面向对象分析与设计实例

——航空公司机票预定系统

**一、面向对象分析**

**1.1 需求说明**

**1.1.1 问题背景**

机票预订系统是某航空公司推出的一款网上购票系统。在信息化时代，为了方便用户购买、预定飞机票，便利出行，某航空公司拟开发一款网上购票系统，用户可以根据该系统方便快捷地查看机票信息，购票；航空公司也可以在其中展示、安排航班信息。

**1.1.2 需求分析**

该系统面向的对象又三类，分别是未登录的用户、已登录的用户和系统管理员。

未登录的用户：查看航班信息；

已登录的用户：查看航班信息，网上购买机票,查看行程,退订机票；

系统管理员：安排系统中的航班信息；

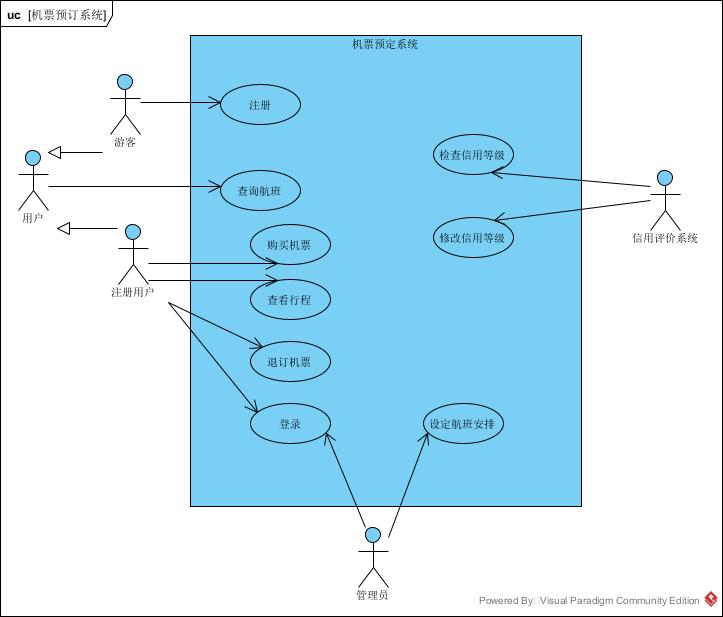
外部交互系统（信用评价系统）：当用户一个月之内退订两次及以上的机票时,需要降低该用户在信用评价系统中的信用等级。当信用等级过低时,则不允许该用户再次购买机票。

**1.2、建立对象模型**

对象模型是面向对象设计的基础，它能够将现实问题抽象为“类和对象”，描述了航空公司机票预定系统的静态数据结构，而对象模型也是之后动态模型和功能模型的基础。所以先为航空公司机票预定系统创建对象模型。

**1.2.1 用例分析**

从需求分析可以看出，系统的参与者主要有用户，系统管理员和信用评价系统三类，可以根据需求分析来画出用例分析，如图1所示：



**1.2.2 类的分类与设计**

图 1 机票预订系统的用例分析

由于该系统是一个完整系统中的子系统，规模较小，功能较为单一，所以不必要划分主题。根据之前建立的用例模型，我们可以将现实生活的具体事务抽象为类和类之间的关系。

购买机票涉及的主体有用户、机票、航班、订单，而管理机票则额外涉及到系统管理员。

在其中：每位用户可以对不同航班下订单，每个订单有几率产生一张机票，而每张机票属于一个航班，一个航班对应多张机票，系统管理员则管理多个航班。

根据以上的对应关系，可以设计出如下的类和属性：

* 用户：身份证号、密码、姓名、电话；
* 订单：订单号、订票时间、订单状态、订单航班号、订单用户；
* 机票：机票号、订单号、座位、机票状态；
* 航班：航班号、起飞站、起飞时间、到达站、到达时间、登机口、起飞状态；
* 管理员：航班号、舱位、票价、余票；

类和行为的分析如下：

* 用户：修改信息、订机票、注册信息、删除账户；
* 订单：生成订单、修改订单状态、获取支付情况、预定机票；
* 机票：生成机票、修改机票状态；
* 航班：修改航班状态、查询航班信息；
* 管理员：修改航班信息、查询航班；

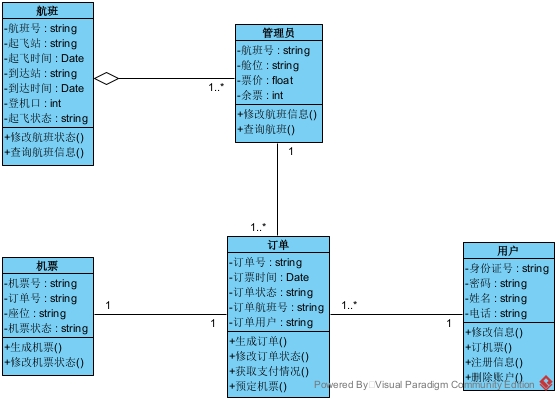
根据类的分类可以画出如下类图，在其中也确定了每个类的数据类型和属性，如图2所示。

图 2 机票预订系统类图

**1.3、建立动态模型**

**1.3.1 编写脚本**

1.3.1.1 用户登录

（用户）1、用户进入登录界面并填写登录表单

（系统）2、系统发送数据表单到数据库

（系统）3、系统确认用户信息的合法性并返回结果

（系统）4、系统显示登录结果

（异常处理）1a.访客输入了不合法的内容

（系统）1、返回提示并等待用户重新输入

1.3.1.2 用户注册

（用户）1、用户进入注册界面并填写注册表单

（系统）2、系统发送数据表单到数据库

（系统）3、系统确认用户信息的合法性并返回结果

（系统）4、系统显示注册结果

（异常处理）1a.访客输入了不合法的内容

（系统）1、返回提示并等待用户重新输入

1.3.1.3 用户预定机票

（用户）1、用户选择一架航班并预定机票

（系统）2、系统生成订单信息并等待用户付款

（用户）3、管理员将对应航班机票数-1并返回结果

（用户）4、用户付款

（系统）5、系统接受到付款请求并发送数据到第三方扣款机构

（系统）6、第三方扣款机构返回扣款信息

（系统）7、发送订单信息至机票管理员

（系统）8、系统将订票结果返回给用户

（异常处理）3a、用户取消付款

（用户）1、用户发送取消付款请求

（系统）2、将订单状态改为“已取消”并关闭等待支付的接口

（用户）3、管理员将对应航班机票数+1

（异常处理）5a、第三方扣款机构扣款失败

（系统）1、得到扣款失败的信息并发送给用户

（用户）2、等待用户再次付款并将订单信息改为“待支付”

（异常处理）3a、机票余座为0

（用户）1、管理员返回不可订票的信息

（系统）2、系统将信息返回给用户并将订单信息改为“预约失败”

1.3.1.4 用户使用机票

（用户）1、用户进入使用机票界面

（用户）2、管理员检查机票状态并返回机票信息

（用户）3、用户使用机票

（系统）4、系统将机票信息改为“已使用”

（异常处理）2a、机票状态为不可用

（用户）1、管理员检查机票信息不可用并返回异常报错

（系统）2、显示异常信息并等待进一步操作

1.3.1.5 用户退订机票

（用户）1、用户退订机票

（系统）2、系统进行退票审核

（用户）3、管理员将对应航班余票+1

（用户）4、信用评价子系统检查退票次数并修改信用值

（系统）5、系统向第三方机构请求退款

（系统）6、系统将订单支付费用按规则返还给用户

（系统）7、系统将机票状态改为“已退款”

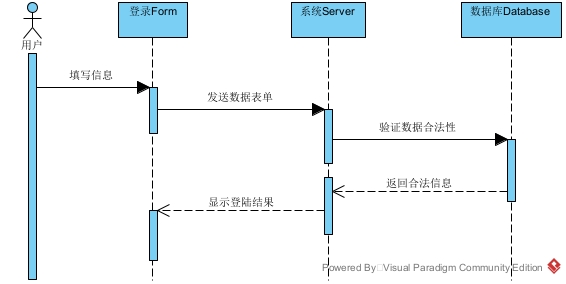
（异常处理）2a、机票不符合退票规则

（系统）1、系统返回失败信息并等待进一步操作

（异常处理）5a、请求退款失败

（系统）1、再次向第三方机构请求退款

**1.3.2 画顺序图**

根据脚本的交互流程，可以设计出每个行为的顺序图，如图3至图7所示：

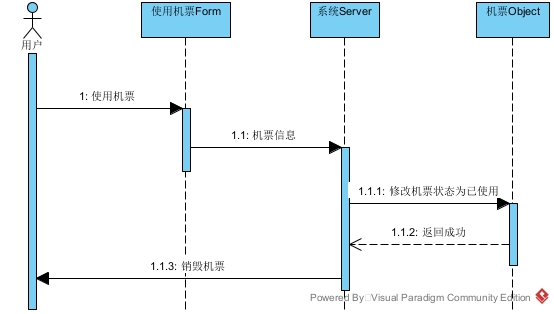


图 4 使用机票的顺序图

图 3 登录的顺序图

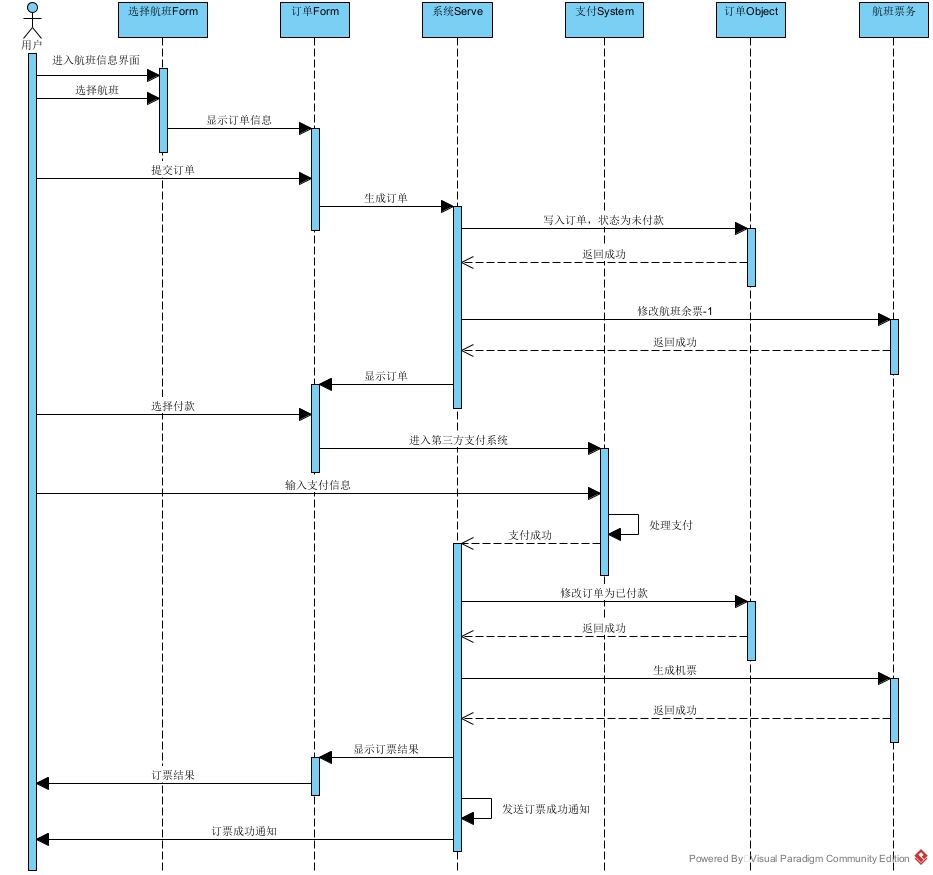
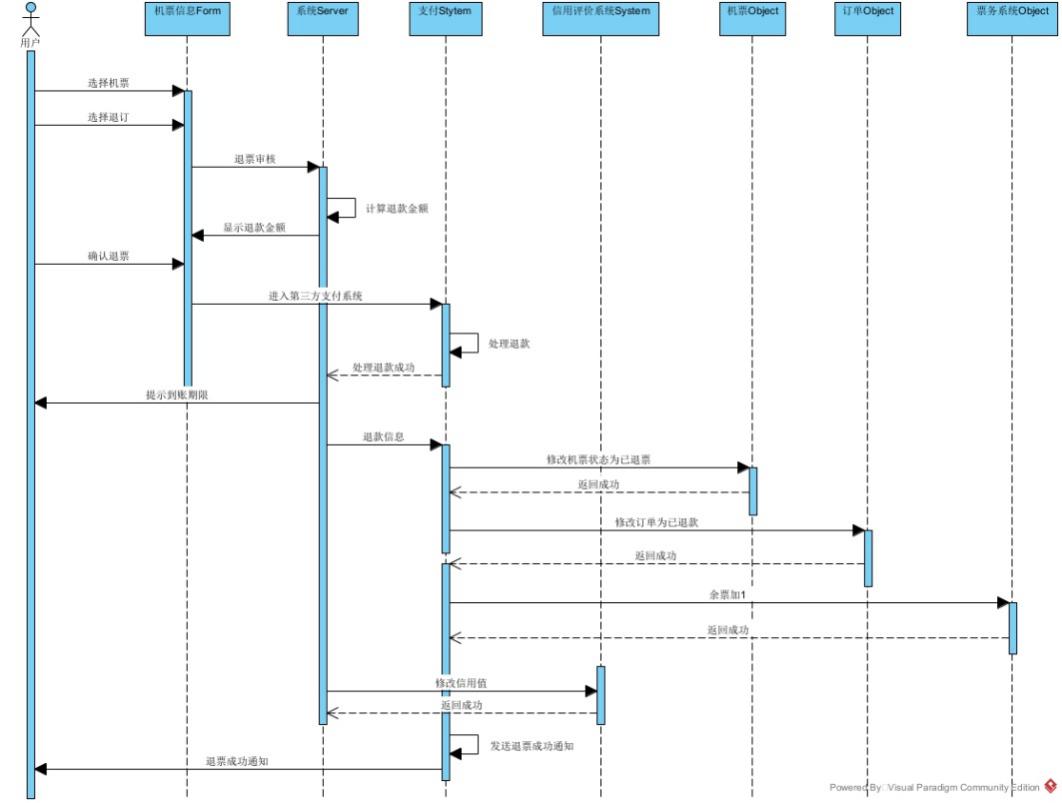


图 5 预定机票的顺序图

图 6 退订机票的顺序图

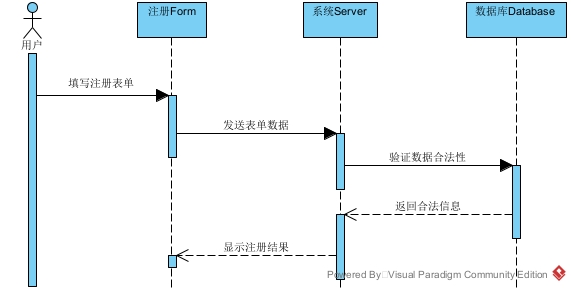


图 7 注册的顺序图

**1.3.3 画状态图**

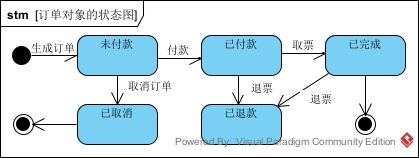
根据前文分析得出的脚本和顺序图，将一个复杂完整的世纪过程抽象为用例和对应的动作。接下来将这个完整的问题抽象为机构简单的子问题，分析每一个子状态机。可以得出对主要用例的状态图，如图8所示。

图 8 订单对象的状态图

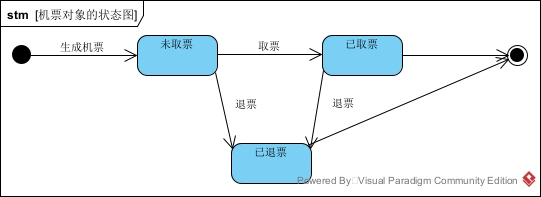


图 9 机票对象的状态图

**1.4、构建功能模型**

该系统的构成比较简单，因此只需要1层数据流。根据前文的分析，可知流程分析应如下：

用户向订单界面发送订票需求，订单界面向系统管理申请余票信息，管理者查看余票数量并-1，返回查找结果，系统管理返回订单结果，显示给用户订单内容并等待付款，如图9所示。

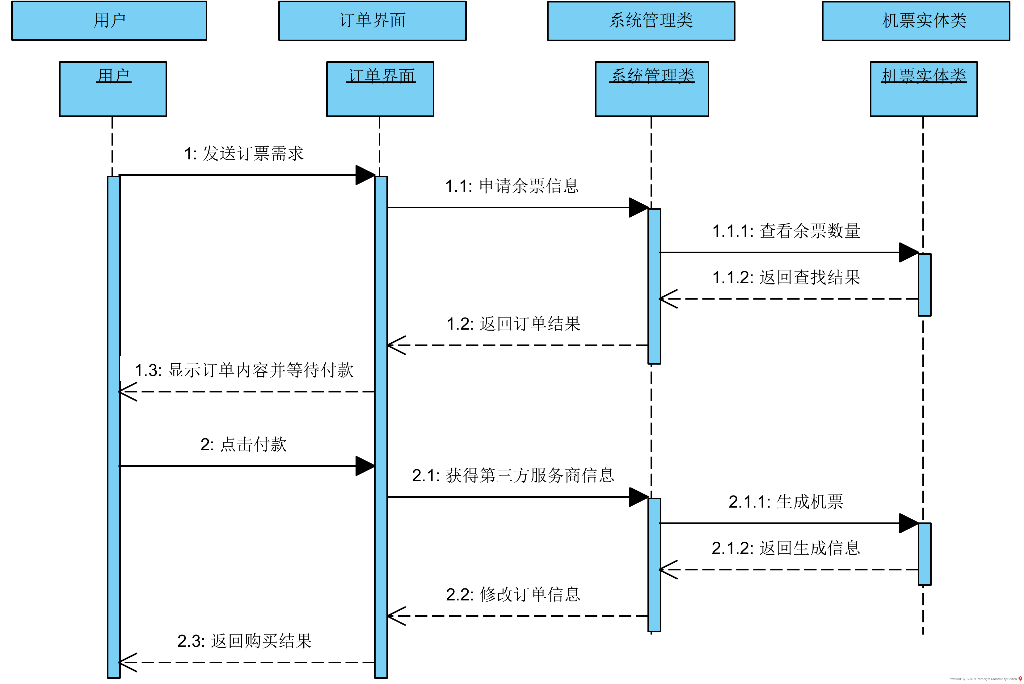
用户点击付款，订单界面发送信息到第三方服务商，并将返回信息发送给系统管理者，管理者查看结果后生成一张机票，系统修改订单信息为“已完成”，订单界面将购买机票结果返回给用户，如图10所示

图 10 购买机票活动的顺序图

**二、面向对象的设计**

**2.1、系统环境模型设计**

设计环境模型就是定义与软件进行交互的外部实体及它们之间的交互方式,外部实体包括人、其他软件系统和硬件设备等。

该系统为航空公司整体购票系统的一个子系统，所以①它需要有可供上级系统调用的接口。②其次，它也和其他的同级系统有交互，如信用评价系统，所以需要用调用同级接口和被调用的同级接口。③之后，它还会调用一下基本的服务，即封装好的子系统，如支付接口，所以需要有调用子系统的接口。④最后，它是给用户使用的，基于硬件设备驱动，所以需要有给用户和硬件的接口。

根据以上的分析，可以画出系统环境模型，如图11所示。

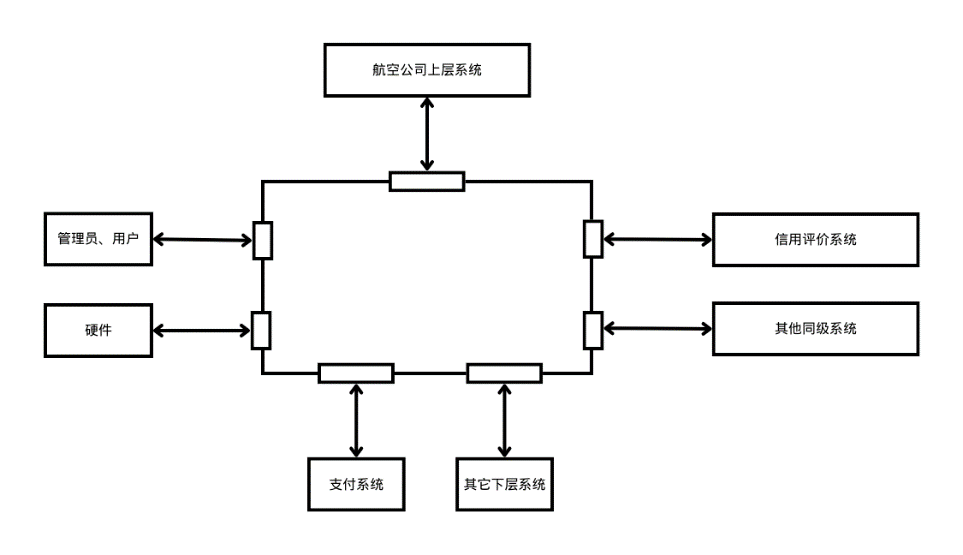


图 11 系统环境模型图

**2.2 软件体系结构设计**

软件有多种体系结构，如管道与过滤器风格、数据抽象与面向兑现风格、仓库风格等。而结合本系统的实际情况和复杂程度，选择C/S风格。

C/S结构是客户机/服务器(Client/Server,C/S)体系结构，它是基于资源不对等且为实现共享而提出的,是 20 世纪 90 年代成熟起来的技术。C/S 体系结构定义了工作站如何与服务器相连,以实现数据和应用分布到多个处理机上。C/S 体系结构有三个主要组成部分:数据库服务器、客户机应用程序和网络。

而C/S架构的分类和本系统的应用场景十分相似，可以很好地协调客户机和服务器数量不对等的情况，同时还可以将功能划分开，保证每部分功能的独立性。而且，采取C/S架构，将数据库独立出来，保证了软件的安全性、可靠性和可移植性。最后，采取C/S架构还可以并行地开发，也支持并行的访问，提高软件的使用体验。

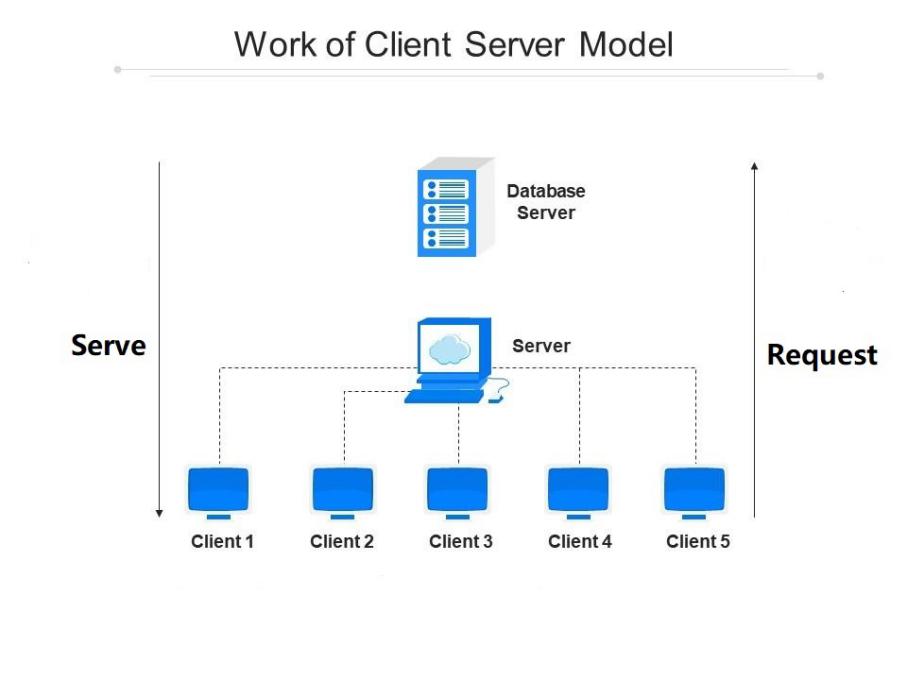
C/S结构的示意图如图12所示。

图 12 C/S结构示意图

**2.3 子系统设计**

大多数系统的面向对象模型,在逻辑上都由 4 部分组成。这 4 部分对应于组成目标系统的4个子系统,它们分别是问题域子系统、人机交互子系统、任务管理子系统和数据管理子系统。

**2.3.1 问题域子系统**

问题域子系统即为软件具体的功能实现部分，即软件的逻辑控制层。这部分的分析在本文1.3 建立动态模型中，已经结合类图和模型的分类详细探讨过，具体可以先实现顺序图中的每个方法，再根据每个函数所负责的功能进行拆分或者合并。

因为内容大部分相同，只需要将抽象出来的类进行复用，方法进行合并即可，所以在此不再赘述，设计时结合1.3的内容设计即可。

**2.3.2 人机交互子系统**

人机交互的主要途径是用户界面进行交互。在本阶段，会根据用户的需求将交互细节加入到界面设计中,包括各种选择菜单、按钮等对象,基于对象的操作,用户输入和显示输出信息的界面布局及操作细节。

主要的页面设计如下：

（1）登录界面

界面主要组件有用户名称、密码的输入框，确认登录的按键及注册的按键。

用户输入用户名和密码后，确认登录按键才会有效。当点击后，系统从输入框获取文本在后端进行验证，并返回结果。如果成功，则跳转到机票列表界面；如果失败，则显示失败信息。

用户点击注册按键后，会跳转到注册界面。

（2）注册界面

界面主要组件有用户手机号、密码、确认密码、验证码的输入框，发送验证码、确定和取消的按键组成。

用户将用户手机号、密码、确认密码的输入框填入有效信息后，点击“发送验证码”按键，会接收到一个验证码，并将其填入输入框。之后点击确认，界面会将数据发至数据库进行验证和添加。如果成功，则返回注册界面，如果失败，返回报错信息。

用户点击取消按键，则跳转到登录界面。

（3）机票列表界面

界面主要由起始地的搜索框、机票展示列表、搜索按键组成。

用户选择起始地后，点击搜索按键，机票展示列表便会显示出符合条件的机票列表信息。用户点击列表页的任意机票即可跳转对应的详情页。

（4）机票详情界面

界面主要由机票信息页、预定按键和返回按键组成。

用户点击返回按键返回机票列表页。

用户点击预定按键，则出发预定机票的流程，跳转到订单界面。

（5）订单界面

界面主要由订单信息页、确认付款按键和返回按键组成。

用户确认订单信息页无误后，点击确认付款按键进行付款，将订单发至后台并申请获得机票，成功后返回详情页面。

用户点击返回按键，返回机票详情页面。

（6）个人信息界面

界面主要由个人信息栏、历史订单栏、查看详细信息按键和修改信息按键组成。

用户点击查看详细信息可以查看自己的历史数据。

用户点击修改信息按键可以修改个人的部分信息。

**2.3.3 任务管理子系统**

在一个系统中，不同的任务有时并发执行，但有时却需要互相依赖。只有在任务管理子系统中将它们按照依赖关系进行设置，才可以保证系统运行的稳定性和正确性。下面将从事件驱动任务、时钟驱动任务、确定优先任务和关键任务等方面进行设计。

（1）事件驱动型任务

用户预订机票：用户在初始界面选择起始地等信息后，单击“查询”,进入机票列表界面。选好机票后，单击列表中的这一项进入航班详细信息的界面。用户点击“购票”,系统进入付款界面,顾客完成付款后,购票结束。

用户使用机票：用户点击“我的订单”进入订单界面，点击要使用的订单进入机票详情展示界面。当机票被取出或使用后，使用机票过程结束。

用户退订机票：用户点击“我都订单”进入订单界面，点击“退款”按键进行退款并进入退款界面。输入退款理由并确认发送后，跳转到订单详情页，并修改订单状态。当信息查验无误后退出订单详细页面，退票成功。

（2）时钟驱动型任务

机票预订系统中的时钟动型任务主要是为了保证资源能够被及时利用，尽量避免资源被无故占用。

在用户预订机票时，如果没有及时完成付款,系统可定时在后台取消此订单,以便将机票释放给其他旅客。

在用户预定号机票后，如果超过了起飞时间，则系统默认将机票作废。

用户订票成功后，系统在距离飞机起飞一段时间前发短信对用户进行提醒等。

（3） 确定优先任务和关键任务

高优先级任务：旅客预订机票时的付款、修改订单信息、取票时的打印机票、退票时计算金额等

低优先级任务：旅客退票时的退款任务

关键任务：为用户分配订单、预定机票时的付款、退票时的计算金额等

**2.3.4 数据管理子系统**

机票预订系统中的航班、旅客和订单等信息需要长期存储在计算机中,而这些数据要求具有完整性、安全性和事务并发操作的一致性,所以一般会采用关系数据库管理系统来存储和管理数据。

数据库中表的建立时根据1.2中抽象出的实体类及其属性来创建的，五个实体对应着五个基本表，其中航班管理的实体类为一对多，旅客发订单的请求为一对多，所以需要在这些相互关联，尤其是一对多的表中设立“外键”将它们关联起来，如1.2的类图所示。

为了维护数据管理子系统的安全性和稳定性，还需要加入数据库日志以及触发器和级联删除的逻辑，保证数据的一致性。万一出现了强制中断，也可以通过日志回滚恢复数据。

**三、参考资料**

1、吕云翔. 2022. 软件工程基础. 清华大学出版社

2、吕云翔，赵天宇. UML面向对象分析、建模与设计（第2版）. 清华大学出版社

【实例出处】3、尹志宇. 软件工程导论方法、工具和案例. 清华大学出版社. P186-P213

**四、总结与反思**

在本次小组作业中，小组成员通过自主学习《软件工程基础》中面向对象分析与设计部分的内容，并利用所学知识修改已有案例，深入理解了在软件设计阶段面向对象的思维方式，并初步具备了编写利用面向对象分析与设计方法的需求说明书的能力。

在作业完成过程中，小组成员感受最深的部分就是在实际编写一份文档时，老师上课所讲授的内容都在其中发挥了作用。在碰到无法理解的部分时，绝大多数情况都可以通过复习课程PPT内容、熟悉概念来帮助解决问题。因此，虽然在上课时觉得所学内容和实际应用有些距离，但是当真正上手去做时，才发现每一步都和课内知识联系紧密，比如顺序图的设计、类的划分等。

同时，大家还从《软件工程基础》这本书中巩固、学习了很多面向对象分析与设计的方法。书中的理论知识很充实，讲解的语言也比较容易让读者理解，同时每一个章节还配有相应的实例，让读者能够通过实例直观的理解知识。在完成本次作业时，主要的理论来源也是《软件工程基础》这本书。小组成员按照书中的分析步骤和要求，重新讨论了原有案例并做出修改，把原来缺失的部分补充上去，使其更加符合面向对象分析与设计的规范。

最后，小组成员还想根据自己的使用体验为《软件工程基础》这本书提出一些改进建议：书中在面向对象分析与设计部分的结构很好，都是先讲理论内容，再以实例作为总结。但是如果能以一个实例贯穿始终，并且按照理论部分所讲步骤去设计，可能更加方便读者理解、掌握各种分析方法的不同和分析设计过程中每个阶段的侧重点。希望这本书能够不断完善，为软件工程专业的学生提供更加规范严谨的教程。