**需求分析规约**



**2151617 郑埴**

**2154314 郑楷**

1. **引言**

1.1. 背景

随着经济全球化和信息技术的快速发展，金融行业面临着更加复杂和多样化的挑战。人们对于个人理财需求的增加、金融产品的多样化以及金融监管政策的不断更新，都要求金融机构提供更加专业、高效和个性化的理财服务。

金融市场竞争日益激烈，金融机构需要不断提高市场占有率和客户满意度，通过推出新的金融产品、提供更好的理财服务来吸引和保留客户。为了实现这些目标，金融机构需要建立起一套高效的理财销售管理系统，以提高销售团队的工作效率、降低成本、提升客户体验。传统的理财销售过程通常繁琐而耗时，包括客户信息采集、风险评估、产品推荐、投资组合管理等环节。通过开发理财销售管理系统，可以对整个业务流程进行优化和自动化，提高工作效率和准确性，减少人为错误和重复劳动，节省时间和资源。同事可以实现对销售业务的数字化管理和监控，将销售数据、客户信息、产品信息等进行集中存储和分析，提供准确的数据支持和决策依据。

1.2. 参考资料

1. Vue.js 官方文档

2. Spring Boot 官方文档

3. MySQL 官方文档

4. GBT18726-2011《现代设计工程集成技术软件接口规范》

5. 网上基金销售信息系统技术指引

1.3. 假定和约束

假定操作员为基金销售业务柜员，熟悉基本的基金业务知识。

此时对于本次系统开发而言，我们开发环境假定如下 ：

开发需要Linux虚拟机，并配备JDK8.0，Mysql 8.0，Zookpeer3.4+，Redis3.X，Git以及实现SSH链接等必要开发工具与此同时，还需要配置好其余本地软件开发环境，例如Java环境、Git、JDK8 以及IDE和SSH客户端等有关程序。

对于前端开发而言，需要配备Edge浏览器以及必要的IDE，并安装好VUE,NODE.JS等。

本系统的运行环境为网页端，采用Windows10系统，并对需要调用的第三行系统提供了相关接口。

本系统的开发期限为3个月。

1.4. 用户的特点

本系统面向的用户为“操作员”（也即“业务员”，在本文档中两者等同），具有一定的金融领域知识，熟悉基本的基金业务知识。

结合2.1节中的适用场景，业务员通常需要频繁进行基金的买卖、清算等操作。因此，系统界面应该简洁明了，操作流程应该直观易懂，以提高业务员的工作效率。

业务员可能需要同时处理多个客户的基金，因此系统应该支持多账户管理和并行操作，确保业务员能够高效地处理大量信息。

**2. 功能需求**

2.1. 系统范围

本系统是一个面向金融机构操作员（业务员）的基金交易和管理系统。其主要目标是简化和优化基金购买、卖出、清算等操作，提高业务员的工作效率，同时支持多账户管理和并行操作。通过软件系统的使用，将传统基金交易过程中的复杂性和庞大性进行简化，从而提高整体业务流程的效率和精确性。

以下将用一个例子来说明本项目的适用情况：

对于一家金融机构，其维护了多项基金（类似于股票），每一个散户可以购买该机构名下的基金。购买基金的基本流程为：1. 散户到达基金公司前台，向前台的“登记员”说明要购买的基金以及相应份额。2. 登记员将相应信息发送给“业务员”。3. 业务员根据收到的信息在系统上完成购买的操作。

传统的基金购买的实现和记录是复杂且庞大的，使用软件系统能够大大简化这一过程。这也便是本系统的设计目的，把“操作员”完成的操作整理为需求并实现。

系统的主要作用范围包括但不限于：

基金交易操作：包括基金的购买、卖出、调仓等交易操作，以及相关的信息记录和确认。

多账户管理：系统支持操作员同时处理多个客户的基金账户，确保操作员能够高效地管理和执行各项任务。

界面设计：系统界面应简洁明了，操作流程直观易懂，以提高业务员的工作效率。界面设计应考虑用户体验，使操作员能够迅速完成所需操作。

本系统是金融机构整体业务系统的一个组成部分。与其他系统的关系主要包括：

前台登记系统：本系统与前台登记系统相连接，接收从登记员处发送的基金购买信息，完成后续交易操作。

整体业务系统：本系统作为整体业务系统的一部分，与其他模块协同工作，确保基金交易流程的顺畅和准确。

2.2. 系统体系结构

项目的总体架构为典型的三层架构。

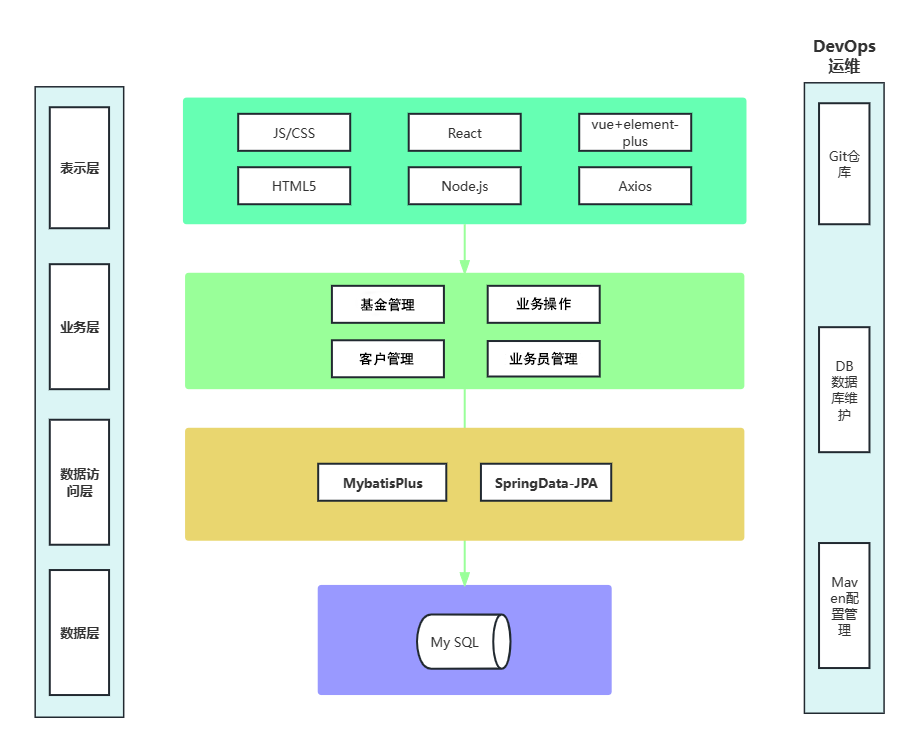
其中，表示层是用户直接与之交互的界面，它包括用户界面元素、UI控件、用户输入、显示和呈现数据等。表示层主要用的技术为：vue+element-plus，Axios等。

业务层包含应用程序的核心逻辑和规则，主要用于完成客户管理、业务操作、基金管理和业务员管理功能。

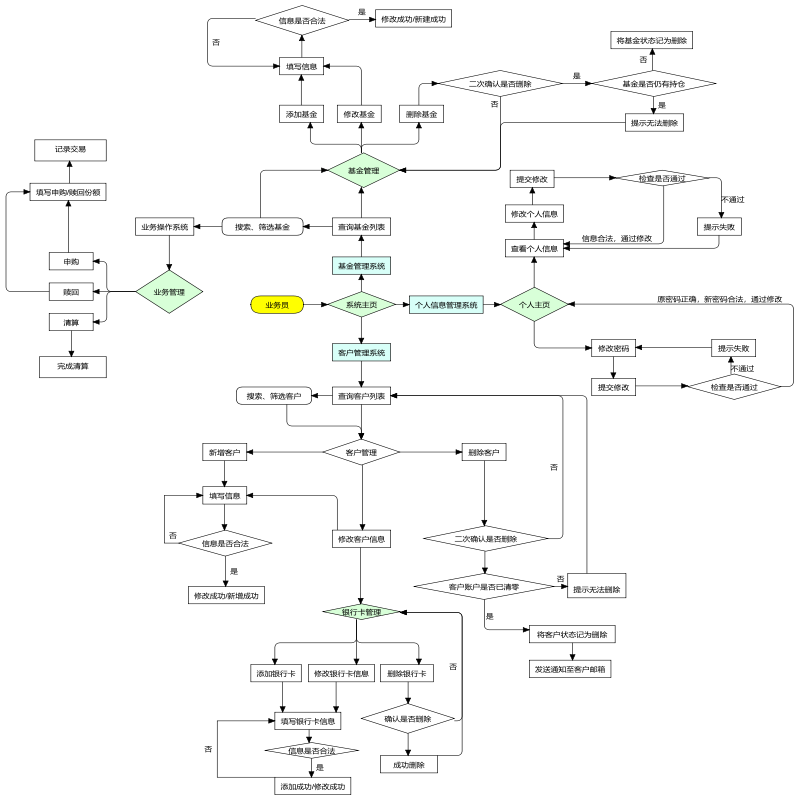
数据访问层负责与数据存储系统进行交互，执行数据的检索、存储、更新和删除操作，主要使用的技术是MybatisPlus、SpringData-JPA，更加方便的和数据库进行交互。

数据库的设计上，大部分的数据存储在Mysql数据库中，在本地进行数据规模较小的数据存储；考虑到图片上传的速度和内存问题，将图片单独存储在阿里云的服务器上，从而更加方便的获取数据，Mysql数据库中仅需存储对应图片的url即可访问。

详情如下图所示：



2.3. 系统总体流程



2.4. 需求分析

2.4.1. 业务员管理子系统

2.4.1.1. 子系统总述

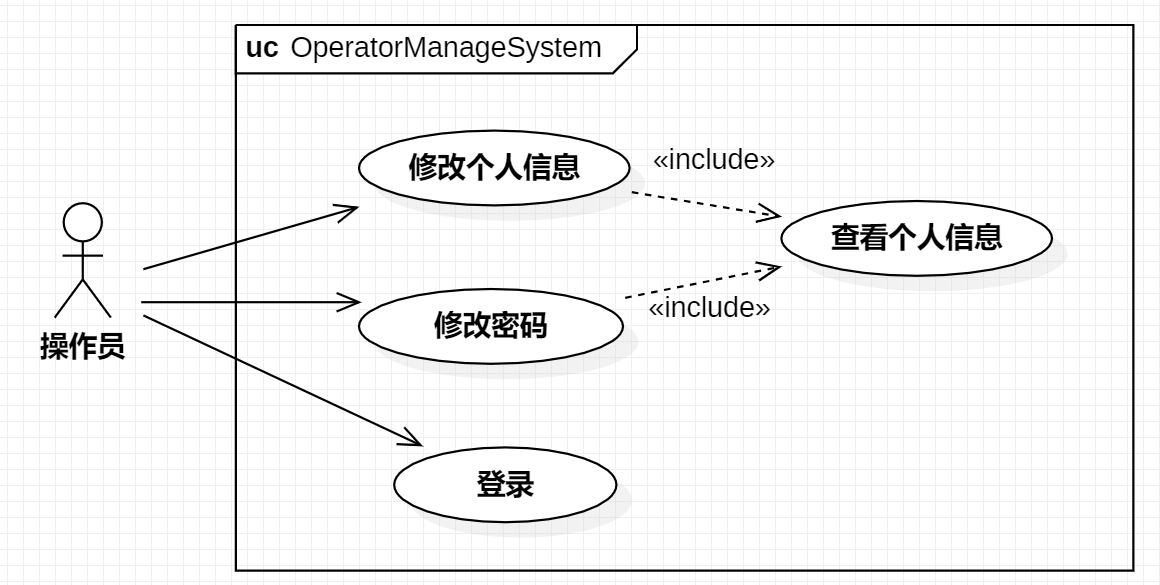
在业务员管理系统中，主要应当完成以下功能：

·业务员登录

·业务员查看个人信息

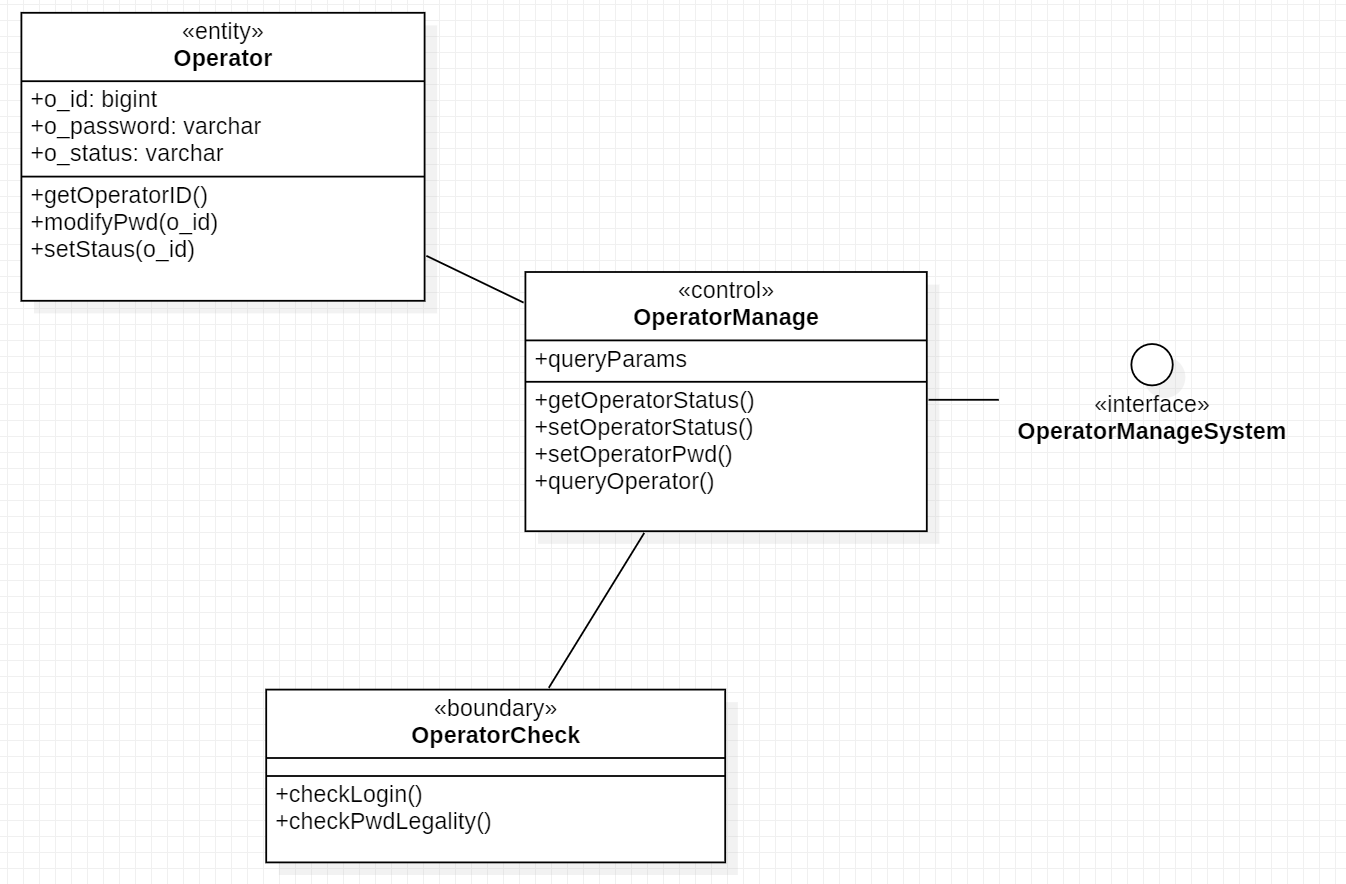
·业务员修改个人信息

·业务员修改密码

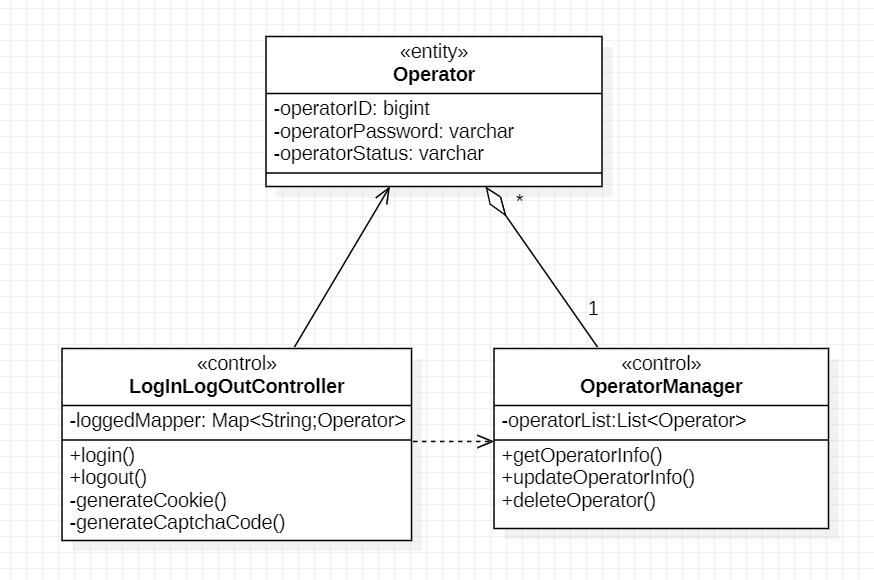


2.4.1.2. 整体数据建模

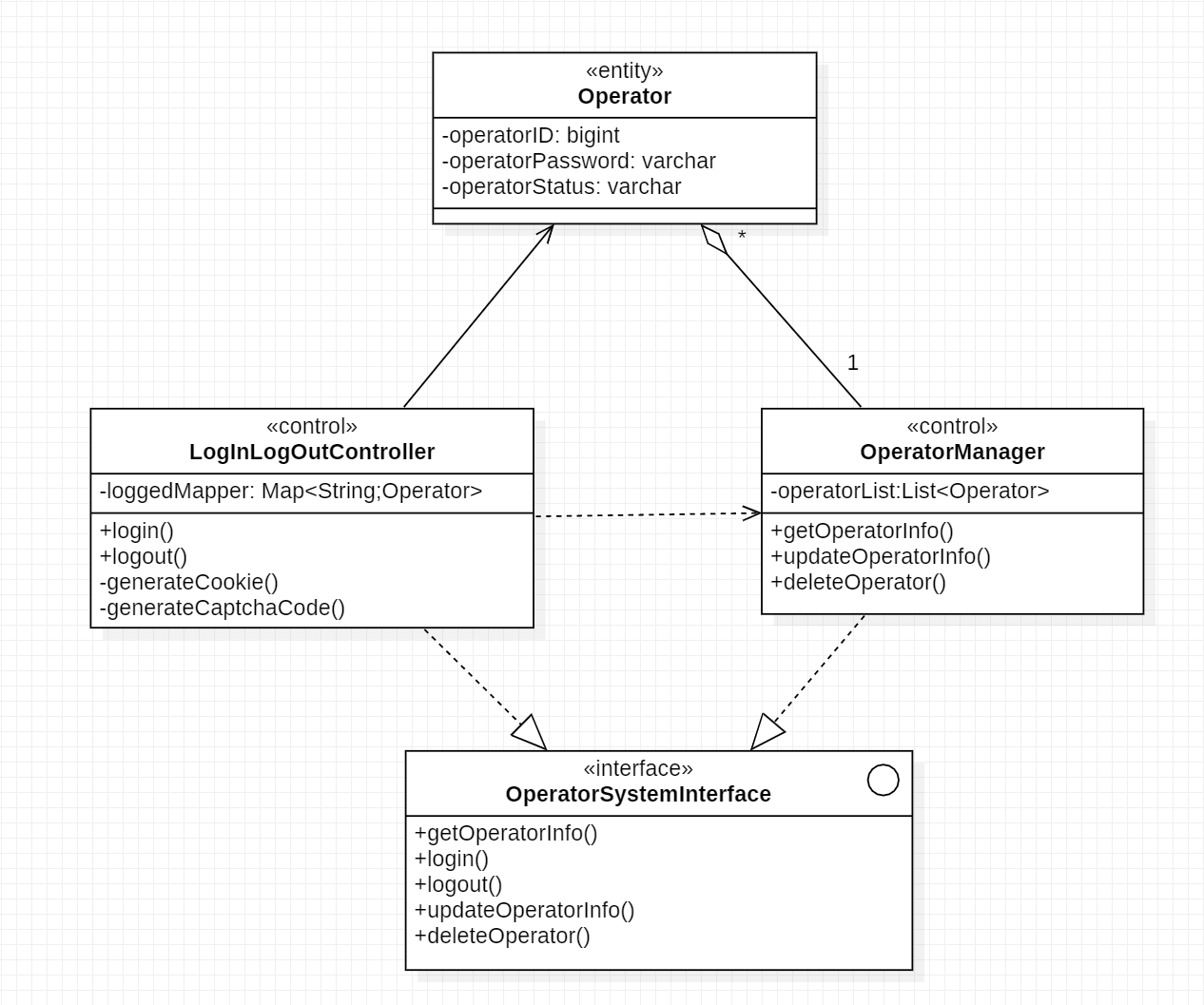
Iteration1:



Iteration2:



Iteration3:

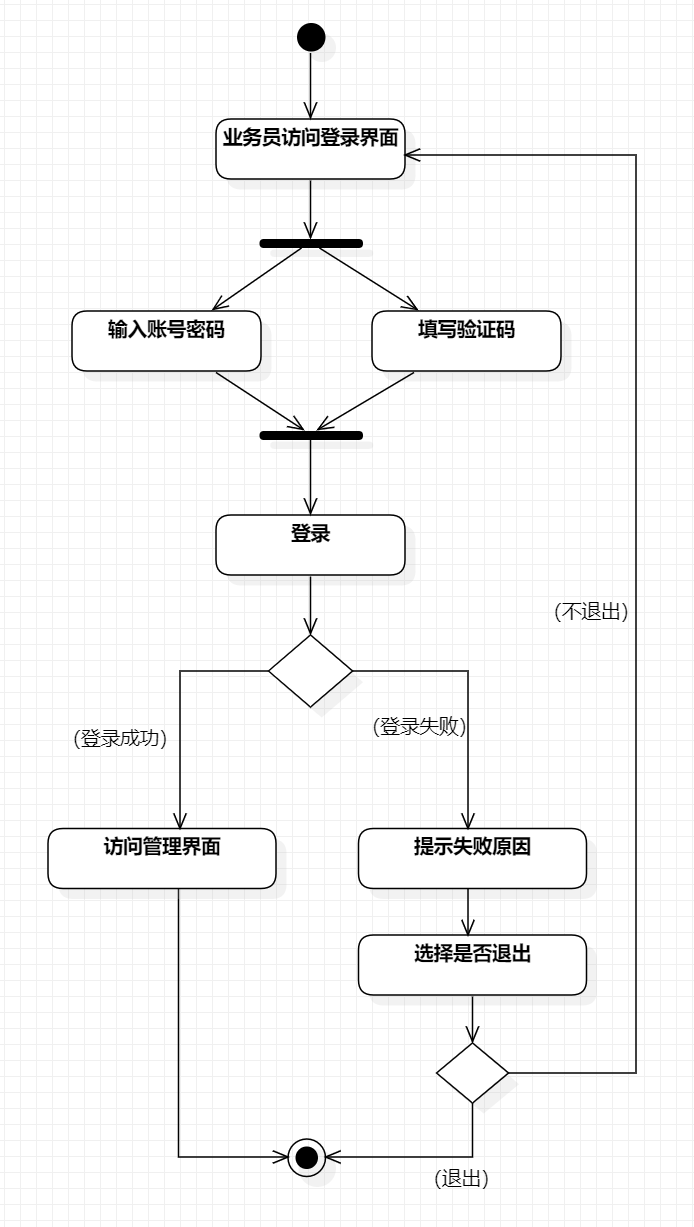


2.4.1.3. 登录用例

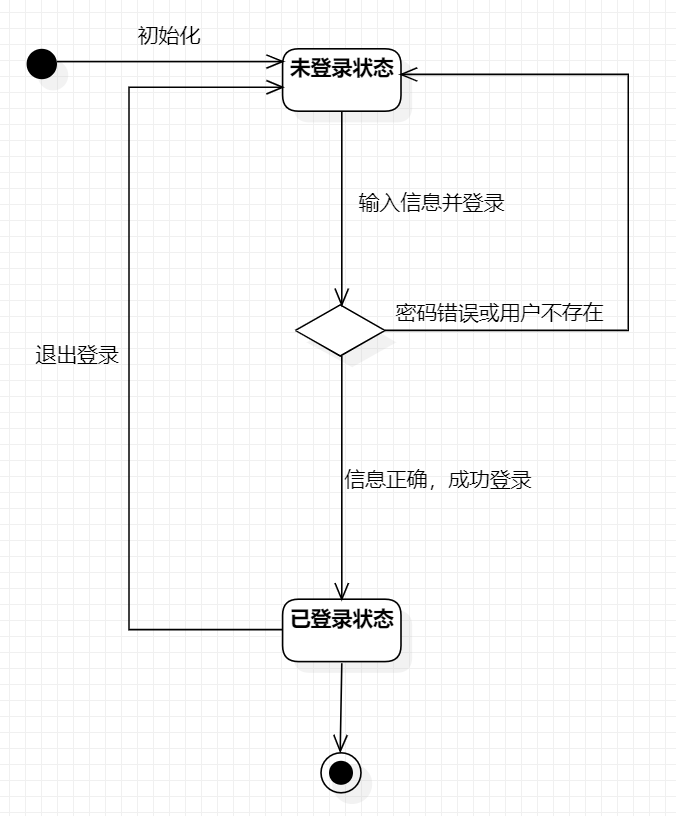
2.4.1.3.1. 用例模型

|  |  |
| --- | --- |
| **描述对象** | **业务员登录** |
| 编号 | UCA01 |
| 说明 | 业务员登录至管理界面 |
| 参与者 | 业务员 |
| 频度 | 1秒1000次 |
| 状态 | 通过审查 |
| 前置条件 | 业务员打开系统网页，进入登录页面 |
| 基本事件流 | 1. 业务员填写账号密码和验证码 2. 业务员登录，系统返回登录状态：   2.1登录成功，进入管理界面  2.2登录失败，显示失败原因，重新回到登录界面 |
| 拓展事件流 | 1. 业务员退出了系统，停止流程 2. 业务员中途点击取消登录，停止流程 3. 业务员点击了验证码图片，刷新了验证码图片 |
| 后置条件 | 无 |

2.4.1.3.2. 功能建模



2.4.1.3.3. 行为建模

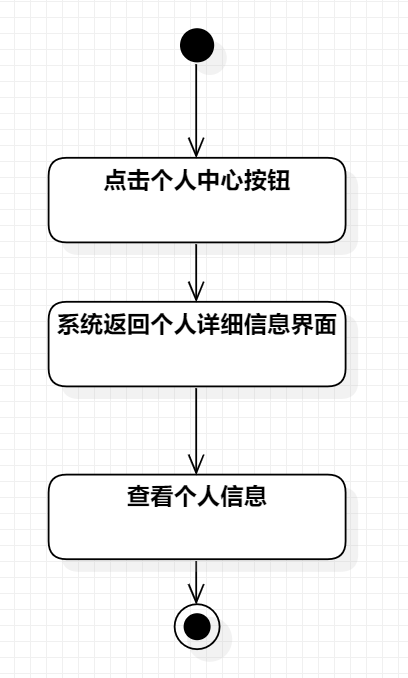


2.4.1.4. 查看个人信息用例

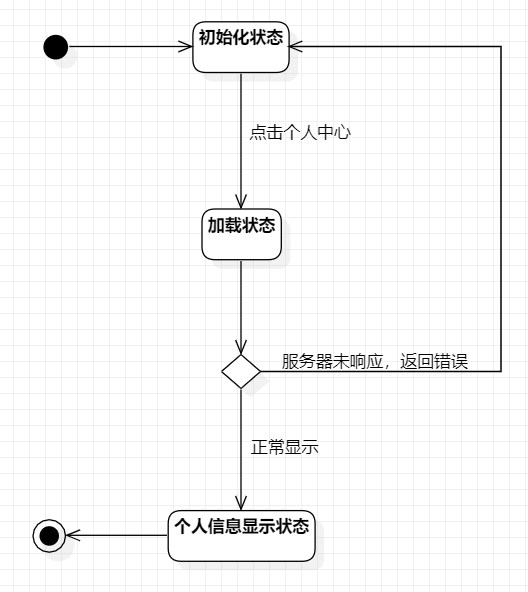
2.4.1.4.1. 用例模型

|  |  |
| --- | --- |
| **描述对象** | **查看个人信息** |
| 编号 | UCA02 |
| 说明 | 业务员查看自己的个人信息 |
| 参与者 | 业务员 |
| 频度 | 1秒500次 |
| 状态 | 通过审查 |
| 前置条件 | 1. 业务员完成登录操作，进入具体操作界面。 |
| 基本事件流 | 1. 业务员进入操作页面 2. 业务员点击查询个人信息 3. 系统返回业务员个人详细信息界面供业务员核对 |
| 拓展事件流 | 1. 业务员退出了系统，停止流程 2. 业务员想要修改密码，进入修改密码用例 3. 业务员想要修改个人信息，进入修改个人信息用例 |
| 后置条件 | 无 |

2.4.1.4.2. 功能建模



2.4.1.4.3. 行为建模

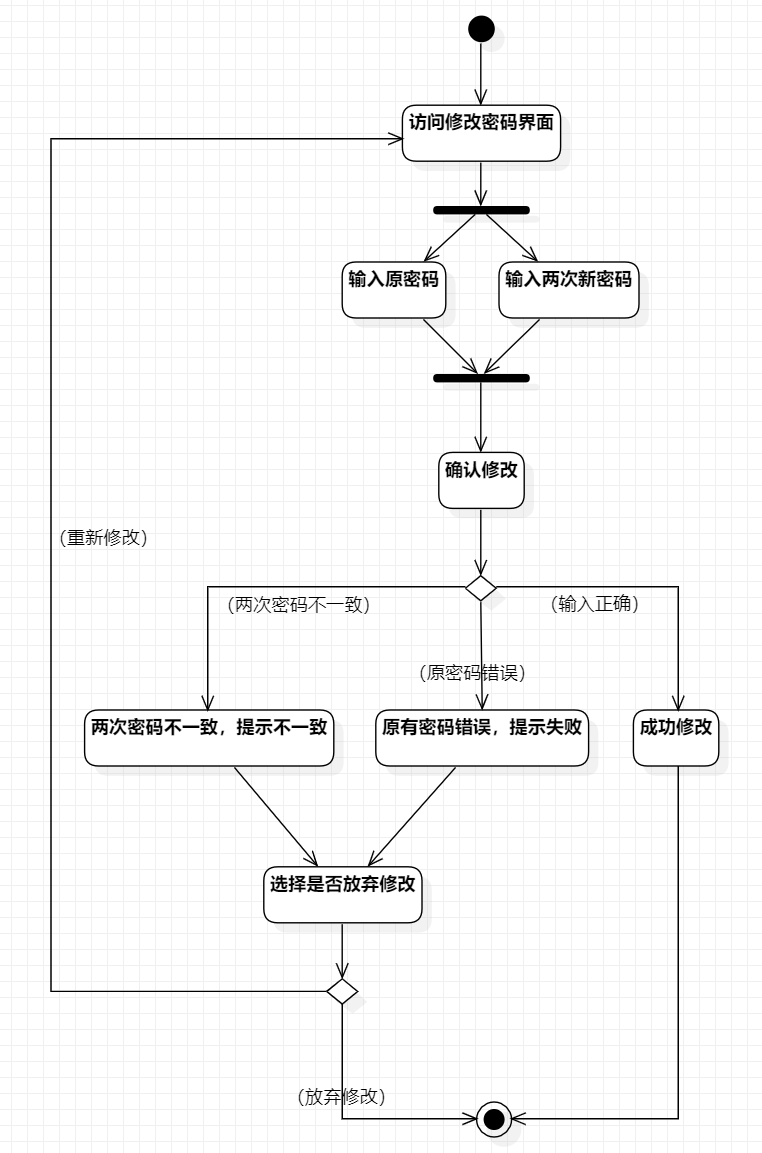


2.4.1.5. 修改密码用例

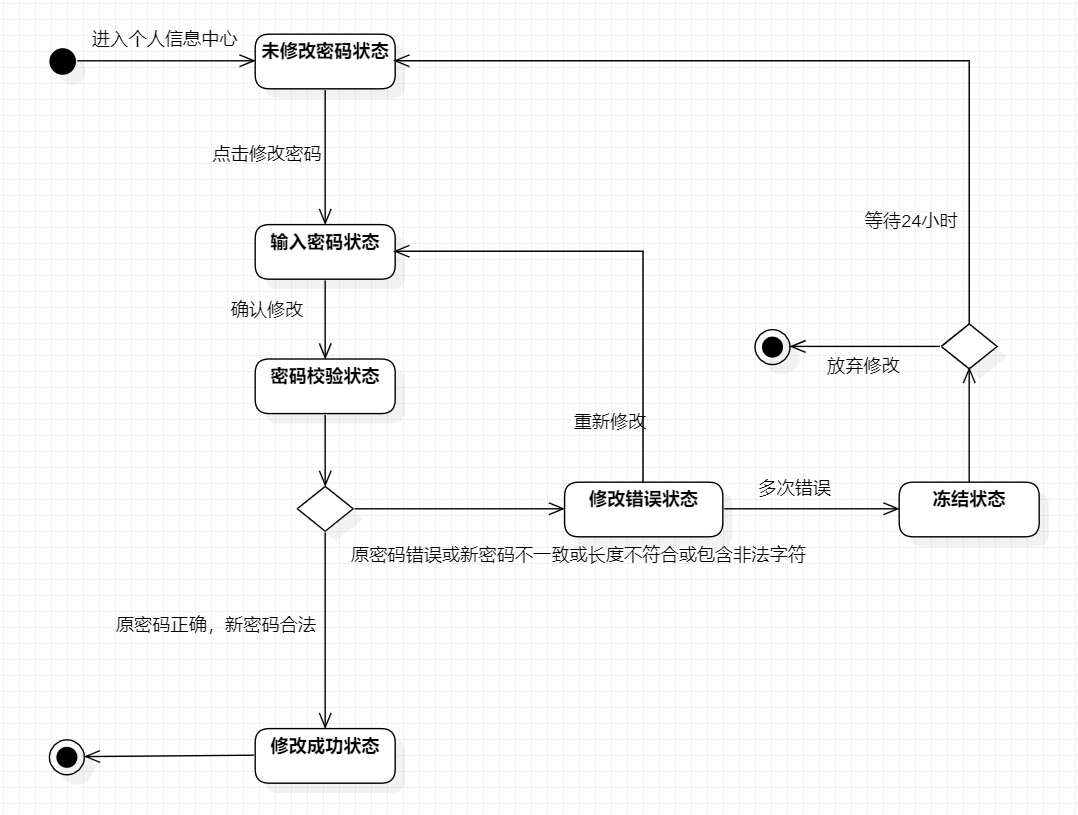
2.4.1.5.1. 用例模型

|  |  |
| --- | --- |
| **描述对象** | **修改密码** |
| 编号 | UCA03 |
| 说明 | 业务员修改自己的账户密码 |
| 参与者 | 业务员 |
| 频度 | 1秒500次 |
| 状态 |  |
| 前置条件 | 1. 业务员完成登录操作，进入具体操作界面。 2. 业务员进入查看个人信息页面 |
| 基本事件流 | 1. 业务员点击修改密码 2. 业务员输入原有密码和新密码，同时重复输入一遍新密码，系统判断原有密码是否正确和两次新密码是否相同：   2.1原有密码错误，提示原密码错误，修改失败，不发生操作  2.2两次输入密码不同，提示新密码不同，修改失败，不发生操作  2.3成功修改，提示修改成功   1. 修改成功后，系统自动刷新页面，回到登录界面 |
| 拓展事件流 | 1. 业务员退出了系统，停止流程 2. 多次修改失败后，修改密码功能将被冻结24小时 |
| 后置条件 | 无 |

2.4.1.5.2. 功能建模



2.4.1.5.3. 行为建模

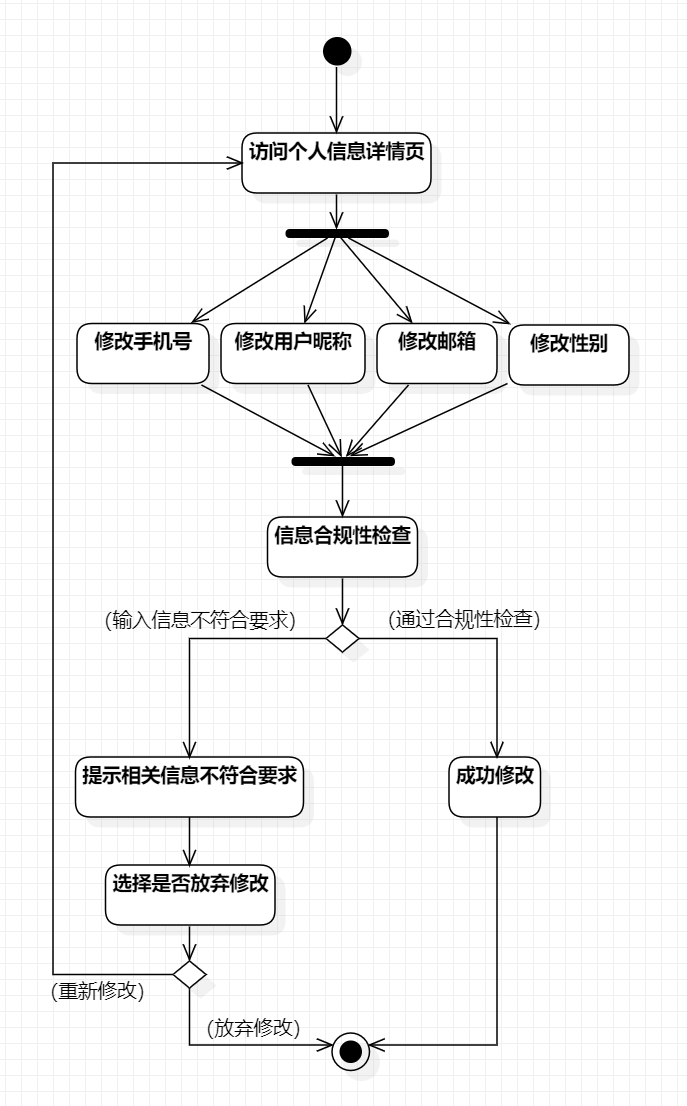


2.4.1.6. 修改个人信息用例

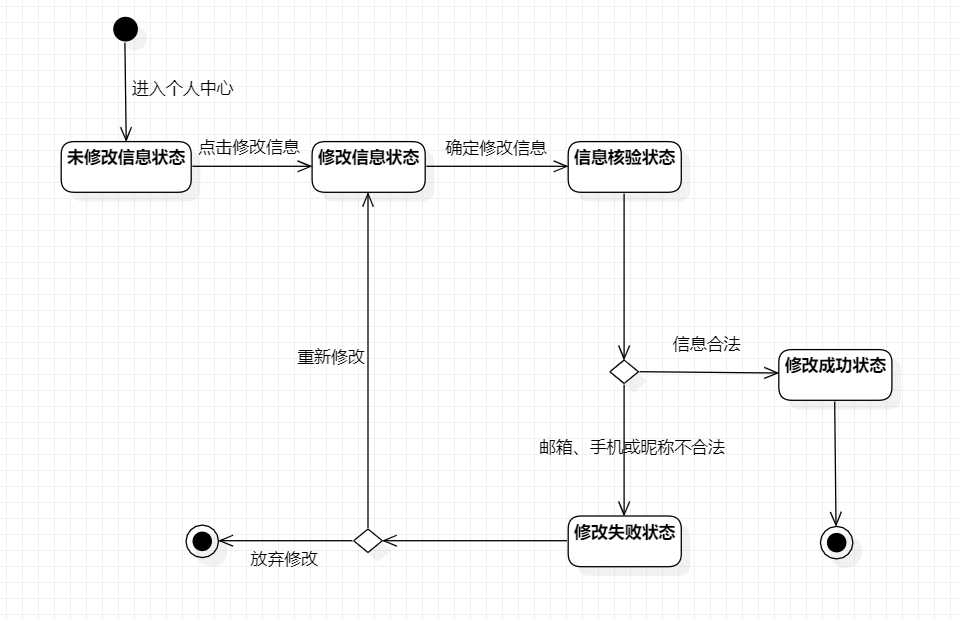
2.4.1.6.1. 用例模型

|  |  |
| --- | --- |
| **描述对象** | **查看个人信息** |
| 编号 | UCA04 |
| 说明 | 业务员修改自己的个人信息 |
| 参与者 | 业务员 |
| 频度 | 1秒500次 |
| 状态 |  |
| 前置条件 | 1. 业务员完成登录操作，进入具体操作界面。 2. 业务员进入查看个人信息页面 |
| 基本事件流 | 1. 业务员进入修改信息页面 2. 业务员在各个想要修改的信息处填写修改后的信息 3. 业务员点击保存按钮 4. 系统核查信息表单   4.1填写了不合理的手机号（空手机号），系统提示  4.2填写了不合理的邮箱，系统提示   1. 系统返回结果提示用户并显示更新后的信息界面 |
| 拓展事件流 | 1. 业务员点击了关闭按钮，退出了个人中心 2. 业务员刷新页面，则该记录丢失并需要重新填写 |
| 后置条件 | 无 |

2.4.1.6.2. 功能建模



2.4.1.6.3. 行为建模



2.4.2. 客户信息管理子系统

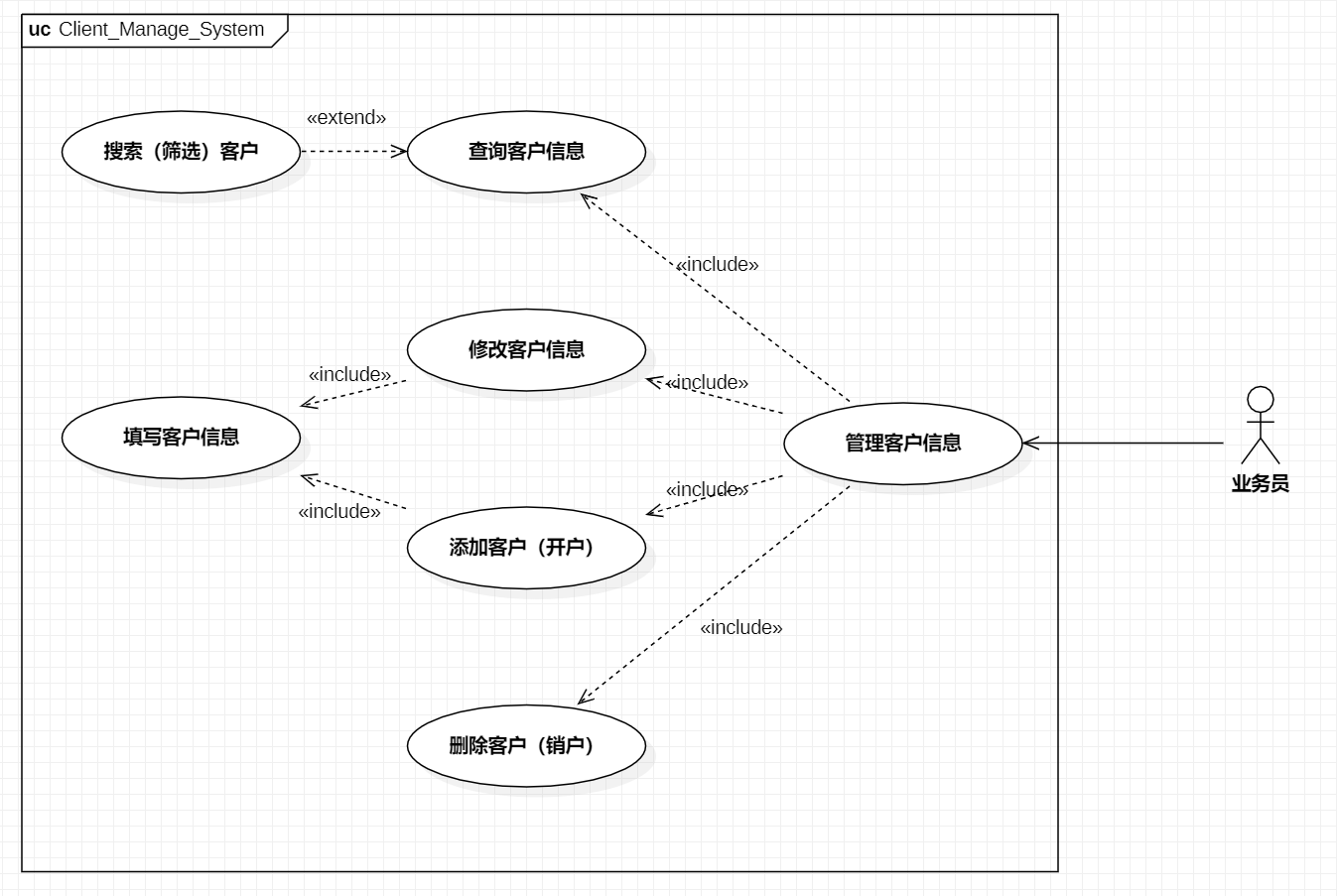
2.4.2.1. 子系统总述

在用户信息管理系统当中，主要应当完成以下功能：

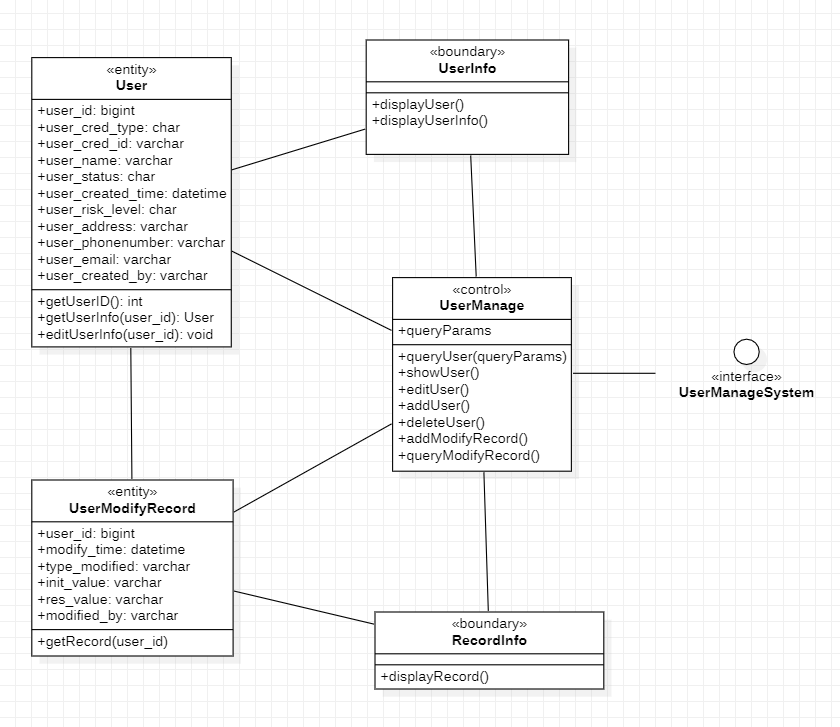
·操作员为客户开户（注册账户）

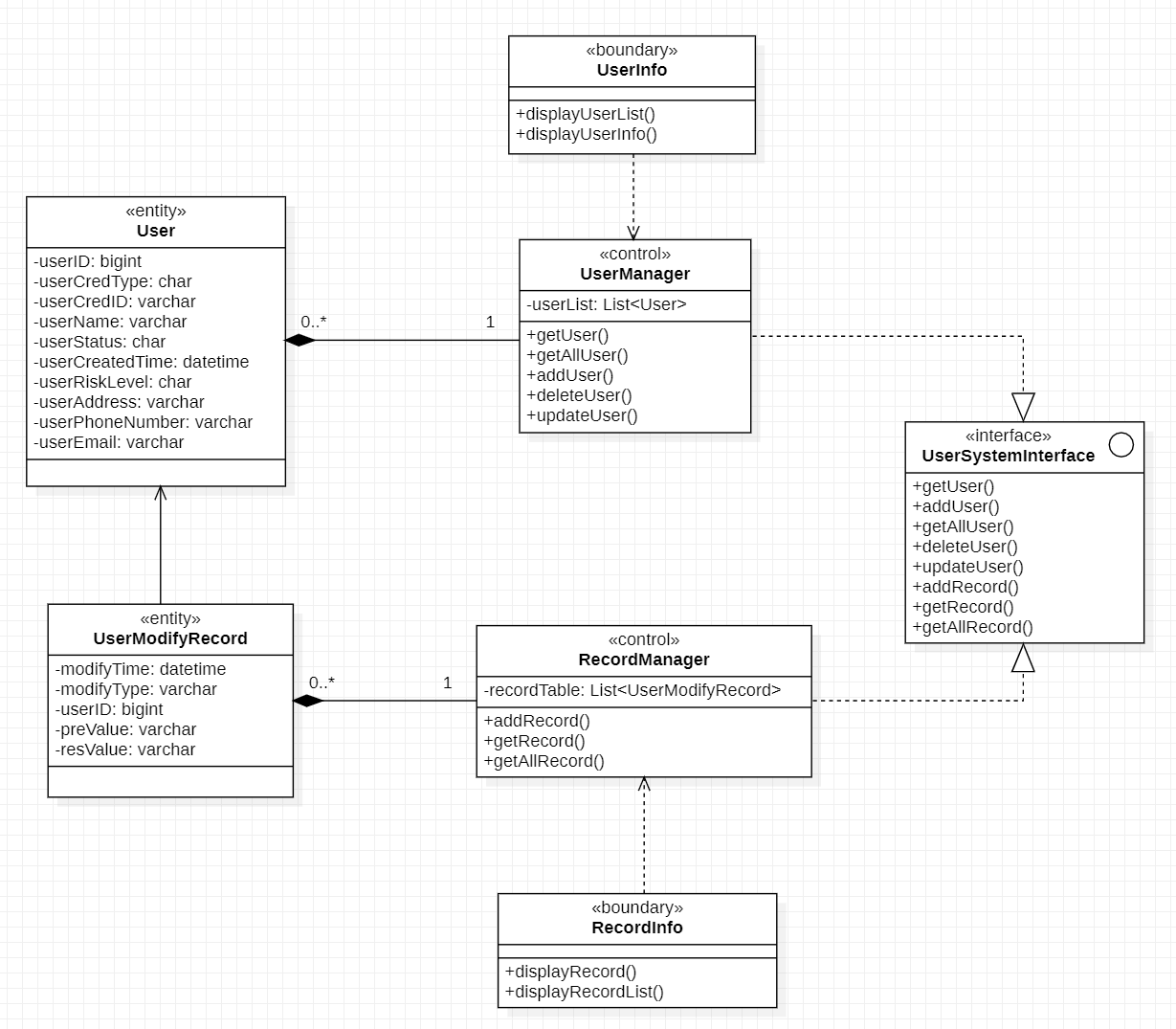
·操作员为客户销户（销毁账户）

·操作员修改客户信息



2.4.2.2. 整体数据建模

Iteration1:

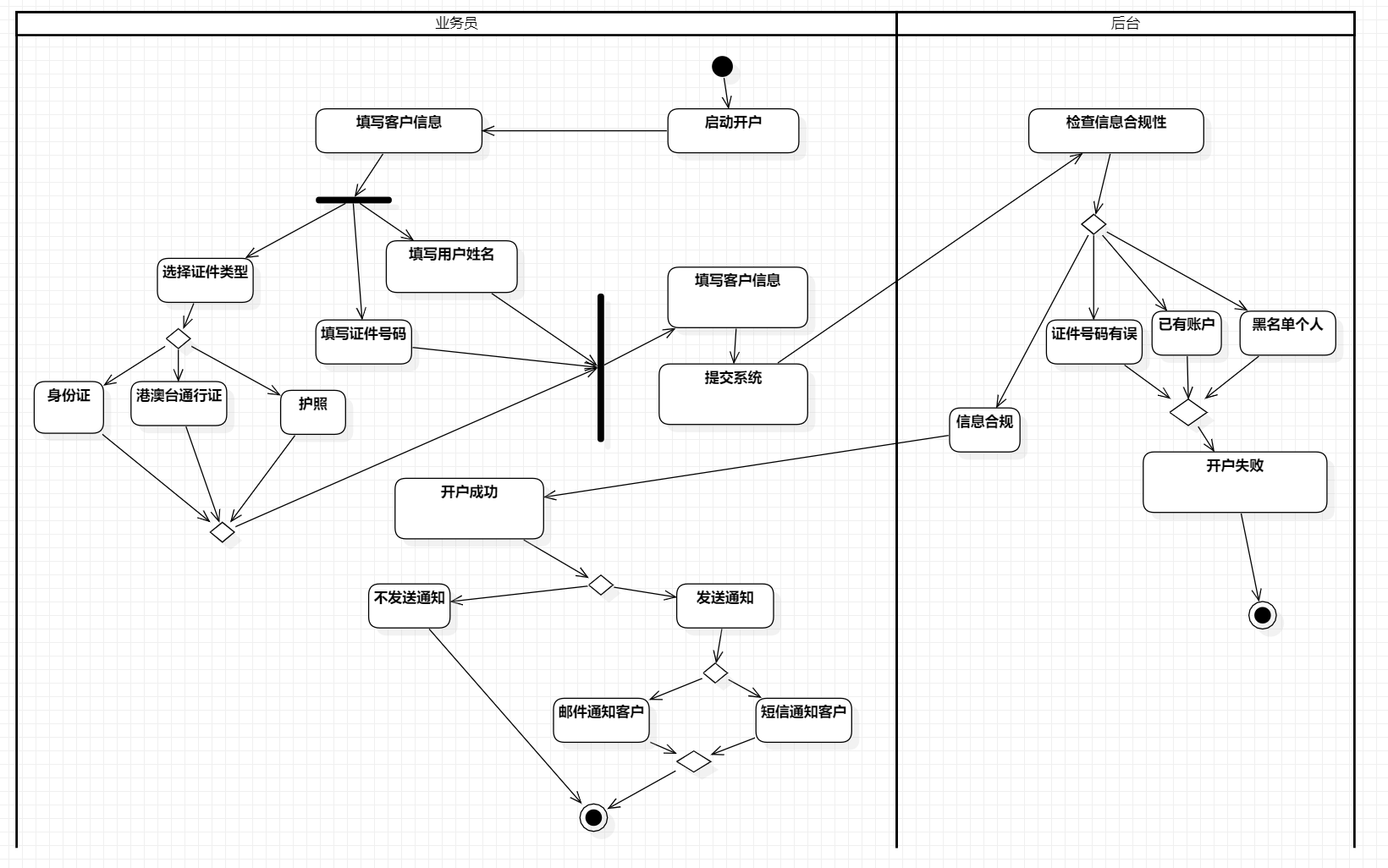
Iteration2:

2.4.2.3. 添加客户（开户）用例

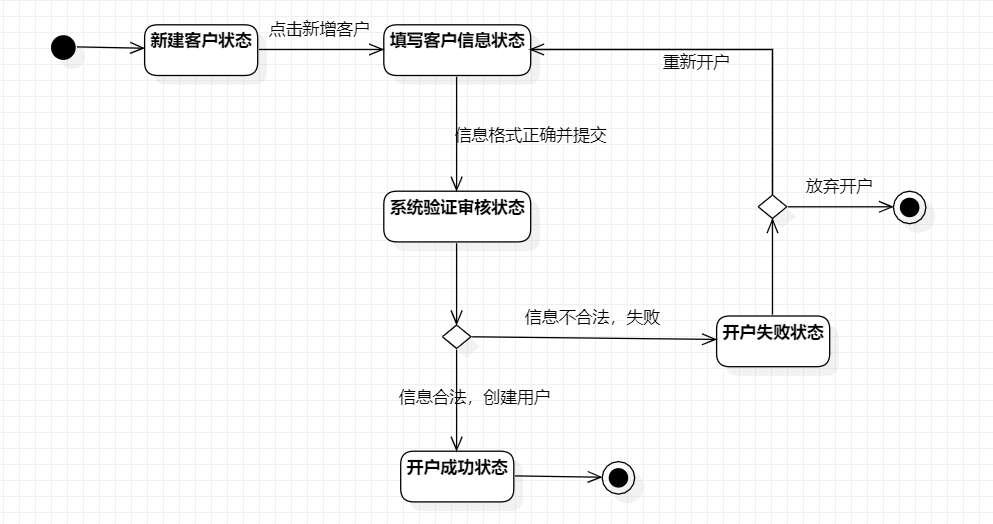
2.4.2.3.1. 用例模型

|  |  |
| --- | --- |
| **描述对象** | **开户** |
| 编号 | UCB01 |
| 说明 | 业务员为用户开户（注册账户） |
| 参与者 | 业务员 |
| 频度 | 1秒300次 |
| 状态 | 通过审核 |
| 前置条件 | 1. 业务员完成登录操作，进入具体操作界面。 2. 数据库中无此用户对应的证件号码/信用代码。 |
| 基本事件流 | 1. 业务员点击“用户管理”，进入管理页面 2. 业务员点击“新增用户”，进入注册页面 3. 业务员输入用户相关信息，如选择证件类型，绑定银行卡，风险等级等 4. 系统判断四要素是否符合前置条件，根据业务规则判断是否符合开户条件，同时返回结果：   4.1开户成功，新增账户完成  4.2开户失败，返回开户界面并提示失败原因   1. 开户成功后，系统记录操作，同时发送通知至客户邮件 |
| 拓展事件流 | 1. 业务员退出了系统，停止流程 2. 业务员中途点击取消开户，停止流程 |
| 后置条件 | 无 |

2.4.2.3.2.功能建模



2.4.2.3.3. 行为建模

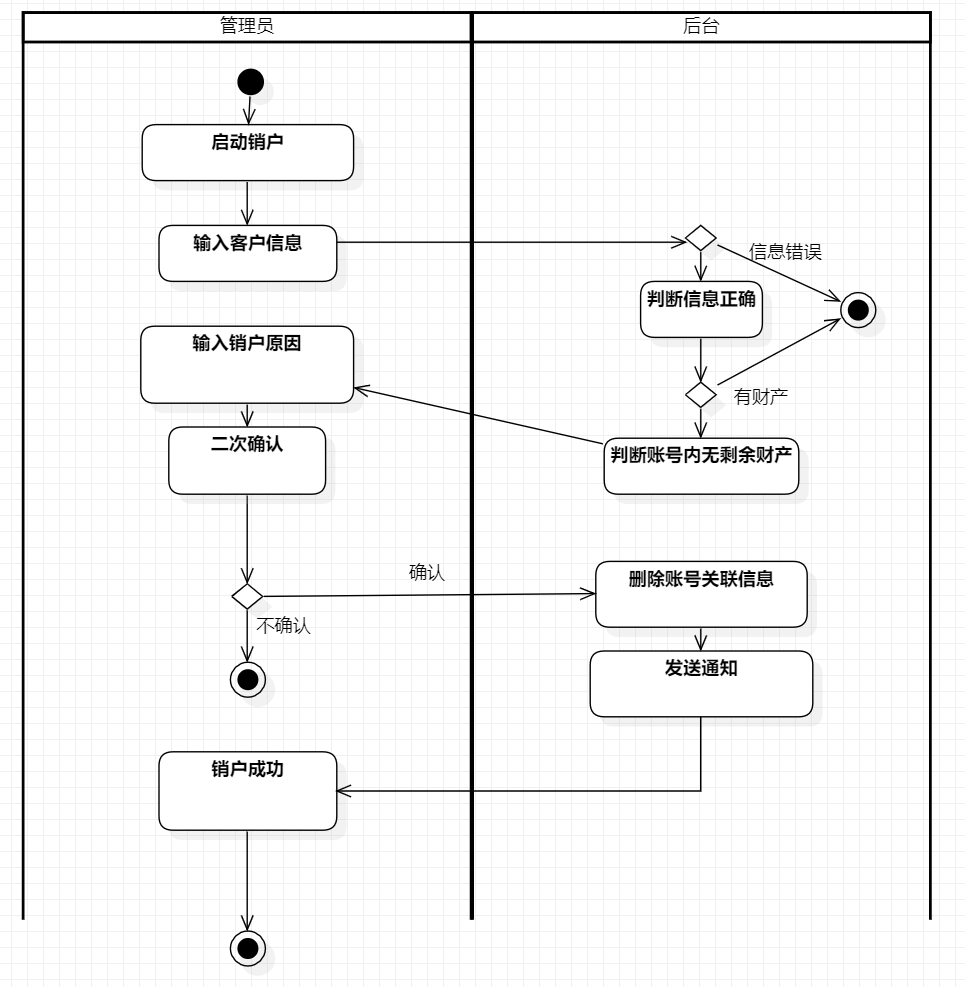


2.4.2.4. 删除客户用例

2.4.2.4.1. 用例模型

|  |  |
| --- | --- |
| **描述对象** | **销户** |
| 编号 | UCB02 |
| 说明 | 业务员输入用户识别信息，并可对识别到的用户进行销户 |
| 参与者 | 业务员 |
| 频度 | 1秒300次 |
| 状态 | 通过审核 |
| 前置条件 | 1. 业务员完成登录操作，进入具体操作界面。 2. 对应客户的账户需完成开户操作，后端数据库中存在。 |
| 基本事件流 | 1. 业务员点击“用户管理”，进入管理页面 2. 业务员点击“销户”，进入销户界面 3. 业务员输入需要进行销户的账户的基本信息，按下确认键即向后端发送请求，得到对应用户是否存在：   3.1若存在，则继续  3.2若不存在，则返回不存在的提示，重新输入信息   1. 对于存在的用户，判断账号内是否有剩余财产：   4.1若有剩余财产，则返回有剩余财产的提示，销户失败  4.2若无财产，则继续   1. 业务员输入销户原因 2. 用户进行身份验证并确认销户：   6.1确认销户，则将账号状态设置为不可用  6.2取消销户，回到管理页面   1. 系统记录操作，同时发送邮件或短信通知用户 |
| 拓展事件流 | 1. 业务员退出了系统，停止流程 2. 业务员中途点击取消销户，停止流程 3. 用户中途点击取消销户，停止流程 4. 用户账户内存在财产，停止流程 5. 用户进行身份认证，进入身份认证用例 |
| 后置条件 | 无 |

2.4.2.4.2. 功能建模

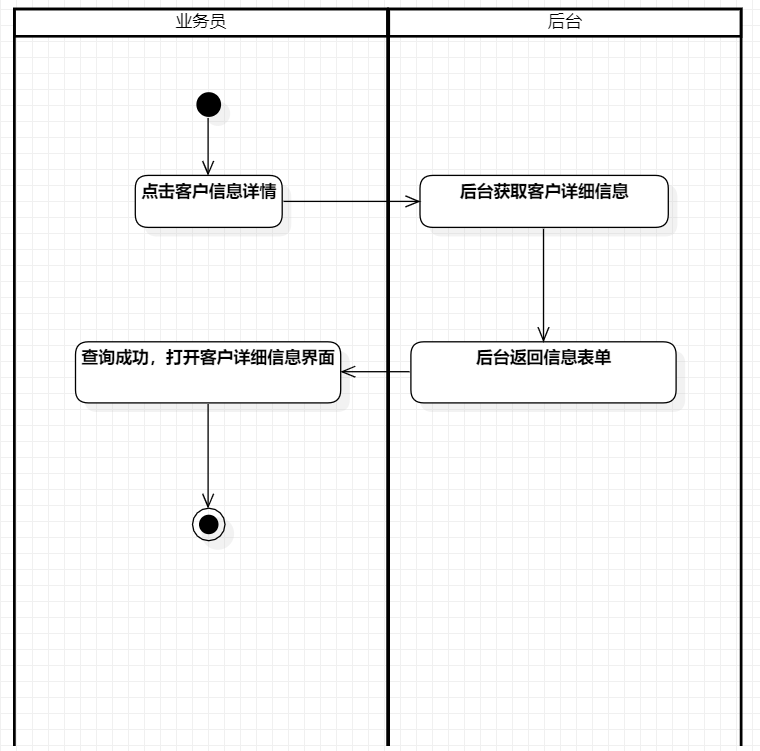


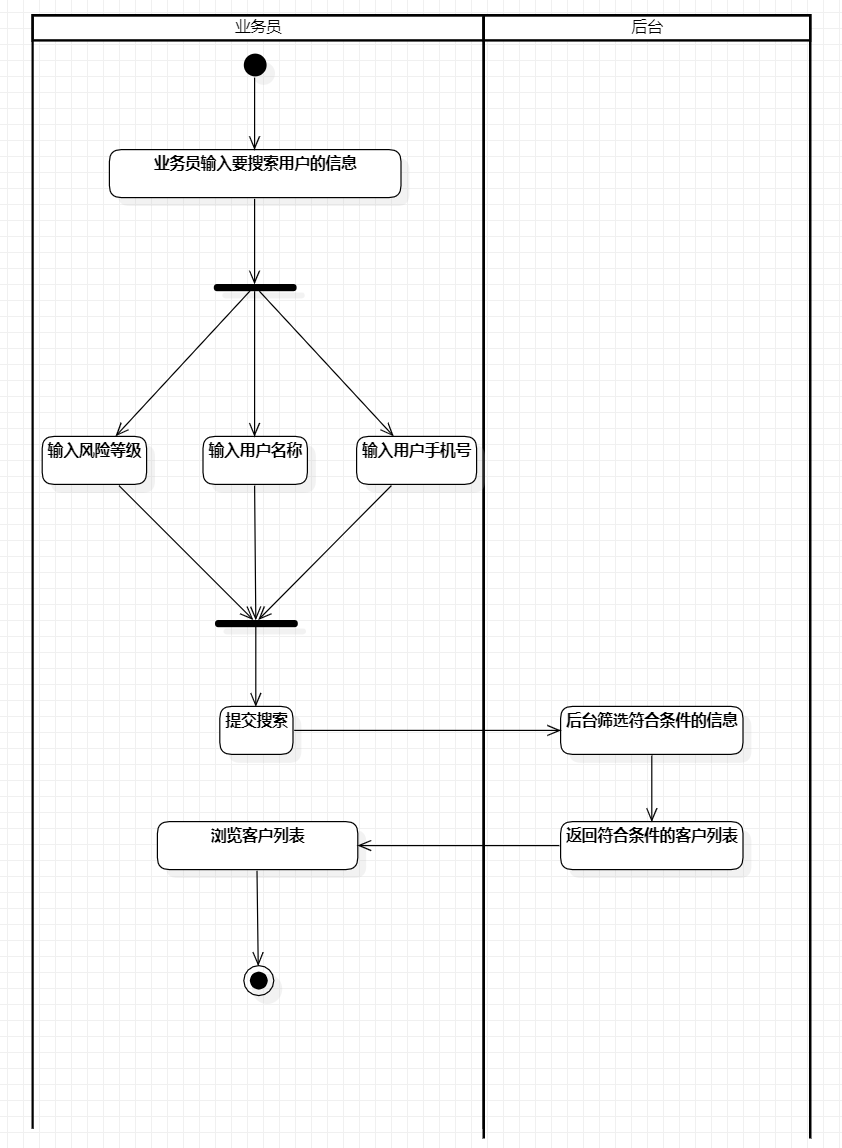
2.4.2.5. 查询客户信息用例

2.4.2.5.1. 用例模型

|  |  |
| --- | --- |
| **描述对象** | **查询客户信息** |
| 编号 | UCB03 |
| 说明 | 业务员查询某一用户详细信息 |
| 参与者 | 业务员 |
| 频度 | 1秒500次 |
| 状态 | 通过审核 |
| 前置条件 | 1. 业务员完成登录操作，进入具体操作界面。 |
| 基本事件流 | 1. 业务员进入管理界面 2. 业务员在用户列表点击用户信息详情 3. 业务员根据结果继续浏览 |
| 拓展事件流 | 1. 业务员退出了系统，停止流程 2. 业务员要修改该客户信息，进入修改客户信息用例 3. 业务员想要在用户列表筛选特定用户或是搜索特定用户，如通过ID、风险类型等进行筛选 |
| 后置条件 | 无 |

2.4.2.5.2. 功能建模





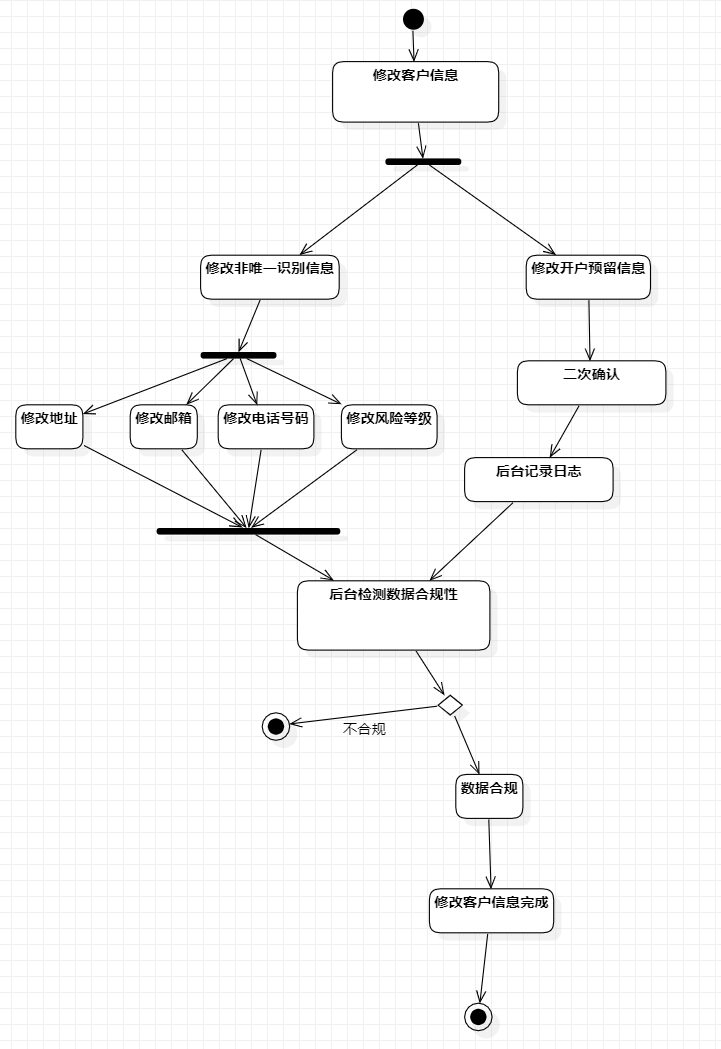
2.4.2.6. 修改客户信息用例

2.4.2.6.1 用例模型

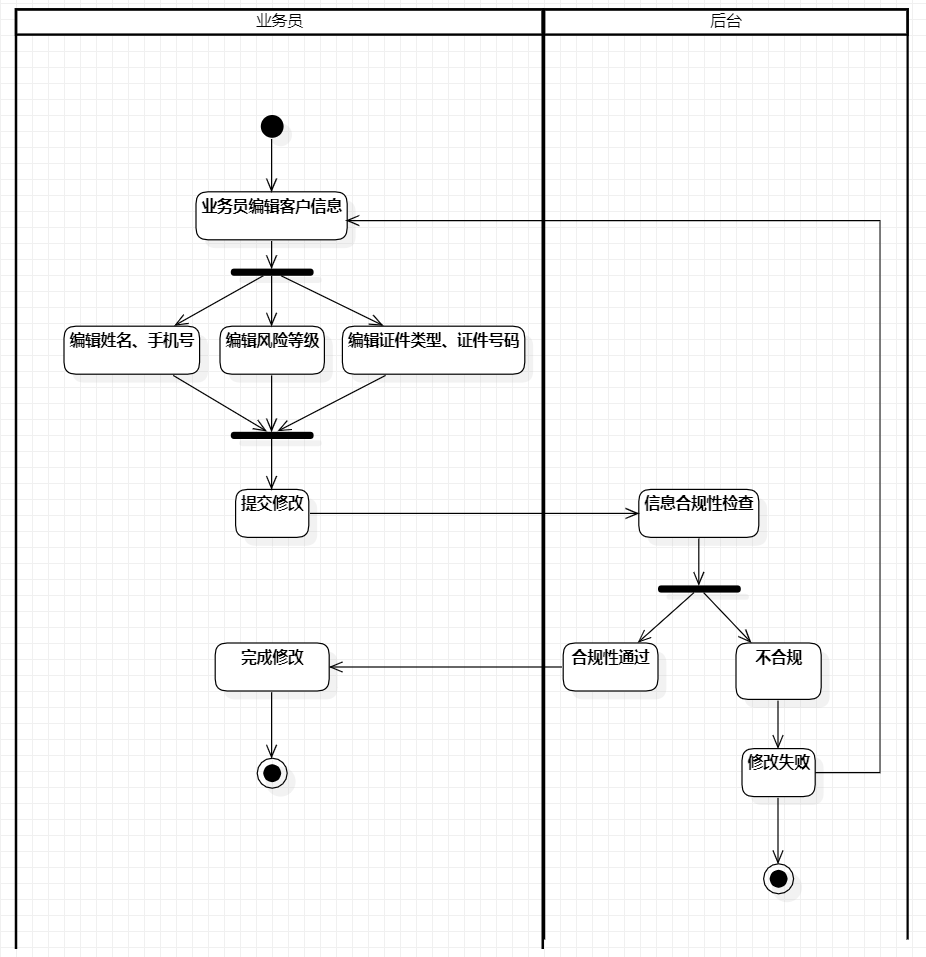
|  |  |
| --- | --- |
| **描述对象** | **修改客户信息** |
| 编号 | UCB04 |
| 说明 | 业务员修改某一用户信息 |
| 参与者 | 业务员 |
| 频度 | 1秒300次 |
| 状态 | 通过审核 |
| 前置条件 | 1. 业务员完成登录操作，进入具体操作界面。 |
| 基本事件流 | 1. 业务员进入某一用户详细信息界面 2. 业务员填写各类信息（包括开户预留和非开户预留信息） 3. 业务员确认修改 4. 系统判断修改的信息是否合法合规：   4.1若合规，则修改完成  4.2若不合规，提示输入信息不合规并回到该用户信息界面   1. 修改完成后，自动刷新并回到客户信息页面，同时若修改了核心信息（Core Info），则需记录操作 |
| 拓展事件流 | 1. 业务员退出了系统，停止流程 2. 业务员修改了核心信息，需要二次确认，并录入系统日志 |
| 后置条件 | 无 |

2.4.2.6.2 功能建模

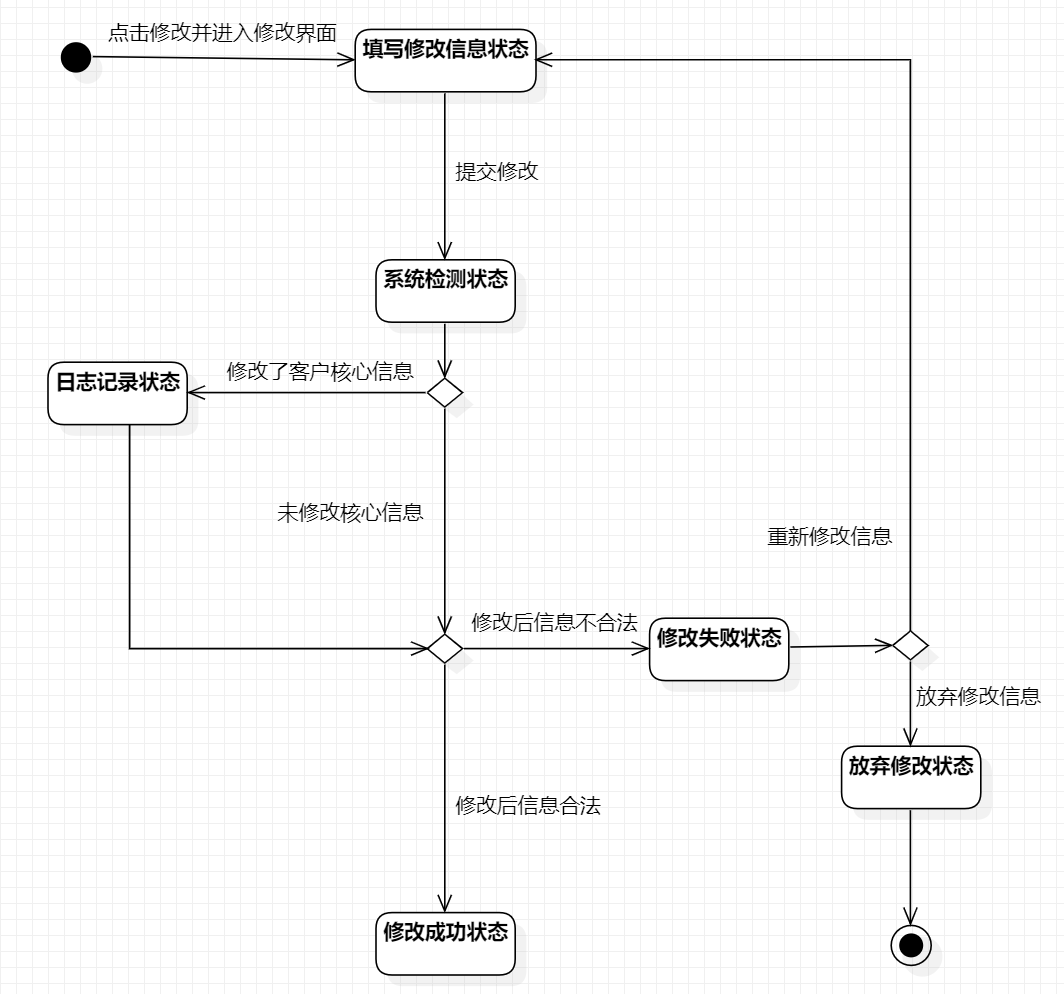
Iteration1:



Iteration2:



2.4.2.6.3 行为建模



2.4.3. 理财产品管理子系统

2.4.3.1. 子系统总述

在理财产品管理系统中，应当实现以下功能：

·业务员查看产品列表

·业务员搜索一款理财产品

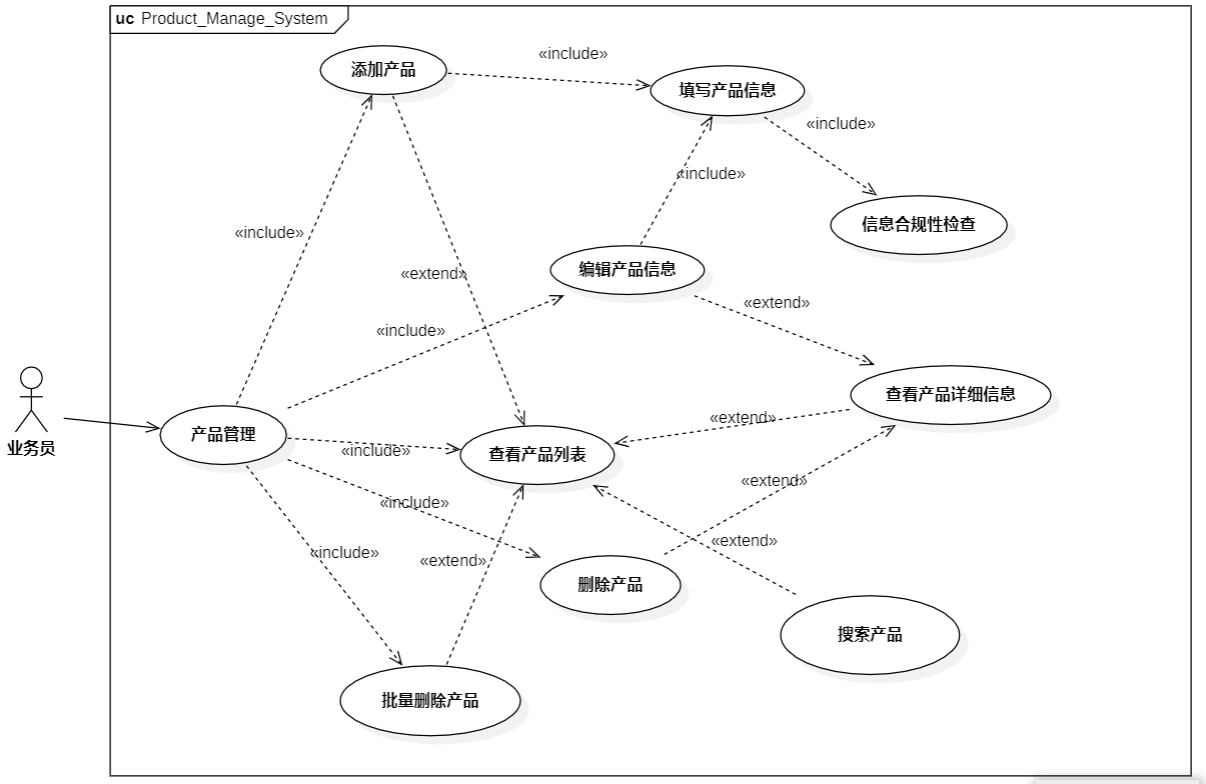
·业务员添加一款理财产品

·业务员查看一款理财产品信息

·业务员修改一款理财产品信息

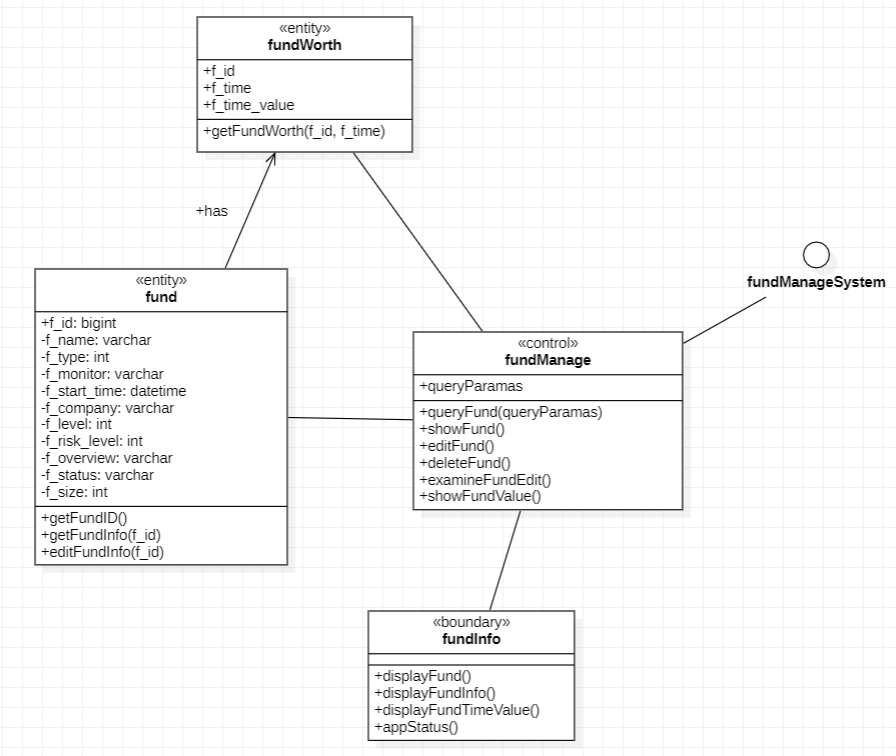
·业务员删除一款理财产品

·业务员批量删除理财产品

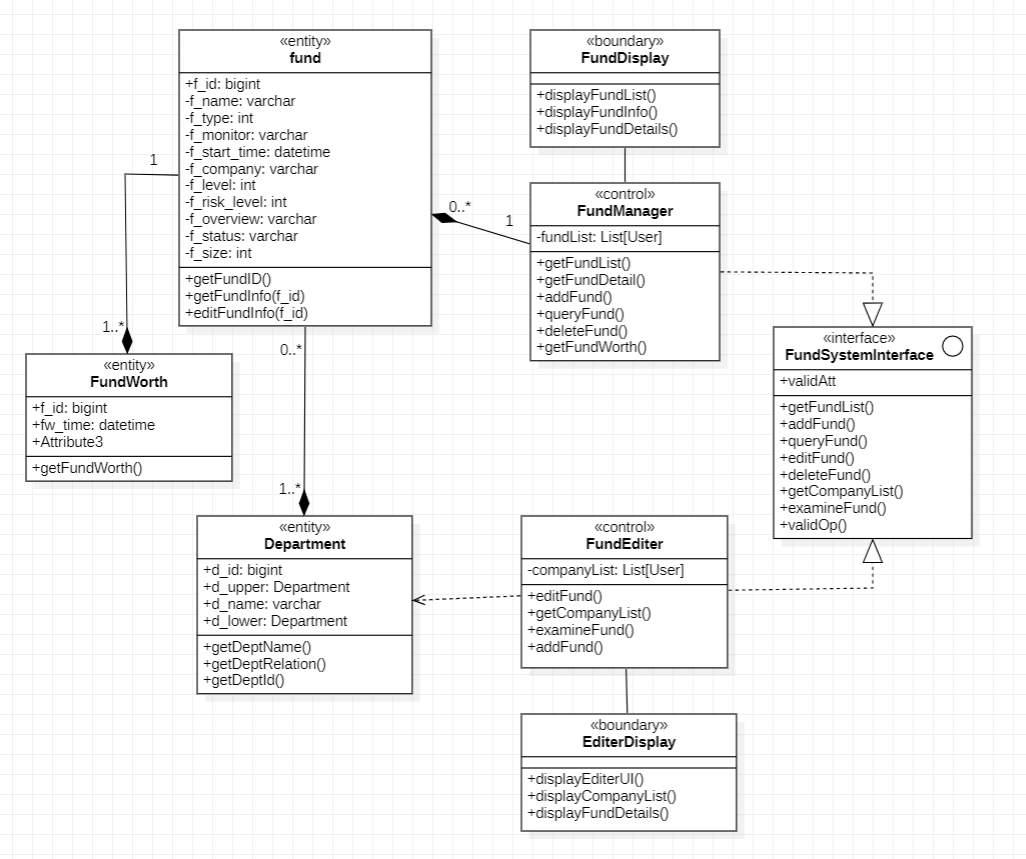


2.4.3.2. 整体数据建模

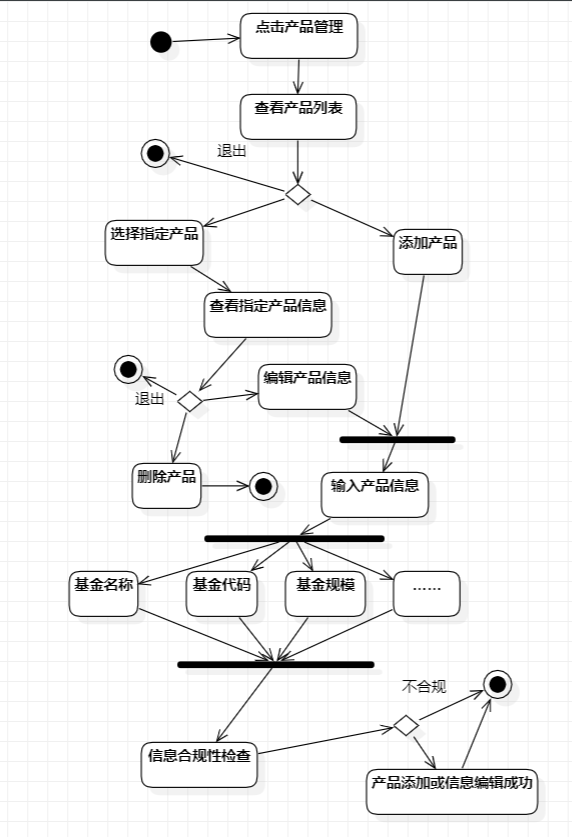
Iteration1:



Iteration2:



2.4.3.3. 整体功能建模

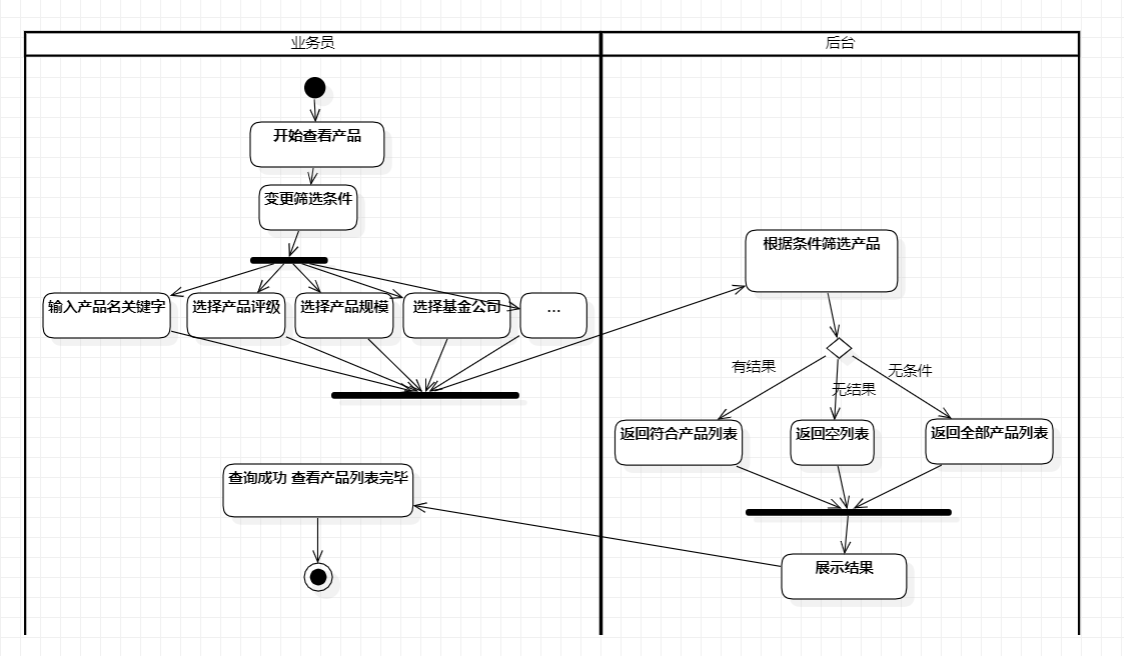


2.4.3.4. 查看产品列表用例

2.4.3.4.1. 用例模型

|  |  |
| --- | --- |
| **名称** | **查看产品列表** |
| 编号 | UCC01 |
| 简述 | 业务员查看产品列表 |
| 执行者 | 业务员 |
| 频度 | 1秒1000次 |
| 状态 |  |
| 前置条件 | 1. 业务员完成登录操作，进入管理界面。 |
| 基本事件流 | 1. 业务员点击理财产品管理 2. 系统返回理财产品列表 3. 业务员继续浏览或执行操作 |
| 拓展事件流 | 1. 业务员退出了系统，停止流程 2. 业务员要添加一款产品，进入添加产品用例 3. 业务员要搜索一款产品，进入搜索产品用例 4. 业务员要查看一款理财产品信息，进入查看产品用例 5. 业务员要批量删除理财产品，进入批量删除产品用例 |
| 后置条件 | 无 |

2.4.3.4.2.功能建模

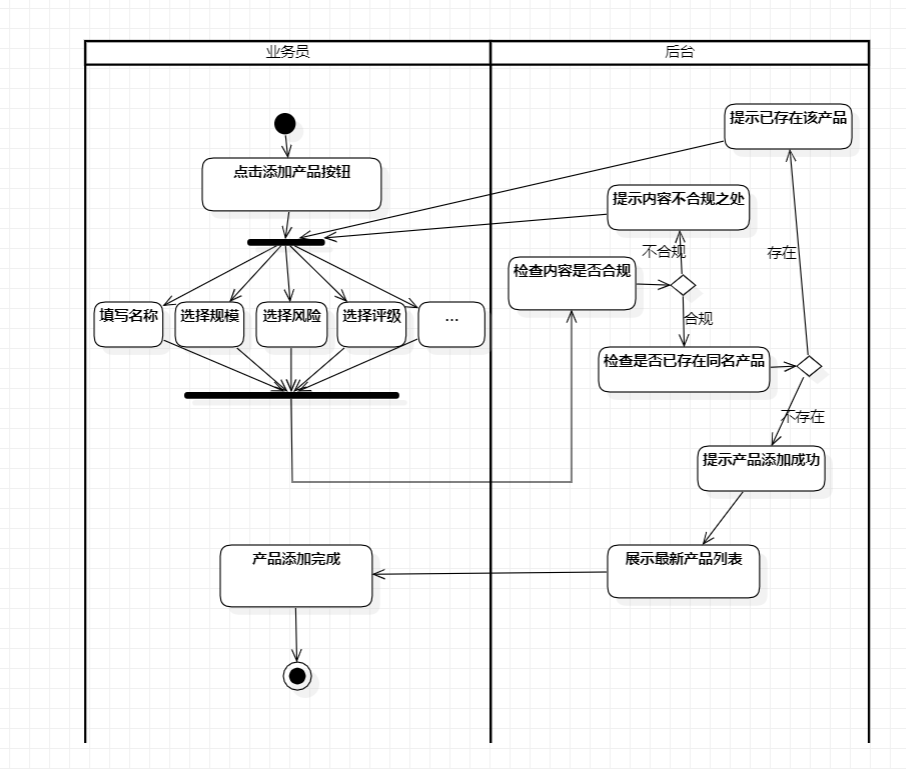


2.4.3.5. 添加产品用例

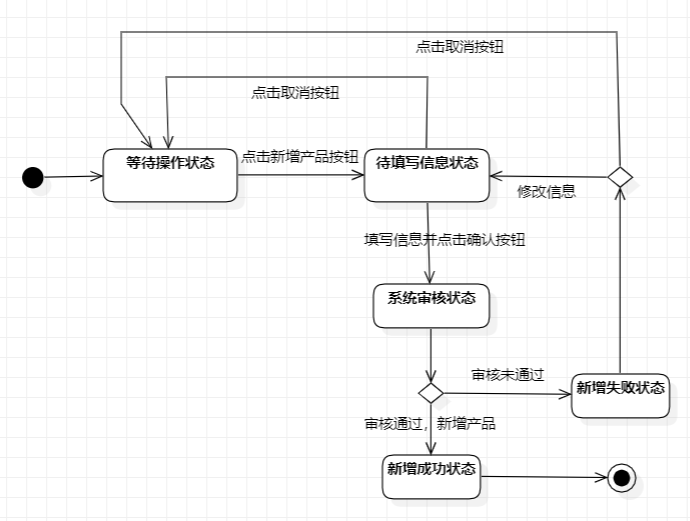
2.4.3.5.1. 用例模型

|  |  |
| --- | --- |
| **名称** | **添加产品** |
| 编号 | UCC02 |
| 简述 | 业务员添加一款理财产品 |
| 执行者 | 业务员 |
| 频度 | 1秒1000次 |
| 状态 |  |
| 前置条件 | 1. 业务员进入了产品管理界面。 2. 该产品在数据库中不存在 |
| 基本事件流 | 1. 业务员点击添加理财产品 2. 业务员填写理财产品信息，如基金名称、基金代码、基金类型、基金规模等，提交给系统 3. 系统判断该款理财产品是否合规：   3.1若合规，添加成功，跳转到该产品信息界面  3.2若不合规，提示添加失败的原因，不发生操作 |
| 拓展事件流 | 1. 业务员退出了系统，停止流程 |
| 后置条件 | 无 |

2.4.3.5.2. 功能建模



2.4.3.5.3. 行为建模

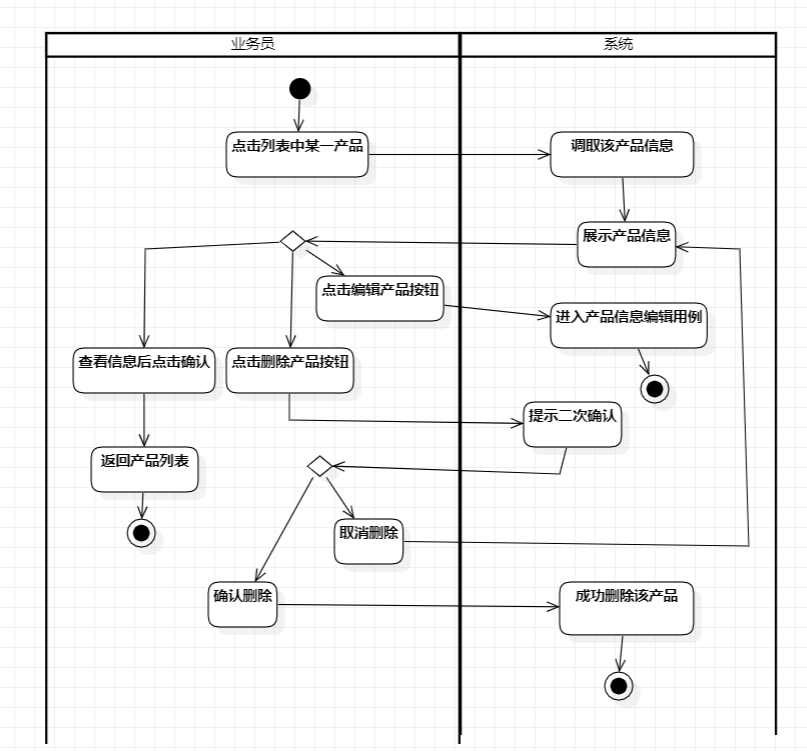


2.4.3.6. 查看产品具体信息用例

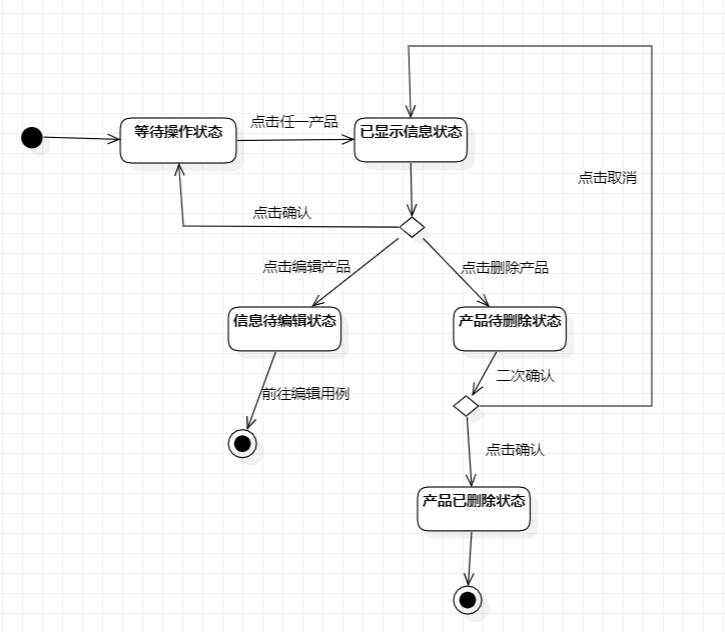
2.4.3.6.1. 用例模型

|  |  |
| --- | --- |
| **名称** | **查看产品具体信息** |
| 编号 | UCC03 |
| 简述 | 业务员查看某一款理财产品信息 |
| 执行者 | 业务员 |
| 频度 | 1秒500次 |
| 状态 |  |
| 前置条件 | 1. 业务员进入了产品管理界面。 2. 该产品在数据库中存在 |
| 基本事件流 | 1. 业务员在系统返回的产品列表中点击一项产品 2. 系统返回对应的产品详细信息 3. 业务员继续浏览或执行操作 |
| 拓展事件流 | 1. 业务员退出了系统，停止流程 2. 业务员选择修改产品信息，进入修改产品信息用例 3. 业务员要删除一款产品，进入删除产品用例 |
| 后置条件 | 无 |

2.4.3.6.2. 功能建模



2.4.3.6.3. 行为建模

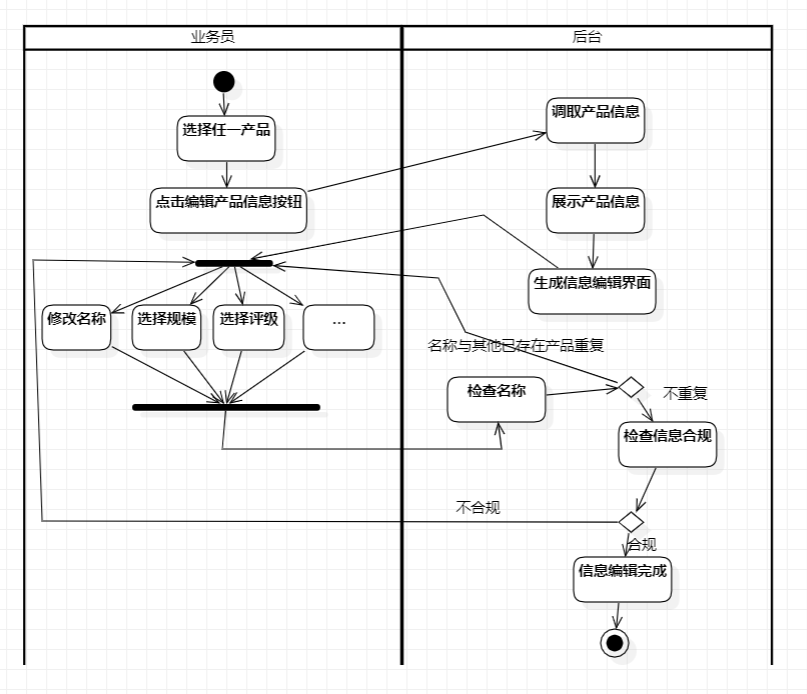


2.4.3.7. 修改产品信息用例

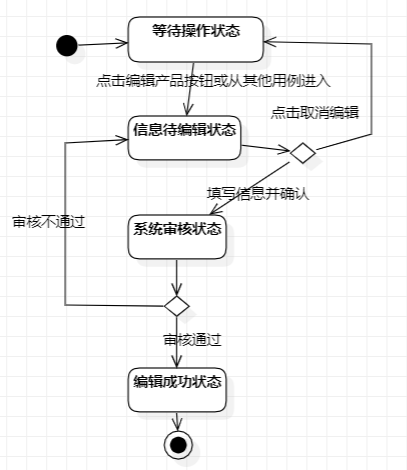
2.4.3.7.1. 用例模型

|  |  |
| --- | --- |
| **名称** | **修改产品信息** |
| 编号 | UCC04 |
| 简述 | 业务员修改某一款理财产品信息 |
| 执行者 | 业务员 |
| 频度 | 1秒500次 |
| 状态 |  |
| 前置条件 | 1. 业务员进入了某一产品信息界面。 2. 该产品在数据库中存在 |
| 基本事件流 | 1. 业务员点击修改信息按钮 2. 业务员填写或修改产品信息后并提交 3. 系统判断信息是否合规：   3.1若合规，则修改成功，跳转回产品信息界面  3.2若不合规，则提示错误原因，不发生操作 |
| 拓展事件流 | 1. 业务员退出了系统，停止流程 2. 业务员要删除一款产品，进入删除产品用例 |
| 后置条件 | 无 |

2.4.3.7.2. 功能建模



2.4.3.7.3. 行为建模



2.4.3.8. 删除产品用例

|  |  |
| --- | --- |
| **名称** | **删除产品** |
| 编号 | UCC05 |
| 简述 | 业务员删除某一款理财产品 |
| 执行者 | 业务员 |
| 频度 | 1秒500次 |
| 状态 |  |
| 前置条件 | 1. 业务员进入了某一产品信息界面。 2. 该产品在数据库中存在 |
| 基本事件流 | 1. 业务员点击删除产品按钮 2. 业务员确认是否删除：   2.1确认，则系统删除该产品，同时记录该操作  2.2取消，回到产品信息界面 |
| 拓展事件流 | 1. 业务员退出了系统，停止流程 |
| 后置条件 | 无 |

2.4.3.8.2. 功能建模

详见2.4.3.6.2.

2.4.3.8.3. 行为建模

详见2.4.3.6.3.

2.4.3.9. 批量删除产品用例

|  |  |
| --- | --- |
| **名称** | **批量删除产品** |
| 编号 | UCC06 |
| 简述 | 业务员批量删除产品列表中的理财产品 |
| 执行者 | 业务员 |
| 频度 | 1秒500次 |
| 状态 |  |
| 前置条件 | 1. 业务员进入了产品管理列表。 2. 被删除的产品在数据库中存在 |
| 基本事件流 | 1. 业务员勾选产品列表中要删除的产品前的复选框 2. 业务员点击批量删除 3. 业务员确认是否删除：   3.1确认，则系统删除被勾选的产品，同时记录该操作  3.2取消，回到产品列表管理界面 |
| 拓展事件流 | 1. 业务员退出了系统，停止流程 |
| 后置条件 | 无 |

3. 非功能需求

3.1. 性能要求

3.1.1. 精度

对于货币和金额的处理要求精确到小数点后四位。

对利率的处理精确到小数点后四位。

对于基金净值的计算，系统要求精确到小数点后四位。

对于日期和时间的处理，系统要求保持到秒级的精度，并且要符合特定的时区。

金融系统生成的报告和文档要求输出的数字精确到小数点后两位，确保用户获得准确的、符合金融行业标准的信息。

3.1.2. 时间特性要求

系统应该能在1秒内响应用户的大部分请求，例如查看产品信息、提交申购/赎回请求等。对于一些可能需要更长时间处理的请求，如复杂查询或生成报告，系统应在5秒内响应。

考虑到每天可能有2000笔交易，系统应能在一个交易日内处理这些交易，即系统应该能处理每秒至少0.02笔交易。考虑到峰值时段可能会有更高的交易量，系统的最大处理能力应至少是这个数值的10倍，即每秒0.2笔交易。同时，系统应该能同时处理至少200个并发用户的请求，而不会显著降低响应时间或数据吞吐量。

3.1.3. 输入输出要求

对于开户：



对于新增银行卡：



对于新增基金：



3.2. 安全及保密性要求

3.2.1. 访问控制

系统应实现严格的访问控制机制，确保只有经过身份验证并授权的业务员才能访问系统功能。对于业务员，应根据其角色分配不同的权限，例如只能访问与基金买卖相关的功能。

3.2.2. 数据加密

敏感数据在传输和存储过程中应进行加密，以防止未经授权的访问和信息泄露。对于金融交易数据、客户个人信息等敏感数据，采用强加密算法，确保数据的保密性。

3.2.3. 安全审计/日志

实施安全审计机制，对系统的关键操作和敏感数据访问进行记录和监控。对于业务员的操作和金融交易，进行详细的审计记录，以便追溯和监测异常操作。

3.3. 灵活性要求

3.3.1. 界面定制

系统应具有界面定制的功能，允许业务员根据个人偏好和工作需求调整界面布局和显示内容，以提高工作效率。

3.3.2. 业务流程配置

系统应支持业务流程的灵活配置，以适应金融业务的变化。业务员可以根据需要调整买卖基金等操作的流程，确保系统适应业务的灵活性。

3.3.3. 操作方式变更

系统应具备易于调整的操作方式，以适应用户的操作习惯和工作流程的变化。例如，当业务员反馈对某个操作流程的不便之处时，系统应能够灵活调整相应的操作界面或流程，提高操作效率。

3.3.4. 运行环境变更

系统应具备对运行环境变化的适应能力。例如，当金融机构需要升级服务器硬件或更换操作系统时，系统应能够平稳过渡，不影响正常运行。

3.3.5. 精度和有效时限的变化

系统应具备灵活的精度和时效性配置，以适应金融领域对数据精度和实时性的不断提升要求。系统应允许根据需求调整数据处理精度，同时确保在计划时限内完成相应的业务操作。

4. 运行环境规定

4.1. 设备

a. 处理器型号及内存容量

处理器型号：支持x86\_64架构的处理器，现有服务器处理器型号为Intel E3-1230.

内存容量：最低8GB RAM

b. 外存容量、联机或脱机、媒体及其存储格式，设备的型号及数量

外存容量：100GB以上，用于存储系统运行所需的软件和数据。

联机/脱机：联机运行，需要网络连接。

媒体及其存储格式：无特定要求，支持常见的存储媒体和文件系统。

c. 输入及输出设备的型号和数量，联机或脱机状况

输入设备：键盘、鼠标

输出设备：显示器、打印机

联机/脱机状况：联机

d. 数据通信设备的型号和数量

数据通信设备：网卡

型号和数量：适配机构网络环境的标准网络接口卡，数量取决于实际部署情况。

e. 功能键及其他专用硬件

功能键：详见系统界面设计。

其他专用硬件：无特殊要求，根据业务需要可考虑支持扫描仪、数字签名设备等。

4.2. 支持软件

操作系统平台：支持 Windows Server 2016 及以上版本，或 Linux 发行版（例如 Ubuntu Server 18.04 及以上）。

数据库系统平台：支持 MySQL 8.0 及以上，或 PostgreSQL 13 及以上。

编译程序和测试支持软件：JDK 11 及以上，Maven 3.6 及以上。

4.3. 接口

系统接口：与其他金融系统进行数据交换，采用标准金融数据格式和协议。

用户接口：提供 Web 用户界面，支持主流浏览器（Chrome、Firefox、Safari）。

4.4. 控制

运行控制方法：通过系统管理界面进行运行控制，包括启动、停止、重启等操作。

控制信号来源：来自系统管理员或授权人员，通过安全身份验证后获得控制权限。