### 4.2.7managebl模块

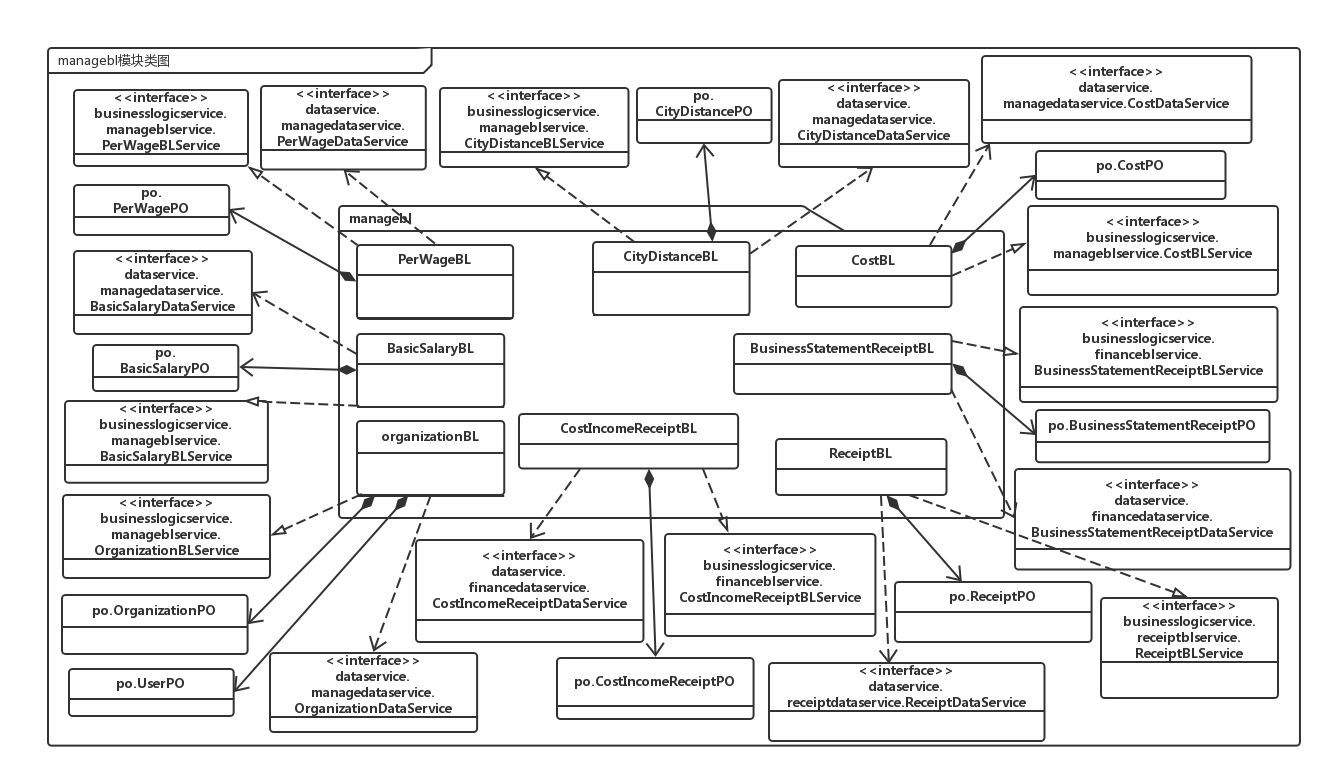
#### 4.2.7.1模块概述

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

managebl模块的职责及接口参见ELS体系结构设计文档表14。

#### 4.2.7.2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.manageblservice.OrganizationBLService、businesslogicservice.manageblservice.BasicSalaryBLService、businesslogicservice.manageblservice.CityDistanceBLService等接口。业务逻辑层和数据层之间添加dataservice.managedataservice.OrganizationDataService、dataservice.managedataservice.BasicSalaryDataService、ataservice.managedataservice.CityDistanceDataService等接口。因为业务逻辑处理较少，根据专家模式没有添加Controller类。OrganizationPO等PO对象是作为总经理管理记录的持久化对象被添加到设计模型中去的。

managebl模块的设计如图7-1所示。

**图7-1 managebl模块各个类的设计**

managebl模块各个类的职责如表7-1所示。

**表7-1 managebl模块各个类的职责**

|  |  |
| --- | --- |
| 模块 | 职责 |
| OrganizationBL | 机构的领域模型对象，拥有机构数据的类别、编号、名称、下属仓库（若是营业厅则无仓库），可以解决总经理机构管理的问题。 |
| ReviewReceiptBL | 单据审批的领域模型对象，可以解决总经理单据审批的问题。 |
| BusinessStatementReceiptBL | 经营情况表的领域模型对象，拥有一个经营情况表所持有的入款记录、付款记录、编号等信息，可以帮助完成查看经营情况表界面所需要的服务。 |
| CostIncomeReceiptBL | 成本收益表的领域模型对象，拥有一个经营情况表所持有的编号、操作员、收入、支出金额等信息，可以帮助完成查看成本收益表界面所需要的服务。 |
| PerWageBL | 每次工资的领域模型对象，拥有每次工资所持有的职业、工资金额等信息，可以解决总经理每次工资管理的问题。 |
| BasicSalaryBL | 基础月薪的领域模型对象，拥有基础月薪所持有的职业、月薪金额等信息，可以解决总经理基础月薪管理的问题。 |
| CityDistanceBL | 城市距离的领域模型对象，拥有城市距离所持有的城市、距离数等信息，可以解决总经理城市距离管理的问题。 |
| CostBL | 运费系数的领域模型对象，拥有运费系数所持有的类型、运费系数等信息，可以解决总经理运费系数管理的问题。 |

#### 4.2.7.3模块内部类的接口规范

OrganizationBL的接口规范如表7-2所示。

**表7-2 OrganizationBL模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| OrganizationBL.addOrganization | 语法 | | public int addOrganization(OrganizationVO organizationvo) |
| 前置条件 | | 经过验证的总经理有添加新机构的权利，且总经理输入了完整有效的新机构信息 |
| 后置条件 | | 系统保存新增的机构信息并更新机构列表 |
| OrganizationBL.deleteOrganization | 语法 | | public int deleteOrganization(OrganizationVO organizationvo) |
| 前置条件 | | 经过验证的总经理有删除机构的权利，且总经理输入了有效的待删除机构信息 |
| 后置条件 | | 系统查找对应的机构信息，并删除该机构对象 |
| OrganizationBL.modifyOrganization | 语法 | | public int modifyOrganization(OrganizationVO organizationvo) |
| 前置条件 | | 经过验证的总经理有修改机构信息的权利，且总经理输入了有效的待修改机构信息 |
| 后置条件 | | 系统更新机构信息 |
| OrganizationBL.findOrganization | 语法 | | public OrganizationVO findOrganization(String organizationID) |
| 前置条件 | | 经过验证的总经理有查询机构的权利，且总经理输入了有效的待查询机构编号 |
| 后置条件 | | 系统根据输入的机构编号查找机构，并返回机构信息 |
| OrganizationBL.chooseDepartment | 语法 | | public int chooseDepartment(String userID, String organizationID) |
| 前置条件 | | 经过验证的总经理有为新员工选择所属机构的权利，且总经理输入了有效的员工编号和机构编号 |
| 后置条件 | | 系统根据输入的员工编号查找对应的员工，并将该员工所属机构修改为机构编号对应的机构 |
| 需要的服务（需接口） | | | |
| 服务名 | | 服务 | |
| ManageDataService.addOrganization(OrganizationPO organizationpo) | | 在管理数据包的机构数据文件中插入一个OrganizationPO对象 | |
| ManageDataService.deleteOrganization(OrganizationPO organizationpo) | | 在管理数据包的机构数据文件中删除一个OrganizationPO对象 | |
| ManageDataService.modifyOrganization(OrganizationPO organizationpo) | | 在管理数据包的机构数据文件中更新一个OrganizationPO对象 | |
| ManageDataService.findOrganization(String OrganizationID) | | 根据输入的机构编号，返回对应的OrganizationPO对象 | |
| ManageDataService.modifyUser(UserPO userpo) | | 在用户数据文件中更新一个UserPO对象 | |

ReviewReceiptBL的接口规范如表7-3所示。

**表7-3 ReviewReceiptBL模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务(供接口) | | | |
| ReviewReceipt.modify | 语法 | Public int modify(String ID) | |
| 前置条件 | 总经理选中了有效单据并输入了有效的修改信息 | |
| 后置条件 | 更新该单据信息 | |
| ReviewReceipt.batch | 语法 | Public int batch(String[] ID) | |
| 前置条件 | 总经理已选中一个或多个单据，还要选择一个或多个单据进行批量审批 | |
| 后置条件 | 更新该单据组信息 | |
| ReviewReceipt.reply | 语法 | Public void reply(String userID) | |
| 前置条件 | 该单据审批完成 | |
| 后置条件 | 该单据制单人收到审批结果信息 | |
| ReviewReceipt.view | 语法 | Public ArrayList<ReceiptVO> view() | |
| 前置条件 | 已有单据记录 | |
| 后置条件 | 返回所有单据信息 | |
| ReviewReceipt.refresh | 语法 | Public ArrayList<ReceiptVO> refresh() | |
| 前置条件 | 已有单据记录 | |
| 后置条件 | 返回所有待审批单据信息 | |
| ReviewReceipt.approve | 语法 | Public int approve(String ID) | |
| 前置条件 | 有单据提交待审批，总经理审批通过 | |
| 后置条件 | 更新该单据信息 | |
| 需要的服务(需接口) | | | |
| 服务名 | | | 服务 |
| FileFactory.getReceiptData | | | 得到Receipt文件的引用 |
| ReceiptDataService.modify(ReceiptPo po) | | | 更新单一持久化对象 |
| ReceiptDataService.modify(ReceiptPo[] po) | | | 更新单一持久化对象组 |
| ReceiptDataService.find(Type type) | | | 根据类型type查找ReceiptPo对象 |
| ReceiptDataService.find(Date date) | | | 根据日期date查找ReceiptPo对象 |
| ReceiptDataService.show() | | | 得到所得的ReceiptPo对象 |

BusinessStatementReceiptBL的接口规范如表7-4所示。

**表7-4 BusinessStatementReceiptBL模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| BusinessStatementReceiptBL.showBSL | 语法 | | public BusinessstatementReceiptVO show(String beginTime,String endTime); |
| 前置条件 | | 用户输入起始日期并要求查看经营情况表 |
| 后置条件 | | 系统显示该段时间内所有入款单和付款单信息 |
| BusinessStatementReceiptBL.showAllBSL | 语法 | | public ArrayList<BusinessstatementReceiptVO> showAllBSList(); |
| 前置条件 | | 用户经验证有查看经营情况表的权限 |
| 后置条件 | | 系统显示所有经营情况的列表 |
| BusinessStatementReceiptBL.export | 语法 | | public int export(BusinessstatementReceiptVO vo); |
| 前置条件 | | 用户要求导出经营情况表的excel |
| 后置条件 | | 系统生成经营情况表的excel |
| 需要的服务（需接口） | | | |
| 服务名 | | 服务 | |
| FinanceDataService.showBSL(String beginTime,String endTime) | | 返回并输出符合时间条件的单一持久化对象BusinessstatementReceiptPO | |
| public ArrayList<BusinessstatementReceiptPO> showAllBSList() | | 返回并输出所有的单一持久化对象BusinessstatementReceiptPO | |
| public int export(BusinessstatementReceiptPO po) | | 导出单一持久化对象BusinessstatementReceiptPO到excel | |

CostIncomeReceiptBL的接口规范如表7-5所示。

**表7-5 CostIncomeReceiptBL模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| CostIncomeReceiptBL.getCostIncmeList | 语法 | | public CostIncomeReceiptVO getCostIncomeList(String s); |
| 前置条件 | | 用户有查看成本收益表的权限 |
| 后置条件 | | 系统返回符合条件的成本收益表 |
| CostIncomeReceiptBL.getAllCostIncomeList | 语法 | | public ArrayList<CostIncomeReceiptVO> getAllCostIncomeList(); |
| 前置条件 | | 用户有查看成本收益表的权限 |
| 后置条件 | | 系统返回所有成本收益表列表 |
| 需要的服务（需接口） | | | |
| 服务名 | | 服务 | |
| FinanceDataService. getNum() | | 得到当天存储的持久化对象CostIncomeReceiptPO的个数 | |
| FinanceDataService. findByID(String ID) | | 根据ID返回对应的一个持久化对象CostIncomeReceiptPO | |

PerWageBL的接口规范如表7-6所示。

**表7-6 PerWageBL模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| PerWageBL.addPerWage | 语法 | | public int addPerWage(PerWageVO perwagevo) |
| 前置条件 | | 经过验证的总经理有添加新的每次工资的权利，且总经理输入了完整有效的新每次工资信息 |
| 后置条件 | | 系统保存新增的每次工资信息并更新每次工资列表 |
| PerWageBL.deletePerWage | 语法 | | public int deletePerWage(PerWageVO perwagevo) |
| 前置条件 | | 经过验证的总经理有删除每次工资的权利，且总经理输入了有效的待删除每次工资信息 |
| 后置条件 | | 系统查找对应的每次工资信息，并删除该每次工资对象 |
| PerWageBL.modifyPerWage | 语法 | | public int modifyPerWage(PerWageVO perwagevo) |
| 前置条件 | | 经过验证的总经理有修改每次工资的权利，且总经理输入了有效的待修改每次工资信息 |
| 后置条件 | | 系统更新每次工资信息 |
| PerWageBL.findPerWage | 语法 | | public double findPerWage(ProfessionType profession) |
| 前置条件 | | 经过验证的总经理有查询每次工资的权利，且总经理输入了有效的对应职业编号 |
| 后置条件 | | 系统根据输入的职业类型查找每次工资，并返回每次工资信息 |
| 需要的服务（需接口） | | | |
| 服务名 | | 服务 | |
| ManageDataService.addPerWage(PerWagePO perwagepo) | | 在管理数据包的每次工资数据文件中插入一个PerWagePO对象 | |
| ManageDataService.deletePerWage(PerWagePO perwagepo) | | 在管理数据包的每次工资数据文件中删除一个PerWagePO对象 | |
| ManageDataService.modifyPerWage(PerWagePO perwagepo) | | 在管理数据包的每次工资数据文件中更新一个PerWagePO对象 | |
| ManageDataService.findPerWage(ProfessionType profession) | | 根据输入的职业类型，返回对应的PerWagePO对象 | |

BasicSalaryBL的接口规范如表7-7所示。

**表7-7 BasicSalaryBL模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| BasicSalaryBL.addBasicSalary | 语法 | | public int addBasicSalary(BasicSalaryVO basicsalaryvo) |
| 前置条件 | | 经过验证的总经理有添加新的基础月薪的权利，且总经理输入了完整有效的新基础月薪信息 |
| 后置条件 | | 系统保存新增的基础月薪信息并更新基础月薪列表 |
| BasicSalaryBL.deleteBasicSalary | 语法 | | public int deleteBasicSalary(BasicSalaryVO basicsalaryvo) |
| 前置条件 | | 经过验证的总经理有删除基础月薪的权利，且总经理输入了有效的待删除基础月薪信息 |
| 后置条件 | | 系统查找对应的基础月薪信息，并删除该基础月薪对象 |
| BasicSalaryBL.modifyBasicSalary | 语法 | | public int modifyBasicSalary(BasicSalaryVO basicsalaryvo) |
| 前置条件 | | 经过验证的总经理有修改基础月薪的权利，且总经理输入了有效的待修改基础月薪信息 |
| 后置条件 | | 系统更新基础月薪信息 |
| BasicSalaryBL.findBasicSalary | 语法 | | public double findBasicSalary(ProfessionType profession) |
| 前置条件 | | 经过验证的总经理有查询基础月薪的权利，且总经理输入了有效的对应职业编号 |
| 后置条件 | | 系统根据输入的职业类型查找基础月薪，并返回基础月薪信息 |
| 需要的服务（需接口） | | | |
| 服务名 | | 服务 | |
| ManageDataService. addBasicSalary(BasicSalaryPO citydistancepo) | | 在管理数据包的基础月薪数据文件中插入一个BasicSalaryPO对象 | |
| ManageDataService.deleteBasicSalary(BasicSalaryPO citydistancepo) | | 在管理数据包的基础月薪数据文件中删除一个BasicSalaryPO对象 | |
| ManageDataService.modifyBasicSalary(BasicSalaryPO citydistancepo) | | 在管理数据包的基础月薪数据文件中更新一个BasicSalaryPO对象 | |
| ManageDataService.findBasicSalary(ProfessionType profession) | | 根据输入的职业类型，返回对应的BasicSalaryPO对象 | |

CityDistanceBL的接口规范如表7-8所示。

**表7-8 CityDistanceBL模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| CityDistanceBL.addCityDistance | 语法 | | public int addCityDistance(CityDistanceVO citydistancevo) |
| 前置条件 | | 经过验证的总经理有添加新的城市距离的权利，且总经理输入了完整有效的新城市距离信息 |
| 后置条件 | | 系统保存新增的城市距离信息并更新城市距离列表 |
| CityDistanceBL.deleteCityDistance | 语法 | | public int deleteCityDistance(CityDistanceVO citydistancevo) |
| 前置条件 | | 经过验证的总经理有删除城市距离的权利，且总经理输入了有效的待删除城市距离信息 |
| 后置条件 | | 系统查找对应的城市距离信息，并删除该城市距离对象 |
| CityDistanceBL.modifyCityDistance | 语法 | | public int modifyCityDistance(CityDistanceVO citydistancevo) |
| 前置条件 | | 经过验证的总经理有修改城市距离的权利，且总经理输入了有效的待修改城市距离信息 |
| 后置条件 | | 系统更新城市距离信息 |
| CityDistanceBL.findCityDistance | 语法 | | public double findCityDistance(String cityA, String cityB) |
| 前置条件 | | 经过验证的总经理有查询城市距离的权利，且总经理输入了有效且存在的两个城市 |
| 后置条件 | | 系统根据输入的两个城市名查找城市距离，并返回城市距离信息 |
| 需要的服务（需接口） | | | |
| 服务名 | | 服务 | |
| ManageDataService.addCityDistance(CityDistancePO citydistancepo) | | 在管理数据包的城市距离数据文件中插入一个CityDistancePO对象 | |
| ManageDataService.deleteCityDistance(CityDistancePO citydistancepo) | | 在管理数据包的城市距离数据文件中删除一个CityDistancePO对象 | |
| ManageDataService.modifyCityDistance(CityDistancePO citydistancepo) | | 在管理数据包的城市距离数据文件中更新一个CityDistancePO对象 | |
| ManageDataService.findCityDistance(String cityA, String cityB) | | 根据输入的两个城市名，返回对应的CityDistancePO对象 | |

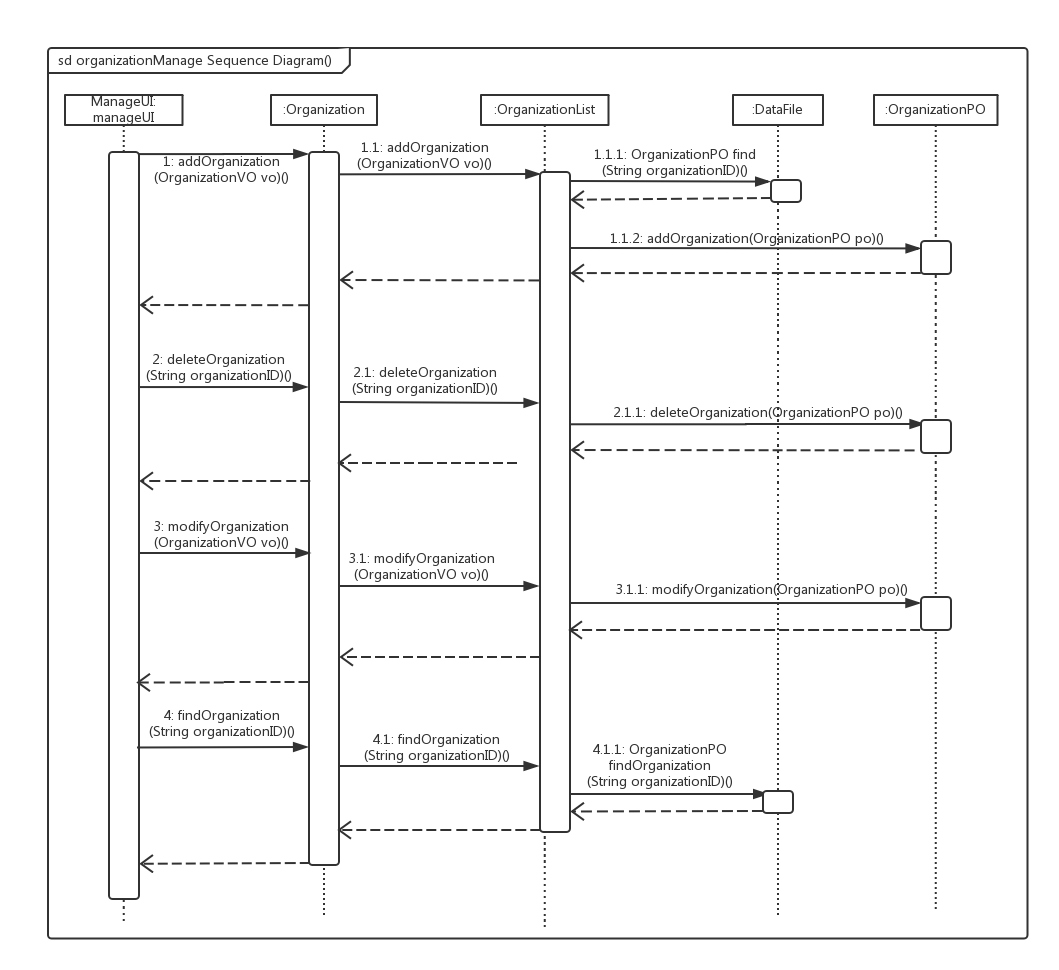
CostBL的接口规范如表7-9所示。

**表7-9 CostBL模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
|  | 后置条件 | | 系统根据输入的两个城市名查找城市距离，并返回城市距离信息 |
| CostBL.addCost | 语法 | | public int addCost(CostVO costvo) |
| 前置条件 | | 经过验证的总经理有添加新运费系数的权利，且总经理输入了完整有效的新运费系数信息 |
| 后置条件 | | 系统保存新增的运费系数信息并更新运费系数列表 |
| CostBL.deleteCost | 语法 | | public int deleteCost(CostVO costvo) |
| 前置条件 | | 经过验证的总经理有删除运费系数的权利，且总经理输入了有效的待删除运费系数信息 |
| 后置条件 | | 系统查找对应的运费系数信息，并删除该运费系数对象 |
| CostBL.modifyCost | 语法 | | public int modifyCost(CostVO costvo) |
| 前置条件 | | 经过验证的总经理有修改运费系数的权利，且总经理输入了有效的待修改运费系数信息 |
| 后置条件 | | 系统更新运费系数信息 |
| CostBL.findCost | 语法 | | public double findCost(ExpressType category) |
| 前置条件 | | 经过验证的总经理有查询运费系数的权利，且总经理输入了有效且存在的类别号 |
| 后置条件 | | 系统根据输入快递类型查找运费系数，并返回运费系数信息 |
| 需要的服务（需接口） | | | |
| 服务名 | | 服务 | |
| ManageDataService.addCost(CostPO costpo) | | 在管理数据包的运费系数数据文件中插入一个CostPO对象 | |
| ManageDataService.deleteCost(CostPO costpo) | | 在管理数据包的运费系数数据文件中删除一个CostPO对象 | |
| ManageDataService.modifyCost(CostPO costpo) | | 在管理数据包的运费系数数据文件中更新一个CostPO对象 | |
| ManageDataService. findCost(ExpressType category) | | 根据输入的快递类型，返回对应的CostPO对象 | |

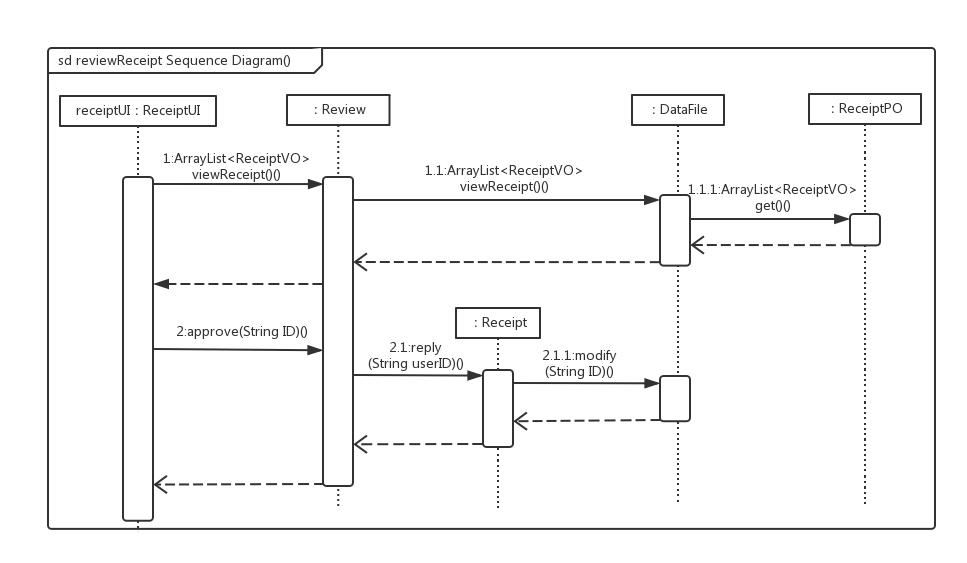
#### 4.7.2.4 业务逻辑层的动态模型

图7-2表明了快递物流系统中，当总经理开始机构管理任务后，机构管理业务逻辑处理的相关对象的协作。



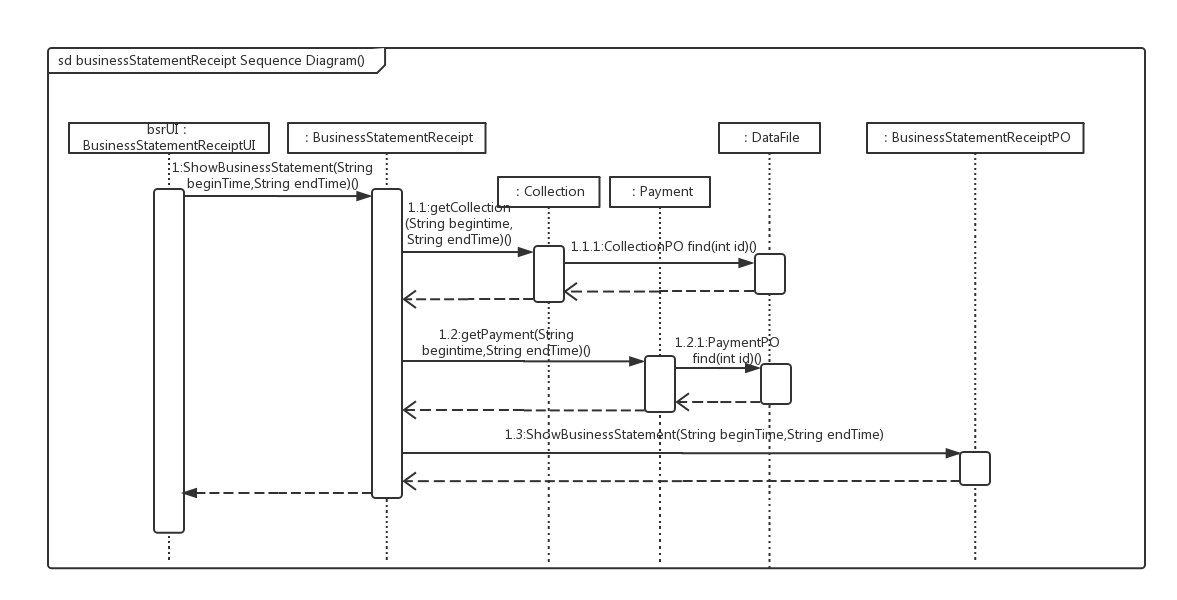
**图7-2 机构管理的顺序图**

图7-3表明了快递物流系统中，当总经理开始单据审批任务后，单据审批业务逻辑处理的相关对象的协作。



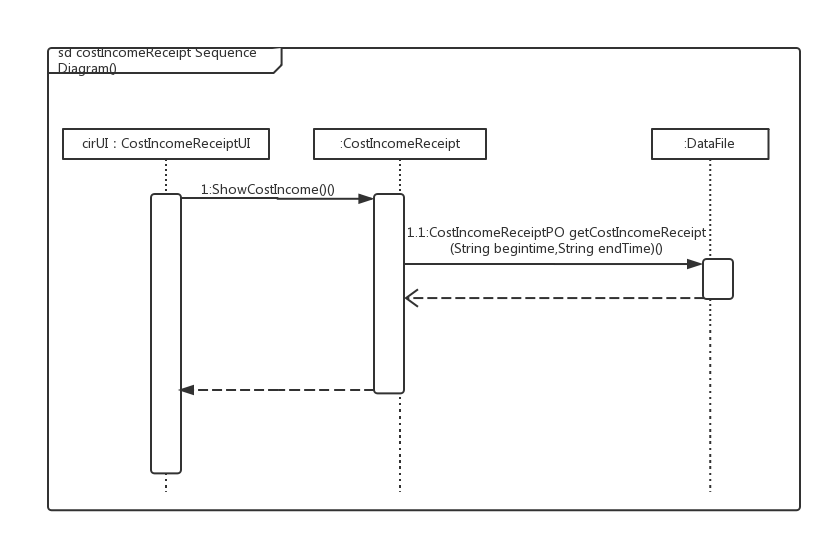
**图7-3 单据审批的顺序图**

图7-4表明了快递物流系统中，当总经理开始查看经营情况表任务后，经营情况表业务逻辑处理的相关对象的协作。



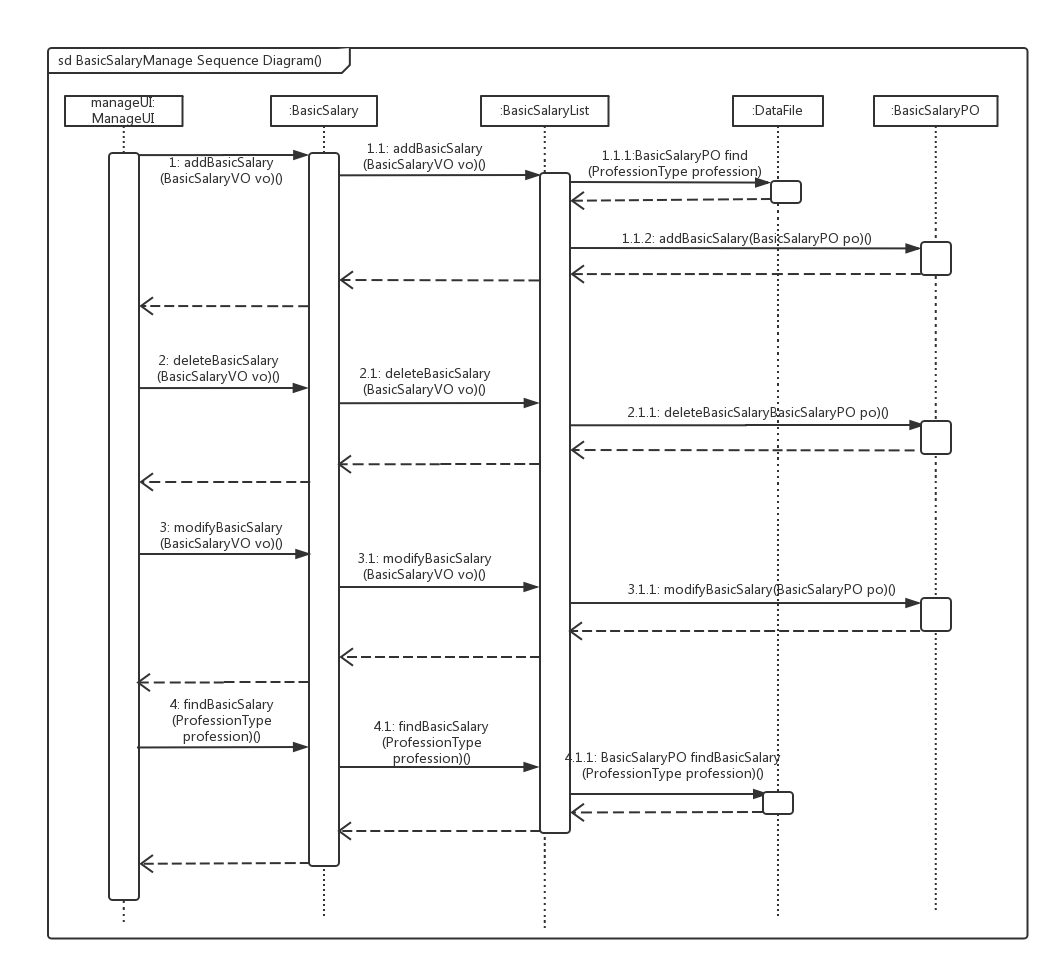
**图7-4 查看经营情况表的顺序图**

图7-5表明了快递物流系统中，当总经理开始查看成本收益表任务后，成本收益表业务逻辑处理的相关对象的协作。



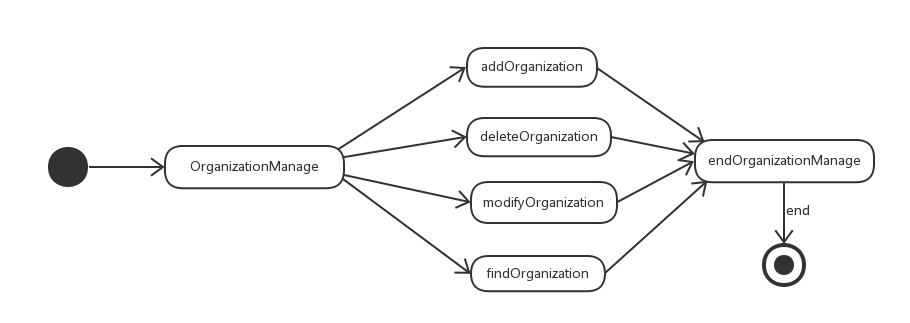
**图7-5 查看成本收益表的顺序图**

图7-6表明了快递物流系统中，当总经理开始查看基础数据设置任务后，基础数据设置业务逻辑处理的相关对象的协作（以基础月薪管理为例）。

****

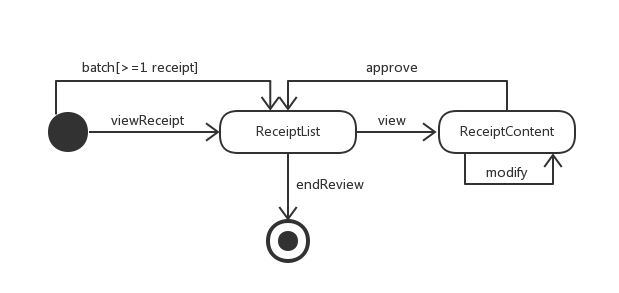
**图7-6 基础月薪管理的顺序图**

如图7-7所示的状态图描述了OrganizationBL对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。随着OrganizationManage方法被UI调用，OrganizationBL进入OrganizationManage状态；之后可以选择添加机构进入addOrganization状态，也可以选择删除机构进入deleteOrganization状态，同理modifyOrganization状态和findOrganization状态。最后结束机构管理进入endOrganizationManage状态。



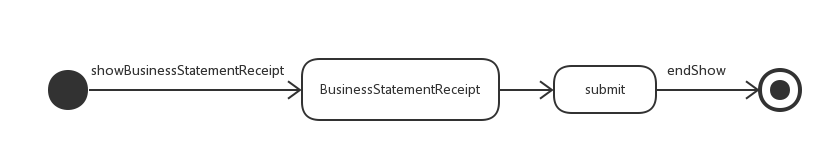
**图7-7 机构管理的状态图**

如图7-8所示的状态图描述了ReceiptBL对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。随着viewReceipt方法被UI调用，ReceiptBL进入ReceiptList状态；之后可以选择详细查看进入ReceiptContent状态，也可以选择批量审批进入ReceiptList状态。最后结束单据审批进入endReview状态。



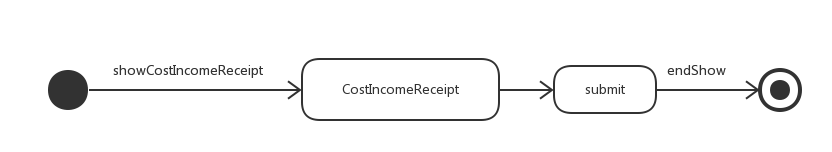
**图7-8 单据审批的状态图**

如图7-9所示的状态图描述了BusinessStatementReceiptBL对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。随着showBusinessStatementReceipt方法被UI调用，BusinessStatementReceiptBL进入BusinessStatementReceipt状态；最后通过submit方法确定查看结束，进入endShow状态。



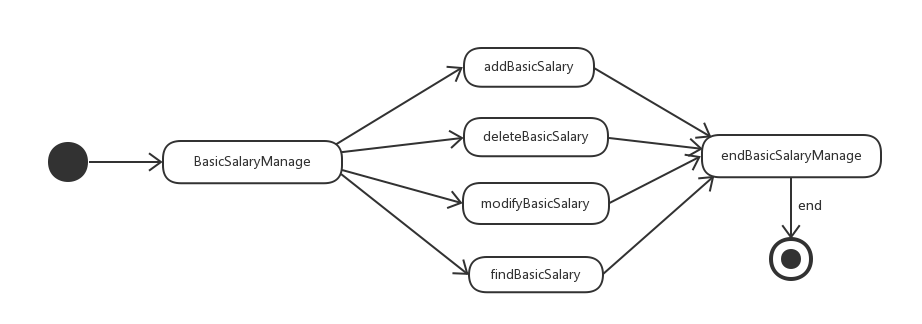
**图7-9 查看经营情况表的状态图**

如图7-10所示的状态图描述了CostIncomeReceiptBL对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。随着showCostIncomeReceipt方法被UI调用，CostIncomeReceiptBL进入CostIncomeReceipt状态；最后通过submit方法确定查看结束，进入endShow状态。



**图7-10 查看成本收益表的状态图**

如图7-7所示的状态图描述了BasicSalaryBL对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。随着BasicSalaryManage方法被UI调用，BasicSalaryBL进入BasicSalaryManage状态；之后可以选择添加基础月薪进入addBasicSalary状态，也可以选择删除基础月薪进入deleteBasicSalary状态，同理modifyBasicSalary状态和findBasicSalary状态。最后结束基础月薪管理进入endBasicSalaryManage状态。

****

**图7-11 基础月薪管理的状态图**