

<任务管理系统>

软件架构设计

版本 <1.0>

修订历史记录

日期	版本	修订说明	作者
<2016-5-1>	<0.2>	<编写引言、系统概述、用例视图、系统逻辑视图部分>	<姜金明>
<2016-5-1>	<0.4>	<编写目前软件系统架构、系统实现视图、系统物理视图部分>	<林封利>
<2016-5-1>	<0.6>	<编写系统运行视图、数据管理设计、其他设计部分>	<钟文才>
<2016-5-1>	<0.8>	<编写系统架构设计目标、边界条件设计部分>	<王哲维>
<2016-5-4>	<1.0>	<整合并对各部分内容进行修订>	<姜金明>

1 引言.....	1
1.1 编写目的.....	1
1.2 适用范围.....	1
1.3 定义.....	1
1.4 参考资料.....	1
1.5 概述.....	1
2 目前软件系统体系架构.....	2
3 软件系统架构设计目标.....	2
4 建议的软件系统架构.....	3
4.1 概述.....	4
4.2 用例视图.....	5
4.3 系统逻辑视图.....	12
4.4 系统运行视图.....	22
4.5 系统实现视图.....	22
4.6 系统物理视图.....	25
4.7 边界条件设计.....	29
4.8 数据管理设计.....	33
4.9 其他设计.....	33

软件需求规约

1 引言

1.1 编写目的

本文档的编写主要是为了更具体地概述任务管理系统软件的软件结构及其实现方式，让开发人员更好地了解系统，并指导开发人员更好地开发系统。本文档的预期读者为任务管理系统的相关开发人员。

1.2 适用范围

本文档适用于与任务管理系统软件开发相关的一切文档，本文档仅适用于任务管理系统。

1.3 定义

HTFMMS：HappyTreeFriends Mission Manage System HTF 任务管理系统

1.4 参考资料

《面向对象软件工程 使用 UML、模式与 Java》，Bernd Bruegge、Allen H. Dutoit 著，清华大学出版社，第 3 版。

1.5 概述

本文档主要包括引言、目前软件系统体系架构、软件系统架构设计目标、建议的软件系统架构三个部分。引言部分主要概述了本文档的编写目的和组织方式；目前软件系统架构部分主要分析了目前已经存在的相关软件的系统架构；软件系统架构设计目标部分概述了本系统将要实现的各项目标；建议的软件系统架构部分则详细描述了本系统的用例、逻辑、实现、运行、物理部署、边界条件和数据管理等相关内容。

2 目前软件系统体系架构

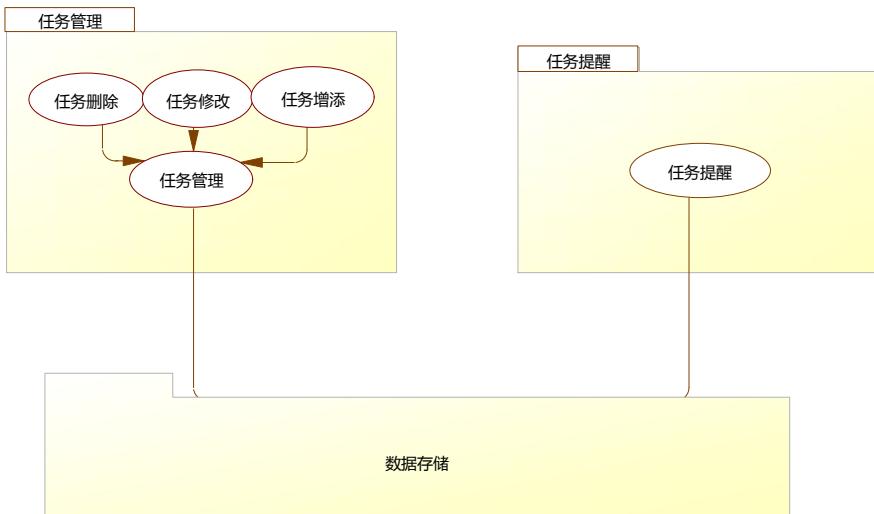
目前系统主要包括：安卓应用商店中的任务 / 计划管理软件(这里以计划管理大师为例)



你没有正在进行中的计划~

轻触屏幕，开始一个计划

目前系统的架构主要由任务管理，数据存储和任务提醒三个子系统组成



目前系统存在的问题:

对于计划管理大师来说，主要存在以下问题：

1. 不具备将任务分类管理功能，无法系统高效的管理任务
2. 不能方便的对各项任务进行紧急程度，重要程度等方面排序
3. 缺乏用户登录功能，不能实现信息在不同设备间的共享
4. 缺乏对任务的完成情况进行更新的功能，不能及时的反应任务完成情况
5. 缺乏对任务进行统计分析的功能，不能使用户方便的获得整个阶段的工作情况
6. 用户界面不够美观，用户体验不够良好

新系统希望得到的改进:

1. 能够将任务分类管理
2. 能够对各项任务进行排序
3. 能够实现用户注册登录功能
4. 能够实现用户更新任务完成情况功能
5. 能够对任务完成情况进行统计分析
6. 改进用户界面

3 软件系统架构设计目标

3.1 安全性

软件系统闭源发布。

3.2 保密性

软件数据库的一切资料在法律范围内仅对使用者本人开放，对其他人保密。

3.3 可移植性

无，不需要顾及此方面。

3.4 软件更新

软件定期自检更新，并且能够方便地用新的组件替换原有组件更新软件。

3.5 开发时间表与团队结构

任务名称	负责人	参与成员
需求分析	姜金明	林封利、钟文才、王哲维
文件编制	钟文才	林封利、姜金明、王哲维
软件设计	林封利	姜金明、王哲维、钟文才
UI 界面	王哲维	姜金明、林封利、钟文才
算法	姜金明	林封利、钟文才、王哲维
系统架构	钟文才	林封利、姜金明、王哲维
测试	林封利	姜金明、王哲维、钟文才
用户培训	王哲维	姜金明、林封利、钟文才

阶段	预定开始日期	预定完成日期	所需资源	里程碑
可行性研究	3. 17	3. 25		
需求分析	3. 26	4. 5		完成第一版文档
设计	4. 5	4. 15	设计师	完成设计文档
编码实现	4. 15	5. 20	开发环境, 程序员	软件成型
测试	5. 20	5. 23		测试成功
移交	5. 24	5. 26		
培训	5. 27	5. 28	培训人员	
安装	5. 29	5. 30		投入使用

3.6 开发工具

Win7 以上版本 Windows 系统电脑 4 台。

3.7 销售

仅在各个应用平台上进行销售

8. 设计与实施策略

精益求精，保证时间表的情况下尽量完美细节。

4 建议的软件系统架构

4.1 概述。

本软件的架构模式是采用的是仓库模式的一个特例：三层客户/服务器模式（C/S 模式）。

第一层接口层包括用户界面子系统，第二层应用逻辑层包括管理账号、管理任务、任务反馈分析三个子系统，第三层存储层包括数据管理子系统。

选择三层 C/S 模式主要是因为我们软件中的子系统之间相互独立，子系统之间的交互都可以通过仓库（数据管理子系统）完成；并且我们的软件还带有经常发生改变的、有复杂数据处理的任务（管理任务、管理账号）。

用户界面子系统的主要功能是与用户进行交互，将用户的请求传输到第二层的各个子系统中，并且显示第二层其他子系统返回的信息。

管理账号子系统的主要功能是进行用户账号的管理，包括账号注册、账号登陆、修改账号密码。

管理任务子系统的主要功能主要是对用户任务的管理，包括增添任务、删除任务、修改任务、更新任务进度。

任务反馈分析子系统的主要功能是给用户分析、反馈某一时间段用户完成任务的情况，同时对任务进行提醒。

数据管理子系统的主要功能是管理和储存整个软件的数据。

本软件重用的中间件主要是数据库及其接口的中间件。

4.2 用例视图

下列是本系统的几个具有代表性的核心功能的用例图：

- **用例名称：** 注册任务管理系统账号
- **范围：** 系统用例
- **级别：** 子功能
- **主要参与者：** 系统用户
- **涉众及其关注点：**
 - 系统用户：希望能够方便、快捷地进行注册，并及时得到系统关于是否注册成功的响应。
- **前置条件：** 手机数据网络处于连通状态。
- **后置条件：** 更新用户账号信息记录，发送注册成功信息；或提示用户注册不成功。
- **主流程：**
 1. 系统用户点击“账号管理”按钮进入“账号管理”界面。
 2. 系统用户点击“注册”按钮进入“注册账号”界面
 3. 系统用户输入所希望注册的账号名。
 4. 系统检测到该账号名未被使用，将此结果显示给用户
 5. 系统用户输入注册密码，点击“完成注册”按钮
 6. 系统将新建账号信息存档，弹出“注册成功”信息框，1s 后返回“登录账号”界面。
- **扩展流程：**

1. 用户账号名检测不通过

在第 4 步，服务器判别用户账号名无效

(1) 提示用户“该用户名无效，请更改”，直至用户输入的用户名可用；

2. 信息存档失败

在第 6 步，系统将信息存档时发生连接中断等异常

(1) 提示用户“注册失败，请稍后重试”

● **特殊需求：**无

● **发生频率：**可能随时发生。

● **用例名称：**增添任务

● **范围：**系统用例

● **级别：**子功能

● **主要参与者：**系统用户

● **涉众及其关注点：**

➤ 系统用户：希望能够方便、快捷地增添任务，并及时得到系统关于是否操作成功的响应。

● **前置条件：**无

● **后置条件：**更新用户任务记录；或提示用户操作不成功。

● **主流程：**

1. 系统用户点击主界面上新建任务按钮。

2. 系统弹出“新建任务表格”窗口。

3. 系统用户填入任务信息（名称、内容、开始时间、结束时间、奖励积分、惩罚积分、类型、提醒方式、提醒时间等信息）并点击“完成”按钮提交。

4. 系统保存相应信息，重新切回主界面，并且在主界面的相应位置显示新建的任务。

● **扩展流程：**

1. 同步任务信息

在第 4 步，若用户已登陆账号：

系统将新建任务信息传输给服务器，并更新服务器上的用户信息

● **特殊需求：**无

● **发生频率：**可能随时发生

- **用例名称:** 修改任务
- **范围:** 系统用例
- **级别:** 子功能
- **主要参与者:** 系统用户
- **涉众及其关注点:**
 - 系统用户：希望能够方便、快捷地修改任务，并及时得到系统关于是否操作成功的响应。
- **前置条件:** 无
- **后置条件:** 更新用户任务记录；或提示用户操作不成功。
- **主流程:**
 1. 系统用户点击主界面上需要修改的任务。
 2. 系统切换到“显示任务 A 详细信息”界面。
 3. 系统用户点击“编辑”按钮。
 4. 系统进入“编辑任务”模式
 5. 系统用户修改该任务的属性，并点击“完成”按钮提交。
 6. 系统保存相关信息，向系统用户反馈修改任务结果，并返回“显示任务 A 详细信息”界面
- **扩展流程:**
 1. 同步任务信息

在第 6 步，若用户已登陆账号：

系统将修改任务信息传输给服务器，并更新服务器上的用户信息
- **特殊需求:** 无
- **发生频率:** 可能随时发生

- **用例名称:** 删除任务
- **范围:** 系统用例
- **级别:** 子功能
- **主要参与者:** 系统用户
- **涉众及其关注点:**

- 系统用户：希望能够方便、快捷地删除任务，并及时得到系统关于是否操作成功的响应。

- **前置条件：**无
- **后置条件：**更新用户任务记录；或提示用户操作不成功。
- **主流程：**
 1. 系统用户点击主界面上需要删除的任务。
 2. 系统切换到“显示任务 A 详细信息”界面。
 3. 系统用户点击“删除”按钮。
 4. 系统弹出“是否确认删除任务 A”对话框。
 5. 系统用户点击“是”按钮提交。
 6. 系统保存相应信息，向系统用户反馈删除任务结果，并返回主界面界面
- **扩展流程：**
 1. 同步任务信息
在第 6 步，若用户已登陆账号：
系统将删除任务信息传输给服务器，并更新服务器上的用户信息

- **用例名称：**更新任务进度
- **范围：**系统用例
- **级别：**子功能
- **主要参与者：**系统用户
- **涉众及其关注点：**
 - 系统用户：希望能够方便、快捷地更新任务进度，并及时得到系统关于是否操作成功的响应。

- **前置条件：**无
- **后置条件：**更新用户任务记录；或提示用户操作不成功。
- **主流程：**
 1. 系统用户在主界面上点击需要更新进度的任务“任务 A”。

2. 系统切换到“显示任务详细信息”界面。
3. 系统用户在“显示任务详细信息”界面上点击“更新进度”按钮。
4. 系统弹出“任务进度”消息框。
5. 系统用户将“任务进度”进行更新。
6. 系统保存相应信息，向系统用户反馈删除任务结果，并返回主界面界面

● 扩展流程：

1. 同步任务信息

在第6步，若用户已登陆账号：

系统将删除任务信息传输给服务器，并更新服务器上的用户信息

● 特殊需求：无

● 发生频率：可能随时发生

● 用例名称：查看任务统计分析

● 范围：系统用例

● 级别：用户目标

● 主要参与者：系统用户

● 涉众及其关注点：

➤ 系统用户：希望能够快速而准确地获得所有任务的信息。

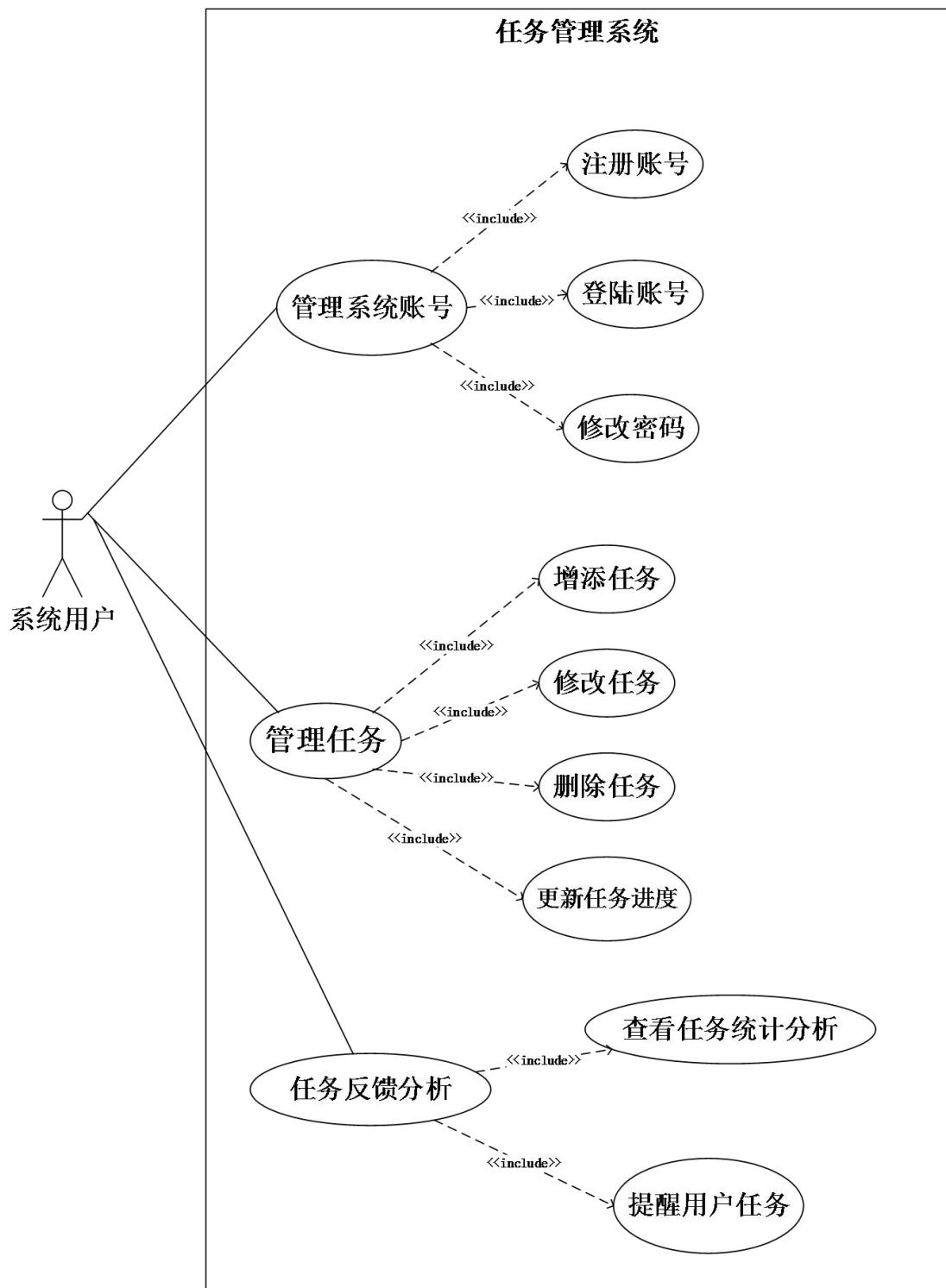
● 前置条件：无

● 后置条件：无

● 主流程：

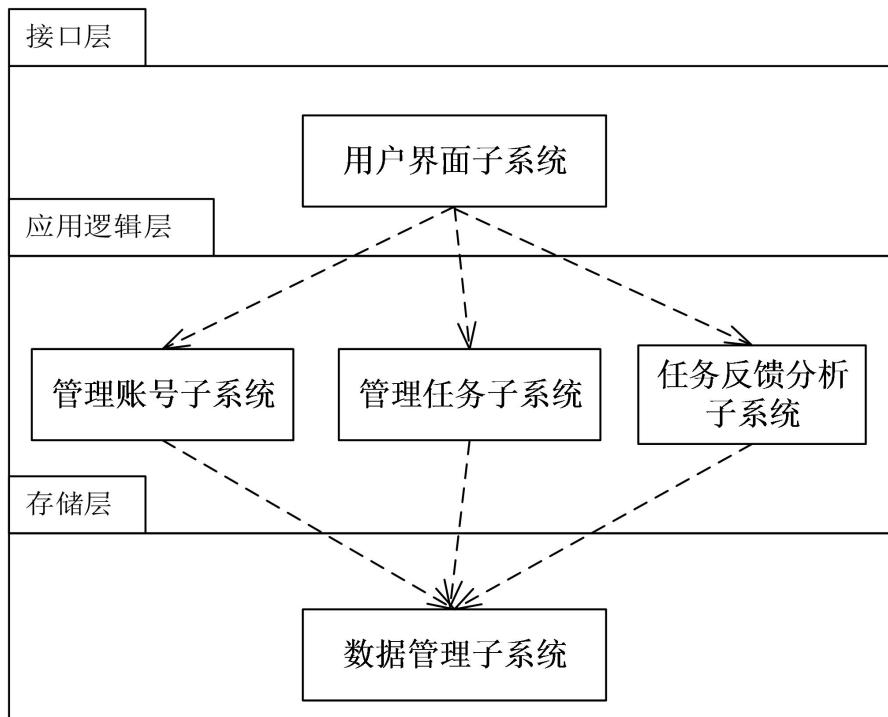
1. 系统用户点击主界面上的“统计分析”按钮。
2. 系统切换到“统计分析”界面，并显示出所有任务的完成进度（百分比）。
3. 系统用户点击“已完成”按钮
4. 系统显示出筛选后的结果
5. 系统用户查看自己所获得的积分
6. 系统用户点击“退出”按钮
7. 系统返回主界面

- 扩展流程：
 - 第4步中用户点击“本周”或“本日”按钮
 - (1) 系统显示出对应的筛选结果
- 特殊需求：无
- 发生频率：可能随时发生



4.3 系统逻辑视图

(1) 系统架构



先由第一层的用户界面子系统接受用户的请求并将用户的请求传递给第二层应用逻辑层的三个子系统：管理账号子系统、管理任务子系统、任务反馈分析子系统；第二层的三个子系统对用户的请求进行处理之后，再传递信息给第三层进行数据存储方面的操作，最后将结果返回给第一层的用户界面子系统，显示给用户。

用户界面子系统的主要功能是与用户进行交互，将用户的请求传输到第二层的各个子系统中，并且显示第二层其他子系统返回的信息。

管理账号子系统的主要功能是进行用户账号的管理，包括账号注册、账号登陆、修改账号密码。

管理任务子系统的主要功能主要是对用户任务的管理，包括增添任务、删除任务、修改任务、更新任务进度。

任务反馈分析子系统的主要功能是给用户分析、反馈某一时间段用户完成任务的情况，同时对任务进行提醒。

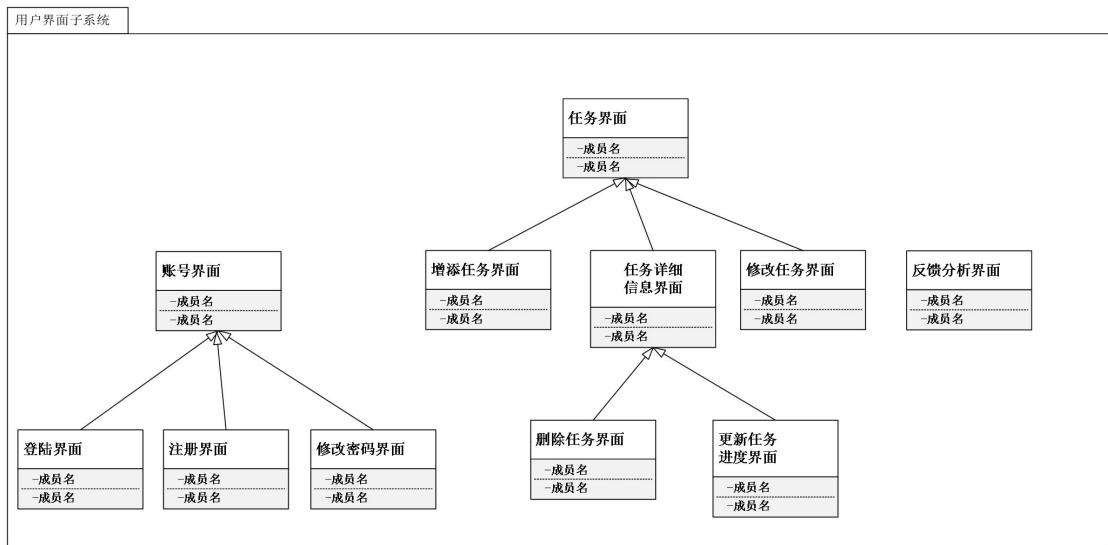
数据管理子系统的主要功能是管理和储存整个软件的数据。

(2) 子系统

用户界面子系统：

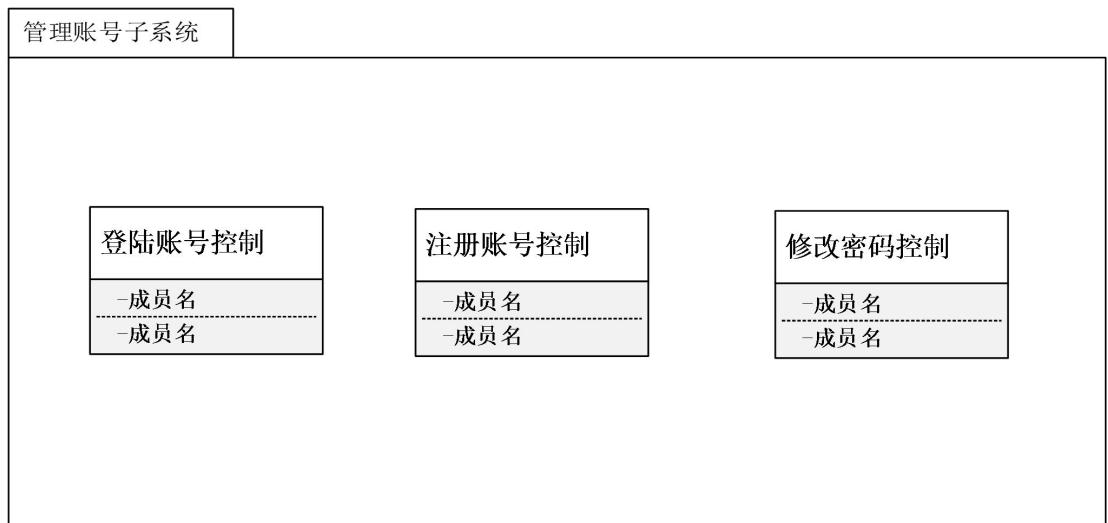
- 子系统功能：
 - 接受用户的请求
 - 将用户的请求传递给第二层的子系统（接口函数：submitAccountInfo()、submitMissionInfo()、submitReportInfo()）

- 接收第二层子系统的返回信息（接口函数：`returnAccountResult()`、`returnMissionResult()`、`returnReportResult()`）
 - 将返回信息显示给用户



管理账号子系统：

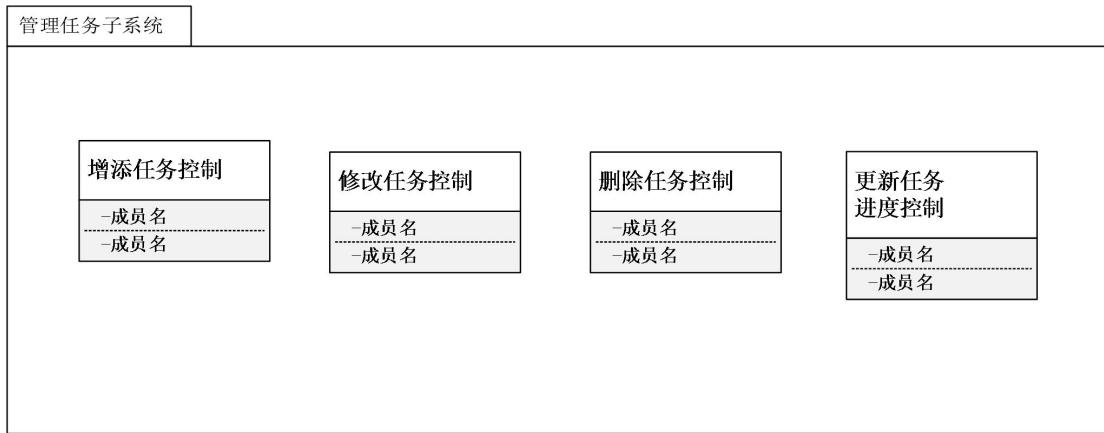
- 子系统功能：
 - 接受接口层传递的管理账号请求及账号表单(接口函数：submitAccountInfo())
 - 处理用户的注册账号、登陆账号、修改密码的请求
 - 将需要更改的账号数据传递给数据管理子系统(接口函数：sendAccountData())
 - 将处理结果返回给用户界面子系统（接口函数：returnAccountResult()）
 - 接收数据管理子系统的数据修改结果(接口函数：receiveAccountResult())



管理任务子系统：

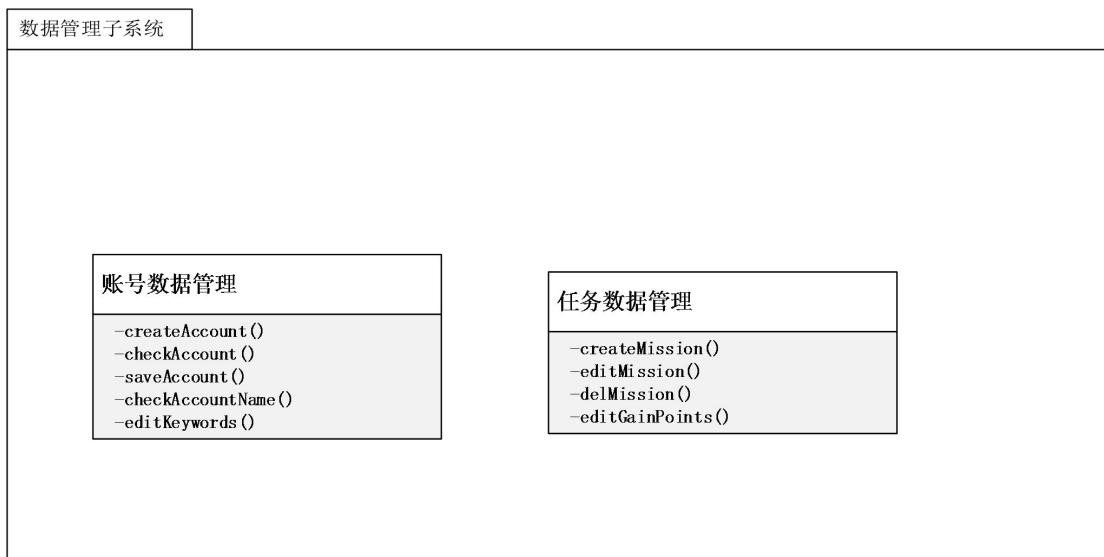
- #### ● 子系统功能:

- 接受接口层传递的管理任务请求及任务表单(接口函数：submitMissionInfo())
- 处理用户的增添任务、修改任务、删除任务、更新任务控制的请求
- 将需要更改的任务数据传递给数据管理子系统(接口函数：sendMissionData())
- 将处理结果返回给用户界面子系统(接口函数：returnMissionResult())
- 接收数据管理子系统的数据修改结果(接口函数：receiveMissionResult())



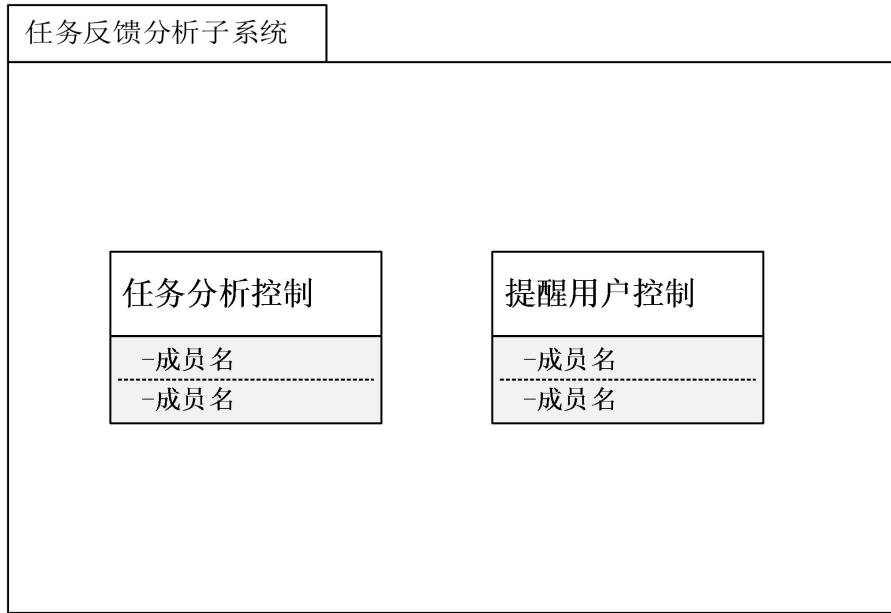
数据管理子系统：

- 子系统功能：
 - 接受第二层子系统的数据处理请求(接口函数：sendAccountInfo()、sendMissionInfo()、sendReportInfo())
 - 将数据处理的结果返回给第二层子系统(接口函数：receiveAccountResult()、receiveMissionResult()、receiveReportResult())
 - 处理账号数据的创建、修改、检查任务；任务数据的创建、修改、删除任务。



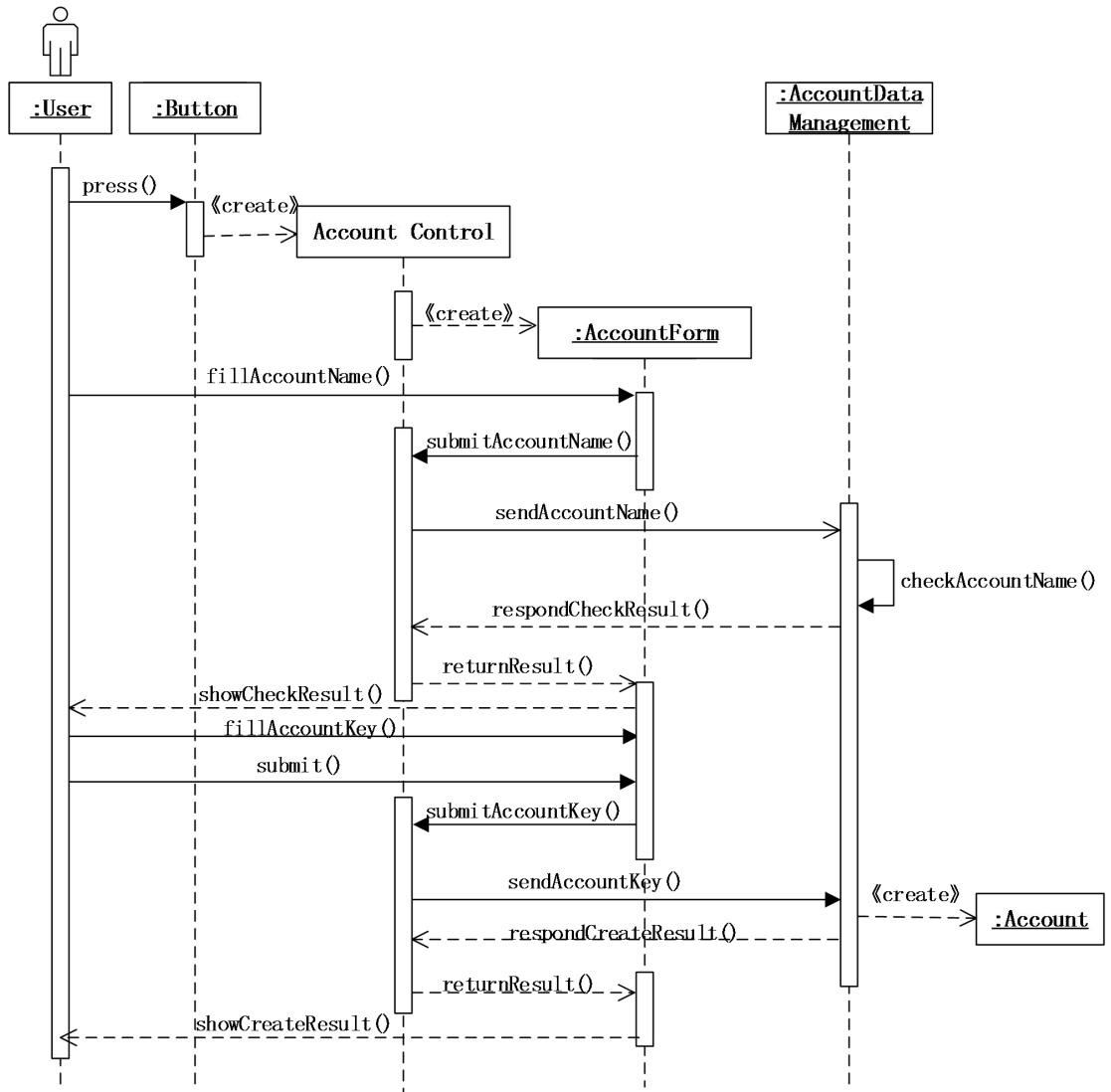
任务反馈分析子系统：

- 子系统功能：
 - 接受接口层传递的获取反馈分析请求(接口函数：submitReportInfo())
 - 对任务及其完成情况进行分析的功能，控制提醒用户任务的功能
 - 将需要获取的反馈分析请求传递给数据管理子系统(接口函数：sendReportData())
 - 将反馈分析结果返回给用户界面子系统 (接口函数：returnReportResult())
 - 接收数据管理子系统的反馈分析数据 (接口函数：receiveReportResult())

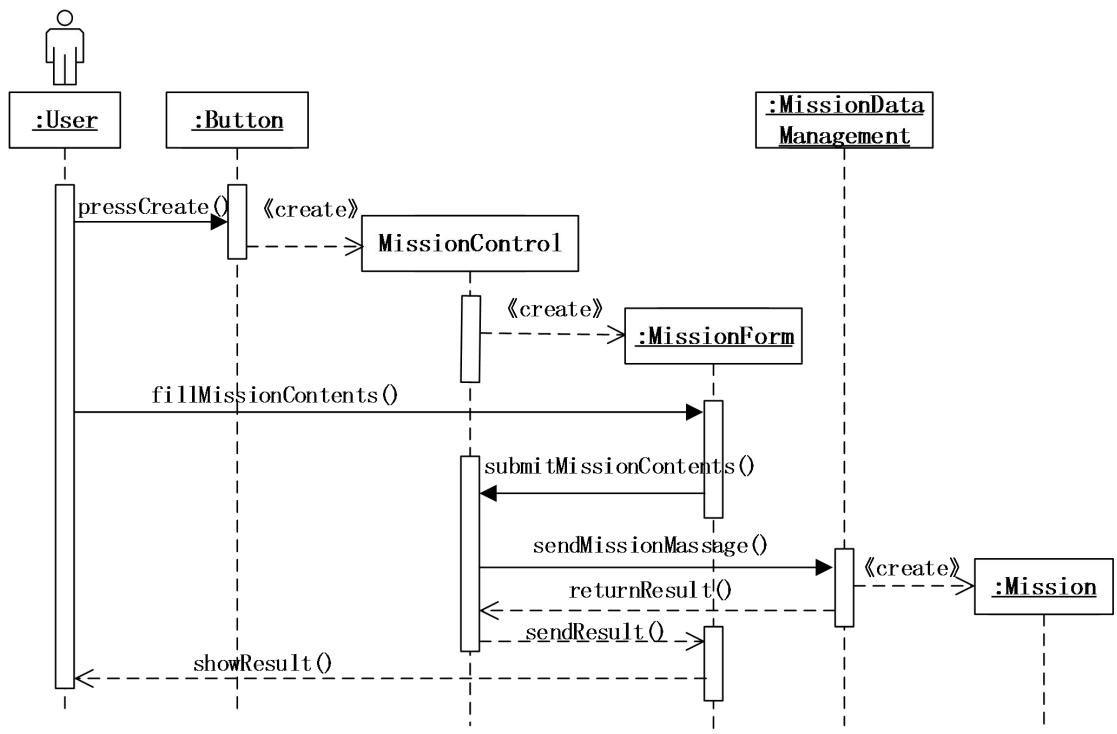


(3) 用例实现

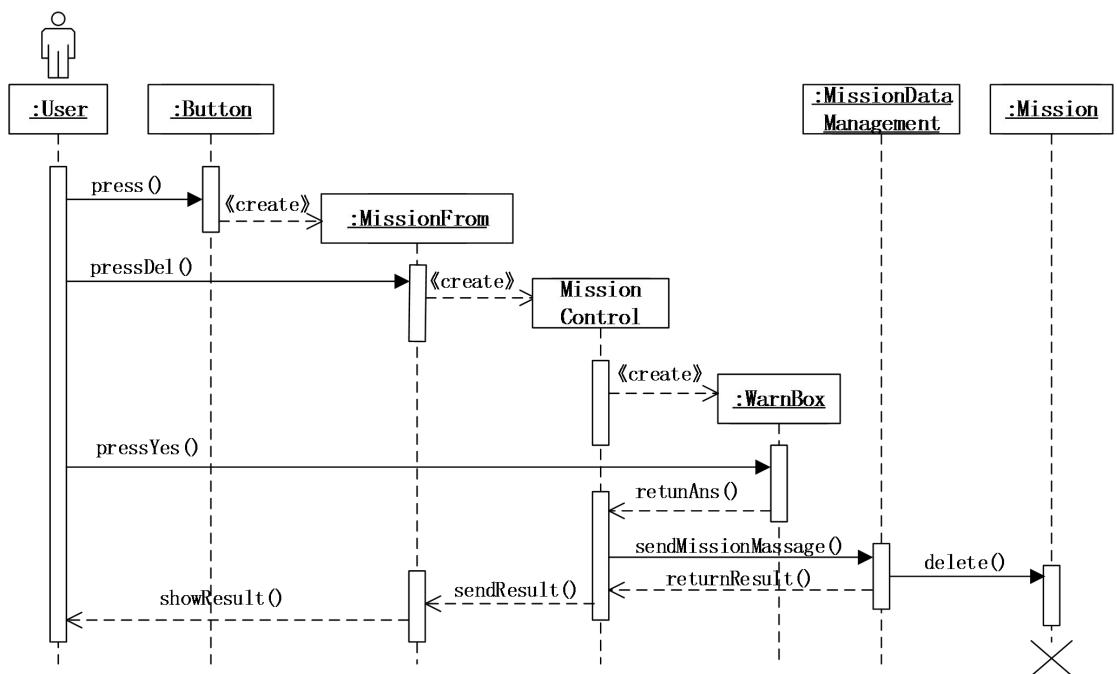
1. 注册任务管理系统账号用例：



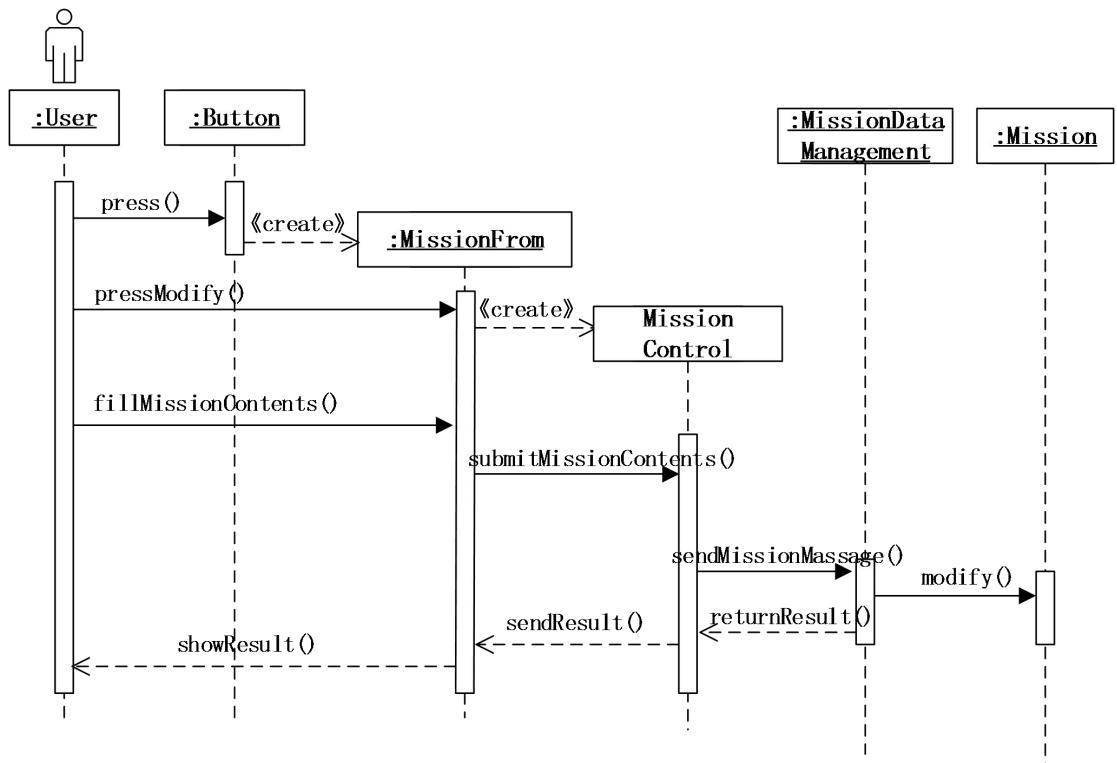
2. 增添任务用例：



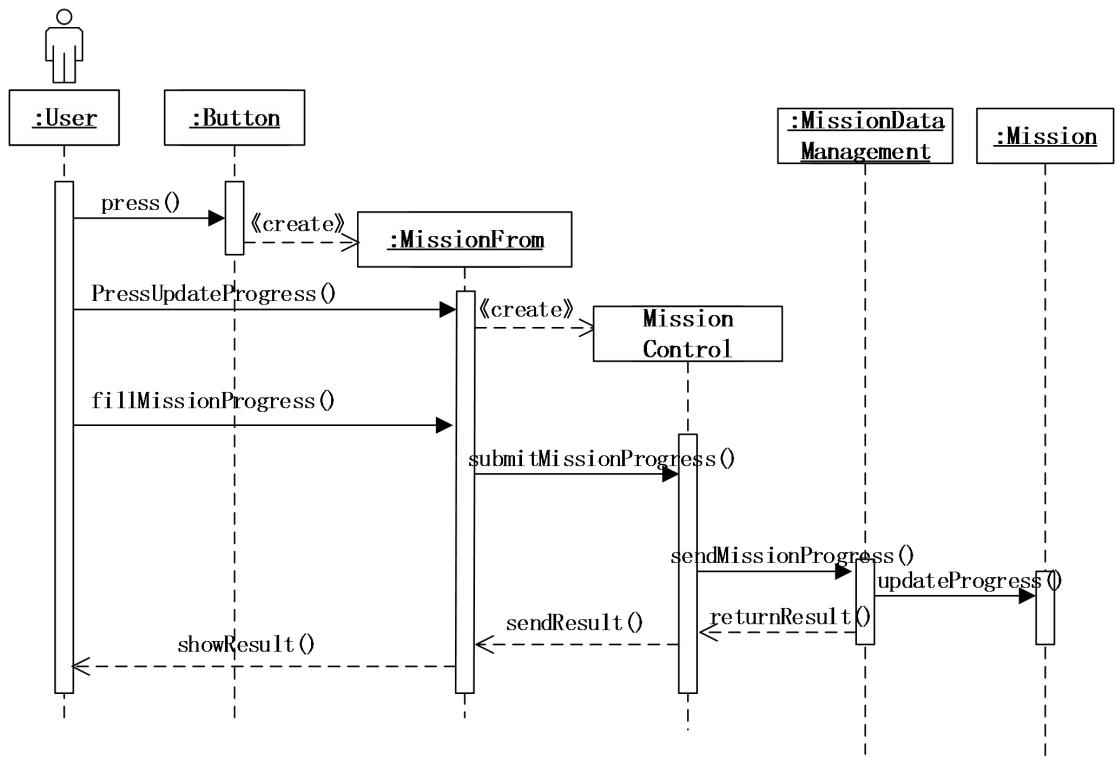
3.删除任务用例:



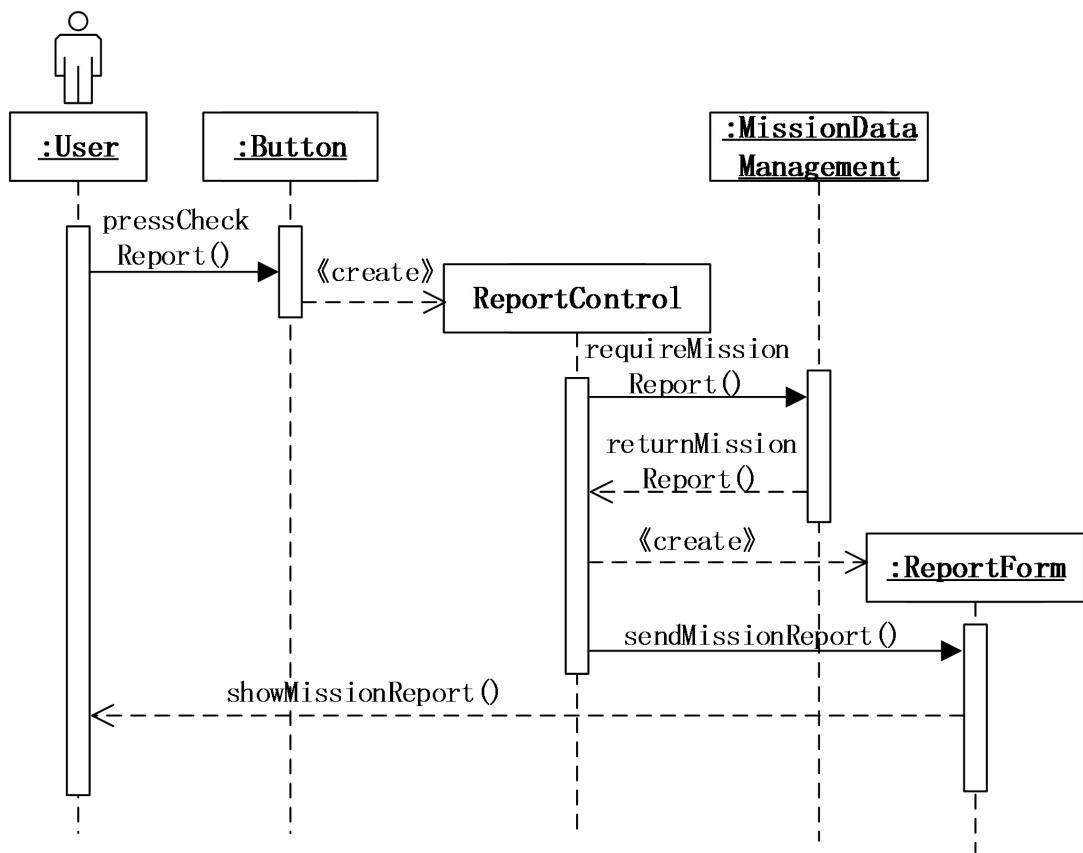
4.修改任务用例:



5.更新任务进度用例:

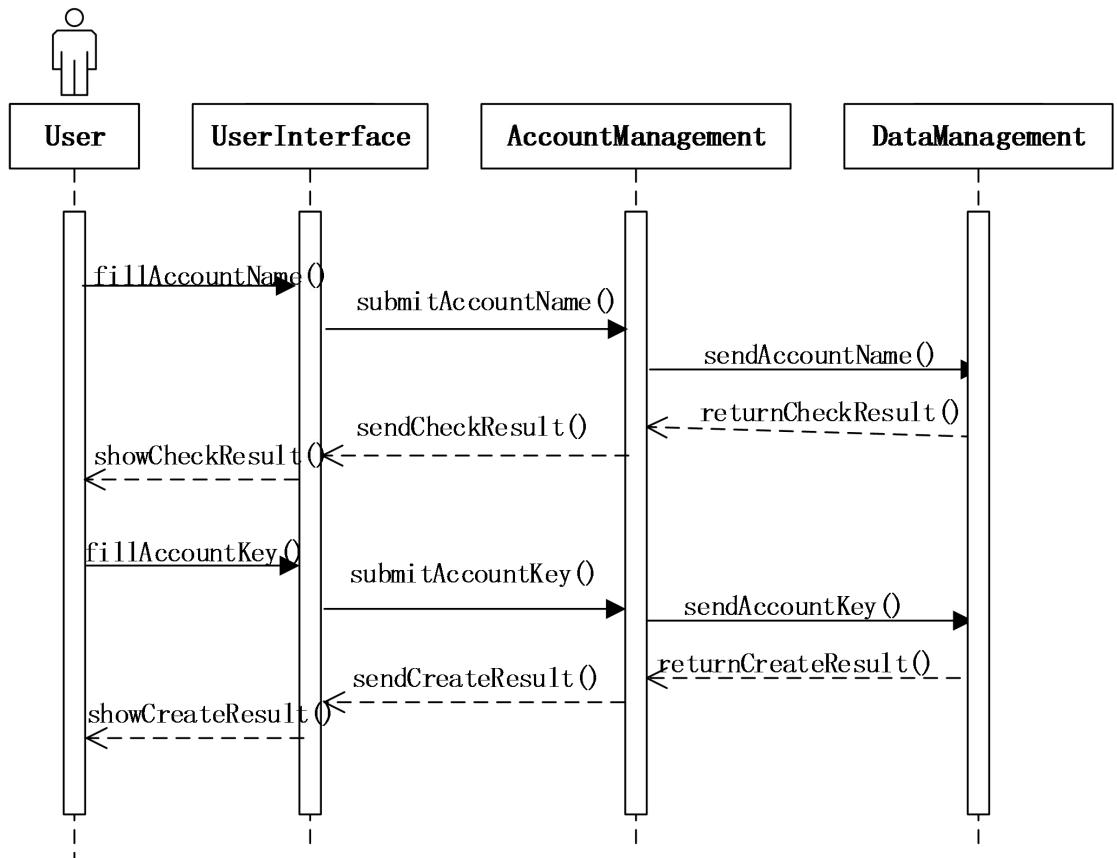


6.查看任务统计分析用例:

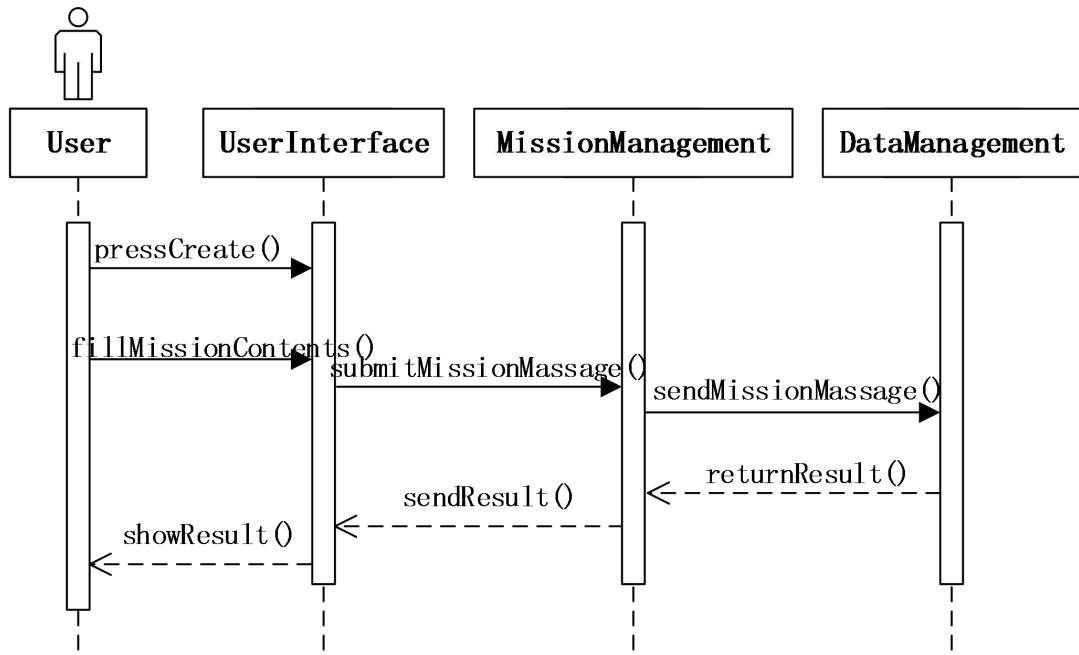


(4) 子系统协作

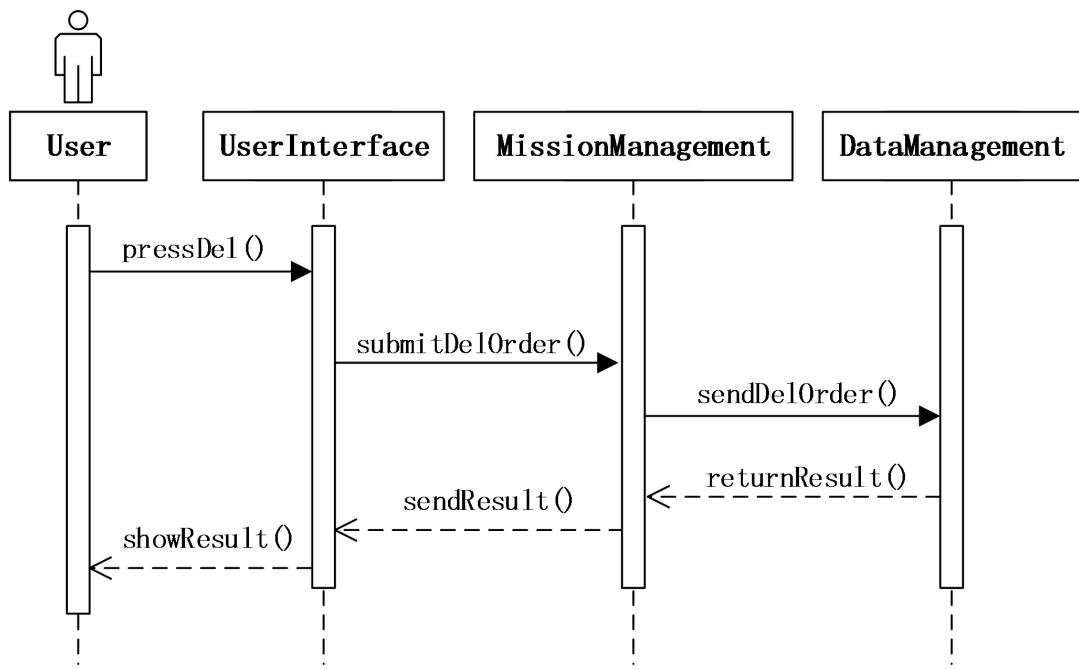
1.注册任务管理系统账号用例：



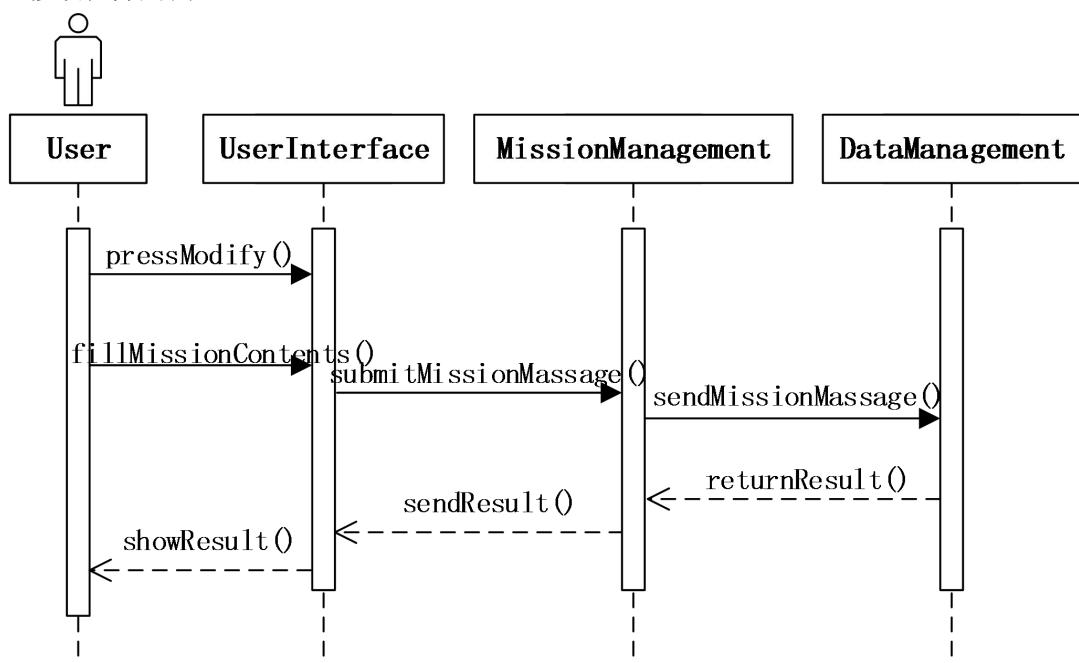
2. 增添任务用例：



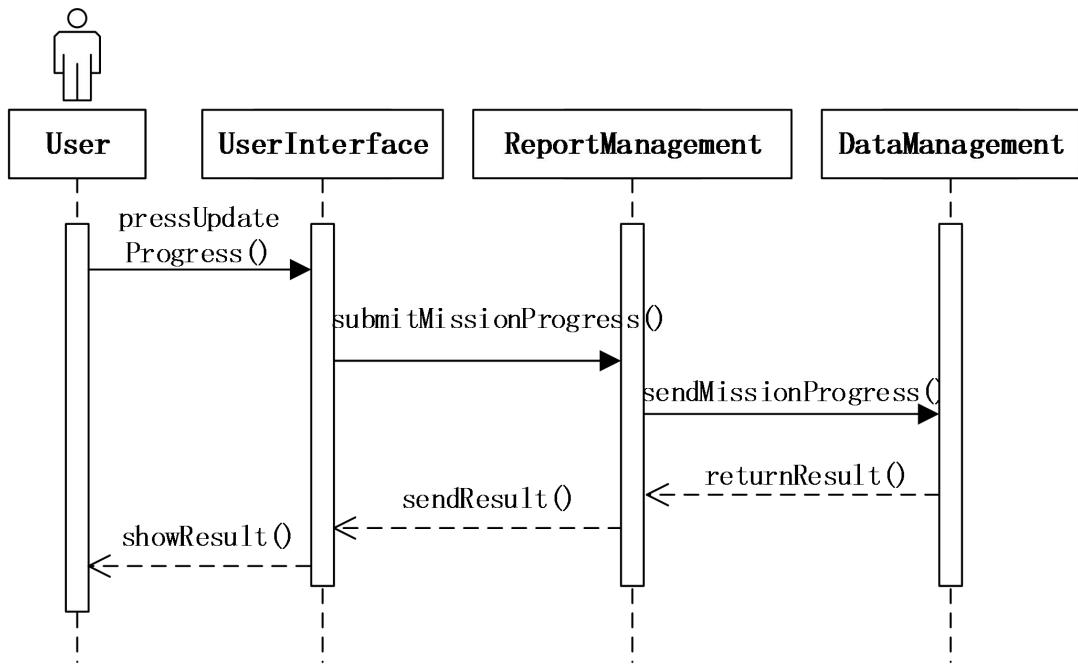
3. 删除任务用例：



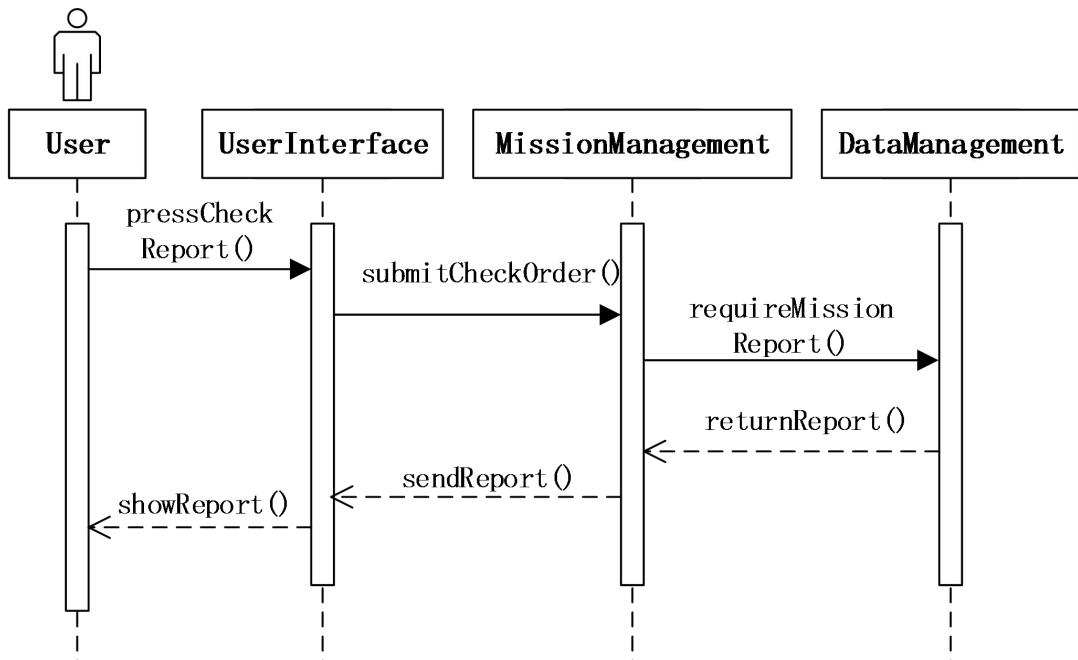
4.修改任务用例:



5.更新任务进度用例:



6. 查看任务统计分析用例:



4.4 系统运行视图

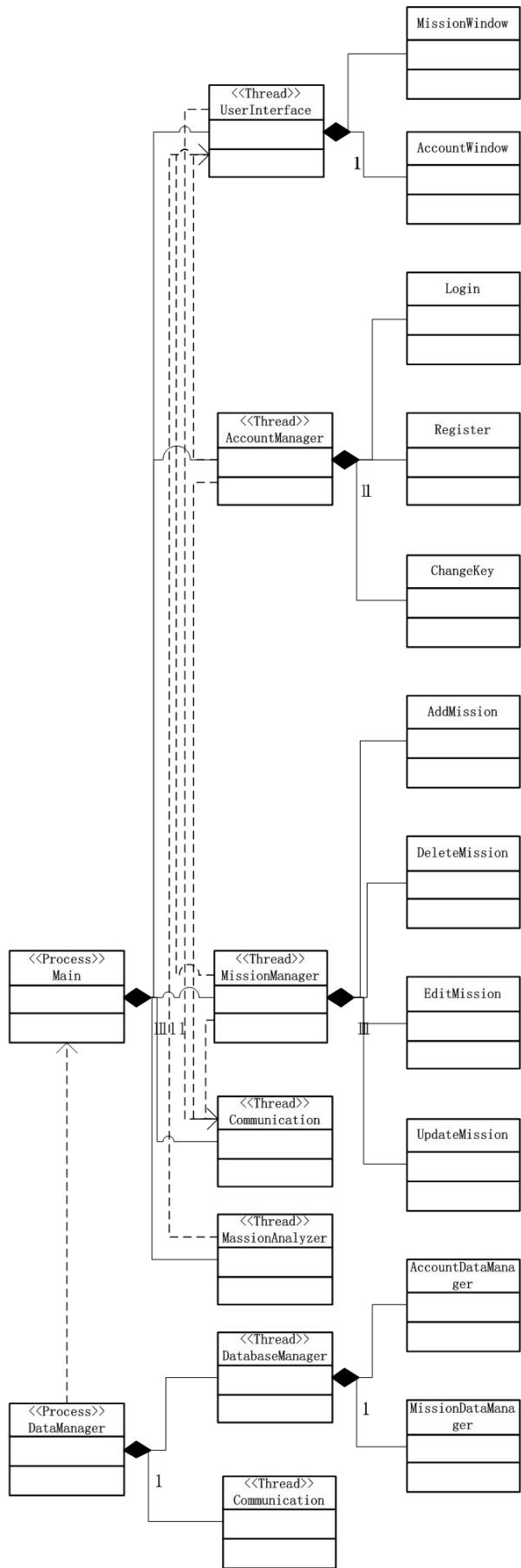
由于本系统主要部分为 Android 客户端，以和用户之间交互为主，因此本系统采用的控制流结构为事件驱动。用户按下按键等操作会调用相应回调函数，作为事件响应。其中，事件源为 Android 界面的组件（如菜单、按键、输入框等），侦听器为 android 组件自带的监

听器，事件处理程序为对应回调函数。

主要事件如下：

事件	响应
点击 app 图标	主进程启动
点击账号管理按钮	账号管理线程启动，切换到账号管理页面
点击管理任务按钮	管理任务线程启动，切换到任务管理页面
点击查看任务统计分析按钮	任务统计分析线程启动，切换到统计分析页面
存储/读取数据	数据管理进程启动，读取/写入数据到数据库

系统运行视图：本系统分为两个进程，分别为主进程、数据管理进程。主进程运行在客户端，在 app 启动的时候开始运行，用户退出的时候结束。数据管理进程运行在服务器上，负责数据存储和读取，生命周期为系统运行的所有时间。其中，主进程包含 UI 线程、通信线程、账号管理线程、任务管理线程、统计分析线程。运行视图如下图所示：



主进程与数据管理进程通过 TCP/IP 协议进行通信，交换数据。

4.5 系统实现视图

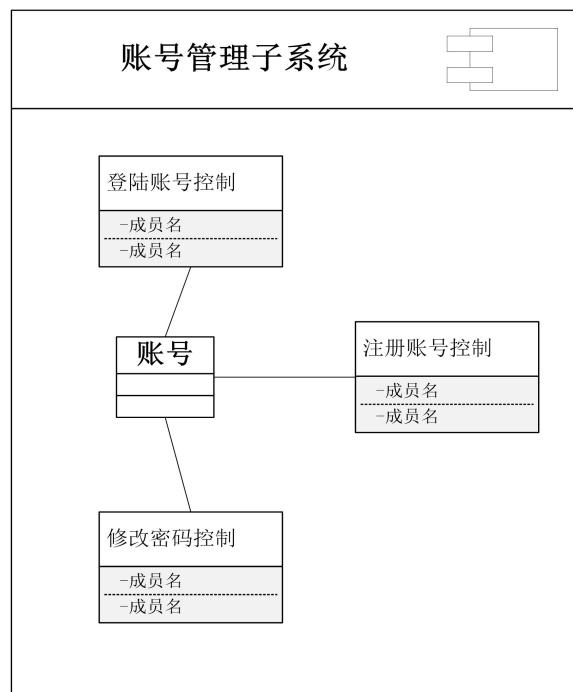
1. 系统开发模型：

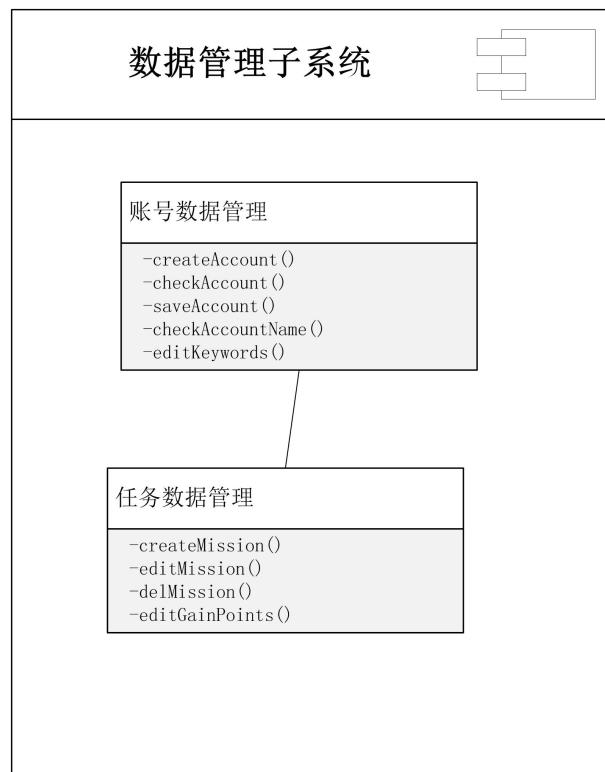
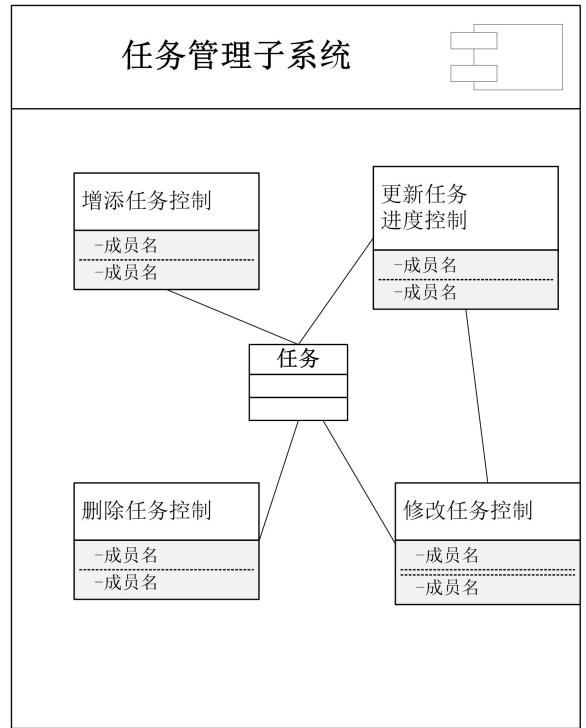
开发环境：AndroidStudio

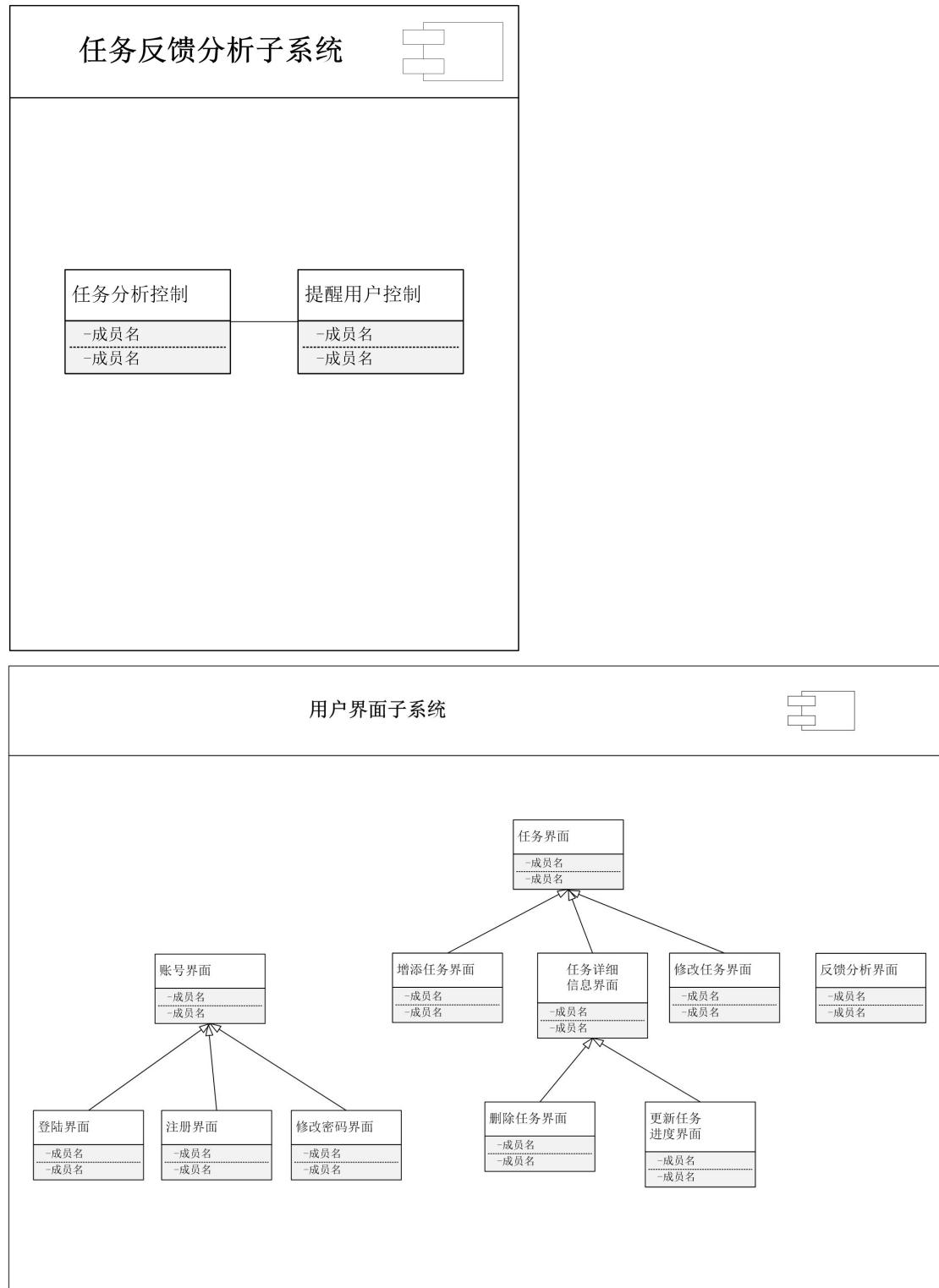
开发语言：Java

版本控制工具：Git

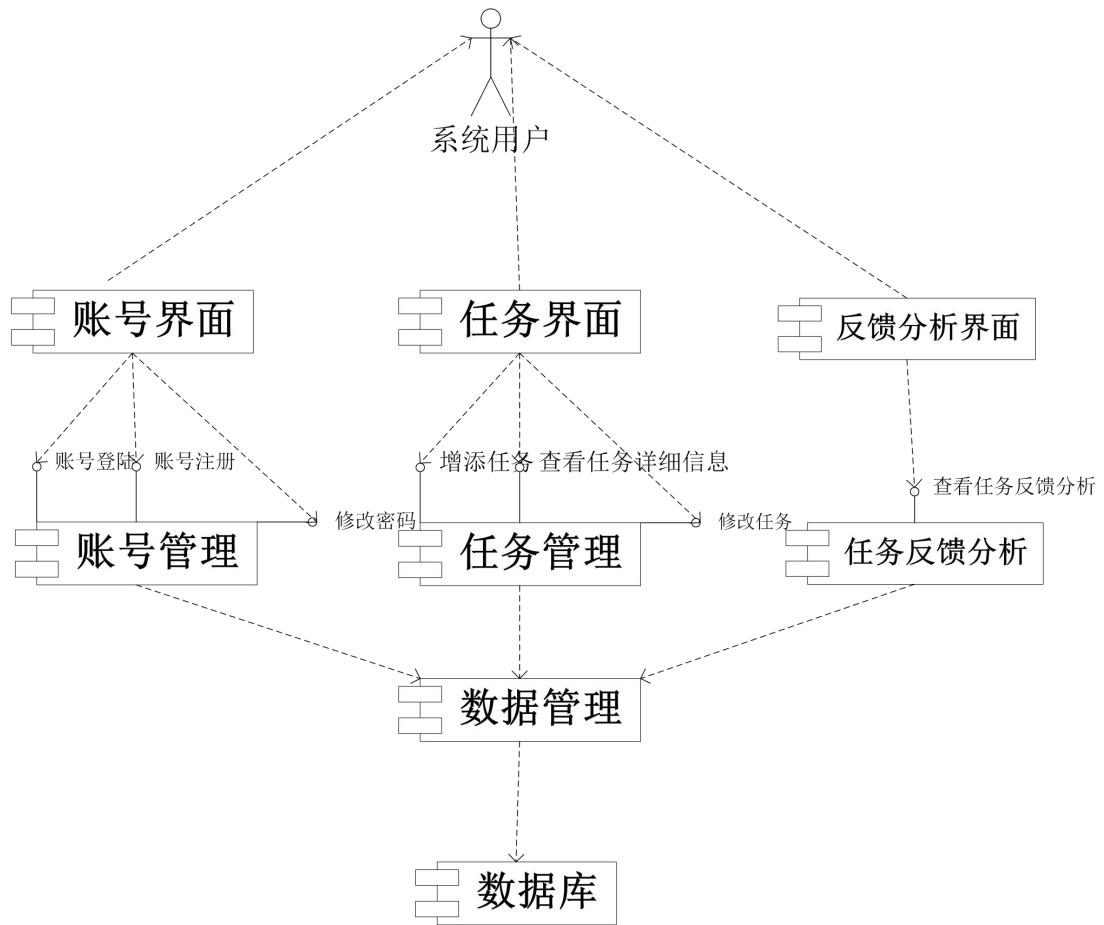
组件图：





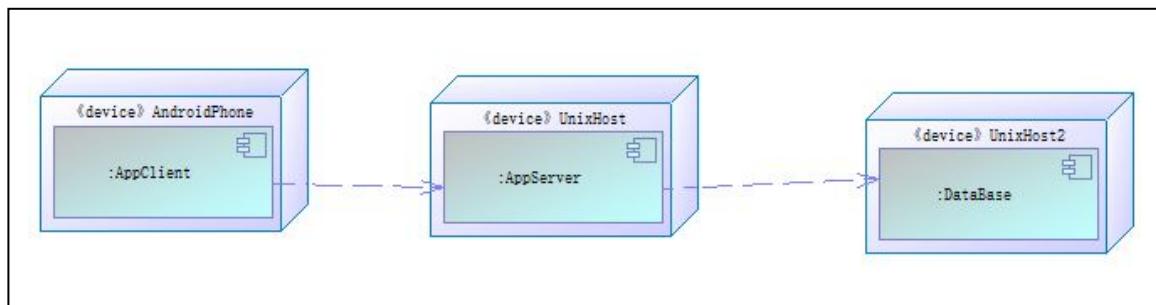


2. 系统实现模型：



4.6 系统物理视图

本章描述了部署和运行软件的物理网络（硬件）配置。



AndroidPhone

用户主要通过安卓手机客户端来访问系统，目前支持安卓 4.0 以上系统。

UnixHost

服务器采用 lampp 架构，运用 laravel 框架搭建后台并与数据库交互。

UnixHost2

数据服务器运行 mysql5.1 数据库。

4.7 边界条件设计

- **用例名称:** 启动系统
- **范围:** 系统用例
- **级别:** 子功能
- **主要参与者:** 系统用户
- **涉众及其关注点:**
 - 系统用户：希望能够方便、快捷地启动系统进行操作。
- **前置条件:** 手机中已经安装了系统
- **后置条件:** 系统成功启动，或者失败退出。
- **主流程:**
 7. 系统用户在手机应用界面单击系统应用图标。
 8. 系统进入启动过程并显示欢迎信息。
 9. 系统启动成功，显示主界面。
- **扩展流程:**
 1. 启动过程出错
 - 弹出对话框显示“启动失败，请重试”
- **特殊需求:** 无
- **发生频率:** 可能随时发生。

- **用例名称:** 关闭系统
- **范围:** 系统用例
- **级别:** 子功能
- **主要参与者:** 系统用户
- **涉众及其关注点:**
 - 系统用户：希望能够方便、快捷地关闭系统结束操作和浏览。
- **前置条件:** 系统以及成功启动

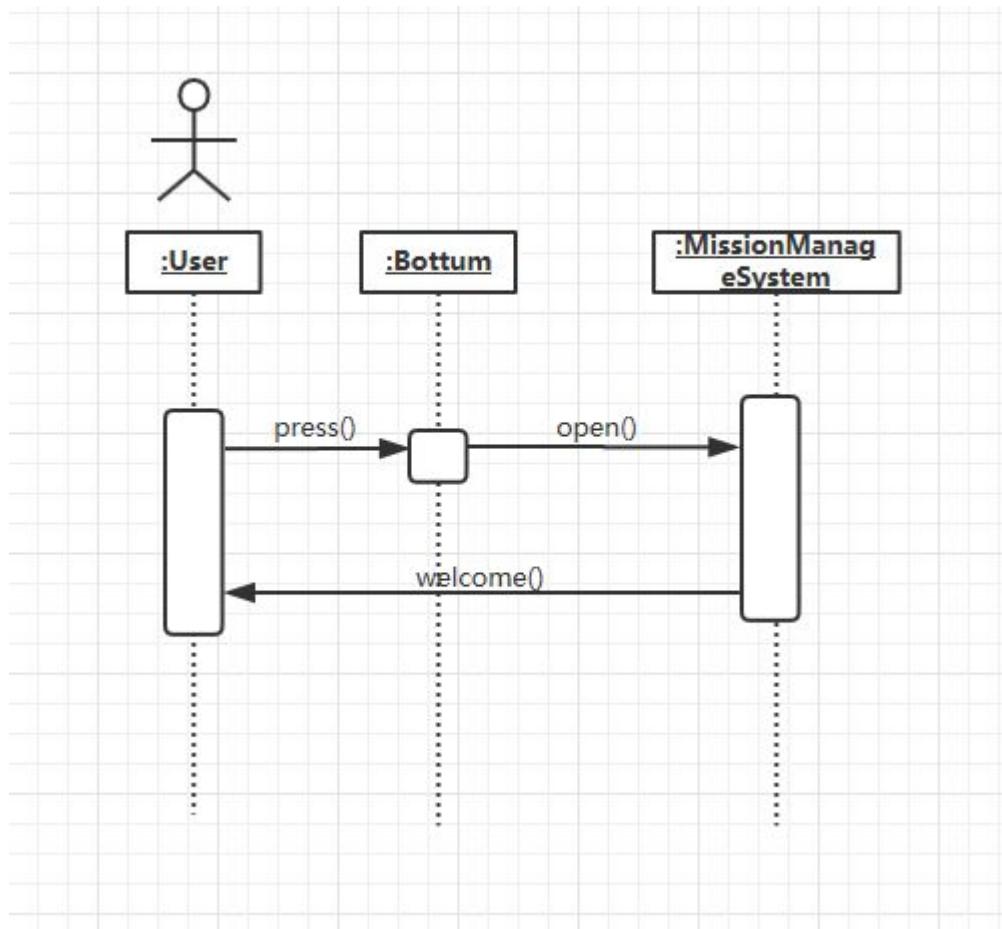
- **后置条件:** 系统关闭。
- **主流程:**
 1. 系统用户在系统主界面单击手机返回键。
 2. 系统弹出窗口询问“再次单击返回键关闭软件”。
 3. 系统用户再次单击返回键。
 4. 显示结束信息，软件关闭，退出系统。
- **扩展流程:**
 1. 退出过程出错
软件强制退出。
- **特殊需求:** 无
- **发生频率:** 可能随时发生。

- **用例名称:** 错误处理
- **范围:** 系统用例
- **级别:** 子功能
- **主要参与者:** 系统用户
- **涉众及其关注点:**
 - 系统用户：希望能够避免类似的错误发生。
 - 系统开发者：希望能够尽可能完善系统。
- **前置条件:** 系统运行中出现错误
- **后置条件:** 系统恢复
- **主流程:**
 1. 系统出错。
 2. 弹出窗口显示“啊哦，系统出错了，是否上传错误报告”。
 3. 用户选择“是”。
 4. 系统存储错误数据，并在下次交互信息的时候上传至服务器。
 5. 系统重启。
- **扩展流程:**
 1. 3处用户选择“否”

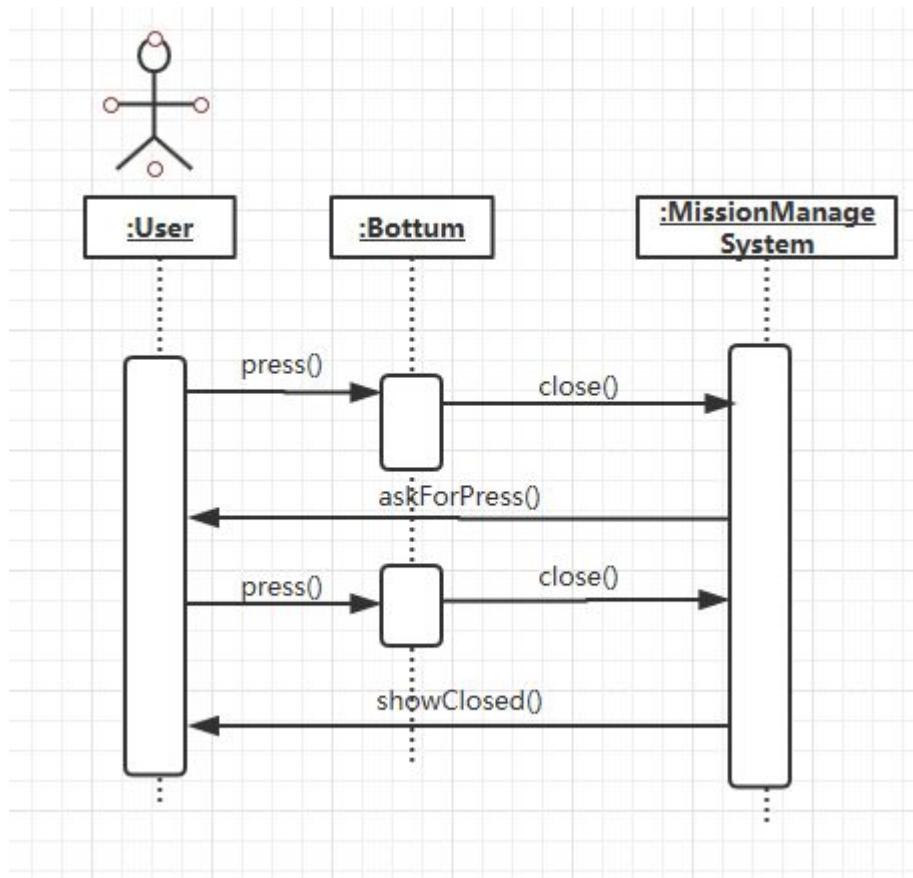
系统不保存数据并重启。

- 特殊需求：无
- 发生频率：可能随时发生。

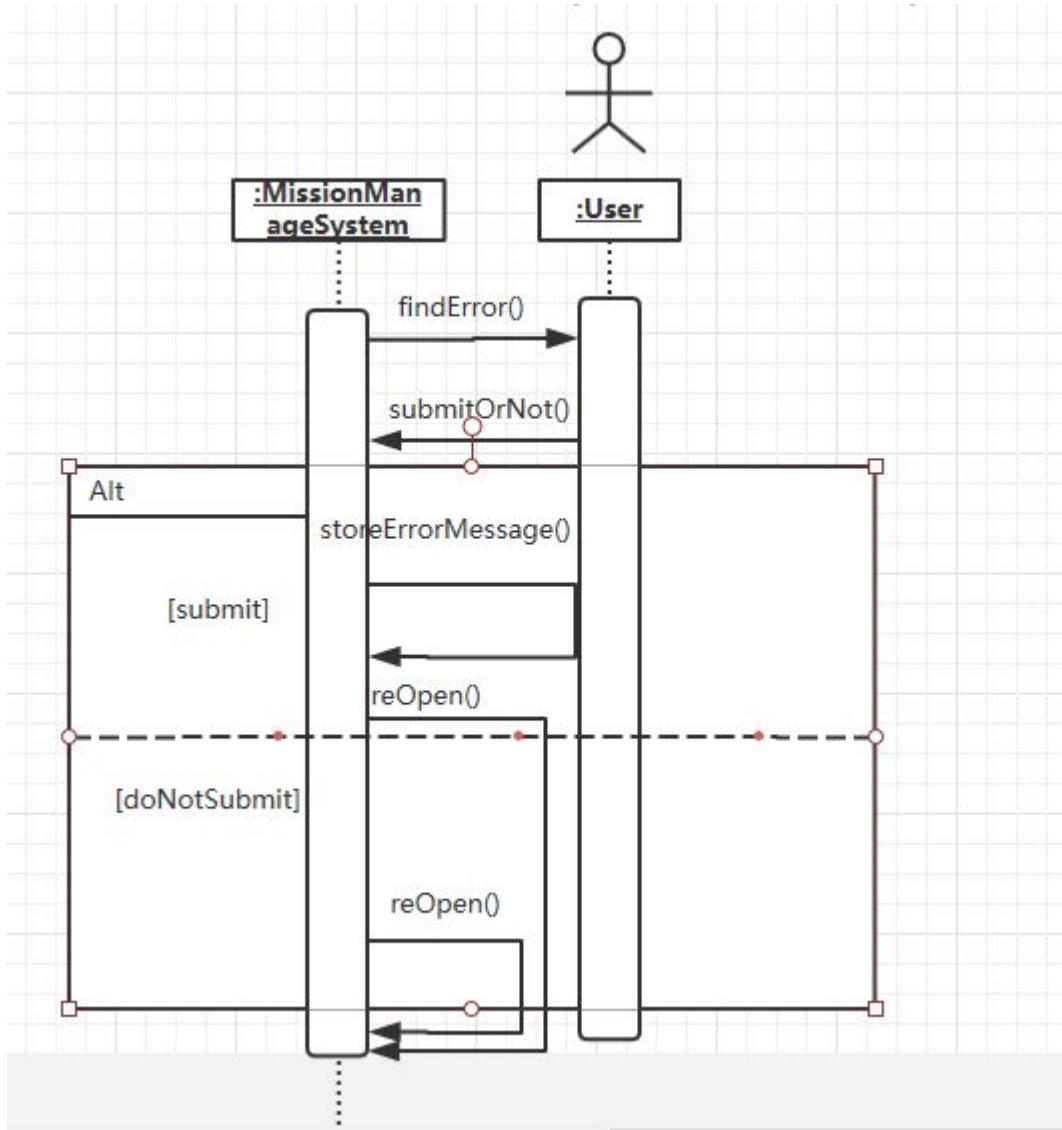
1.启动系统用例



2.关闭系统用例

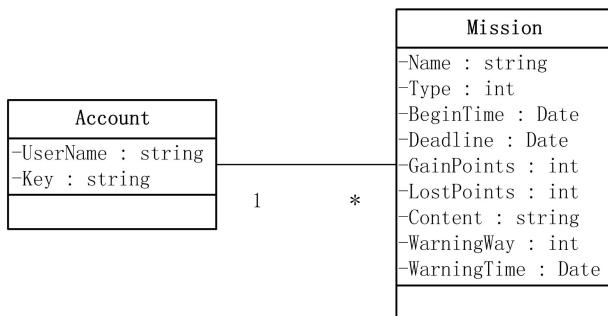


3. 错误处理用例



4.8 数据管理设计

需要持久化的数据有两类，分别是用户账号和任务。类图如下：



数据保存方式采用数据库保存。数据库系统实现整体数据的结构化，这是数据库的主要特征之一，也是数据库系统与文件系统的本质区别。数据库系统中，数据是面向整个系统，

数据可以被多个用户、多个应用共享使用，减少了数据冗余。数据库系统中，通过 DBMS 的两级映象实现了数据的物理独立性和逻辑独立性，把数据的定义从程序中分离出去，减少了应用程序的维护和修改。便于数据的存储和访问。

考虑到需要存储的数据量不是很大，也比较简单，采用 MySQL 数据库管理系统。

Account 数据表为：

```
account_tbl(
    account_id  INT NOT NULL AUTO_INCREMENT,
    user_name   VARCHAR(25) NO NULL,
    key VARCHAR(25) NO NULL,
);
```

account_id:int	user_name:text[25]	key:text[25]

Mission 数据表为：

```
Mission_tbl(
    mission_id  INT NOT NULL AUTO_INCREMENT,
    user        VARCHAR(100) NO NULL,
    name        VARCHAR(100) NO NULL,
    type        INT NOT NULL,
    begin_time  DATETIME,
    deadline    DATETIME,
    gain_points INT NOT NULL,
    lost_points INT NOT NULL,
    content     TEXT,
    warning_way INT NOT NULL,
    warning_time DATETIME,
);
```

mission_id:int	user:text[25]	name:text[25]	type:int	begin_time:date	deadline:date

gain_points:int	lost_points:int	content:text[1000]	warning_way:int	warning_time:date

4.9 其他设计

访问控制：用户仅能访问用户文件，无法访问系统文件。对于存储在数据库中的账户和任务信息，访问权限如下：

操作者	账号数据管理	任务数据管理
未登录用户	无权限	无权限
登录认证用户	读取数据权限、修改密码	读取、修改权限
系统管理员	管理权限	管理权限

用户认证方式：账号密码认证。

将用户关键信息（如账号密码等）存储在服务器上而不是本地，有助于统一管理，防止

信息泄露造成安全隐患。