval领域对象 |
| 后置条件 | | 调用Formapproval领域对象的PaymentFormbl.judge方法 |
| FormapprovalblController.ReceiptFormbl.judge | 语法 | | Public boolean judge (receiptFormVO vo) |
| 前置条件 | | 已创建一个Formapproval领域对象 |
| 后置条件 | | 调用Formapproval领域对象的ReceiptFormbl.judge方法 |
| 需要的服务（需接口） | | | |
| Formapprovalbl.OderFormbl.judge(oderFormVO vo) | | 审批一个寄件单对象 | |
| Formapprovalbl.OfficeArrivalFormbl.judge(officeArrivalFormVO vo) | | 审批一个营业厅到达单对象 | |
| Formapprovalbl.CarOfficeFormbl.judge(carOfficeFormVO vo) | | 审批一个装车单对象 | |
| Formapprovalbl.DeliveryFormbl.judge(deliveryFormVO vo) | | 审批一个派件单对象 | |
| Formapprovalbl.StoreArrivalFormbl.judge(storeArrivalFormVO vo) | | 审批一个中转中心到达单对象 | |
| Formapprovalbl.TransferFormbl.judge(transferFormVO vo) | | 审批一个中转单对象 | |
| Formapprovalbl.CarInputFormbl.judge(carInputFormVO vo) | | 审批一个中转中心装车单对象 | |
| Formapprovalbl.StorageOutForm.judge(storageOutFormVO vo) | | 审批一个出库单对象 | |
| Formapprovalbl.StorageInFormbl.judge(storageInFormVO vo) | | 审批一个入库单对象 | |
| Formapprovalbl.PaymentFormbl.judge(paymentFormVO vo) | | 审批一个付款单对象 | |
| Formapprovalbl.ReceiptFormbl.judge(receiptFormVO vo) | | 审批一个收款单对象 | |

**表4.1.7-3 formapproval模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| Formapprovalbl.OderFormbl.judge | 语法 | | Public boolean judge (oderFormVO vo) |
| 前置条件 | | 启动一个审批寄件单任务 |
| 后置条件 | | 更改寄件单状态为已审批，更改寄件单货运状态 |
| Formapprovalbl.OfficeArrivalFormbl.judge | 语法 | | Public boolean judge (officeArrivalFormVO vo) |
| 前置条件 | | 启动一个审批营业厅到达单任务 |
| 后置条件 | | 更改营业厅到达单状态为已审批，更改寄件单货运状态 |
| Formapprovalbl.CarOfficeFormbl.judge | 语法 | | Public boolean judge (carOfficeFormVO vo) |
| 前置条件 | | 启动一个审批营业厅装车单任务 |
| 后置条件 | | 更改营业厅装车单状态为已审批 |
| Formapprovalbl.DeliveryFormbl.judge | 语法 | | Public boolean judge (deliveryFormVO vo) |
| 前置条件 | | 启动一个审批派件单任务 |
| 后置条件 | | 更派件单状态为已审批，更改寄件单货运状态 |
| Formapprovalbl.StoreArrivalFormbl.judge | 语法 | | Public boolean judge (storeArrivalFormVO vo) |
| 前置条件 | | 启动一个审批中转中心到达单任务 |
| 后置条件 | | 更改中转中心到达单状态为已审批，更改寄件单货运状态 |
| Formapprovalbl.TransferFormbl.judge | 语法 | | Public boolean judge (transferFormVO vo) |
| 前置条件 | | 启动一个审批中转单任务 |
| 后置条件 | | 更改中转单状态为已审批 |
| Formapprovalbl.CarInputFormbl.judge | 语法 | | Public boolean judge (carInputFormVO vo) |
| 前置条件 | | 启动一个审批中转中心装车单任务 |
| 后置条件 | | 更改中转中心装车单状态为已审批 |
| Formapprovalbl.StorageOutForm.judge | 语法 | | Public boolean judge (storageOutFormVO vo) |
| 前置条件 | | 启动一个审批出库单任务 |
| 后置条件 | | 更改出库单状态为已审批 |
| Formapprovalbl.StorageInFormbl.judge | 语法 | | Public boolean judge (storageInFormVO vo) |
| 前置条件 | | 启动一个审批入库单单任务 |
| 后置条件 | | 更改入库单状态为已审批 |
| Formapprovalbl.PaymentFormbl.judge | 语法 | | Public boolean judge (paymentFormVO vo) |
| 前置条件 | | 启动一个审批付款单任务 |
| 后置条件 | | 更改付款单状态为已审批 |
| Formapprovalbl.ReceiptFormbl.judge | 语法 | | Public boolean judge (receiptFormVO vo) |
| 前置条件 | | 启动一个审批收款单任务 |
| 后置条件 | | 更改收款单状态为已审批，更改寄件单货运状态 |
| 需要的服务（需接口） | | | |
| DatabaseFactory.getFormapprovalDatabase | | 得到Formapproval数据库的服务的引用 | |
| FormapprovalDataService.updata(receiptFormPO po) | | 更新单一持久化对象 | |
| FormapprovalDataService.updata(paymentFormPO po) | | 更新单一持久化对象 | |
| FormapprovalDataService.updata(oderFormPO po) | | 更新单一持久化对象 | |
| FormapprovalDataService.updata(officeArrivalFormPO po) | | 更新单一持久化对象 | |
| FormapprovalDataService.updata(carOfficeFormPO po) | | 更新单一持久化对象 | |
| FormapprovalDataService.updata(deliveryFormPO po) | | 更新单一持久化对象 | |
| FormapprovalDataService.updata(storeArrivalFormPO po) | | 更新单一持久化对象 | |
| FormapprovalDataService.updata(transferFormPO po) | | 更新单一持久化对象 | |
| FormapprovalDataService.updata(carInputFormPO po) | | 更新单一持久化对象 | |
| FormapprovalDataService.updata(storageOutFormPO po) | | 更新单一持久化对象 | |
| FormapprovalDataService.updata(storageInFormPO po) | | 更新单一持久化对象 | |

#### **4.1.7.4业务逻辑层的动态模型**



#### **4.1.7.5业务逻辑的设计原理**

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的对象。

## 4.2界面层的分解

### 4.2.1.导航设计

#### 4.2.1.1快递员

快递员的任务主要有三个：建立寄件单、订单信息查询、输入收件信息。可以据此建立菜单导航。

##### 4.2.1.1.1建立寄件单

针对建立寄件单任务，依据建立场景和规格说明需求，可以设计下面独立界面或界面组件：

● 建立寄件单：进行建立寄件单任务导航的主要部分

● 寄件人姓名输入：建立寄件单任务界面中的一个独立组件

● 寄件人住址输入：建立寄件单任务界面中的一个独立组件

● 寄件人单位输入：建立寄件单任务界面中的一个独立组件

● 寄件人电话输入：建立寄件单任务界面中的一个独立组件

● 寄件人电话输入错误提示：独立界面

● 寄件人手机输入：建立寄件单任务界面中的一个独立组件

● 寄件人手机输入错误提示：独立界面

● 收件人姓名输入：建立寄件单任务界面中的一个独立组件

● 收件人住址输入：建立寄件单任务界面中的一个独立组件

● 收件人单位输入：建立寄件单任务界面中的一个独立组件

● 收件人电话输入：建立寄件单任务界面中的一个独立组件

● 收件人电话输入错误提示：独立界面

● 收件人手机输入：建立寄件单任务界面中的一个独立组件

● 收件人手机输入错误提示：独立界面

● 托运货物原件数输入：建立寄件单任务界面中的一个独立组件

● 托运货物实际重量输入：建立寄件单任务界面中的一个独立组件

● 托运货物体积输入：建立寄件单任务界面中的一个独立组件

● 内件品名输入：建立寄件单任务界面中的一个独立组件

● 快递种类输入：建立寄件单任务界面中的一个独立组件

● 包装费输入：建立寄件单任务界面中的一个独立组件

● 订单条形码号输入：建立寄件单任务界面中的一个独立组件

● 订单条形码号输入错误提示：独立界面

● 揽件日期输入：建立寄件单任务界面中的一个独立组件

● 货运状态输入：建立寄件单任务界面中的一个独立组件

● 运费显示：建立寄件单任务界面中的一个独立组件

● 费用合计显示：建立寄件单任务界面中的一个独立组件

● 预计到达日期显示：建立寄件单任务界面中的一个独立组件

基于上述独立界面或者独立组件，可以建立新建寄件单交互过程的导航——对话结构，如图所示：

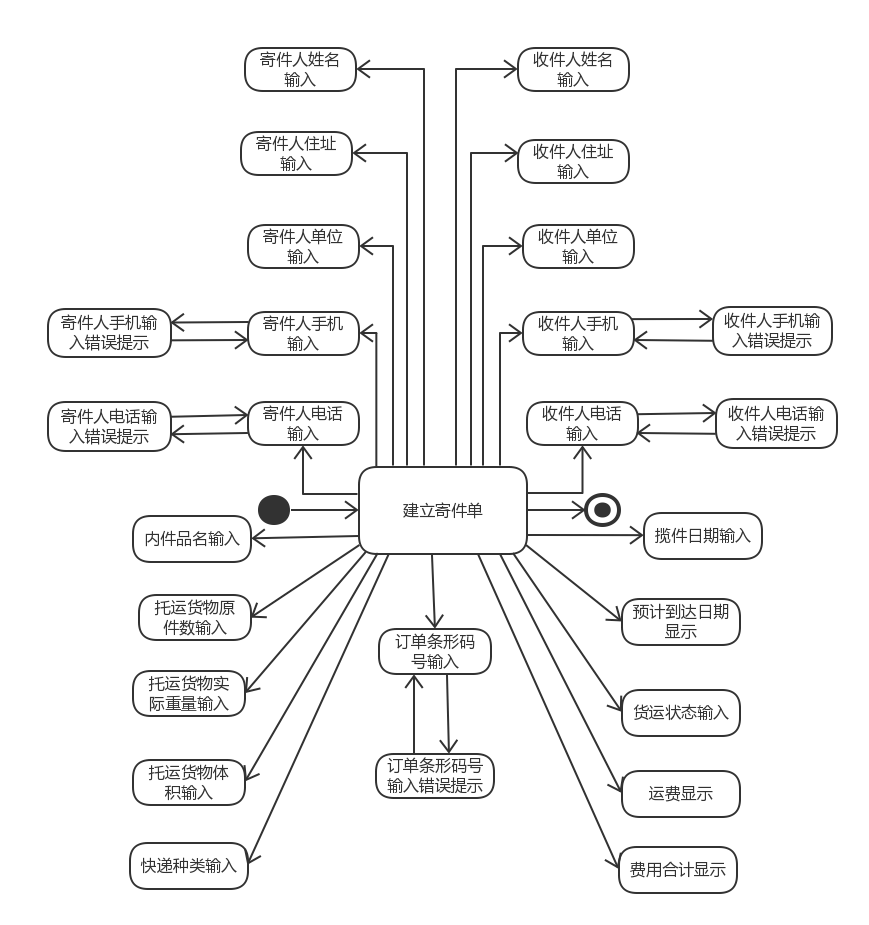


图1.1.1-1 建立寄件单对话结构

##### 4.2.1.1.2订单信息查询

针对订单信息查询任务，依据订单信息查询场景和规格说明需求，可以设计下列独立界面或界面组件：

● 订单信息查询：进行订单信息查询任务导航的主要部分

● 托运订单条形码号输入：订单信息查询任务界面中的一个独立组件

● 托运订单条形码号输入错误提示：独立界面

● 寄件单信息显示：独立界面

基于上述独立界面或者独立组件，可以建立订单信息查询交互过程的导航——对话结构，如图所示：

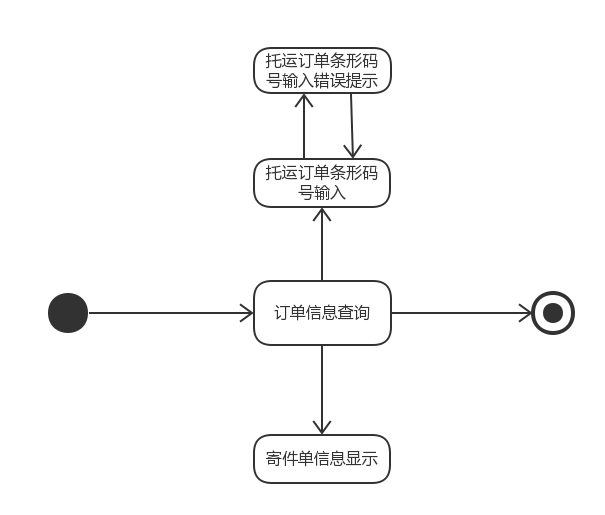


图1.1.2-1 订单信息查询对话结构

##### 4.2.1.1.3输入收件信息

针对输入收件信息任务，依据输入收件信息场景和规格说明需求，可以设计下列独立界面或界面组件：

● 输入收件信息：进行输入收件信息任务导航的主要部分

● 托运订单条形码号输入：输入收件信息任务中的一个独立组件

● 托运订单条形码号输入错误提示：独立界面

● 寄件单信息显示：独立界面

● 收件人姓名输入：输入收件信息任务中的一个独立组件

● 收件时间输入：输入收件信息任务中的一个独立组件

基于上述独立界面或者独立组件，可以建立输入收件信息交互过程的导航——对话结构，如图所示：

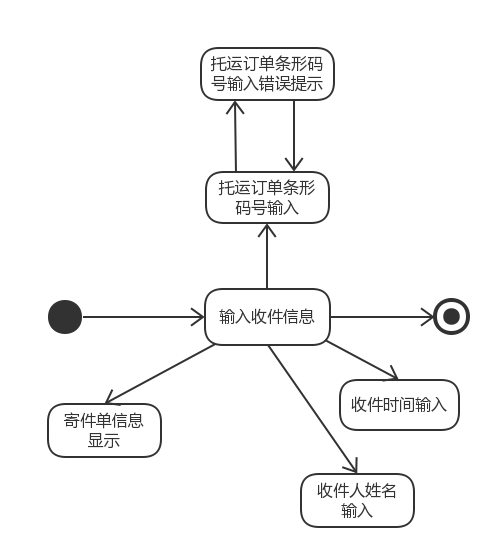


图1.1.3-1 输入收件信息对话结构

#### 4.2.1.2营业厅业务员

营业厅业务员的任务有：建立营业厅到达单、生成派件单、录入营业厅装车单、建立收款单。可以据此建立菜单导航。

##### 4.2.1.2.1建立营业厅到达单

针对建立营业厅到达单任务，依据建立场景和规格说明需求，可以设计下面独立界面或界面组件：

●建立营业厅到达单：进行建立营业厅到达单任务导航的主要部分。

●到达日期输入：建立营业厅到达单任务界面中的一个独立组件。

●中转单编号输入：建立营业厅到达单任务界面中的一个独立组件。

●中转单编号输入错误提示：独立界面。

●出发地输入：建立营业厅到达单任务界面中的一个独立组件。

●出发地输入错误提示：独立界面。

●货物到达状态选择：建立营业厅到达单任务界面中的一个独立组件。

基于上述独立界面或者独立组件，可以建立新建营业厅到达单交互过程的导航——对话结构，如图所示：

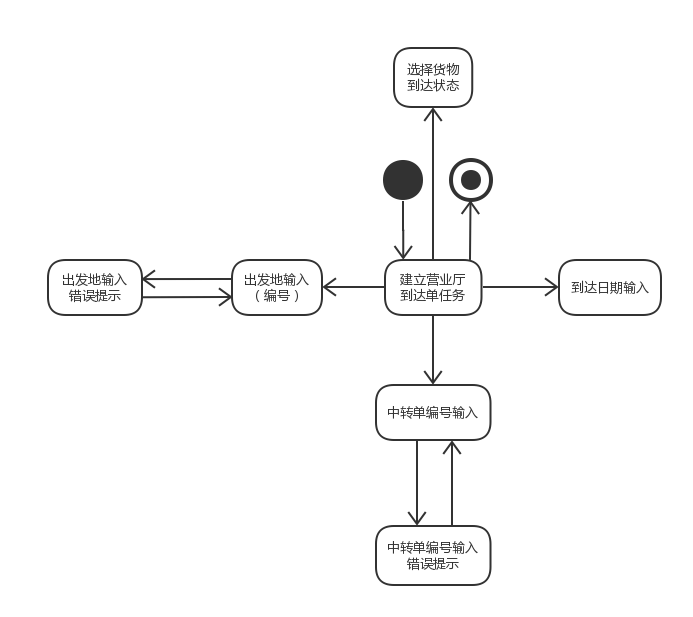


图1.2.1建立营业厅到达单

##### 4.2.1.2.2生成派件单

针对生成派件单任务，依据生成派件单场景和规格说明需求，可以设计下列独立界面或界面组件：

●生成派件单：进行生成派件单任务导航的主要部分。

●到达日期输入：生成派件单任务界面中的一个独立组件。

●订单号输入：生成派件单任务界面中的一个独立组件。

●订单号输入错误提示：独立界面。

●派送员（快递员编号）输入：生成派件单任务界面中的一个独立组件。

●派送员（快递员编号）输入错误提示：独立界面。

基于上述独立界面或者独立组件，可以建立生成派件单交互过程的导航——对话结构，如图所示：

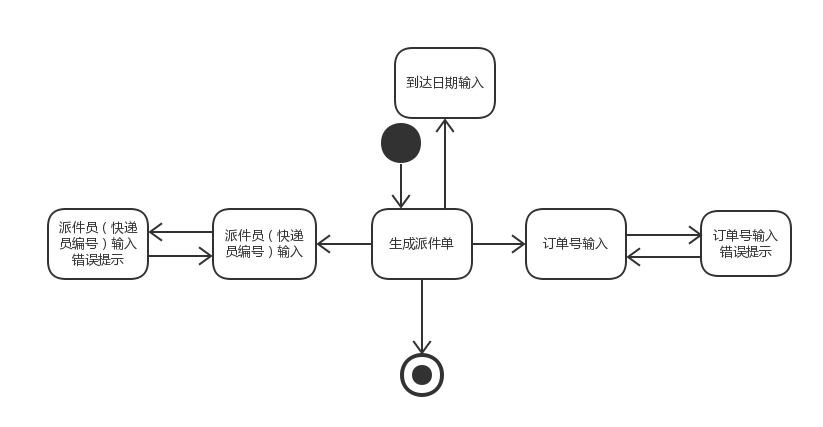


图1.2.2生成派件单

##### 4.2.1.2.3录入营业厅装车单

针对录入营业厅装车单任务，依据录入营业厅装车单场景和规格说明需求，可以设计下列独立界面或界面组件：

●录入装车单：进行录入装车单任务导航的主要部分。

●装车日期输入：录入营业厅装车单任务界面中的一个独立组件。

●本营业厅编号：录入营业厅装车单任务界面中的一个独立组件。

●汽运编号：录入营业厅装车单任务界面中的一个独立组件。

●到达地输入：录入营业厅装车单任务界面中的一个独立组件。

●到达地输入错误提示：独立界面。

●车辆代号输入：录入营业厅装车单任务界面中的一个独立组件。

●车辆是否空闲状态显示：录入营业厅装车单任务界面中的一个独立组件。

●监装员（编号）输入：录入营业厅装车单任务界面中的一个独立组件。

●押运员（编号）输入：录入营业厅装车单任务界面中的一个独立组件。

●订单号输入：录入营业厅装车单任务界面中的一个独立组件。

●订单号输入错误提示：独立界面。

●运费：录入营业厅装车单任务界面中的一个独立组件。

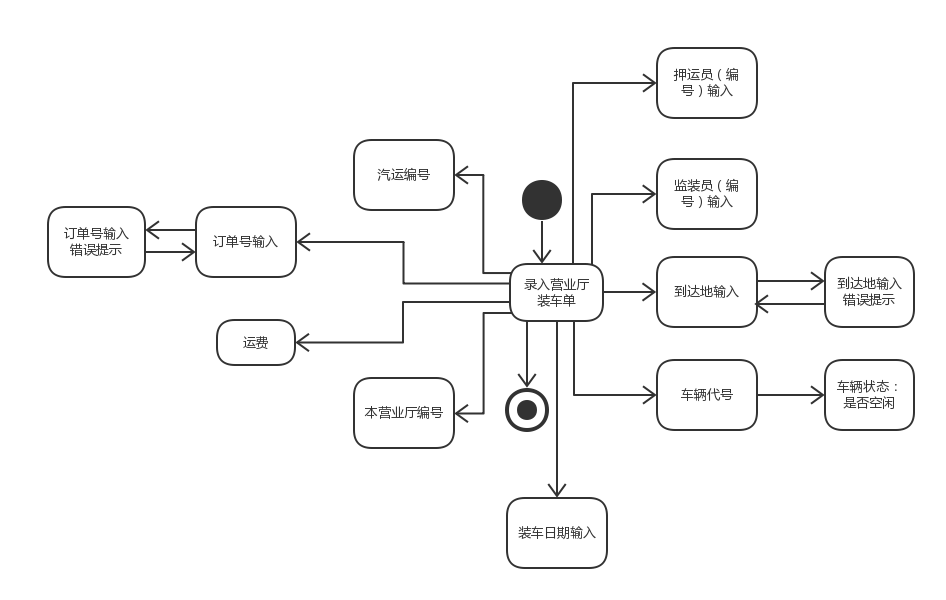
基于上述独立界面或者独立组件，可以建立录入营业厅装车单交互过程的导航——对话结构，如图所示：

图1.2.3录入营业厅装车单

##### 4.2.1.2.4建立收款单

针对建立收款单任务，依据建立收款单场景和规格说明需求，可以设计下列独立界面或界面组件：

●建立收款单：进行建立收款单任务导航的主要部分。

●收款日期输入：建立收款单任务界面中的一个独立组件。

●收款金额输入：建立收款单任务界面中的一个独立组件。

●收款快递员（编号）输入：建立收款单任务界面中的一个独立组件。

●收款快递员（编号）输入错误提示：独立界面。

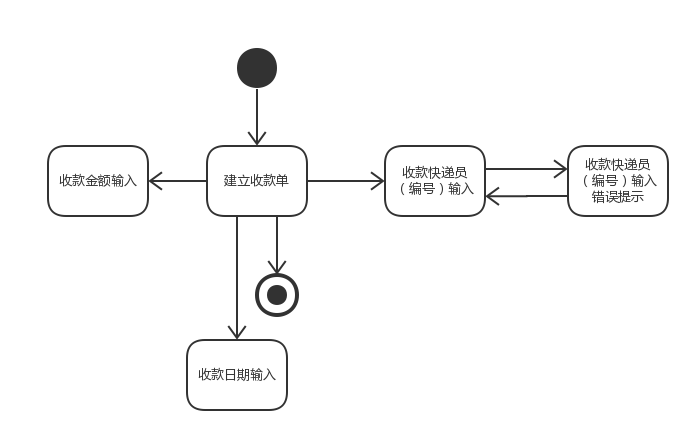


图1.2.4建立收款单

##### 4.2.1.2.5维护车辆信息

针对维护车辆信息任务，依据维护车辆信息场景和规格说明需求，可以设计下列独立界面或界面组件：

●维护车辆信息：进行维护车辆信息任务导航的主要部分（增删改查）。

●新车代号（城市编号+营业厅编号+三位数字）输入：独立界面或者维护车辆信息界面中的一个独立组件。

●新车代号（城市编号+营业厅编号+三位数字）输入错误提示：独立界面。

●车牌号输入：维护车辆信息界面中的一个独立组件。

●发动机号输入：维护车辆信息界面中的一个独立组件。

●底盘号输入：维护车辆信息界面中的一个独立组件。

●购买时间输入：维护车辆信息界面中的一个独立组件。

●服役时间输入：维护车辆信息界面中的一个独立组件。

●车辆信息显示：独立界面。

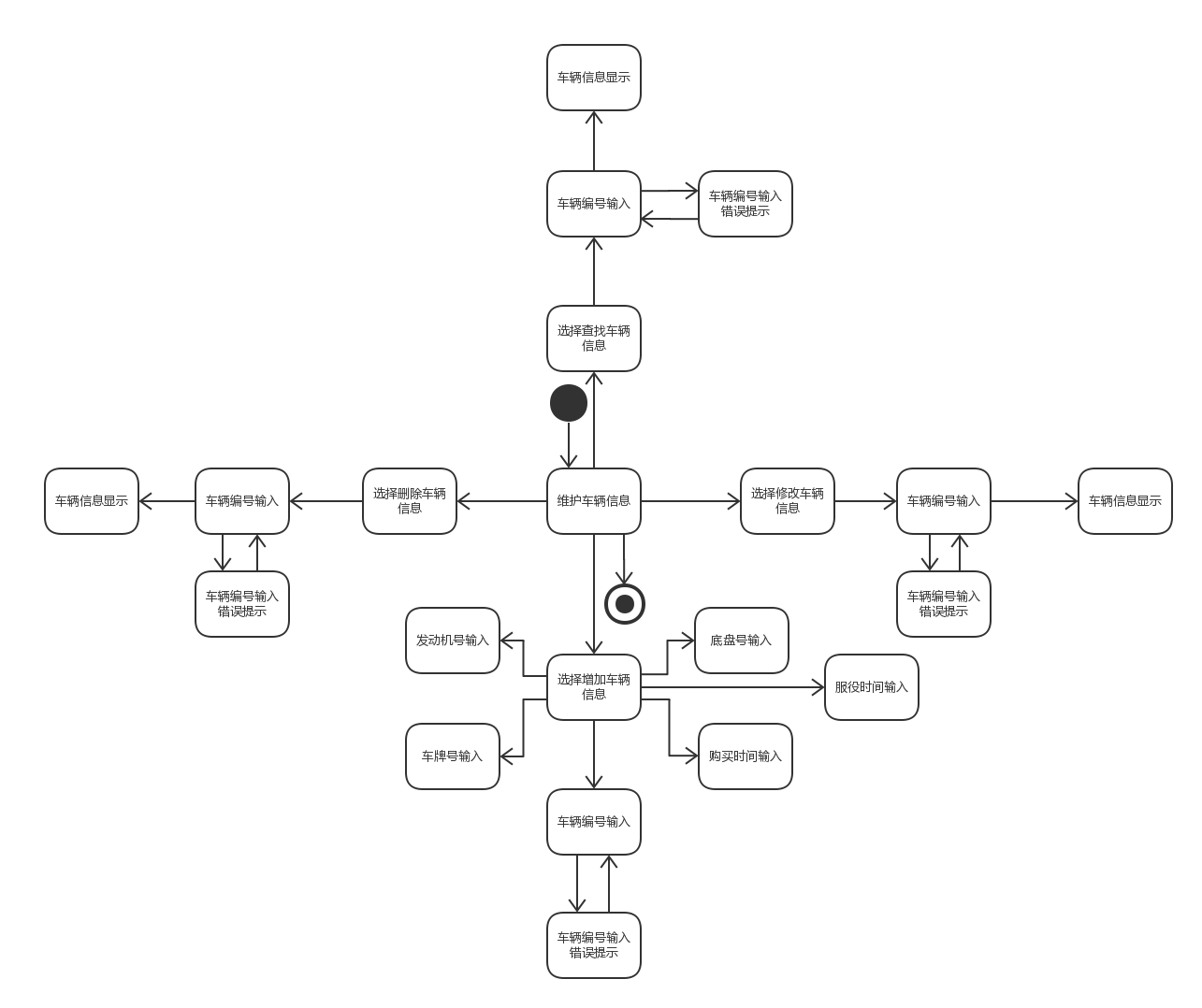


图1.2.5维护车辆信息

##### 4.2.1.2.6维护司机信息

针对维护司机信息任务，依据维护司机信息场景和规格说明需求，可以设计下列独立界面或界面组件：

●维护司机信息：进行维护司机信息任务导航的主要部分（增删改查）。

●新司机的编号（城市编号+营业厅编号+三位数字）输入：独立界面或者维护司机信息界面中的一个独立组件。

●新司机的编号（城市编号+营业厅编号+三位数字）输入错误提示：独立界面。

●姓名输入：维护司机信息界面中的一个独立组件。

●出生日期输入：维护司机信息界面中的一个独立组件。

●身份证号输入：维护司机信息界面中的一个独立组件。

●性别输入：维护司机信息界面中的一个独立组件。

●手机输入：维护司机信息界面中的一个独立组件。

●行驶证期限输入：维护司机信息界面中的一个独立组件。

●司机信息显示：独立界面。

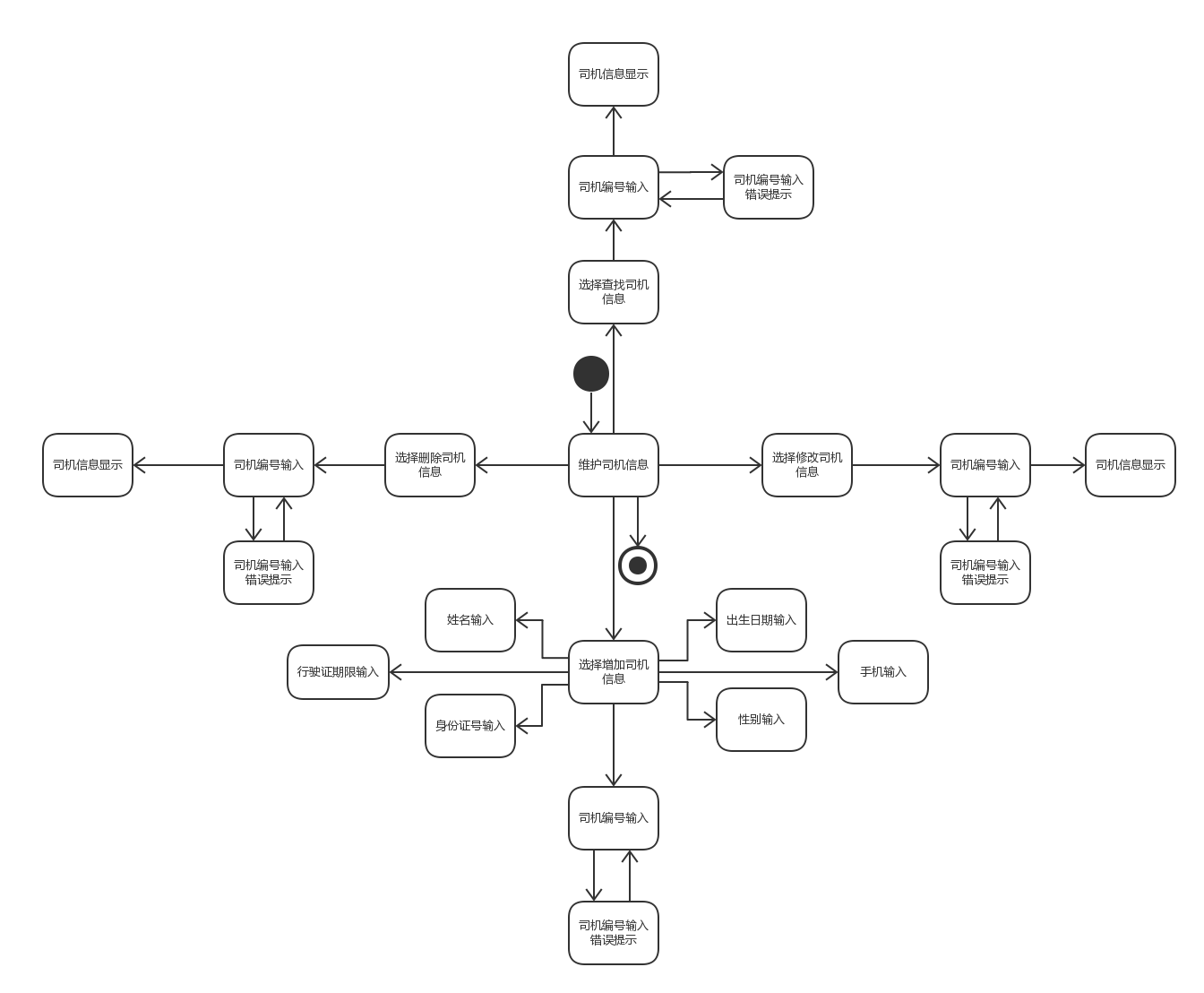


图1.2.6维护司机信息

#### 4.2.1.3中转中心业务员

中转中心业务员的任务主要有三个：生成中转中心到达单、录入中转单、录入中转中心装车单。可以据此建立菜单导航。

##### 4.2.1.3.1生成中转中心到达单

针对生成中转中心到达单任务，依据生成中转中心到达单场景和规格说明需求，可以设计下列独立界面或界面组件：

●生成中转中心到达单：进行生成中转中心到达单任务导航的主要部分。

●出发地输入：生成中转中心到达单任务界面中的一个独立组件。

●货物到达状态选择：生成中转中心到达单任务界面中的一个独立组件。

●到达时间输入：生成中转中心到达单任务界面中的一个独立组件。

●显示中转中心编号：生成中转中心到达单任务界面中的一个独立组件。

●显示中转单编号：生成中转中心到达单任务界面中的一个独立组件。

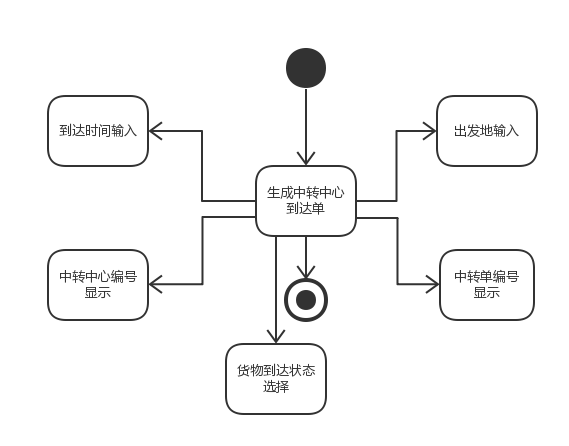


图1.3.1生成中转中心到达单

##### 4.2.1.3.2录入中转单

针对录入中转单任务，依据录入中转单场景和规格说明需求，可以设计下列独立界面或界面组件：

●录入中转单：进行录入中转单任务导航的主要部分。

●装车日期输入：录入中转单任务界面中的一个独立组件。

●运输工具选择：录入中转单任务界面中的一个独立组件。

●出发地输入：录入中转单任务界面中的一个独立组件。

●到达地输入：录入中转单任务界面中的一个独立组件。

●监装员（编号）输入：录入中转单任务界面中的一个独立组件。

●货柜号输入：录入中转单任务界面中的一个独立组件。

●订单号输入：录入中转单任务界面中的一个独立组件。

●订单号输入错误提示：独立界面。

●运费显示：录入中转单任务界面中的一个独立组件。

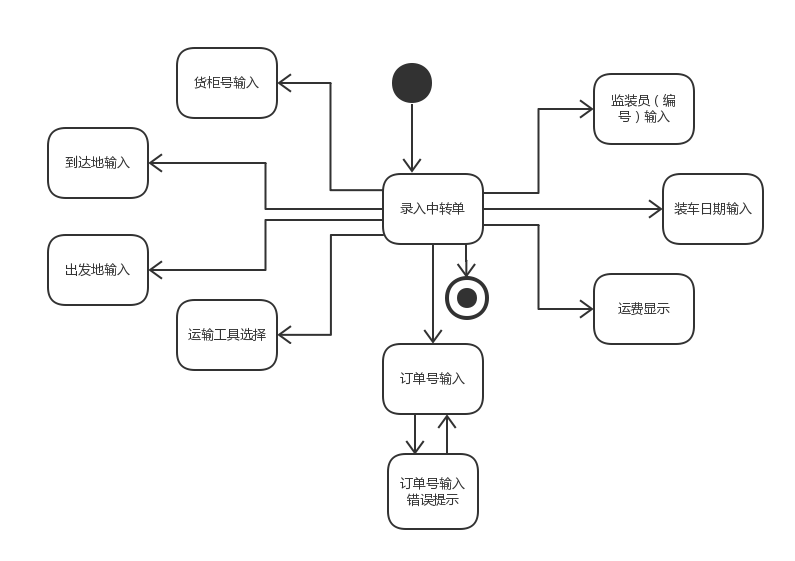


图1.3.2录入中转单

##### 4.2.1.3.3录入中转中心装车单

针对录入中转中心装车单任务，依据录入中转中心装车单场景和规格说明需求，可以设计下列独立界面或界面组件：

●录入中转中心装车单：进行录入中转中心装车单任务导航的主要部分。

●装车日期输入：录入中转中心装车单任务界面中的一个独立组件。

●到达地输入：录入中转中心装车单任务界面中的一个独立组件。

●车辆代号输入：录入中转中心装车单任务界面中的一个独立组件。

●监装员（编号）输入：录入中转中心装车单任务界面中的一个独立组件。

●押运员（编号）输入：录入中转中心装车单任务界面中的一个独立组件。

●订单号输入：录入中转中心装车单任务界面中的一个独立组件。

●订单号输入错误提示：独立界面。

●运费显示：录入中转中心装车单任务界面中的一个独立组件。



图1.3.3录入中转中心装车单

#### 4.2.1.4中转中心仓库管理人员

中转中心仓库管理人员的任务有：入库、调整库存分区、库存查看、库存盘点。可设置库存警戒值。

##### 4.2.1.4.1建立入库单

针对建入库，依据建立入库单场景和规格说明需求，可以设计下面独立界面或界面组件：

●建立入库单：进行建立入库单任务导航的主要部分。

●入库日期输入：建立入库单任务界面中的一个独立组件。

●订单编号输入：建立入库单任务界面中的一个独立组件。

●订单编号输入错误提示：独立界面。

●目的地输入：建立入库单任务界面中的一个独立组件。

●库存位置的选择：建立入库单任务界面中的一个独立组件。

基于上述独立界面或者独立组件，可以建立入库单交互过程的导航——对话结构，如图所示：

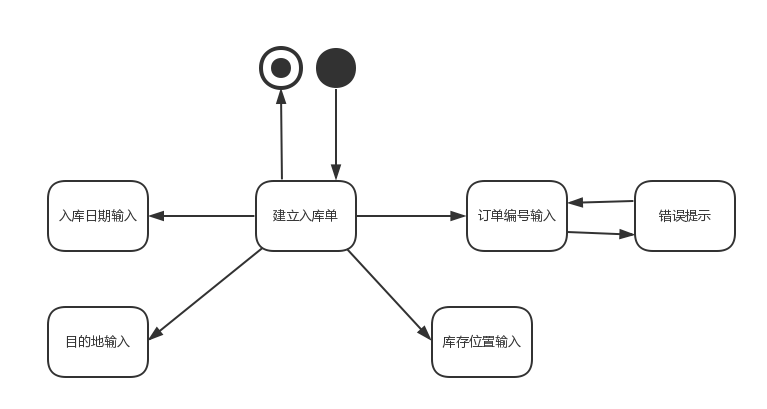


图1.4.1建立入库单

##### 4.2.1.4.2调整库存分区

针对调整库存分区，依据调整库存分区场景和规格说明需求，可以设计下面独立界面或界面组件：

●库存分区调整：进行库存分区调整任务导航的主要部分。

●库存警戒显示：库存分区任务界面中的一个独立组件。

●保存库存分区：库存分区任务界面中的一个独立组件。

●初始化库存分区：库存分区任务界面中的一个独立组件。

●保存成功提示：独立界面

●初始化成功提示：独立界面

基于上述独立界面或者独立组件，可以建立库存分区交互过程的导航——对话结构，如图所示：



图1.4.2调整库存分区

##### 4.2.1.4.3库存查看

针对库存查看任务，依据库存查看场景和规格说明需求，可以设计下面独立界面或界面组件：

●库存查看任务：进行库存查看任务导航的主要部分。

●输入开始时间：库存查看任务界面中的一个独立组件。

●输入结束时间：库存查看任务界面中的一个独立组件。

●库存明细单：独立界面。

基于上述独立界面或者独立组件，可以建立录入营业厅装车单交互过程的导航——对话结构，如图所示：

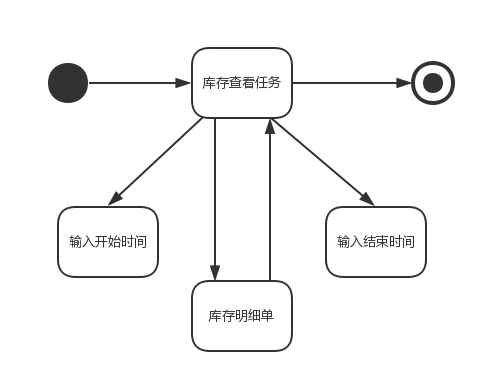


图1.4.3库存查看

##### 4.2.1.4.4库存盘点

针对库存盘点任务，依据库存盘点场景和规格说明需求，可以设计下面独立界面或界面组件：

●库存盘点任务：进行库存盘点任务导航的主要部分。

●区号输入：库存盘点任务界面中的一个独立组件。

●库存显示：独立界面

基于上述独立界面或者独立组件，可以建立库存盘点交互过程的导航——对话结构，如图所示：

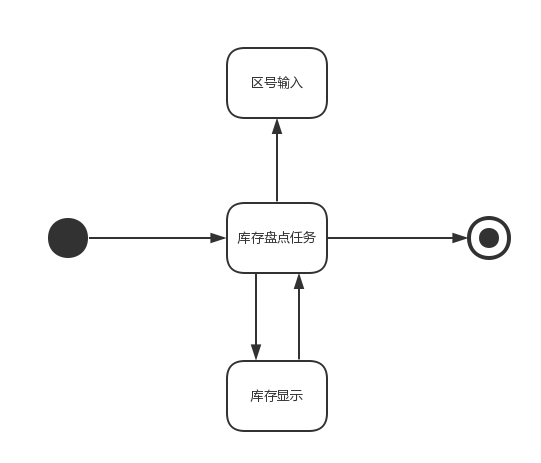


图1.4.4库存盘点

##### 4.2.1.4.5设置库存警戒值

针对设置库存警戒值任务，依据设定库存警戒值场景和规格说明需求，可以设计下面独立界面或界面组件：

●设定库存警戒值任务：设定库存警戒值任务导航的主要部分。

●输入库存警戒值：设定库存警戒值任务界面中的独立组件

●保存提示：独立界面

●提交提示：独立界面

●保存库存警戒值：设定库存警戒值任务界面中的独立组件

●提交库存警戒值：设定库存警戒值任务界面中的独立组件

基于上述独立界面或者独立组件，可以画出设定库存警戒值交互过程的导航——对话结构，如图所示：

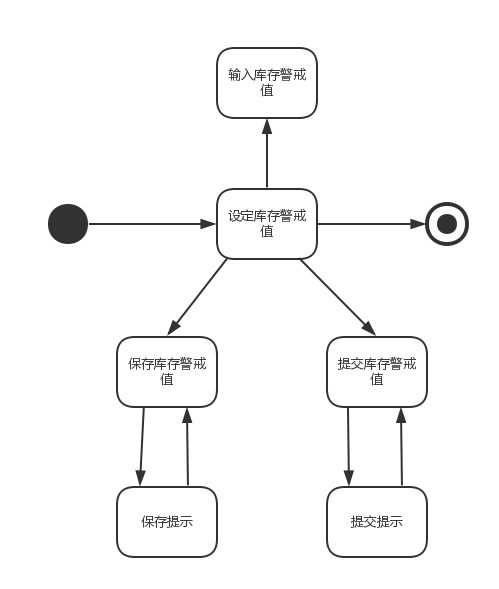


图1.4.5设定库存警戒值

##### 4.2.1.4.6建立出库单

针对出库任务，依据建立出库单场景和规格说明需求，可以设计下面独立界面或界面组件：

●建立出库单：出库任务导航的主要部分

●订单号输入：出库任务界面中的一个独立组件。

●保存出库单：建立收款单任务界面中的一个独立组件。

●提交出库单：建立收款单任务界面中的一个独立组件。

●撤回出库单：建立收款单任务界面中的一个独立组件。

●删除出库单：建立收款单任务界面中的一个独立组件。

●保存提示：独立界面

●提交提示：独立界面

●订单号错误提示：独立界面

●撤回提示：独立界面

●删除提示：独立界面

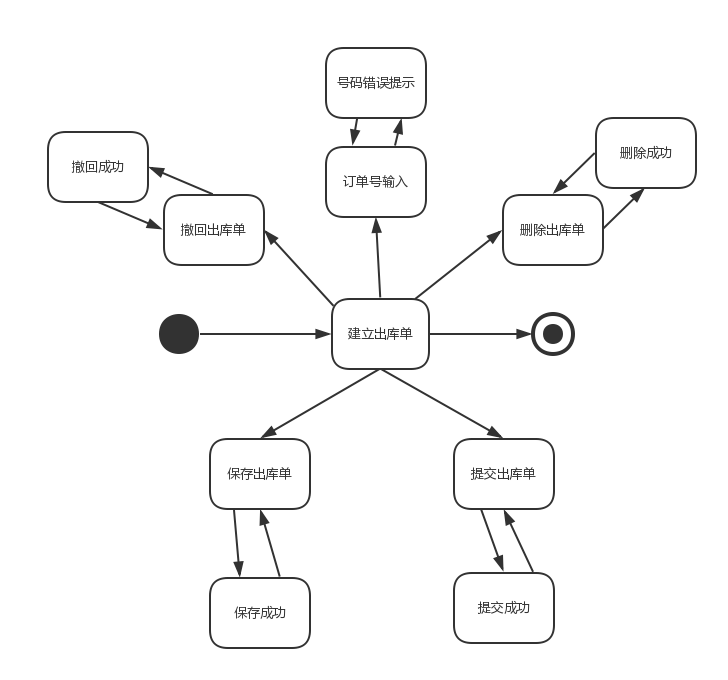
基于上述独立界面或者独立组件，可以画出出库交互过程的导航——对话结构，如图所示：

图1.4.6建立出库单

#### 4.2.1.5财务人员

财务人员的任务有：收款单汇总、生成付款单、生成成本收益表、生成经营情况表、增删改查账户、期初建账、查看成本收益表、查看经营情况表、查看日志。可以据此建立菜单导航。

##### 4.2.1.5.1收款单汇总

针对收款单汇总任务，依据建立场景和规格说明需求，可以设计下面独立界面或界面组件：

●收款单汇总：进行收款单汇总任务导航的主要部分。

●系统新建收款单汇总报表：收款单汇总任务界面中的一个独立组件。

●选择收款单汇总方式：收款单汇总任务界面中的一个独立组件。

●选择汇总的区间：收款单汇总任务界面中的一个独立组件。

●保存收款单记录汇总报表：收款单汇总任务界面中的一个独立组件。

●系统保存提示：独立界面。

●取消收款单汇总：收款单汇总任务界面中的一个独立组件。

●删除收款单记录汇总报表：收款单汇总任务界面中的一个独立组件。

基于上述独立界面或者独立组件，可以生成经营情况表汇总交互过程的导航——对话结构，如图所示：

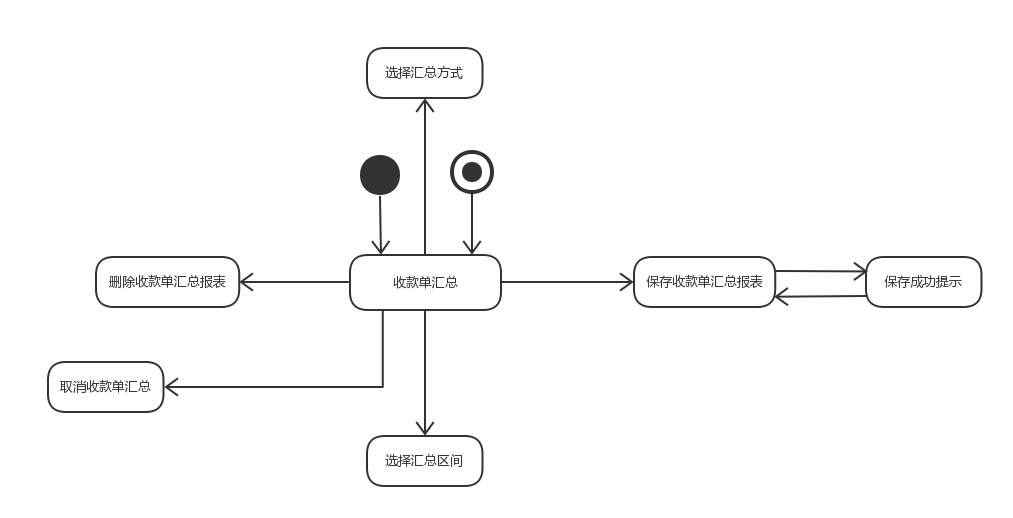


图1.5.1收款单汇总

##### 4.2.1.5.2生成付款单

针对生成付款单任务，依据生成付款单场景和规格说明需求，可以设计下列独立界面或界面组件：

●新建付款单：进行生成付款单任务导航的主要部分。

●付款日期输入：生成付款单任务界面中的一个独立组件。

●付款金额输入：生成付款单任务界面中的一个独立组件。

●付款人输入：生成付款单任务界面中的一个独立组件。

●付款账号输入：生成付款单任务界面中的一个独立组件。

●租金输入：生成付款单任务界面中的一个独立组件。

●租金年份输入：生成付款单任务界面中的一个独立组件。

●运费输入：生成付款单任务界面中的一个独立组件。

●运单号输入：生成付款单任务界面中的一个独立组件。

●人员工资输入：生成付款单任务界面中的一个独立组件。

●工资月份输入：生成付款单任务界面中的一个独立组件。

●保存付款单：生成付款单任务界面中的一个独立组件。

●保存提示：独立界面。

●提交付款单：生成付款单任务界面中的一个独立组件。

●提交提示：独立界面。

●取消新建付款单：生成付款单任务界面中的一个独立组件。

●撤回付款单：生成付款单任务界面中的一个独立组件。

●删除付款单：生成付款单任务界面中的一个独立组件。

基于上述独立界面或者独立组件，可以建立生成付款单交互过程的导航——对话结构，如图所示：

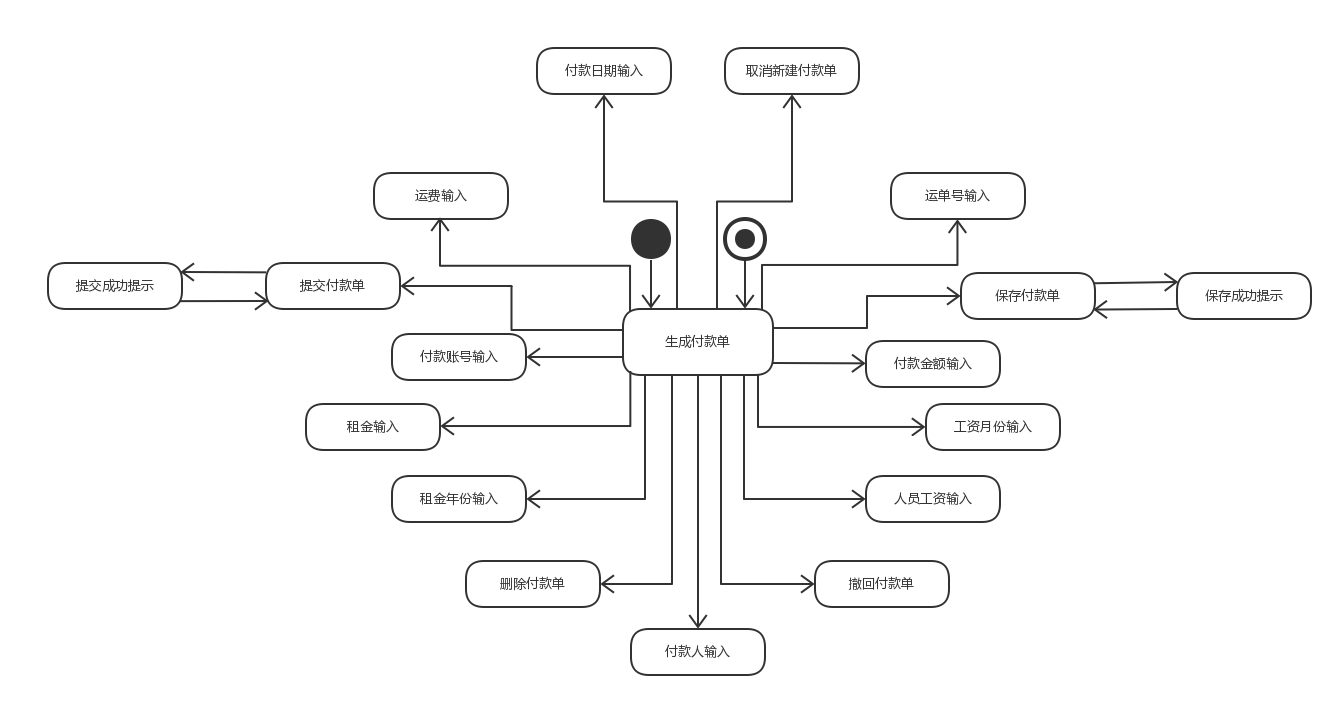


图1.5.2生成付款单

##### 4.2.1.5.3生成成本收益表

针对生成成本收益表任务，依据生成成本收益表场景和规格说明需求，可以设计下列独立界面或界面组件：

●生成成本收益表：进行生成成本收益表任务导航的主要部分。

●保存成本收益表：录入生成成本收益表任务界面中的一个独立组件。

●保存提示：独立界面。

●取消生成成本收益表：生成成本收益表任务界面中的一个独立组件。

基于上述独立界面或者独立组件，可以建立生成成本收益表交互过程的导航——对话结构，如图所示：

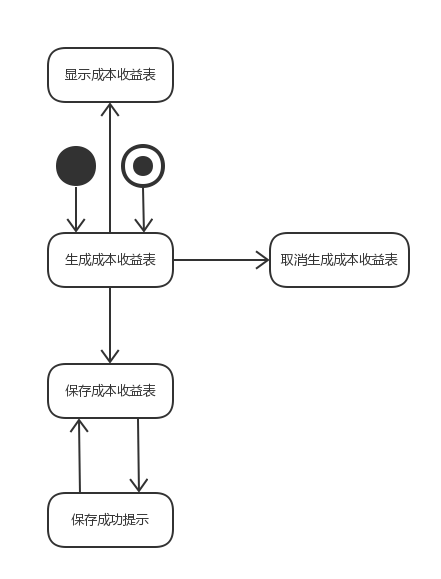


图1.5.3生成成本收益表

##### 4.2.1.5.4生成经营情况表

针对生成经营情况表任务，依据生成经营情况表场景和规格说明需求，可以设计下列独立界面或界面组件：

●生成经营情况表：进行生成经营情况表任务导航的主要部分。

●保存经营情况表：生成经营情况表任务界面中的一个独立组件。

●保存提示：独立界面。

●开始日期输入：查看经营情况表任务界面中的一个独立组件。

●结束日期输入：查看经营情况表任务界面中的一个独立组件。

●日期错误提示：独立界面。

●取消生成经营情况表：生成经营情况表任务界面中的一个独立组件。

基于上述独立界面或者独立组件，可以建立生成经营情况表交互过程的导航——对话结构，如图所示：

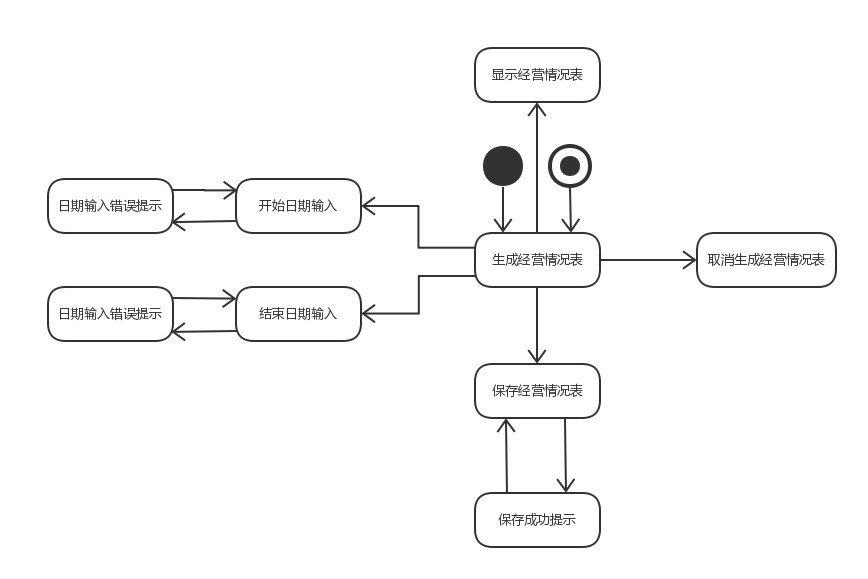


图1.5.4生成经营情况表

##### 4.2.1.5.5增删改查账户

针对增删改查账户任务，依据增删改查账户场景和规格说明需求，可以设计下列独立界面或界面组件：

●管理账户：进行增删改查账户任务导航的主要部分。

●增加账户：增删改查账户任务界面中的一个独立组件。

●删除账户：增删改查账户任务界面中的一个独立组件。

●修改账户：增删改查账户任务界面中的一个独立组件。

●查询账户：增删改查账户任务界面中的一个独立组件。

●确认删除提示：独立界面。

●账户名输入：增加账户界面中的一个独立组件。

●银行账号输入：增加账户界面中的一个独立组件。

●金额输入：增加账户界面中的一个独立组件。

●保存账户：增改账户任务界面中的一个独立组件。

●保存提示：独立界面。

●提交提示：独立界面。

基于上述独立界面或者独立组件，可以建立增删改查账户交互过程的导航——对话结构，如图所示：

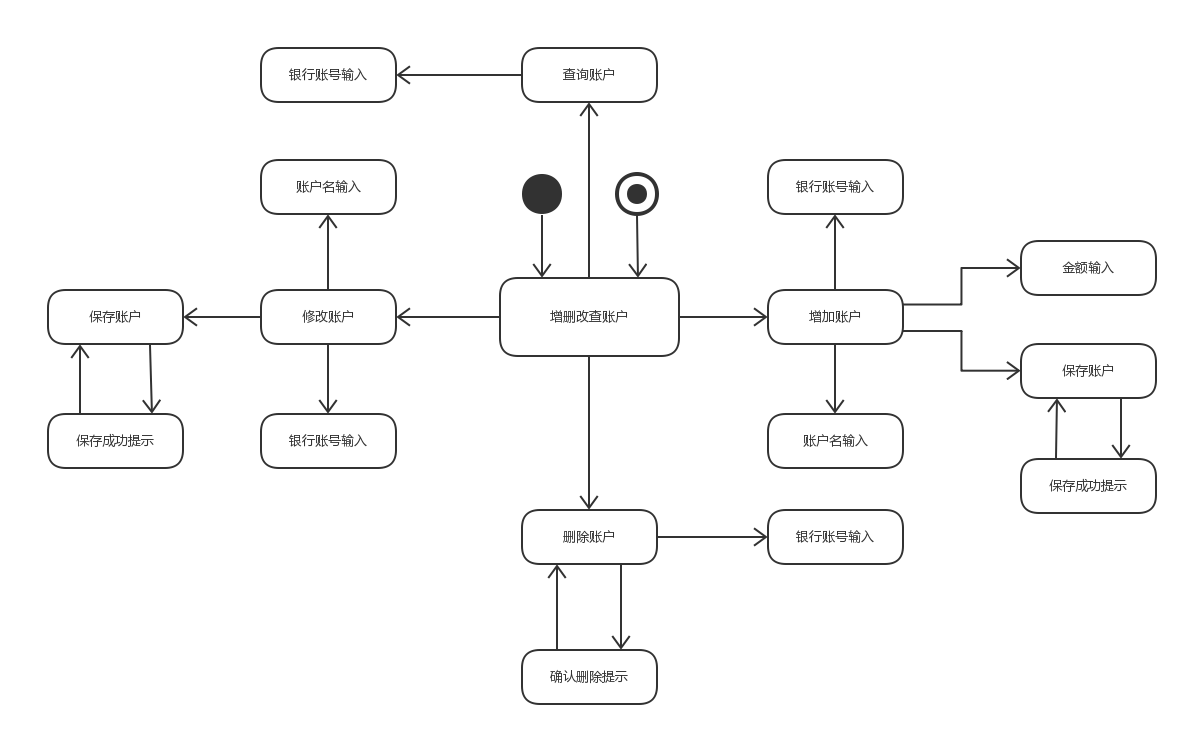


图1.5.5增删改查账户

##### 4.2.1.5.6期初建账

针对期初建账任务，依据期初建账场景和规格说明需求，可以设计下列独立界面或界面组件：

●期初建账：进行期初建账任务导航的主要部分。

●操作时间错误提示：独立界面。

基于上述独立界面或者独立组件，可以建立期初建账交互过程的导航——对话结构，如图所示：

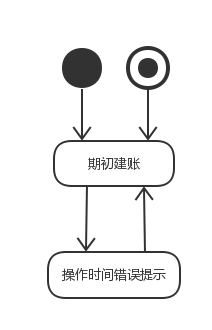


图1.5.6期初建账

##### 4.2.1.5.7查看成本收益表

针对查看成本收益表任务，依据查看成本收益表场景和规格说明需求，可以设计下列独立界面或界面组件：

●查看成本收益表：进行查看成本收益表任务导航的主要部分。

基于上述独立界面或者独立组件，可以建立查看成本收益表交互过程的导航——对话结构，如图所示：

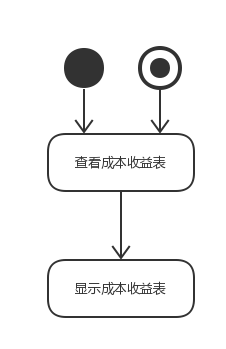


图1.5.7查看成本收益表

##### 4.2.1.5.8查看经营情况表

针对查看经营情况表任务，依据查看经营情况表场景和规格说明需求，可以设计下列独立界面或界面组件：

●查看经营情况表：进行查看经营情况表任务导航的主要部分。

●开始日期输入：查看经营情况表任务界面中的一个独立组件。

●结束日期输入：查看经营情况表任务界面中的一个独立组件。

●日期错误提示：独立界面。

基于上述独立界面或者独立组件，可以建立查看经营情况表交互过程的导航——对话结构，如图所示：

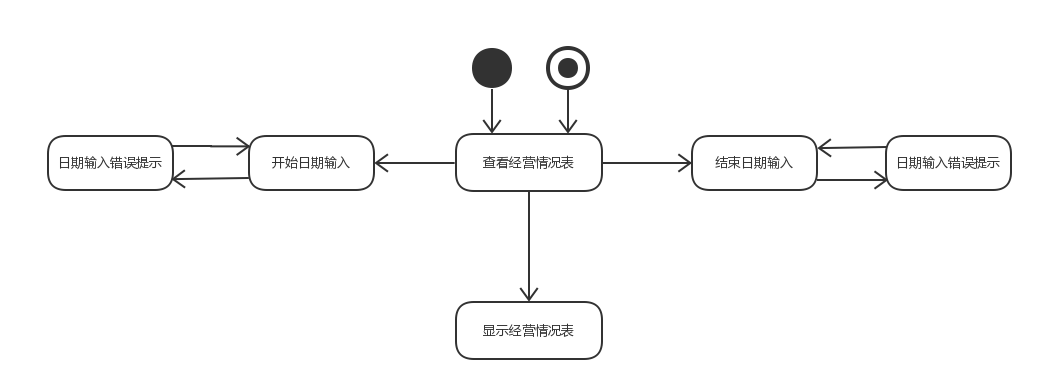


图1.5.8查看经营情况表

##### 4.2.1.5.9查看日志

针对查看日志任务，依据查看日志场景和规格说明需求，可以设计下列独立界面或界面组件：

●查看日志：进行查看日志任务导航的主要部分。

●日期输入：查看日志任务界面中的一个独立组件。

●日期错误提示：独立界面。

基于上述独立界面或者独立组件，可以建立查看经营情况表交互过程的导航——对话结构，如图所示：

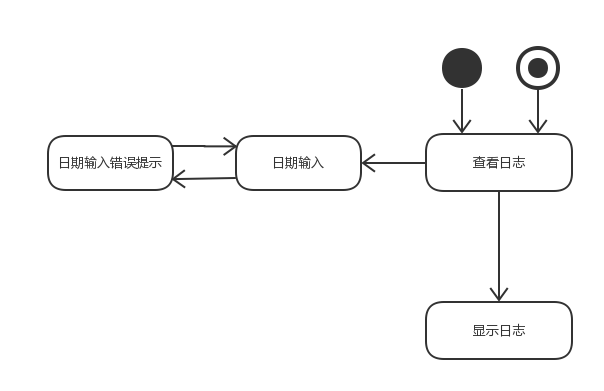


图1.5.9查看日志

#### 4.2.1.6总经理

总经理的任务主要有七个：制定薪资标准、制定价格常量、审批单据、查看账户、查看成本收益表、查看经营情况表、查看日志。可以据此建立菜单导航。

##### 4.2.1.6.1制定薪资标准

针对制定薪资标准任务，依据建立场景和规格说明需求，可以设计下面独立界面或界面组件：

● 制定薪资标准：进行制定薪资标准任务导航的主要部分

● 快递员的揽件提成输入：制定薪资标准界面的一个独立组件

● 快递员的派件提成输入：制定薪资标准界面的一个独立组件

● 司机市内计次输入：制定薪资标准界面的一个独立组件

● 司机跨市计次输入：制定薪资标准界面的一个独立组件

● 业务员月薪输入：制定薪资标准界面的一个独立组件

基于上述独立界面或者独立组件，可以建立制定薪资标准交互过程的导航——对话结构，如图所示：

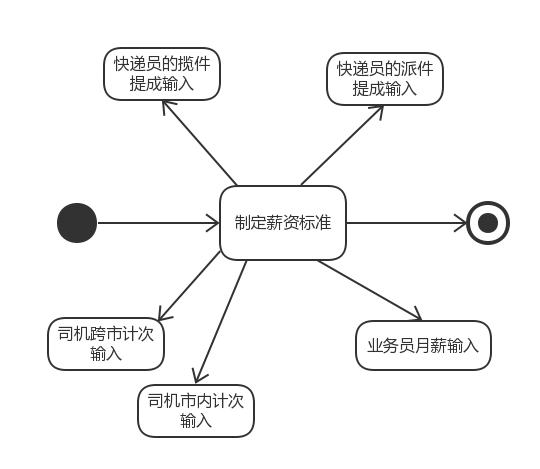


图1.6.1 制定薪资标准对话结构

##### 4.2.1.6.2制定价格常量

针对制定价格常量任务，依据制定价格常量场景和规格说明需求，可以设计下列独立界面或界面组件：

● 制定价格常量：进行制定价格常量任务导航的主要部分。

● 距离价格常量输入：制定价格常量界面的一个独立组件

基于上述独立界面或者独立组件，可以建立制定价格常量交互过程的导航——对话结构，如图所示：

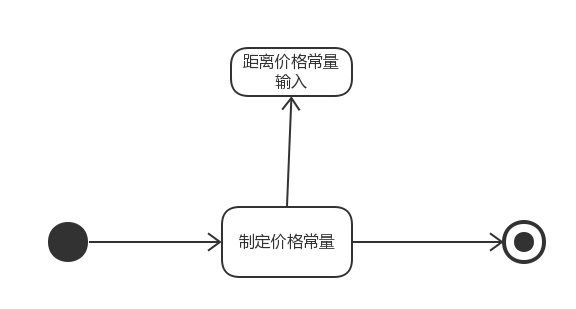


图1.6.2制定价格常量对话结构

##### 4.2.1.6.3审批单据

针对审批单据任务，依据审批单据场景和规格说明需求，可以设计下列独立界面或界面组件：

● 审批单据：进行审批单据任务导航的主要部分

● 所有状态为提交的单据显示：审批单据任务界面中的一个组件

● 单据查看输入：审批单据任务界面中的一个组件

● 单据信息显示：独立界面

● 批量审批输入：审批单据任务界面中的一个组件

基于上述独立界面或者独立组件，可以建立审批单据交互过程的导航——对话结构，如图所示：

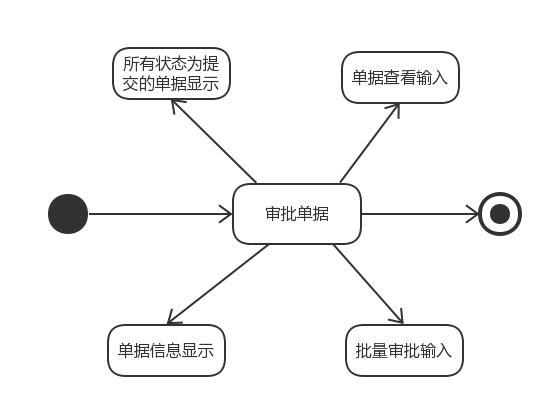


图1.6.3 审批单据对话结构

##### 4.2.1.6.4查看账户

针对查看账户任务，依据查看账户场景和规格说明需求，可以设计下列独立界面或界面组件：

● 查看账户：进行查看账户任务导航的主要部分

● 银行账号输入：查看账户任务界面的一个独立组件

● 银行账号输入错误提示：独立界面

● 账户信息显示：独立界面

基于上述独立界面或者独立组件，可以建立查看账户交互过程的导航——对话结构，如图所示：

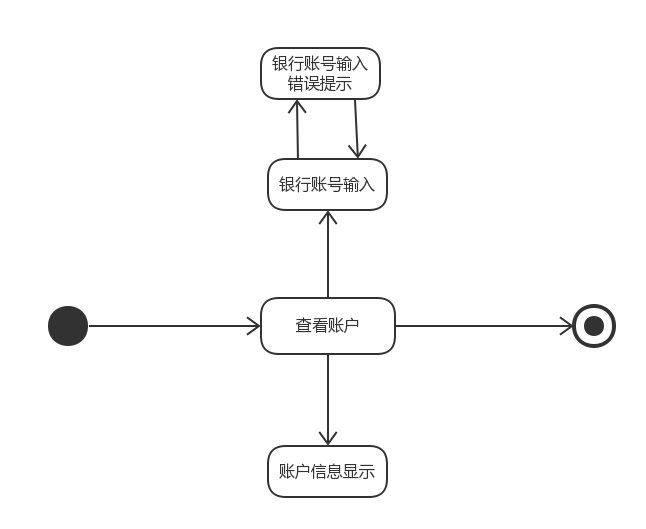


图1.6.4 查看账户对话结构

##### 4.2.1.6.5查看成本收益表

针对查看成本收益表任务，依据查看成本收益表场景和规格说明需求，可以设计下列独立界面或界面组件：

● 查看成本收益表：进行查看成本收益表任务导航的主要部分

● 所有成本收益表显示：查看成本收益表任务界面的一个独立组件

● 成本收益表查看输入：查看成本收益表任务界面的一个独立组件

● 成本收益表显示：独立界面

基于上述独立界面或者独立组件，可以建立查看成本收益表交互过程的导航——对话结构，如图所示：

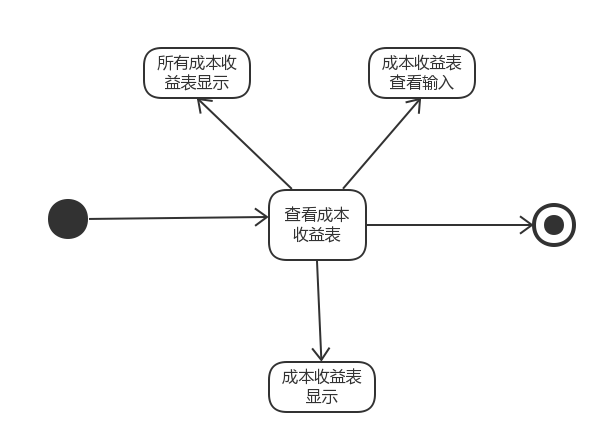


图1.6.5 查看成本收益表对话结构

##### 4.2.1.6.6查看经营情况表

针对查看经营情况表任务，依据查看经营情况表场景和规格说明需求，可以设计下列独立界面或界面组件：

● 查看经营情况表：进行查看经营情况表任务导航的主要部分

● 开始日期输入：查看经营情况表任务界面的一个独立组件。

● 开始日期输入错误提示：独立界面

● 结束日期输入：查看经营情况表任务界面的一个独立组件。

● 结束日期输入错误提示：独立界面

基于上述独立界面或者独立组件，可以建立查看经营情况表交互过程的导航——对话结构，如图所示：

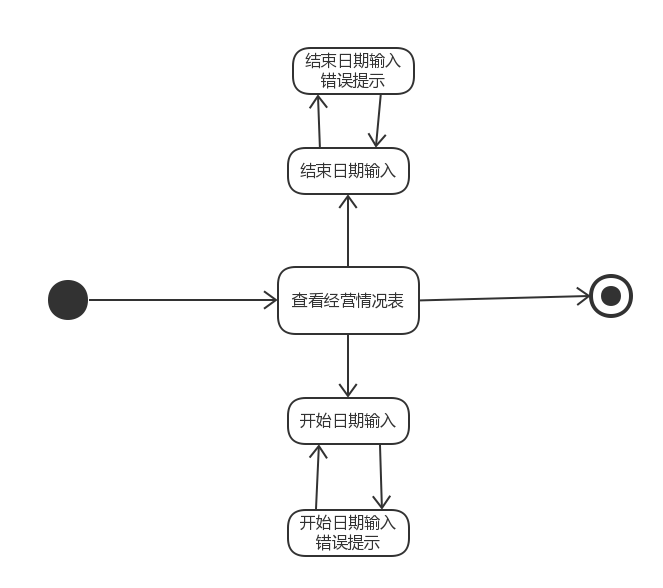


图1.6.6 查看经营情况表对话结构

##### 4.2.1.6.7查看日志

针对查看日志任务，依据查看日志场景和规格说明需求，可以设计下列独立界面或界面组件：

● 查看日志：进行查看日志任务导航的主要部分

● 日期输入：查看日志任务界面的一个独立组件。

● 日期输入错误提示：独立界面

● 日志显示：独立界面

基于上述独立界面或者独立组件，可以建立查看日志交互过程的导航——对话结构，如图所示：

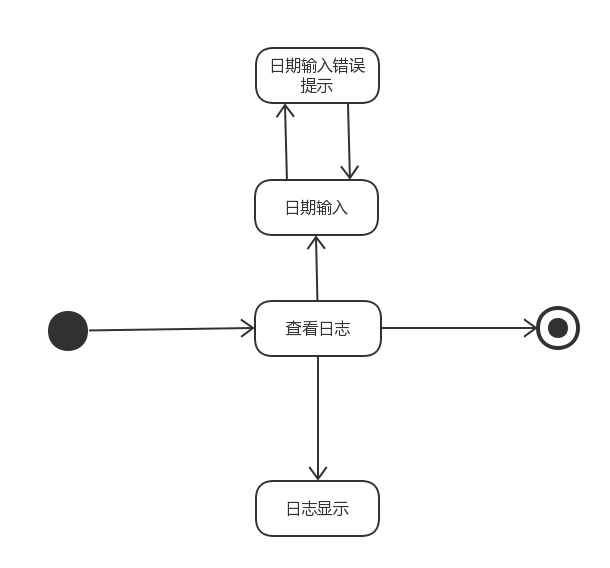


图1.6.7 查看日志对话结构

#### 4.2.1.7管理员

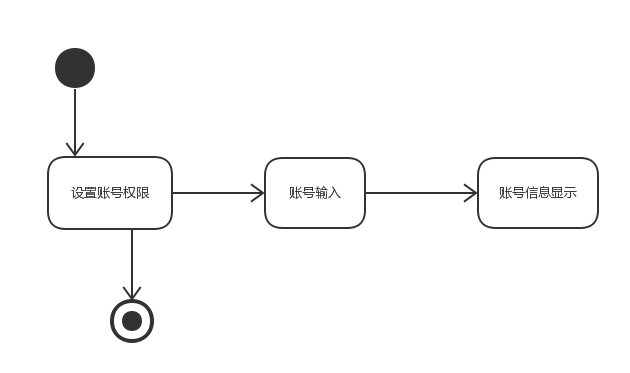
##### 4.2.1.7.1设置账号权限

针对设置账号权限任务，依据设置账号权限场景和规格说明需求，可以设计下列独立界面或界面组件：

●设置账号权限：进行设置账号权限任务的主要导航部分。

●账号输入：设置账号权限任务界面的一个独立组件。

●账号信息显示：独立界面



##### 4.2.1.7.2增删查账号

针对增删查账号任务，依据增删查账号场景和规格说明需求，可以设计下列独立界面或界面组件：

●增删查账号：进行增删查账号任务的主要导航部分。

●新增账号选择：增删查账号任务界面中的一个独立组件。

●删除账号选择：增删查账号任务界面中的一个独立组件。

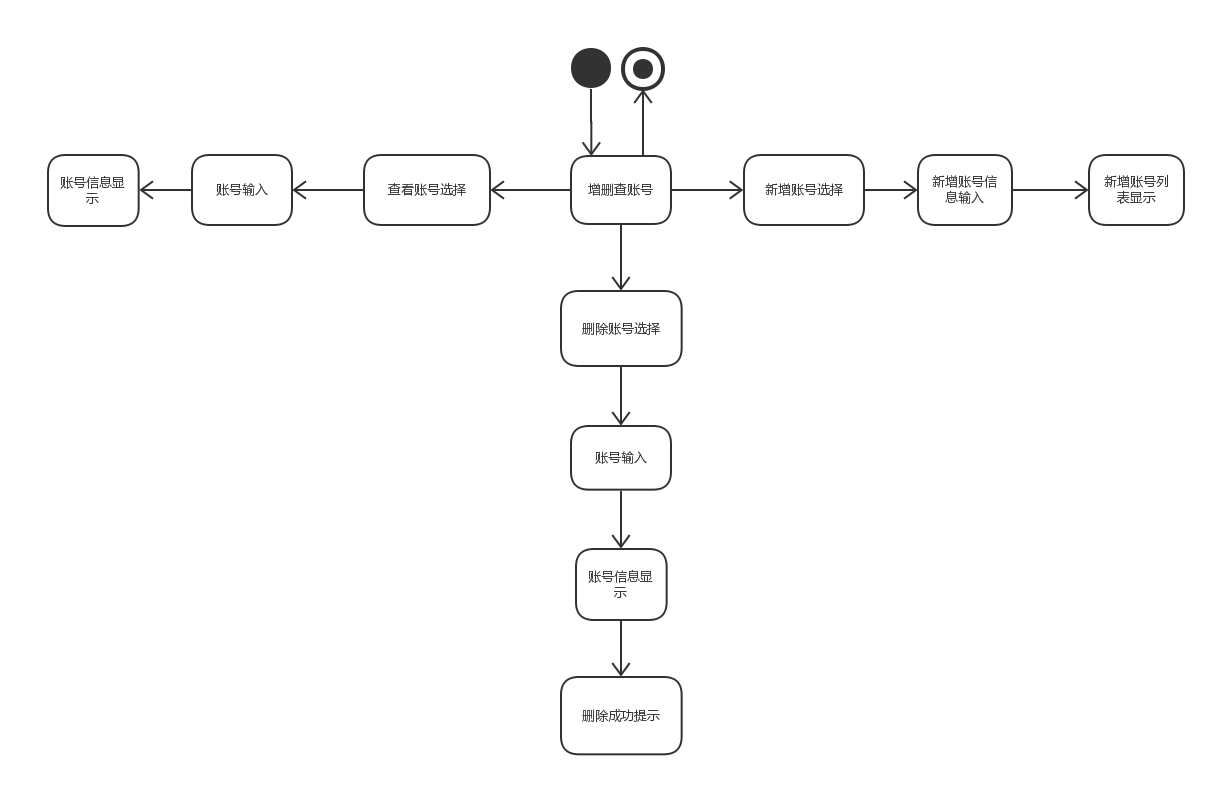
●查询账号选择：增删查账号任务界面中的一个独立组件。

●账号信息显示：独立界面。

●新增账号信息输入：独立界面。

●新增账号列表显示：独立界面。

●删除成功提示：独立界面。



### 4.2.2.界面设计

#### 4.2.2.1快递员



图2.1-1快递员处理寄件单、输入收件信息界面设计

#### 4.2.2.2营业厅业务员



图2.2-1营业厅业务员维护司机信息界面设计

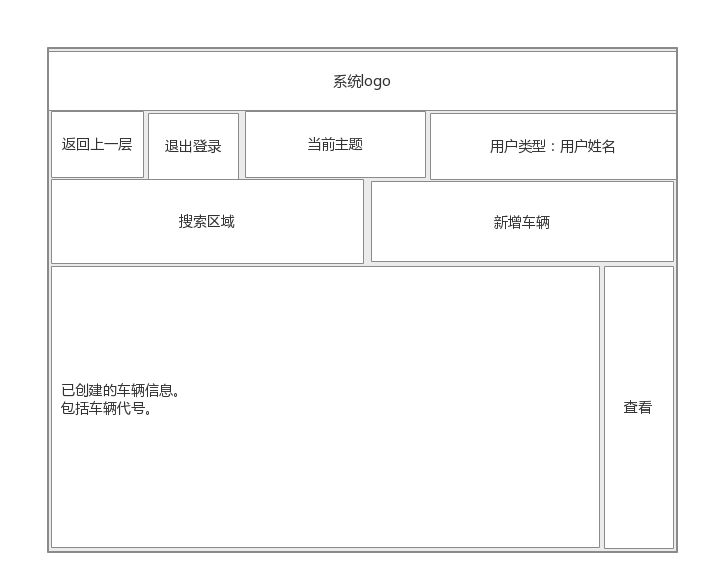


图2.2-2营业厅业务员维护车辆信息界面设计

营业厅业务员建立营业厅到达单、生成派件单、录入装车单界面设计与图2.1-1类似

#### 4.2.2.3中转中心业务员

中转中心业务员建立中转中心到达单、中转单、装车单界面设计与图2.1-1类似

#### 4.2.2.4中转中心仓库管理人员

中转中心仓库管理人员建立入库单、出库单界面设计与图2.1-1类似

#### 4.2.2.5财务人员



图2.2-3营业厅业务员维护银行账户信息界面设计

财务人员生成付款单界面设计与图2.1-1类似

#### 4.2.2.6管理员



图2.6-1 管理员维护账号信息首页界面设计

#### 4.2.2.7参与者主界面



### 4.2.3.界面原型化

#### 4.2.3.1快递员

##### 4.2.3.1-1 快递员首页界面设计



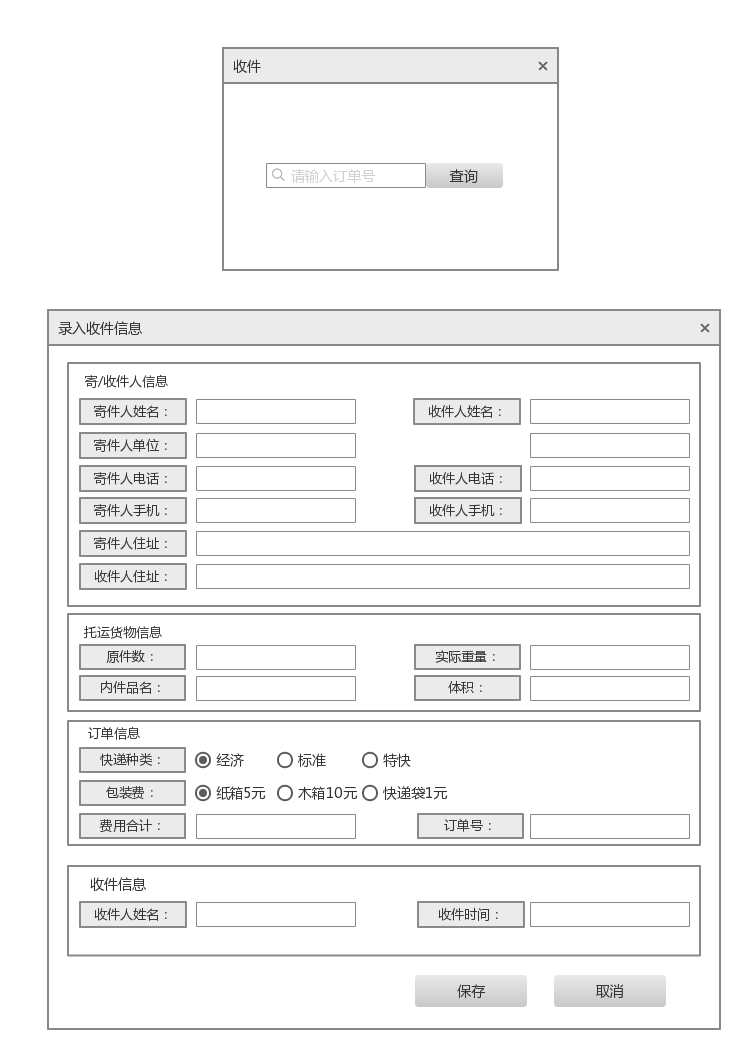
##### 4.2.3.1-2 快递员建立寄件单界面原型



##### 4.2.3.1-3 快递员订单信息查询界面原型



##### 4.2.3.1-4 快递员输入收件信息界面原型



（注：只有收件信息框中的输入框是可以填写的，上方的信息是灰的只显示，不可修改）

#### 4.2.3.2营业厅业务员

##### 4.2.3.2-1营业厅业务员首页界面设计



##### 4.2.3.2-2营业厅业务员建立营业厅到达单界面原型



##### 4.2.3.2-3营业厅业务员生成派件单界面原型



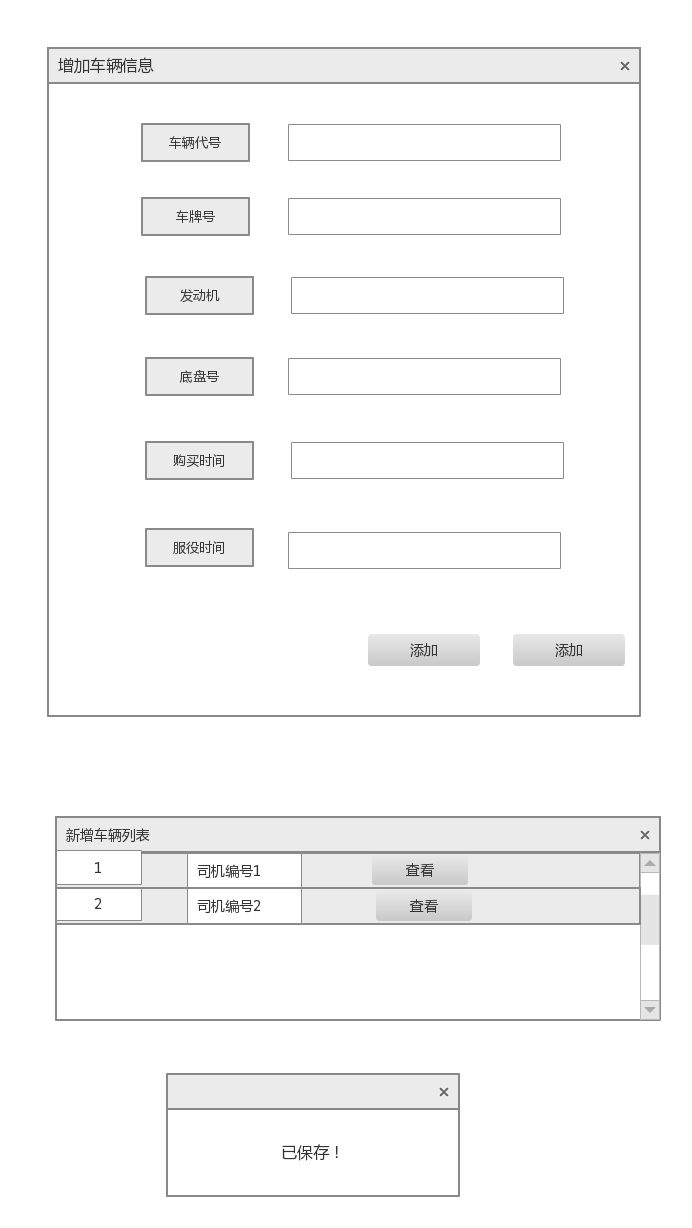
##### 4.2.3.2-4营业厅业务员建立营业厅装车单界面原型

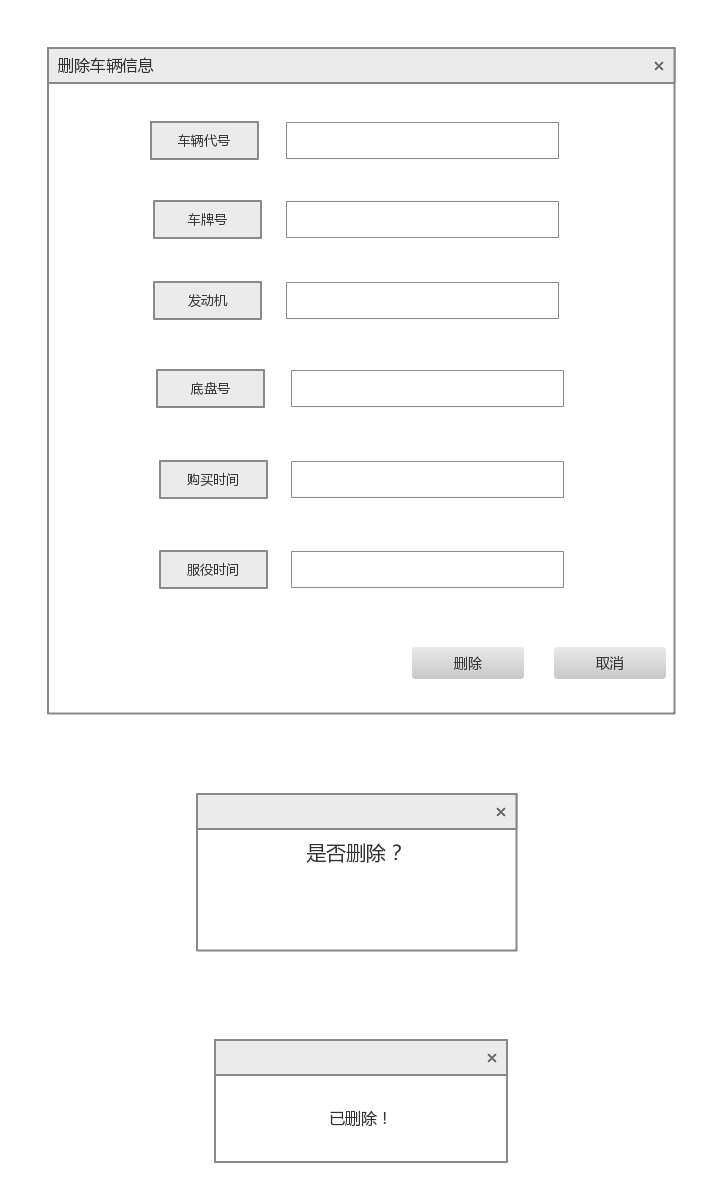


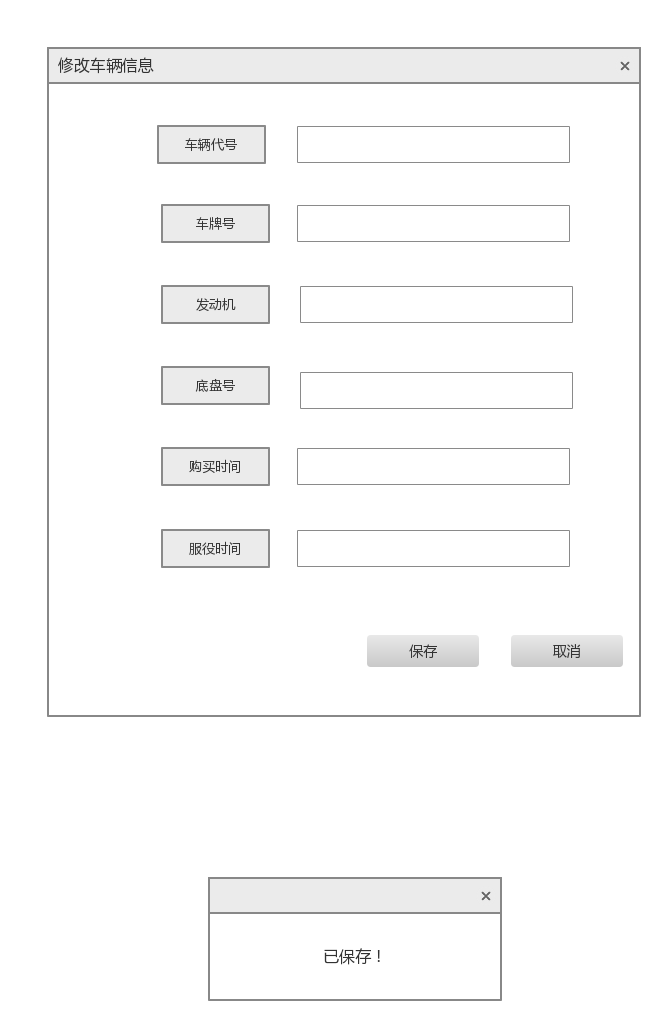
##### 4.2.3.2-5营业厅业务员建立收款单界面原型

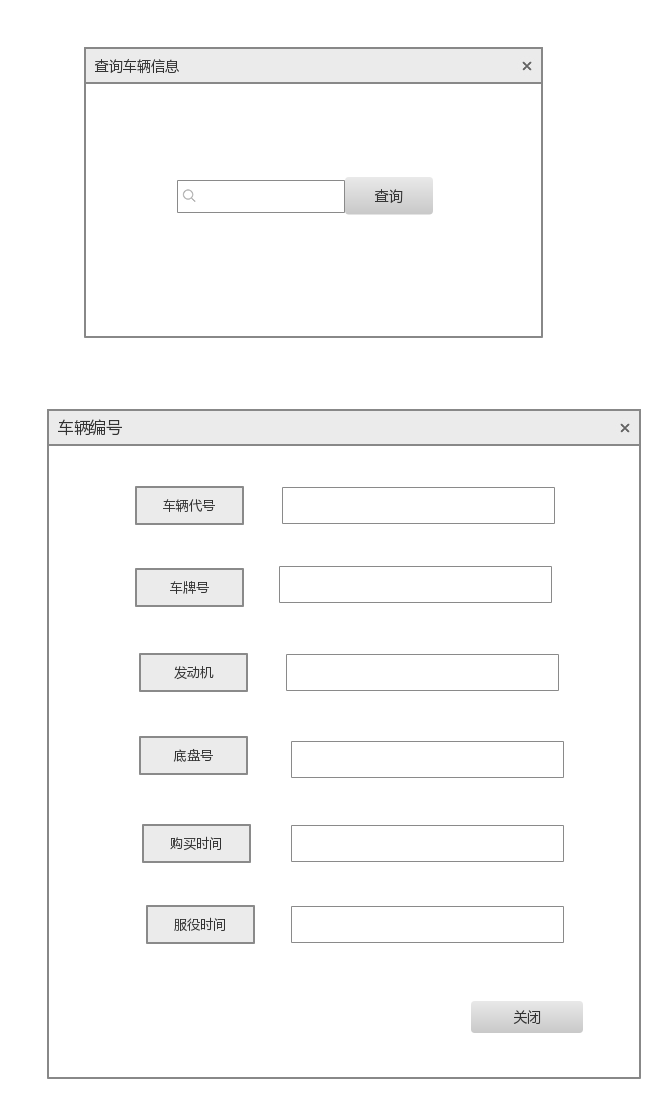


##### 4.2.3.2-6营业厅业务员维护车辆信息界面原型

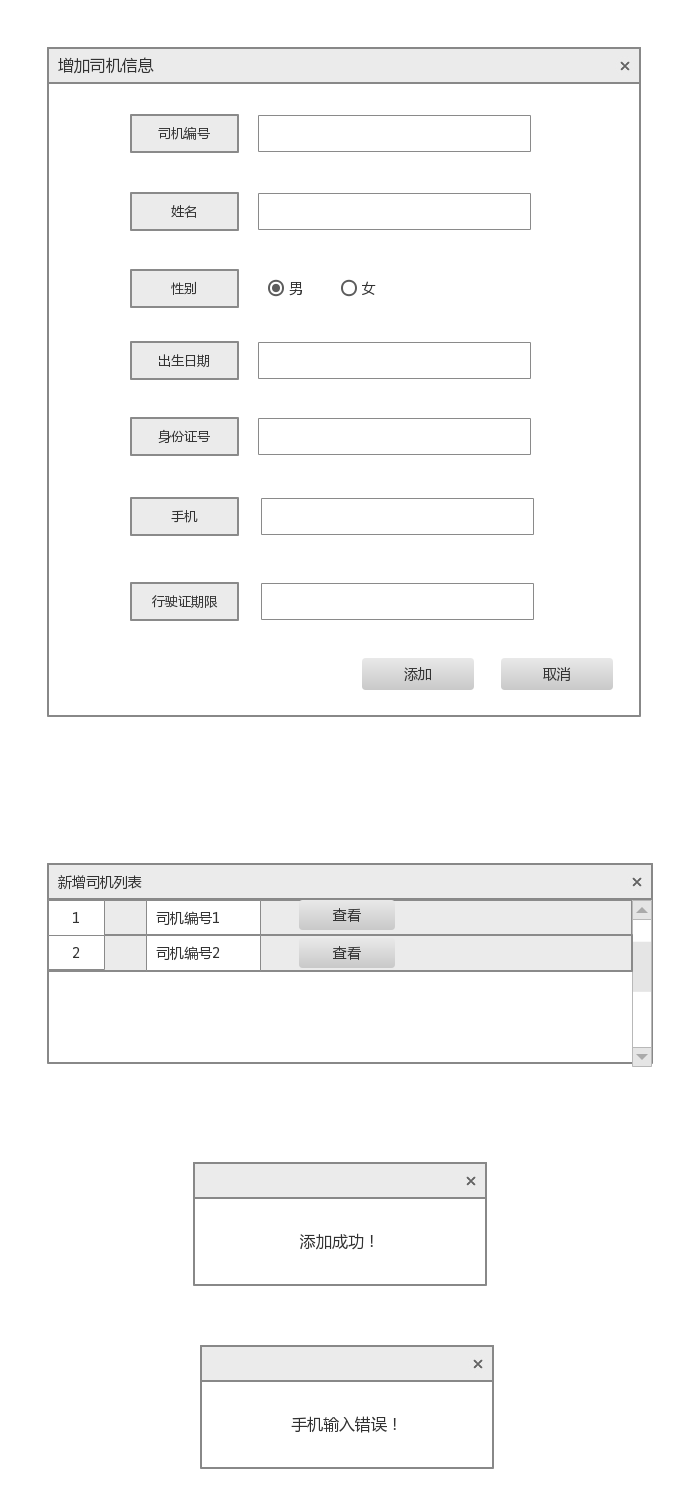






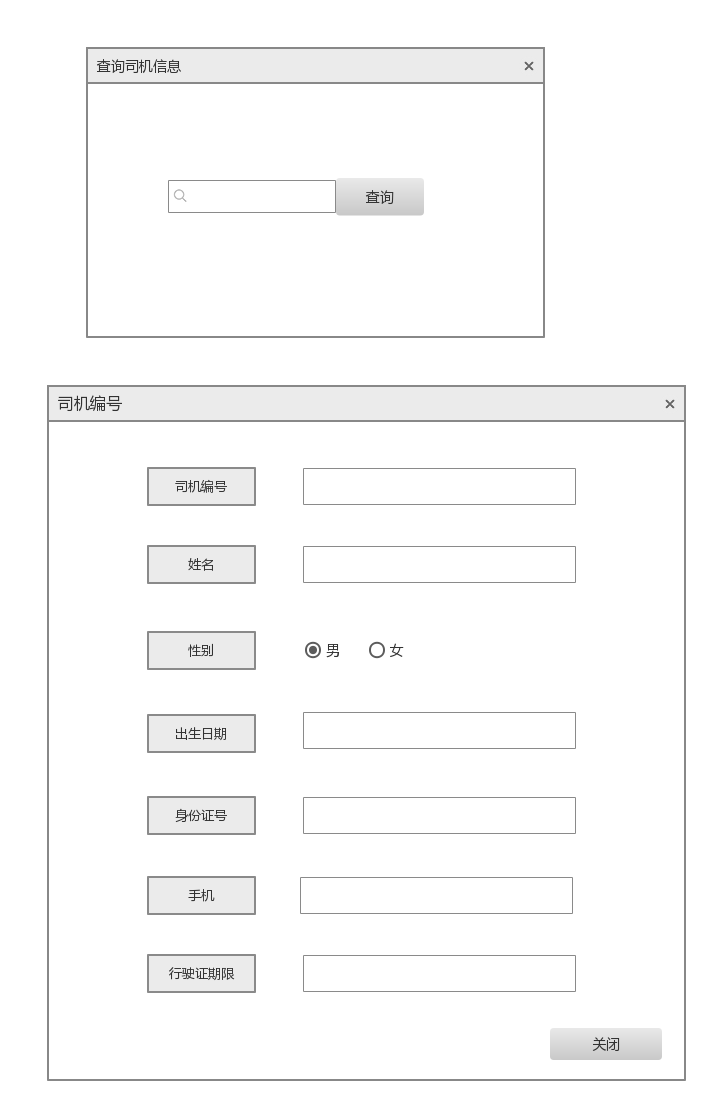


##### 4.2.3.2-7营业厅业务员维护司机信息界面原型









#### 4.2.3.3中转中心业务员

##### 4.2.3.3-1中转中心业务员首页界面设计



##### 4.2.3.3-2中转中心业务员生成中转中心到达单界面原型



##### 4.2.3.3-3中转中心业务员录入中转单界面原型



##### 4.2.3.3-4中转中心业务员录入中转中心装车单界面原型



#### 4.2.3.4中转中心仓库管理人员

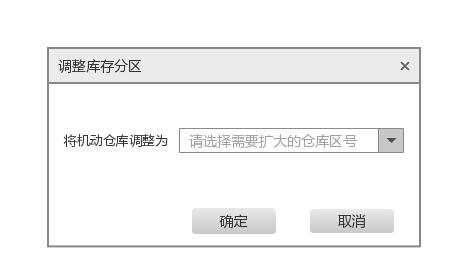
##### 4.2.3.4-1中转中心仓库管理人员首页界面设计



##### 4.2.3.4-2中转中心仓库管理人员建立入库单界面原型



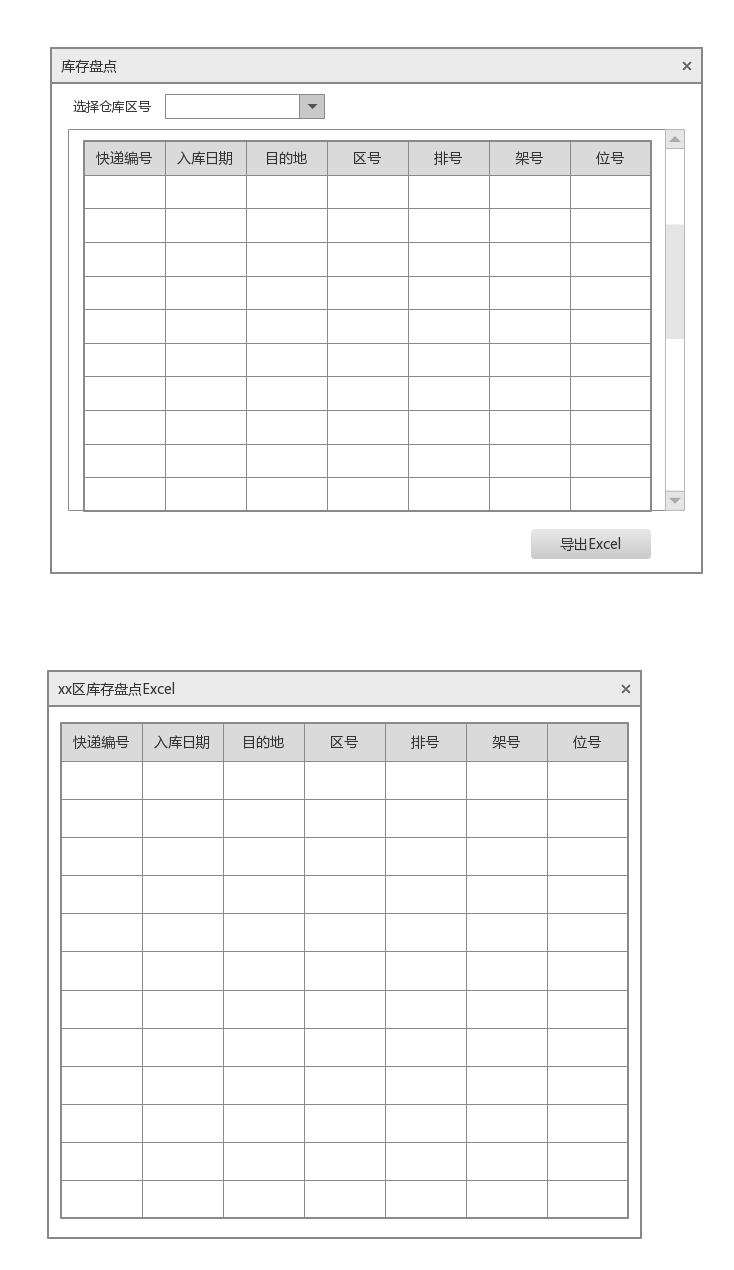
##### 4.2.3.4-3中转中心仓库管理人员调整库存分区界面原型



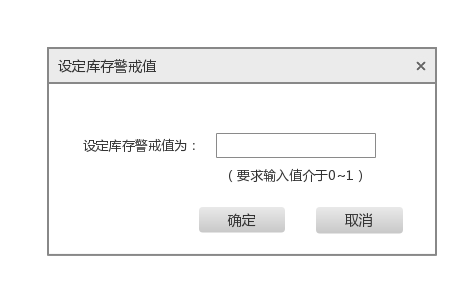
##### 4.2.3.4-4中转中心仓库管理人员库存查看界面原型



##### 4.2.3.4-5中转中心仓库管理人员库存盘点界面原型



##### 4.2.3.4-6中转中心仓库管理人员设置库存警戒值界面原型



##### 4.2.3.4-7中转中心仓库管理人员建立出库单界面原型



#### 4.2.3.5财务人员

##### 4.2.3.5-1普通财务人员首页界面设计



##### 4.2.3.5-2高级财务人员首页界面设计



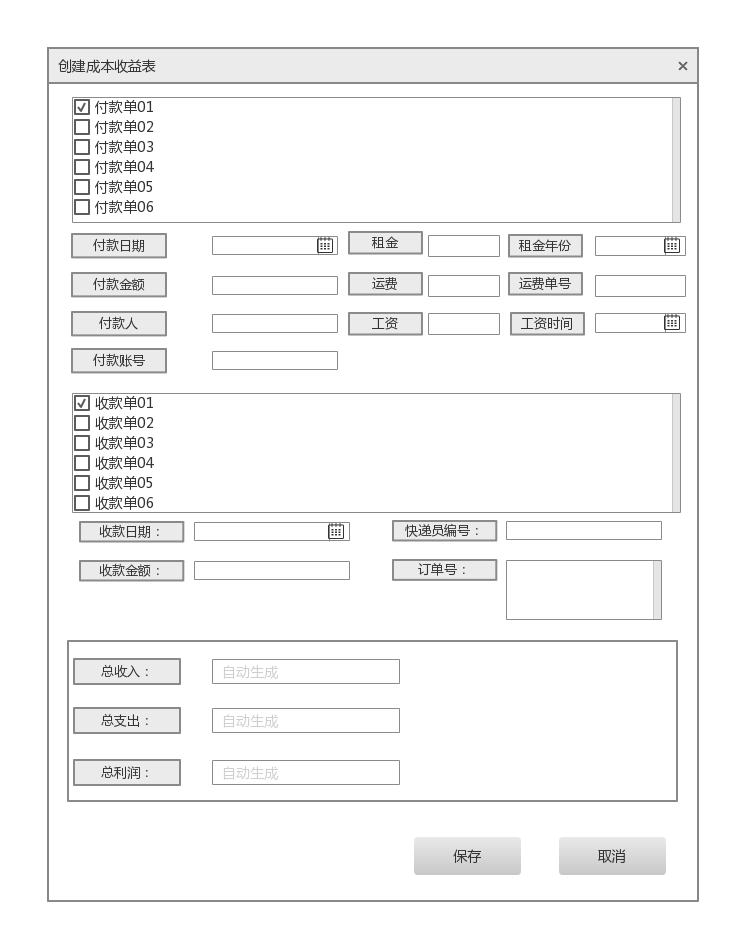
##### 4.2.3.5-3财务人员收款单汇总界面原型



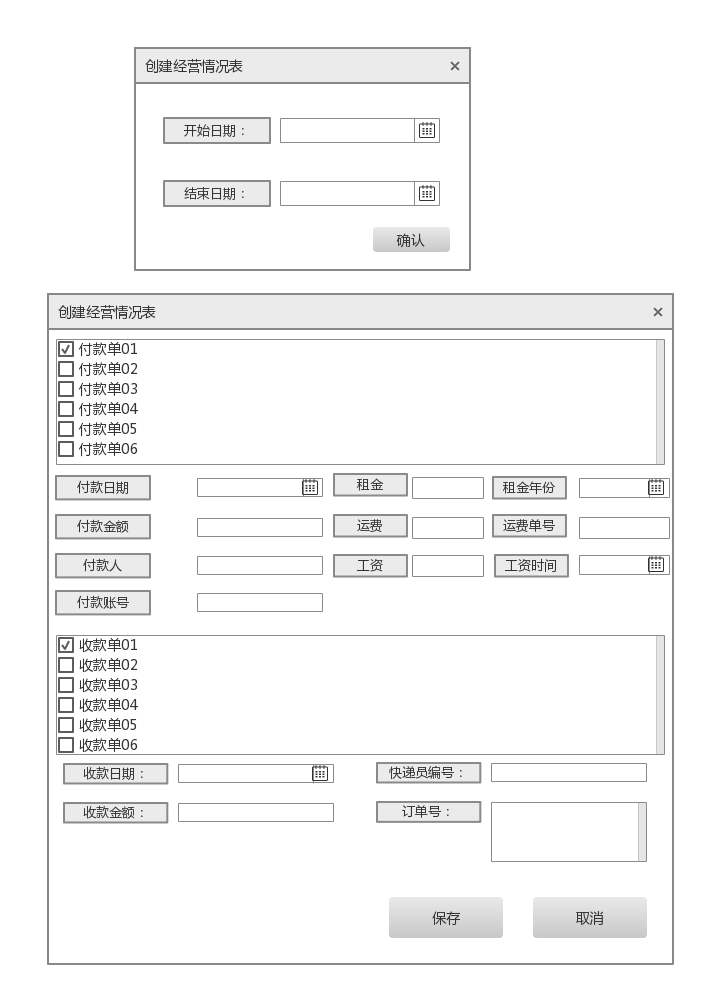
##### 4.2.3.5-4财务人员生成付款单界面原型



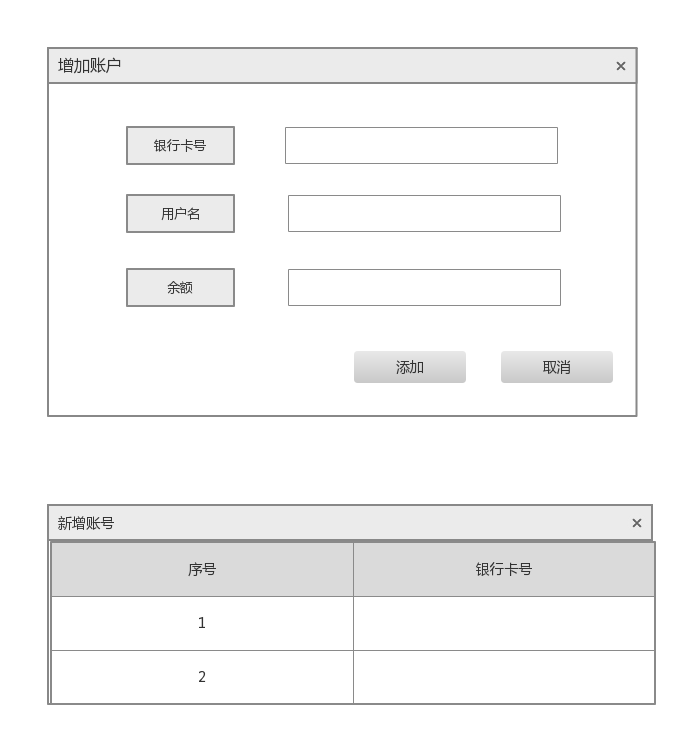
##### 4.2.3.5-5财务人员生成成本收益表界面原型

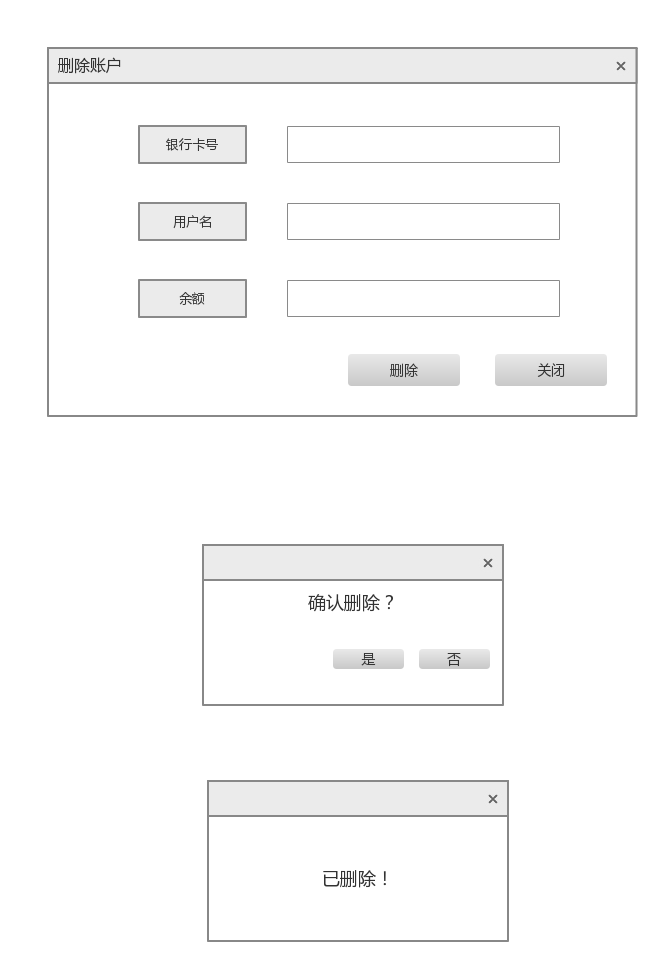


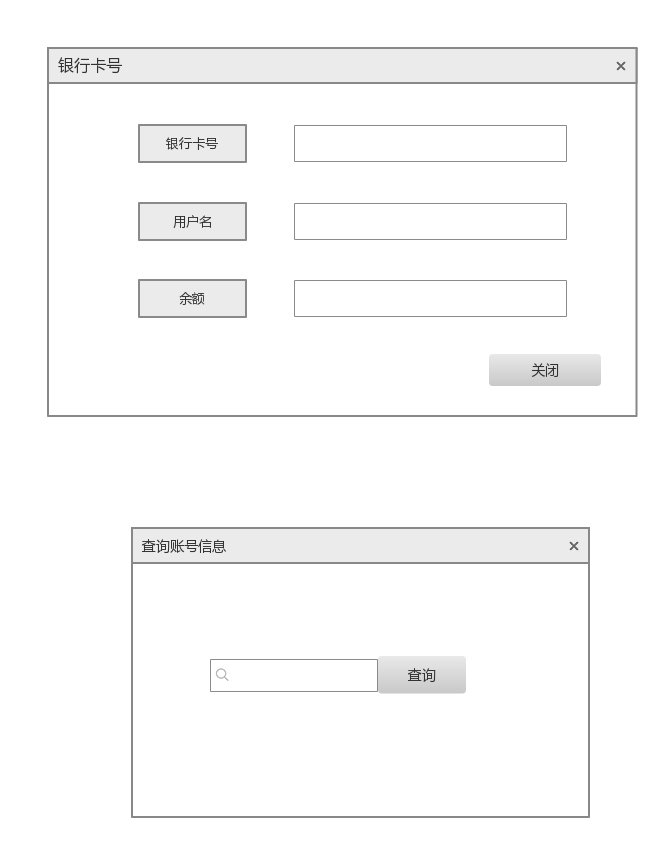
##### 4.2.3.5-6财务人员生成经营情况表界面原型



##### 4.2.3.5-7财务人员增删查账户界面原型



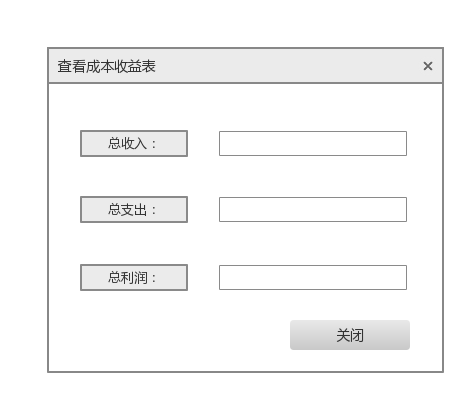




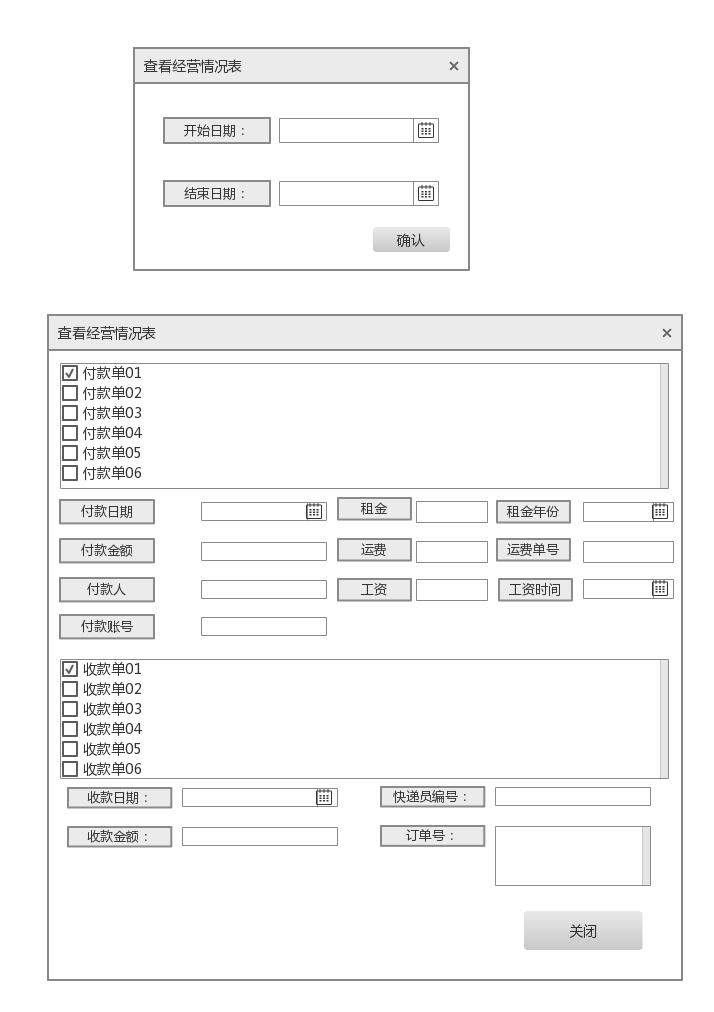
##### 4.2.3.5-8财务人员期初建账界面原型



##### 4.2.3.5-9财务人员查看成本收益表界面原型



##### 4.2.3.5-10财务人员查看经营情况表界面原型

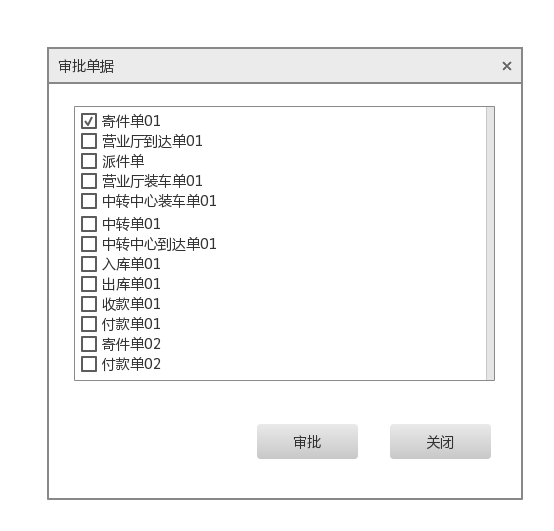


#### 4.2.3.6总经理

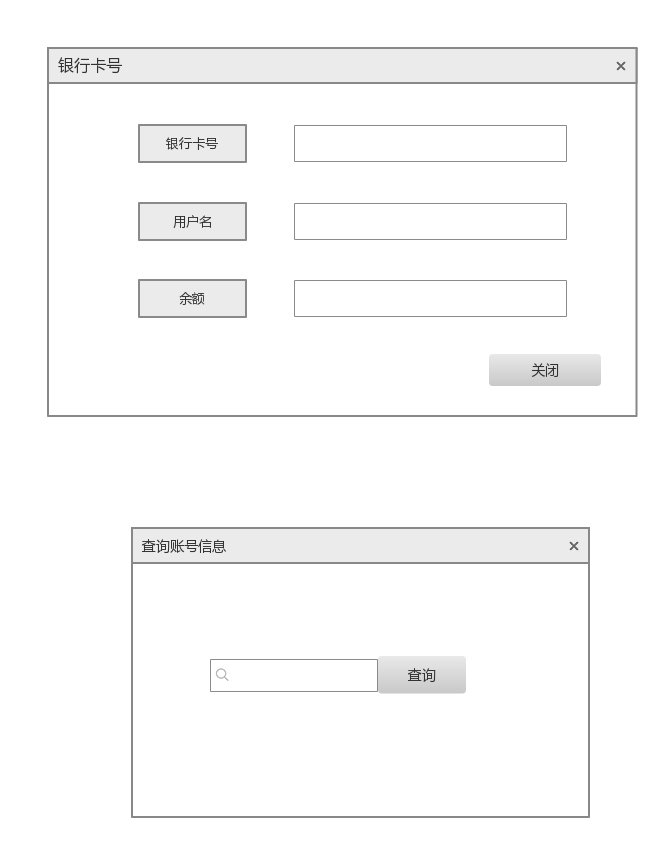
##### 4.2.3.6-1总经理首页界面设计



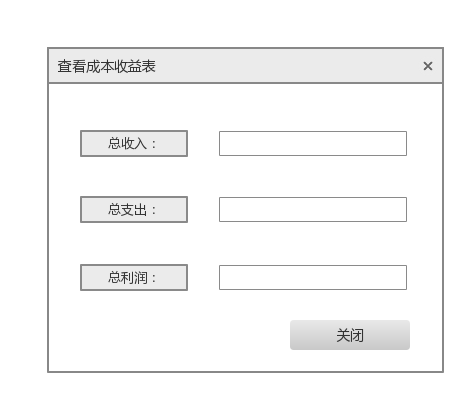
##### 4.2.3.6-2总经理审批单据界面原型



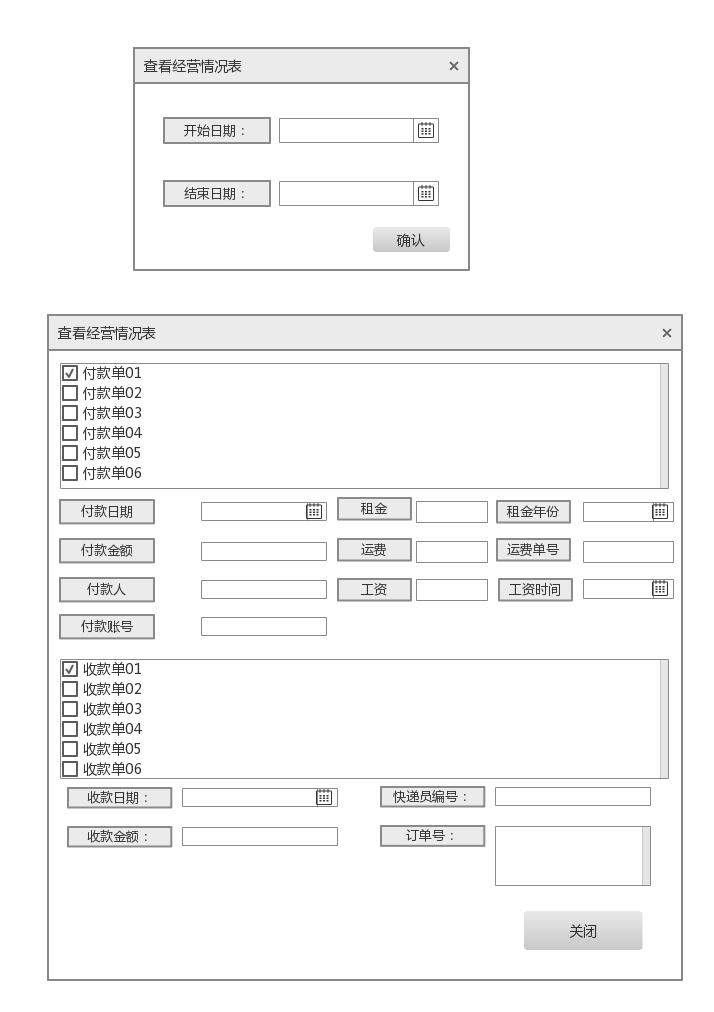
##### 4.2.3.6-3总经理查看账户界面原型



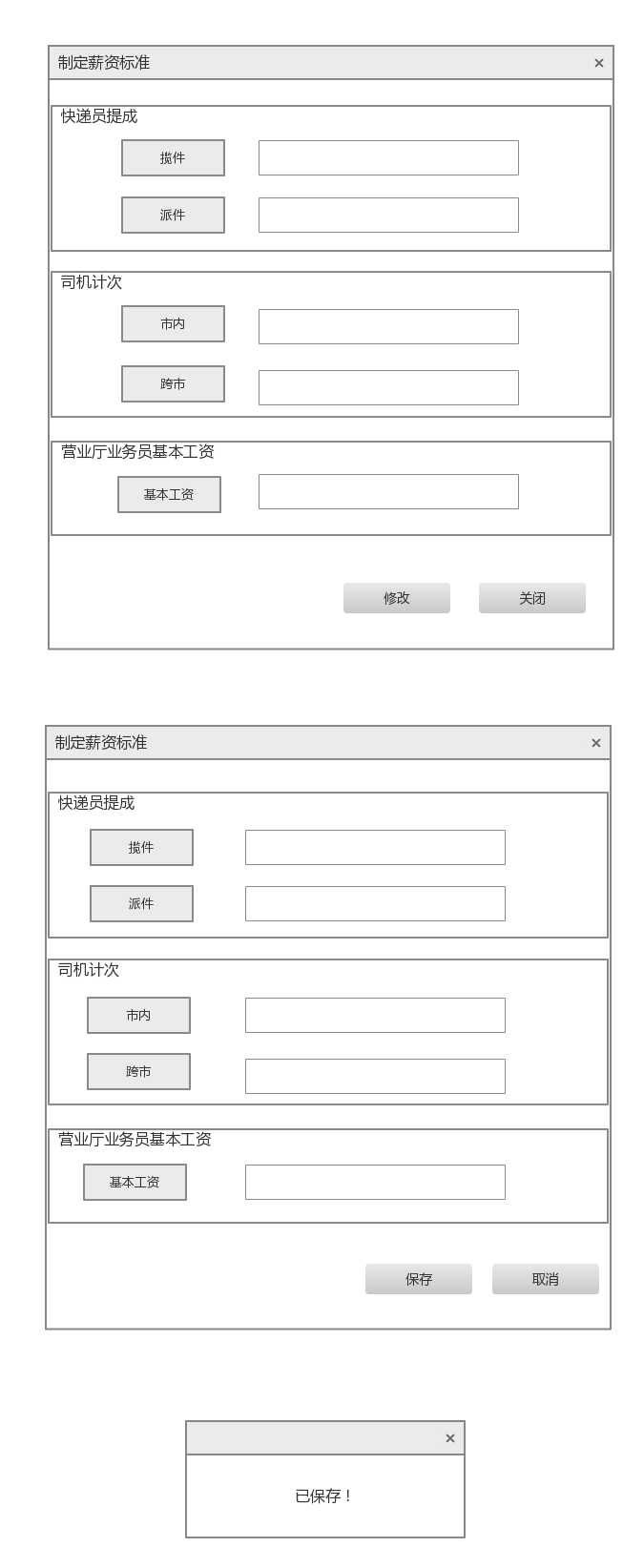
##### 4.2.3.6-4总经理查看成本收益表界面原型



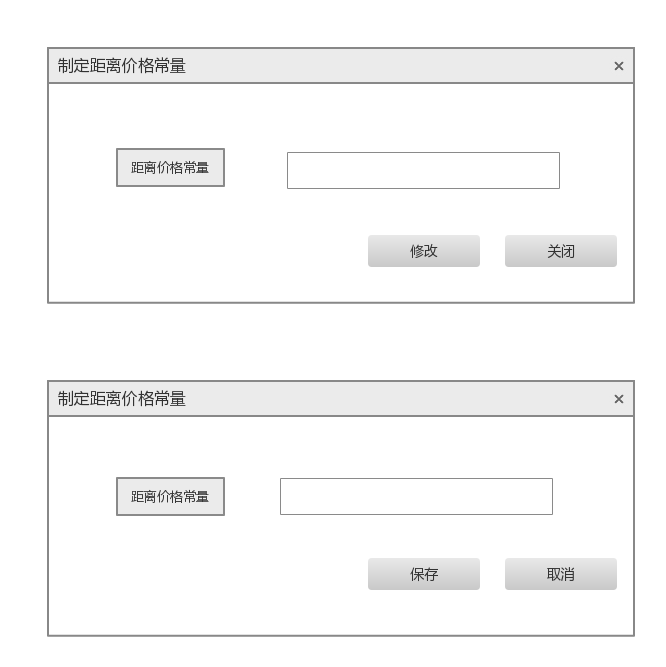
##### 4.2.3.6-5总经理查看经营情况表界面原型



##### 4.2.3.6-6总经理制定薪资标准界面原型



##### 4.2.3.6-7总经理制定价格常量界面原型



#### 4.2.3.7管理员

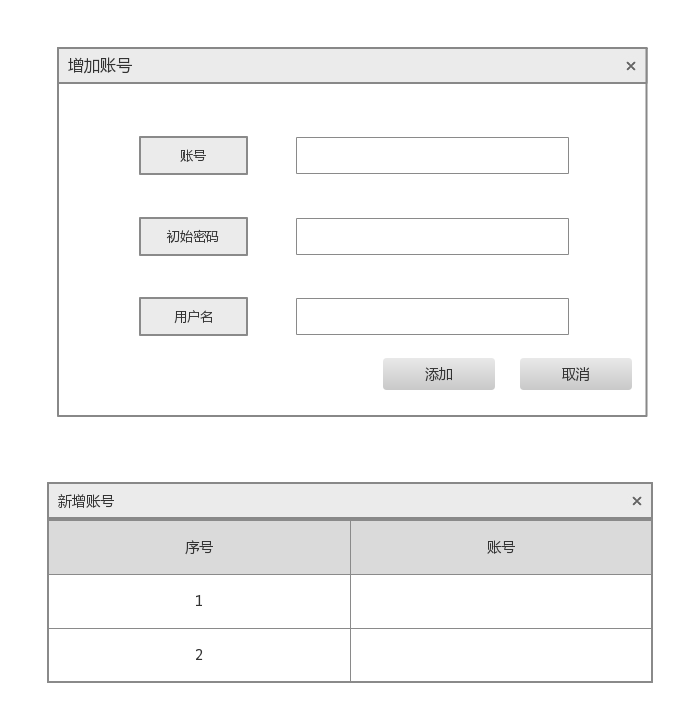
##### 4.2.3.7-1管理员首页界面设计

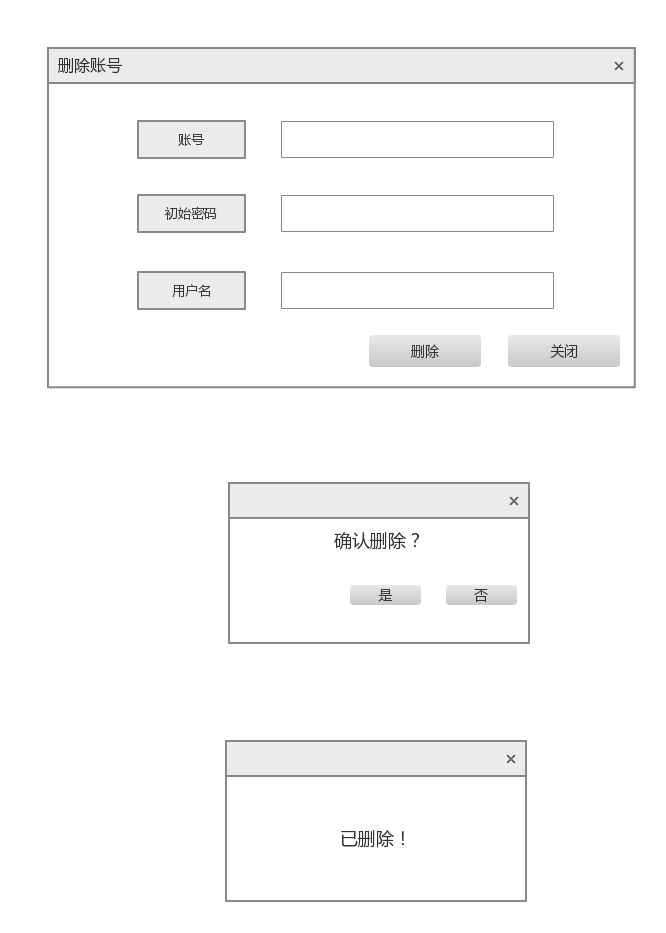


##### 4.2.3.7-2管理员设置账号权限界面原型



##### 4.2.3.7-3管理员增删查账号界面原型







#### 4.2.3.8登录

##### 4.2.3.8-1登录首页界面设计

