

打车软件系统

体系结构设计文档

2.0

2019.12.06

唐宝钰
软件学院
2017211851

软件工程设计
2019 年秋

修订历史

日期	描述	作者	备注
2019/12/6	版本 1.0	唐宝钰	完成文本描述信息
2019/12/10	版本 2.0	唐宝钰	添加附图
2019/12/13	版本 3.0	唐宝钰	添加附录

文档批准

以下需求分析报告已经被以下机构人员批准并认可：

签字	打印姓名	标题	日期

目录

一、引言.....	5
1.1 编写目的.....	5
1.2 范围.....	6
1.3 术语定义.....	6
1.4 参考文献.....	8
二、体系模式概述.....	8
2.1 相关体系结构模式.....	8
2.1.1 C/S 模式.....	9
2.1.2 代理器模式.....	11
2.1.3 MVC 模式.....	12
2.2 体系结构描述视角.....	13
三、目标与约束，.....	14
3.1 功能目标.....	14
3.2 性能目标.....	14
3.3 体系约束.....	15
3.4 体系结构目标.....	16
四、用例视角.....	17
4.1 架构主要用例.....	17
4.1.1 信息管理与验证.....	22
4.1.2 下单叫车.....	23
4.1.3 付款.....	23
4.1.4 接单.....	23
4.1.5 收款.....	23
4.1.6 处理警报信息.....	24
4.1.7 监控异常交易.....	24
4.1.8 处理警报信息.....	24
4.1.9 管理用户信息.....	24
4.1.10 注册信息审核.....	24
五、逻辑视角.....	24
5.1 架构概述-系统分层.....	25
5.1.1 表示层.....	25
5.1.2 客户端层.....	25
5.1.3 业务逻辑层.....	26
5.1.4 数据管理层.....	26
5.1.5 分层的优点.....	26
5.2 架构概述-包/模块的组成分布.....	26
5.2.1 模块概述.....	26
5.2.2 模块分布.....	27
5.2.3 架构概述-模块中类的分布.....	28
六、进程视角.....	32

6.1 进程流程图.....	33
6.1.1 乘客打车进程流程图.....	33
6.1.2 司机接单进程流程图.....	34
6.1.3 业务监测人员工作流程图.....	36
6.1.4 系统管理人员工作流程图.....	37
.....	37
6.2 进程有关类图.....	38
6.2.1 乘客打车进程有关类图.....	38
6.2.2 司机接单进程有关类图.....	39
6.2.3 业务监测人员进程有关类图.....	40
6.2.4 系统管理人员进程有关类图.....	42
七、部署视角.....	43
7.1 智能手机.....	43
7.2 客户端.....	43
7.3 无线访问点.....	43
7.4 以太网.....	44
7.5 服务器.....	44
7.6 数据库.....	44
八、规格和性能.....	44
8.1 性能与效率.....	44
8.1.1 用户总量.....	44
8.1.2 用户同时使用该软件时.....	44
8.1.3 乘客同时发出打车需求时.....	44
8.1.4 司机接收订单时.....	44
8.1.5 业务监测人员处理警报时.....	45
九、质量需求.....	45
9.1 可靠性.....	45
9.1.1 导航信息的可靠性.....	45
9.1.2 付款系统的可靠性.....	45
9.1.3 系统可靠性.....	45
9.2 易用性.....	45
9.3 安全性.....	46
9.4 可维护性.....	46
9.5 可移植性.....	46
9.6 可用性：.....	46
9.7 工程需求：.....	47
9.8 数据库需求：.....	47
十、附录.....	47

一、引言

本章将介绍该体系结构设计文档的编写目的、涉及范围、参考文献、该文档涉及到的名次术语解释以及该文档的基本组织结构。

1.1 编写目的

该文档是针对打车软件系统编写的体系结构设计文档。该文档是基于对打车软件系统的全面需求分析，在明确了打车软件系统所要应该具备的基本功能、基本性能之后编写的文档。

本文档旨在清楚阐述打车软件系统的总体结构（包括逻辑设计、物理结构等），细致分析打车软件系统的体系结构需求（包括约束条件、设计遵循的标准、非功能性需求等），最终给出打车软件体系结构设计的解决方案并分析建模，并进行体系结构的质量分析和评估。

本文档作为产品立项和产品开发过程的参考文档，给出了打车软件系统详细的功能要求，系统功能块组成及联系，进程部署和硬件要求等，有益于提高软件开发过程中的能见度，便于软件开发过程中的控制与管理。此体系结构设计文档是进行软件项目设计开发的基础，也是编写测试用例和进行系统测试的主要依据，它对开发的后续阶段性工作起着指导作用。同时此文档也可作为软件用户、软件客户、开发人员等各方进行软件项目沟通的基础。

本文档的预期读者有：

软件用户：从本文档了解到软件系统的功能和整体结构。

开发人员：从本文档中了解到软件系统预期的功能需求、性能要求以及架构标准，并以此为依据对系统进行设计和开发。

测试人员：从文档中了解到软件系统需要测试哪些部分，以及各部分的测试标准。

维护人员：根据本文档中确定的体系结构进行软件系统维护。

项目经理：根据本文档预估软件系统开发的大致时间成本和技术人员需求，制定相应的时间流程和开发计划。

其他人员：该文档作为撰写其他系统文档的参考，为系统开发、测试、维护、转移等各个阶段的相关文档编写提供依据。

1.2 范围

1. 系统名称：打车软件系统。
2. 系统目标用户：出租车/快车司机，乘客。
3. 系统开发目的：解决在流量大，交通秩序混乱或较为偏僻的地点，乘客打车难，出租车/快车司机接单难的问题。
4. 系统组成：该系统应当分为三个部分，分别为面向用户的移动端手机软件、面向业务监测人员的业务监测网站、面向系统管理人员你的信息管理网站。
5. 系统功能：系统对于不同的使用者提供不同的功能：
对于司机，系统提供司机接单，导航，收款等基本功能；
对于乘客，系统提供下单打车，付款，获得导航信息，一键报警等基本功能。
对于业务监测人员，系统提供接收并处理警报信息，接收并处理举报信息，实时监测业务状况发现异常等基本功能。
对于系统管理人员，系统提供查看并管理用户数据，用户信息审核等基本功能。
6. 文档内容：本体系结构设计文档概括地描述了打车软件系统的主要功能与总体应用结构，说明了系统的总体设计策略，给出了体系结构设计的解决方案并分析建模，最后进行体系结构的质量分析和评估。
7. 文档应用范围：本软件体系结构设计文档适用于打车软件系统的总体应用结构，撰写目的是满足打车软件系统的质量要求和可信赖性要求，以及为系统未来的测试、移交、维护、运行和更新升级等提供统一参考。

1.3 术语定义

Table 1.1—术语解释表

术语	解释
系统	为方便叙述，用“系统”来代替“打车软件系统”。
项目	为方便叙述，用“项目”来代替“打车软件系统开发项目”。
用户	这里特指注册使用该软件的乘客、出租车司机和快车司机，不包括系统管理人员和业务监测人员。
客户端	供用户使用的安装在手机上的客户端软件，是本系统的一个组成部分，但是独立运行。
司机	这里指这册使用该软件的所有司机，包含出租车司机和快车司机。
乘客	注册并使用该打车系统的乘客。
系统管理员	管理本系统用户信息，对系统进行日常维护的管理员。
业务监测人员	指对业务订单进行监控，发现并处理订单异常，实时处理举报和警报的监测人员。
功能性需求	功能性需求规定开发人员必须在产品中实现的软件功能，用户利用这些功能来完成任务，满足业务需求。
非功能性需求	非功能性需求是指依一些条件判断系统运作情形或其特性，而不是针对系统特定行为的需求。
用户需求	关于系统服务和约束的自然语言加上方块图表述。为客户撰写。
用例	是软件工程或系统工程中对系统如何反应外界请求的描述，是一种通过用户的使用场景来获取需求的技术。每个用例提供了一个或多个场景，该场景说明了系统是如何和最终用户或其它系统互动，也就是谁可以用系统做什么，从而获得一个明确的业务目标。
体系结构	软件体系结构是具有一定形式的结构化元素，即构件的集合，包括处理构件、数据构件和连接构件。处理构件负责对数据进行加工，数据构件是被加工的信息，连接构件把体系结构的不同部分组合连接起来。

Table 1.2—缩写解释表

缩写	名词解释
----	------

USPMS	Urban Shared-Parking Management System 城市共享停车管理系统
E-R 图	Entity Relationship Diagram 实体-联系图 提供了表示实体类型、属性和联系的方法，用来描述现实世界的概念模型。
UML	Unified Modeling Language 统一建模语言或标准建模语言 为软件开发的所有阶段提供模型化和可视化支持，由需求分析到规格，再到构造和配置。
SDL	Specification and Description Language 规格和描述性语言 描述一个现实世界中特定事件的发生过程的时序关系以及步骤。
DFD	Data Flow Diagrams 数据流图 数据流图从功能的角度对系统建模，追踪数据的处理有助于全面地理解系统，数据流图也可用于描述系统和外部系统之间的数据交换。
C/S	Client/Server 客户端/服务器
MVC	Model-view-controller 模型-视图-控制器 是软件工程中的一种软件架构模式，把软件系统分为三个基本部分：模型、视图和控制器。目的是实现一种动态的程序设计，使后续对程序的修改和扩展简化，并且使程序某一部分的重复利用成为可能。
USE CASE	用例图 用例图是指由参与者、用例以及它们之间的关系构成的用于描述系统功能的静态视图。

1.4 参考文献

[1] 王安生.软件工程化[M].北京：清华大学出版社，2014

[2] Craig Larman.UML 和模式应用(原书第三版)[M].北京：机械工业出版社，2006

二、体系模式概述

本章对打车软件系统采用的基本体系架构模式进行规定和概述。并简要概述本文档描述体系模式采用的不同视角。

2.1 相关体系结构模式

本节简要介绍软件系统采用的三种模式。

2.1.1 C/S 模式

2.1.1.1 选用 C/S 模式的原因

在打车软件系统中，需要乘客的订单信息实时播报到范围内的司机对应的手机软件上，需要司机的位置信息实时同步到乘客的手机软件上，要求业务监测人员能够实时监控所有交易信息，要求导航信息和位置信息等公共信息得到同步的更新。

该系统要求为每个用户提供的手机端的软件都能够实时接收到来自其他用户的实时信息，并能够做到导航信息，位置信息的实时更新与同步，以此来满足软件的功能需求和用户的基本需要。因此软件选择使用的基本模式必须支持大量信息的共享和调度的同步。

该系统要求不同用户的手机软件彼此独立，并在不同的进程中运行。两个部件之间的通信并不一定完全对等。一个发起通信请求，期待另一个提供服务。即，多个客户端部件虽然使用同一个服务，但确期望产生出不同的结果。

2.1.1.2 C/S 模式结构描述

客户端-服务器(C/S---Client-Server)模式是满足以上两种要求的一种结构模式。明显区别出客户端和服务端，客户端请求服务器上的信息或服务。服务器优化如何为多个客户服务。客户端和服务端都必须具有密安性、事务处理和系统管理的功能。

本系统选择采用三层体系结构，如图 2.1 所示，

- 客户端层：负责数据的表现，接收用户的请求，控制用户界面。
- 应用逻辑层（或称应用服务层）：负责实现应用逻辑（或业务逻辑）。他既给客户端提供服务，也会依据客户端服务请求向第三层（数据）服务器提出服务请求。
- 数据服务层：负责向第二层提供服务，例如数据库中的数据存储、与老系统

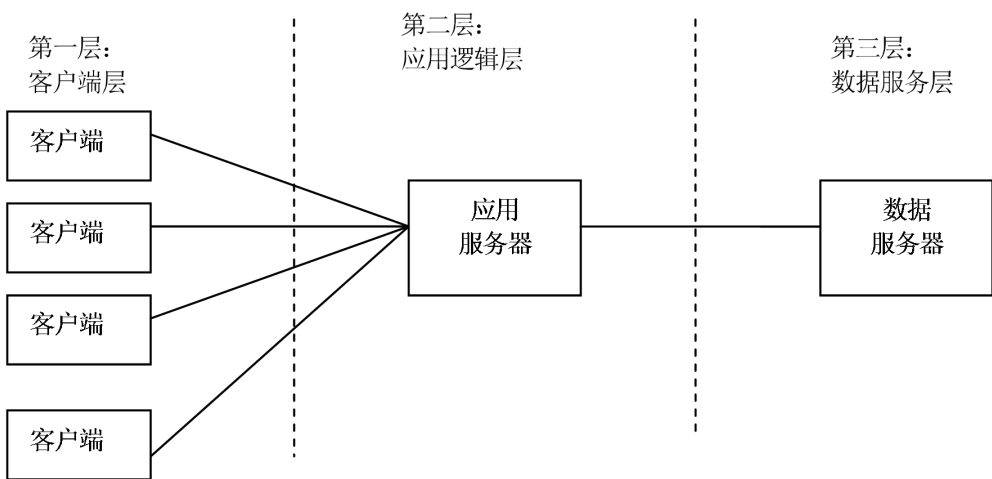


Figure2.1—三层体系结构示意图

的的连接等。

2.2.1.3 C /S 模式特征

C/S 模式有如下特征：

1) 分割：

表现层、业务层和数据处理逻辑被分割到不同的层面；

2) 各层之间异步通信：

层与层之间的通信是异步的“请求-响应(request-reply)”。请求是单方向的，从客户层开始，经过 Web 和业务逻辑层，再到数据管理层。每层都等待其它层的处理响应。

3) 部署灵活：

分层结构不限制多层应用的部署方式。所有层可以运行在一台机器上，或者，每层部署到一台独立的机器上。

4) 标准化：

分层可以实现各层的标准化，例如，在 Web 应用中，客户端可以使用不同厂家的浏览器，兼容地对不同厂家 Web 服务器访问。

2.1.1.4 C/S 模式对质量属性的影响

选用 C/S 模式对打车软件系统的质量属性的可能影响如下：

Table 2.1—C/S 模式对质量属性的影响

质量属性	对质量的影响
可使用性	每层的服务器可以部署多个服务器，互为备份，因此，一个服务器出现故障，其它服务器仍可使用。虽然会降低服务性能，但仍可用。
故障处理	如果客户端与服务器的通信失败，手机客户端和应用服务器可以实现透明的失效备援。因此客户端的请求可以重新定位到运行着的备份服务器上，而客户并不知晓。并不影响客户的使用体验。
可修改性	分割增强了可修改性。表现层、业务和数据管理逻辑清晰地得到了封装。每个层面都有其内部逻辑，修改不会影响其它层。

性能	这种结构的性能已经得到了证明。关键的问题是要考虑每个服务器支持的并发线程数量、各层之间的连接速度、以及数据传递的速度。对于分布式系统来说，降低了为完成每个请求所需要的层与层之间的调用时间。
可伸缩性	各层中的服务器可以有备份，多个服务运行在同一个或多个不同服务器上，体系结构的规模可以得到很好地提升。在实际中，数据管理层往往会成为系统能力的瓶颈。

2.1.2 代理器模式

2.1.2.1 选用代理器模式的原因

分布式软件系统开发者面临的许多困难比单处理器软件更多。一种原因是网络传输是不可靠的。另一种原因是，要将异构部件集成为一个有条理的应用系统，并使之可以有效地使用网络资源。如果开发者不能很好地处理这些问题，就会降低系统的运行效率。

2.1.2.2 代理器模式结构描述

代理器(Broker)模式将应用系统与分布式系统的通信功能分割开来。代理器隐藏和协调对象或系统部件之间的通信。一个代理器由客户端的请求器(Requestor)和一个服务端的调用器(Invoker)组成。请求器发出调用，调用器负责调用目标的远程对象操作。通信路径上的两端各有一个装配器(Marshaller)操纵请求的转换，并将编程语言本地数据类型转换为能在传输媒介上传送的字节组。代理器模式的示例图如图 2.2 所示。

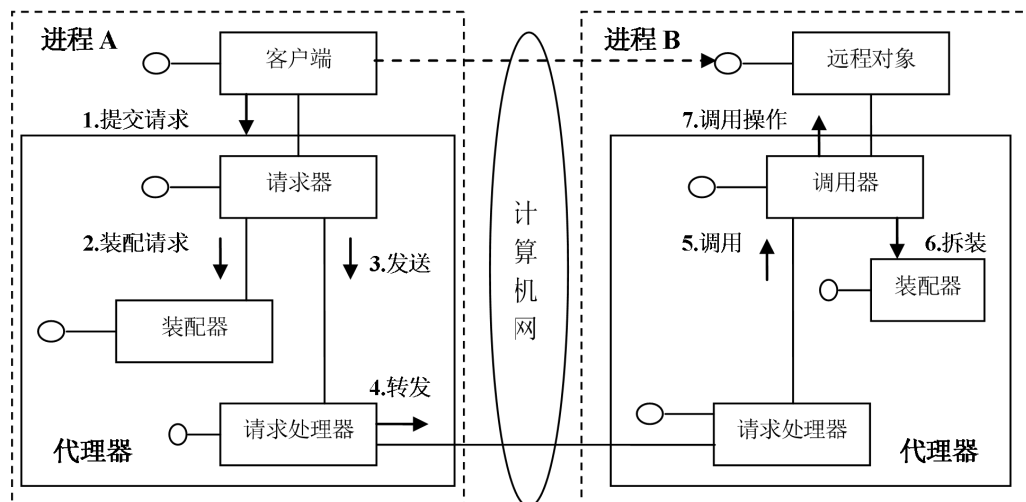


Figure 2.2—代理器模式示例图

2.1.3 MVC 模式

2.1.3.1 选用 MVC 模式的原因

一个系统往往需要提供多个用户界面,而每个用户界面可能只需要反应一部分应用数据。当这些数据改变时,要能够自动和灵活地反应到不同的用户界面上。这就需要能够很容易地修改其中的一些用户界面,而不需要修改与数据相关联的应用逻辑。

2.1.3.2 MVC 模式的结构描述

MVC 模式将系统划分为三个部分:

- 1) 一个模型(Model): 封装应用数据及其对这些数据的操作, 与用户界面独立开;
- 2) 一个或多个视图(Views): 向用户展示指定的数据;
- 3) 一个控制器(Controller): 与每个视图关联起来, 接收用户的输入, 并将它翻译成对 Model 的请求。

View 和 Controller 组成了用户界面。依据 MVC 模型,更改所有 View 和 Controller 的通知机制可以用“发布-订阅(Publish-Subscribe)”模式实现。所有控制器和视图从模型接收到的订阅都要发布出通知。用户只通过 View 及其 Controller 进行交互,而不依赖于 Model,反过来,将更改情况通知给所有的不同用户。MVC 模式实例如下图 2.3。

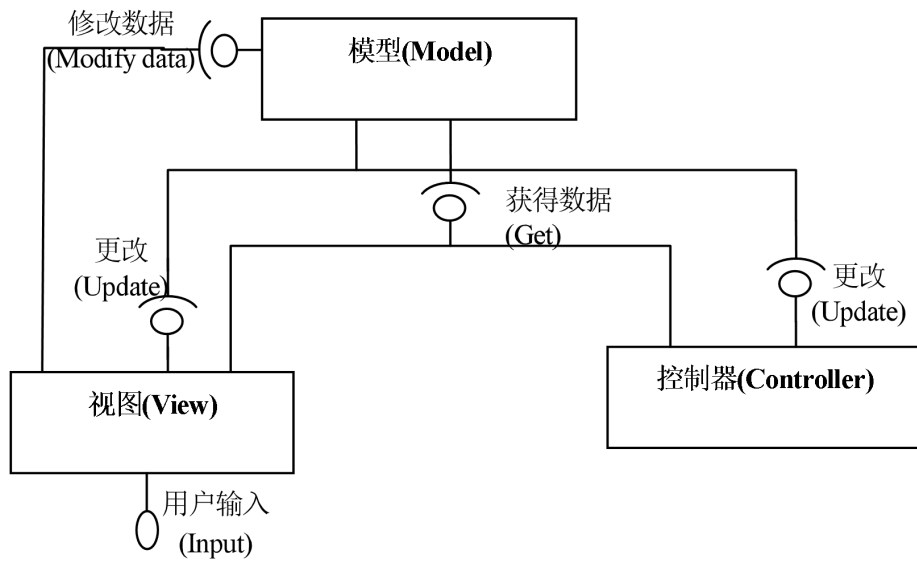


Figure 2.3—MVC 模式实例图

2.2 体系结构描述视角

该体系结构文档为打车软件系统提供了一个可供参考的通用开发体系结构。

本文档根据“4+1”视图模型，从五个视角分析呈现体系结构：

用例视图：客户、需求分析人员、测试人员关注的体系特性。见文档第四章。

逻辑视角：最终用户、设计者关注的系统功能。见文档第五章

进程视角：系统集成人员关注的系统性能，伸缩性，吞吐量等。见文档第六章。

实现视角：程序员们关注的软件项目的组织与管理。见文档第七章。

部署视角：系统运维工程师关注的系统拓扑结构，交付流程，安装方式等。见文档第八章。

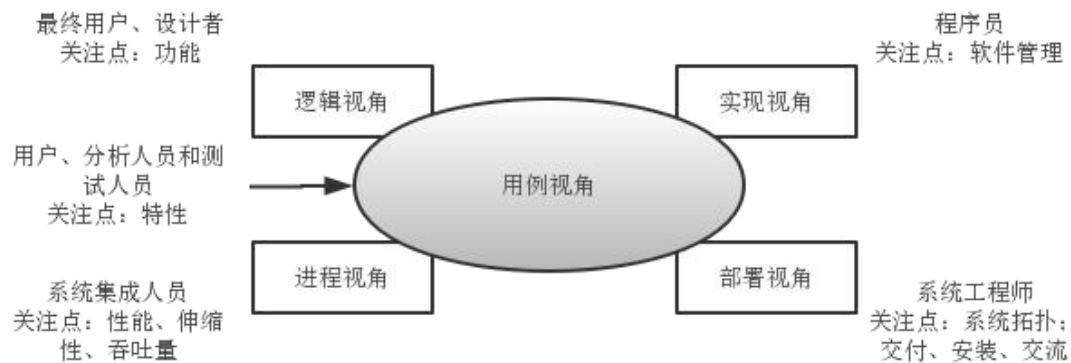


Figure 2.4—视图模型

三、目标与约束，

3.1 功能目标

1) 对于注册本软件的乘客

- 能够通过本软件在自己的所在地发出打车需求。
- 在范围内的司机接下订单后，能够看到接单司机和其车辆的基本信息
- 在交易结束后，乘客能够通过本软件进行付款，不需要跳转到其他软件。
- 交易结束后，乘客能够对司机进行匿名评价。
- 该软件能够为乘客推荐公交车路线
- 该软件能够为用户提供附近共享单车位置和推荐骑行路线。
- 该软件为用户提供快速报警功能和司机举报机制。

2) 对于注册本软件的出租车/快车司机：

- 能够接收到范围内的乘客的订单信息，并抢单。
- 能够准确获得乘客的所在位置，并得到去往乘客所在地的导航路线。
- 交易结束后，由司机设置付款金额，自动根据乘客选择的免密支付方式进行收款。
- 交易结束后，允许司机对用户进行举报。对损坏车内设施等极端行为，核实后会要求用户进行赔偿。

3) 对该系统的业务监测人员：

- 能够实时得知注册该系统的每辆出租车/快车的位置。
- 能够实时接收的用户的报警信息。
- 能够接收到乘客和司机的举报信息并进行审核给出反馈

4) 对系统的管理人员：

- 能够管理乘客和司机信息。
- 能够对申请注册该软件的司机信息进行审核并给出结果。

5) 除此之外，系统应当提供：

- 用户的注册，登录，退出功能。
- 计算距离范围并将乘客的下单信息通告给范围内的司机的功能。
- 处理不同情况下，司机或用户终止交易的功能。
- 处理不同情况下，用户付款失败的功能。
- 提供用户为司机提供红包促使司机接单，并将红包信息告知司机的功能。

3.2 性能目标

可伸缩性：

- 系统必须能存储 7000 万用户的注册信息，系统最好能够存储 1 亿用户的注册息。

性能与效率：

- 系统必须支持 3000 万用户同时使用该软件，且 95% 以上的用户操作的系统响应间不超过 1s。

- 系统必须在 500 万乘客同时发出打车请求的情况下仍然能够成功且正确地处理打车请求给出相应反馈，并且能够将 90%以上乘客的打车有关操作的反应时间控制在 1.5s 以内。
- 系统必须能够在系统同时处理订单数量为 500 万条时，将 97%的用户的订单信息能够在 2s 内发布给符合范围条件的司机。
- 系统必须能够将 1000 条不同乘客同时发布的警报信息在 1s 内传递到业务监测相关系统并给出明显的提示，且确保 100%的警报信息不出现错误或缺漏。

可靠性：

- 导航信息的可靠性：要求导航系统提供的位置信息误差在 300 内。提供的路径信息准确率最低达到 95%。

付款系统的可靠性：

- 要求付款系统具由高可靠性，99.8%的付款操作必须能够正常进行，不存在错误扣款，重复扣款或漏扣款的情况。

系统可靠性：

- 系统必须保证在系统出现繁忙，系统故障或网络故障时，若所有的订单信息在传送和处理过程中不出现信息错误。

可移植性：

- 该系统必须能够在安卓系统和 IOS 系统的智能手机上安装并运行。且当能够和其他软件共存于一个平台上。

可维护性：

- 开发人员在实现此系统时必须根据此系统地业务模块和运行周境，设计各个模块之间以及系统与第三方系统之间的有效接口和连接实现方法，便于系统的修改更新和扩展。

可使用性：

- 系统必须 24*365 运行，可使用性达到 99%以上。

安全性：

- 对不同用户的权限进行控制，对于重要数据进行数据加密。
- 数据库中的信息必须要得到良好的保护，并严格限制修改数据库的权限。
- 数据库应当定时备份，具有较强的抗干扰能力。

3.3 体系约束

Table3.1—体系约束表

要素	约束
成本	1,000,000 元人民币
开发工期	2 年左右。
平台	用于实现计步的手机客户端至少应支持 Android 系统和 IOS 系统，并支持其 1 年内发布的各个版本。
实现约束	1) 采用 C/S 架构。

	2) 服务器需要部署在 Linux 操作系统上。 3) 使用 Mysql 数据库, 要做到数据库容量可扩展。
设计约束	1) 本系统要求应用 Java 语言进行编写, 并且所有变量的命名规范符合 Java 语言命名规范。 2) 本系统手机端应用框架采用 SSM 框架进行系统开发, 业务监测网页端和系统管理网页端使用标准 HTML5 技术来实现。
数据库要求	1) 数据库数据存储应尽量节省空间。 2) 从未来可扩展的角度上要求数据库可以移植到 oracle 上。 3) 本系统对于数据库的访问操作要有相应的并发处理和恢复机制, 并且保证访问获取到数据信息的正确性
其他方面	开发此软件时, 必须严格按照有关法律和政策执行。

3.4 体系结构目标

软件体系结构为软件系统提供了一个结构、行为和属性的高级抽象, 由构成系统的元素描述、元素相互作用、指导元素集成的模式以及这些模式的约束组成。该文档描述的打车软件体系结构应该有以下五个特性:

- 1) 体系结构应是适宜的: 该文档给出的体系结构应当是对于打车软件可通用的体系结构, 能够实现打车软件的大部分功能性需求和非功能性需求。打车软件的体系构造可以以该文档描述的体系为参考依据或参考标准。
- 2) 体系结构应是概念完整的: 该体系结构应当从完整的视角和观点出发, 能够从不同且全面的角度反映相关利益方对软件体系结构的要求和期望。包括从逻辑、实现、进程、部署等视角描述该体系结构。
- 3) 体系结构应是易于维护和升级的: 该软件体系结构可能会随用户量的扩充、软件硬件设备的升级、功能扩充修改等需要升级或更新, 对于体系结构的设计应当为升级或修改留下接口。
- 4) 体系结构应是便于移植的: 鉴于打车软件可能会用于不同地区, 不同平台, 被部署在不同的服务器上, 连接不同的第三方系统, 因此该软件体系结构应当是便于移植的。

5) 体系结构应是理性化的。该软件体系结构应当逻辑清楚，阐明清晰无二义性，使读者易于理解并能够找到实现方法和解决方案。

这五个质量要素体现了体系结构作为早期设计决策对系统需求的支持、实现的约束，管理的组织。

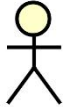


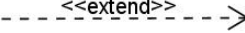

四、用例视角

本章是对该打车软件体系架构中主要用例的描述。

用例图是由参与者，功能用例以及它们之间的关系构成的图，其目的是描述系统功能，通过用例图呈现参与者和他们之间的关系构成的图，就可以更清晰地了解用户对系统、子系统以及各项功能的使用和行为。

从用例视角看待软件的体系结构，可以作为软件迭代开发的过程中选择作为迭代焦点的场景集和用例集的重要依据。用例图描述了一组场景和用例，表示一些重要的中心功能。用例图还描述了一组覆盖范围大的，在体系结构中起中心作用的场景，这些场景强调或说明体系结构的功能特点和结构特性。

本打车软件系统的主要参与者包括司机、乘客、业务监测人员和系统管理人员。对不同的用户本系统提供同的功能。本章针对这四类主要用户角色，给出了系统整体用例图和各参与者分别对应的用例图。并对主要用例给与详细描述。用例图各部分图解如下表 4.1。

Table 4.1—用例图解表	
图形	含义
	参与者
	用例
	关联关系
	扩展关系
	系统边界

4.1 架构主要用例

本打车软件系统体系结构的主要用例如下：

- 信息管理与验证：登录注册信息管理
- 打车：下单叫车、取消订单、评价、举报、发送警报信息
- 付款：设置付款方式
- 接单：接收订单、取消订单、举报乘客
- 收款：设置收款方式，设置收款金额
- 处理警报信息
- 监控异常交易：发现，处理
- 处理举报信息：核实，反馈
- 审核用户信息
- 管理用户信息：增删改查

以上用例的主要参与者为司机、乘客、业务监测人员和系统管理人员。其中打车，接单等用例还与第三方导航系统有关，付款，收款等用例还与第三方收款系统有关。架构整体用例图如下图 4.1：

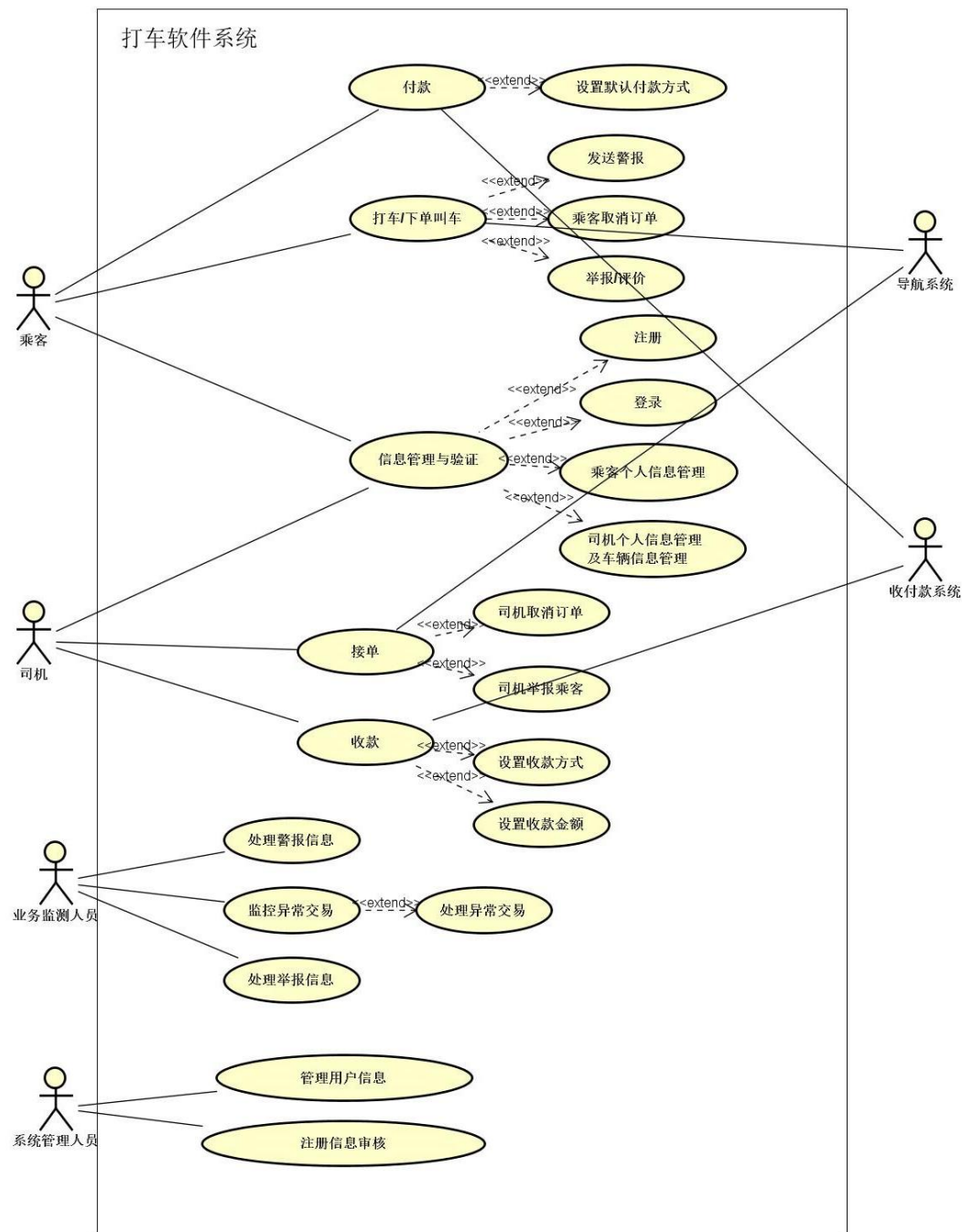


Figure 4.1—整体用例图

乘客主要参与的用例有管理乘客个人信息，乘客的注册登录，乘客打车，评价举报司机，发送警报信息，取消订单，设置付款方式等。

司机主要参与的用例有管理乘客个人信息和车辆信息，司机的注册登录，司机接单，举报乘客，取消订单，设置收款方式和收款金额等。

业务监测人员主要参与的用例有处理警报信息，监控并处理异常交易，审核并处理举报信息等。

系统管理人员主要参与的用例有审核注册软件的司机和乘客信息，对用户信息进行管理，包括信息的增添，删除，修改，查找等。

乘客，司机，业务监测人员，系统管理人员分别对应的部分用例图如下图 4.2：

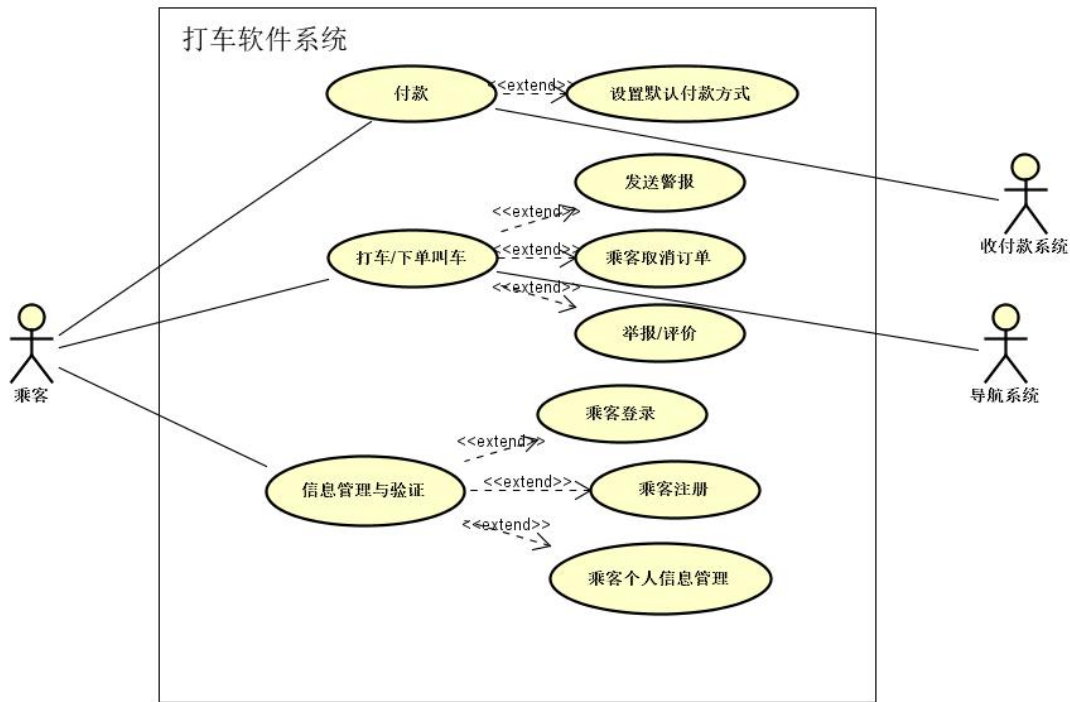


Figure4.2—乘客用例图

乘客能够借助客户端进行下单叫车，在用户加单参与交易的过程中，乘客遇到危险可以一键发送警报，乘客可以取消订单，在交易结束后可以对司机进行评价。乘客可以借助客户端软件进行付款操作，用户设置默认免密的付款方式，由司划款。乘客还可以进行注册，登录和个人信息管理等操作。

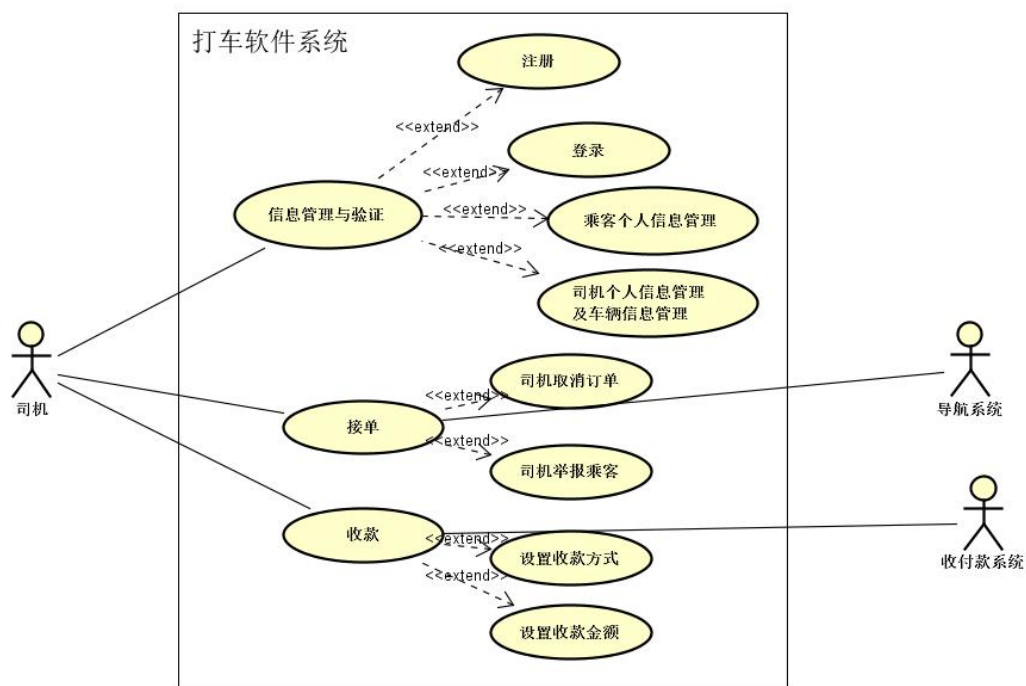


Figure 4.3—司机用例图

司机可以借助系统的手机客户端收到用户的下单信息并选择性接单，在交易过程中，司机可以取消订单。在交易结束后，司机可以举报乘客。司机可以通过本系统的手机客户端软件进行收款，允许司机设置收款方式和每个订单的收款金额。允许司机进行登录，注册操作，允许司机管理个人的基本信息和车辆的基本信息。

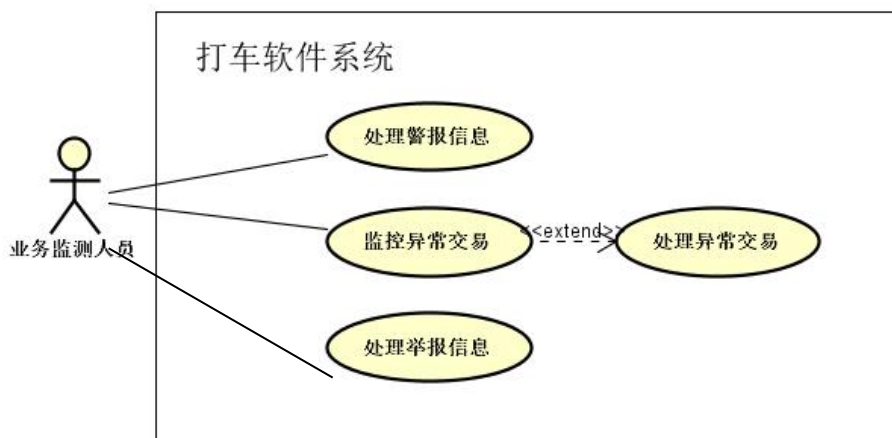


Figure 4.4—业务监测人员用例图

业务监测人员借助系统提供的业务监测网页能够进行警报信息的处理，能够实时监控所有交易信息并发现并处理异常交易，能够对乘客和司机的举报信息进行处理。

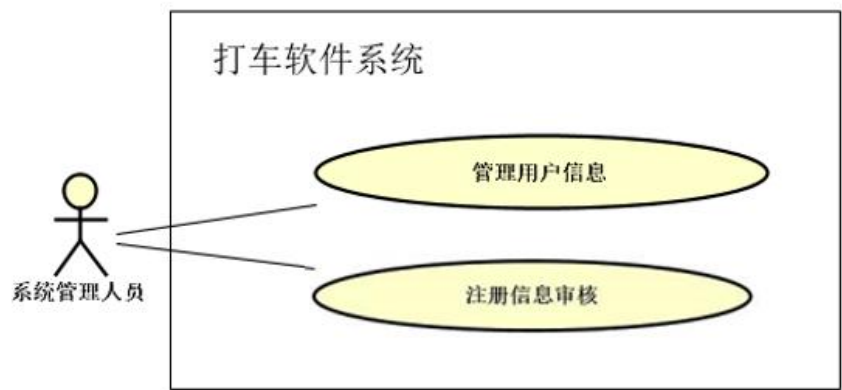


Figure4.5—系统管理人员用例图

系统管理人员借助系统提供的系统管理页面能够对所有用户的信息和权限进行管理，能够对司机的注册信息进行审核并给出反馈。

以下将分别对体系各个主要用例进行简要描述，包括该用例的主要参与者，主要功能以及该用例扩展用例的介绍。

4.1.1 信息管理与验证

此用例主要参与者为司机和乘客，主要处理与用户信息相关部分功能，包含司机和乘客的登录，注册和个人信息编辑管理部分功能

乘客注册：主要参与者为乘客。需要乘客设置昵称，密码，填写实名制个人信息包括手机号码等。

司机注册：主要参与者为司机。司机注册需要系统管理人员对相关信息进行审核。需要司机在注册时提供真实详细的个人信息以及司机车辆的基本信息。包含司机的姓名，性别，身份证号，手机号，驾驶证号，行车证号，车牌号，车辆型号等等基本信息。需要司机上传相关资质证明，车况证明等等。司机提交信息后等待审核与反馈。

登录：主要参与者为乘客和司机。根据乘客和司机输入的账号和密码登录系统，需要提供找回密码功能。

乘客个人信息管理：主要参与者为乘客，为乘客提供修改管理个人昵称，电话，密码等基本信息的功能。

司机个人信息管理及车辆信息管理：主要参与者为司机，为司机提供修改个人基本信息和车辆基本信息的功能，司机对基本信息的修改同样需要审核，

4.1.2 下单叫车

此用例主要参与者为乘客，主要处理用户下单叫车部分功能，包括用户选择叫车类型，选择出发地与目的地等，并将订单信息发布给范围内符合条件的车辆。订单信息首先发送给 2 公里内符合选择条件的车辆，若 2 公里内车辆不足 5 辆或 2 分钟内没有司机接单，则将地点范围扩展 2 公里，以此类推。该用例有发送警报信息，取消订单，举报评价三个扩展用例。

发送警报：主要参与者为乘客。当用户在交易过程中遇到危险，可以使用该功能发送警报信息共业务监测人员处理。

乘客取消订单：主要参与者为乘客。司乘客在有司机接单前或司机接单 2 分钟内可以无条件取消订单。若司机接单 2 分钟后乘客取消订单，应按照司机的行驶距离给与司机一定补偿。

举报/评价：主要参与者为乘客。为乘客提供对司机进行评价和举报的功能，举报信息会统一有业务监测人员核实处理并给出反馈。

4.1.3 付款

此用例主要参与者为乘客，主要提供用户在交易结束后向司机付款的功能。为保障司机权益，要求用户实名制注册并绑定至少一种默认付款方式，在交易结束后由司机设置交易金额，从默认付款方式里自动划款。当乘客默认付款方式余额不足或付款失败时，应当提供重新付款功能。该用例包含扩展用例设置默认付款方式。

设置默认付款方式：主要参与者为乘客。为用户提供设置、添加和更改默认付款方式的功能。

4.1.4 接单

此用例主要参与者为司机，主要处理司机接收乘客订单部分功能，包括司机选择性接单，获得订单信息和导航信息并将接单司机的基本信息反馈给乘客。该用例有司机取消订单，司机举报乘客两个扩展用例。

司机取消订单：主要参与者为司机。司机在接单后 2 分钟内可以无理由取消订单。若司机在接单 2 分钟后取消订单，视为违规操作，会扣除司机信誉分。若司机一星期内违规取消订单次数超过 10 次，将会给与司机一星期封号处理。若司机两个月内违规取消订单次数超过 100 次，将给与永久封号处理。司机取消订单后哦订单信息将会被重新发布给订单出发地范围内符合条件的司机。

司机举报乘客：主要参与者为司机。为司机提供举报乘客的功能。若乘客出现恶意违规，恶意损坏车内设施等操作，司机可对乘客进行举报。经业务监测人员审核后给与相应处理。

4.1.5 收款

此用例主要参与者为司机，主要提供司机在交易结束后面向用户设置收款金额并收款的功能。当司机设置金额错误，收款过程出错或失败时，应提供重新收款功能。

该功能应杜绝收款出现错误的情况。该用例有司机设置收款方式，设置收款金额两个扩展用例。

设置默认收款方式：主要参与者为司机。为司机提供设置、添加和更改默认收款方式的功能。

设置收款金额：主要参与者为司机。为司机提供设置收款金额的功能。为保障用户权益，快车，拼车的车费由系统自动根据距离和路况计算，不需要司机设置。出租车司机根据出租车计价表金额设置收款金额。

4.1.6 处理警报信息

此用例主要参与者为业务监测人员，主要提供业务监测人员接收，查看并处理警报信息的功能。

4.1.7 监控异常交易

此用例主要参与者为业务监测人员，主要提供业务监测人员实时监控各个交易情况，系统发现异常的交易信息，包括当车辆路线偏移给定导航路线超过 10 公里时监控程序自动报警等功能。该用例有处理异常交易的扩展用例。

处理异常交易：主要参与者为业务监测人员，主要为业务监测人员提供异常交易的详细信息，异常车辆的实时位置，提供异常交易的处理反馈功能。

4.1.8 处理举报信息

此用例主要参与者为业务监测人员，主要提供业务监测人员接收，查看并处理举报信息的功能。提供业务监测人员向举报用户反馈举报信息处理结果的功能。

4.1.9 管理用户信息

此用例主要参与者为系统管理人员，主要对系统管理人员提供对注册系统的所有用户信息的增添、查找、删除、更改功能，提供用户权限的分配功能。

4.1.10 注册信息审核

此用例主要参与者为系统管理人员，主要对系统管理人员提供审核注册该系统的司机提供的注册信息并给出反馈的功能，当司机提出修改个人信息申请时向系统管理人员提供审核反馈功能。

五、逻辑视角

本章是对打车软件体系结构的逻辑视图描述。描述体系中重要的类的部署，子系统的分层，子系统中包和模块的组织。

打车软件系统基于 C/S 模式和 MVC 模式分为 4 个主要层，从上到下分别为表示层，客户端层，业务逻辑层和数据管理层。

进一步将系统分为 9 个包，对应 9 个不同的功能模块，分布在 4 个层中。分别为人机交互模块，司机信息管理模块，乘客信息管理模块，打车处理模块，接单处理模块，付款模块，业务监测模块，用户信息审核管理模块和导航模块。

本章还将在层次结构图和包图中对系统主要类的归属进行简要规划和解释说明。

5.1 架构概述-系统分层

打车软件系统采用基于 C/S 模式的分层架构，将系统划分为 4 个层，结构如下图 5.1。

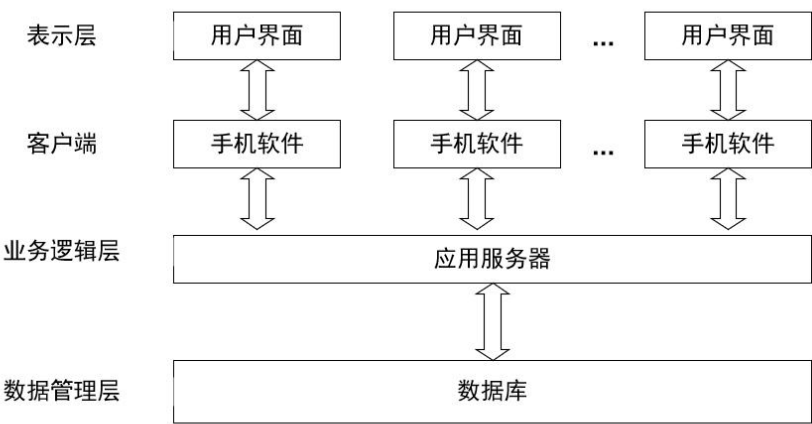


Figure 5.1—系统分层图

5.1.1 表示层

表示层负责与用户交互的用户界面有关逻辑，负责对用户的各个需求输入进行接收和传递，对服务器给出的响应消息及时反馈给用户。表示层降低了系统之间的耦合，使用户界面的开发大大简化，表示层开发人员只注重人机界面的设计，不必关心业务逻辑和数据库的访问。

5.1.2 客户端层

客户端层负责对表示层接受的消息进行基本的处理，负责与表示层和作为服务器的业务逻辑层进行交互。负责与第三方付款系统进行基于客户端的交互，负责对用户的手机软件即客户端中进行的一系列动作进行响应并给出响应处理，负责对从服务器接收到的指令或信息进行处理或传递。

5.1.3 业务逻辑层

系统的服务器部署在业务逻辑层，负责对从表示层接收到的数据信息进行处理，计算出反馈动作回馈给表示层。该层负责本软件绝大多数的逻辑运算，提供客户端程序调用的业务逻辑规则，以完成其业务操作，业务逻辑改变时，表示层的客户端界面可以不作变化。

5.1.4 数据管理层

数据管理层提供对数据库进行各种操作的方法，被中间业务逻辑层调用完成业务逻辑。主要参与系统中导航信息的提供，用户信息的查看、管理和审核等于数据库有交互的操作。

5.1.5 分层的优点

1. 使系统的可伸缩性好。C/S 模式和业务逻辑层与客户端层的分割使系统可在本地网和广域网等复杂的网络环境之间进行信息的传递和交互。
2. 使系统效率提高。经过合理的层次布局，使通过网络的传输数据量大大减少，使同一时间客户端进行的计算操作大大减少，提高了网络效率和系统运行效率。
3. 可管理性强。系统客户端层和表示层基本实现“零维护”，主要管理工作集中在业务逻辑层，而业务逻辑的修改对客户端层没有影响。
4. 可重用性好。按可提供的服务构筑应用，每种服务可以被不同的应用重用。由于采用面向对象的组件构成，进一步增加了系统的可重用性。
5. 安全性较好。降低了系统各个模块之间的耦合性。

5.2 架构概述-包/模块的组成分布

5.2.1 模块概述

该打车软件体系结构将系统内部的功能被细化 10 个功能模块，对应 10 个包，分别为：

- 1) 司机信息管理模块：司机管理其个人信息和自己的车辆信息
- 2) 乘客信息管理模块：乘客管理自己的个人信息。
- 3) 打车操作模块：与乘客交互，部署在客户端上负责乘客的下单功能的用户操作与系统反馈等处理。
- 4) 打车处理模块：负责乘客的下单，播报订单信息，提供导航等服务器处理打车操作的有关工作。
- 5) 接单操作模块：与司机交互，部署在客户端上负责司机的接单功能的用户操作与系统反馈等处理。
- 6) 接单处理模块：负责对司机抢单，接单，提供导航信息等服务器处理接单操作的有关工作。
- 7) 付款模块：与第三方付款系统相连，负责司机和乘客之间的金额结算。

- 8) 业务监测模块：与业务监测人员交互，负责接收警报，处理举报和发现异常。
- 9) 用户信息审核管理模块：与信息管理人员交互，负责司机信息审核和用户信息管理。
- 10) 导航模块：与第三方导航系统交互，负责为司机，乘客和业务监测人员提供导航信息。
- 11) 人机交互模块：负责用户和客户端软件之间人机交互的处理和相应显示。

5.2.2 模块分布

该 11 个模块对应的包在系统 4 个层中对应的分布如下图 5.2，图中虚线表示对应模块之间有信息传递或模块间有重要关联：

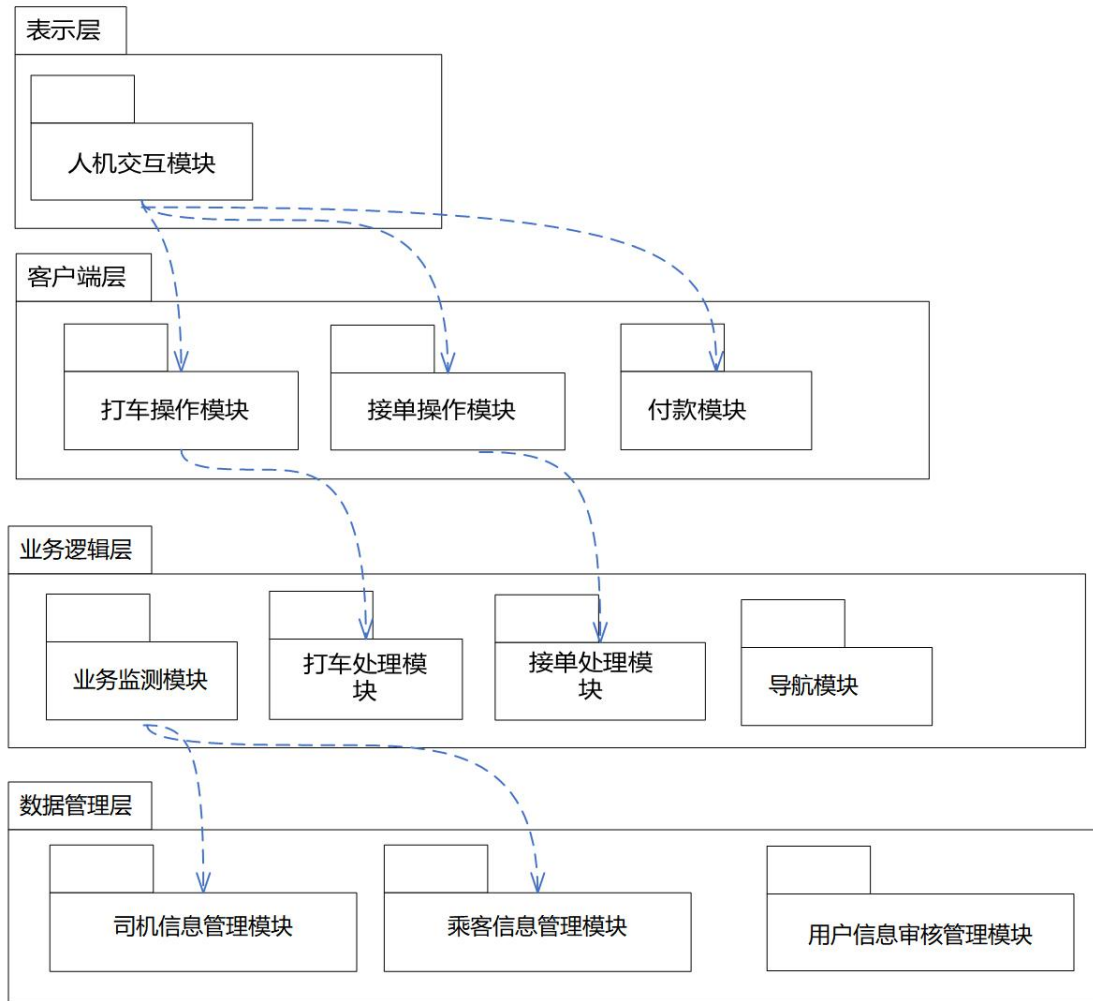


Figure5.2—系统包图

- 表示层中包含 1 个包，实现人机交互模块所需的功能
- 客户端层中包含 3 个包，包含客户端对乘客打车操作进行处理的打车操作模块，客户端对司机接单操作进行处理的接单操作模块，和乘客与用户进行收付款操作的付款模块。

- 业务逻辑层中包含 4 个包，分别为业务监测人员进行业务监测的处理模块，处理打车请求对应的打车处理模块，处理接单请求对应的接单处理模块，以及和第三方导航系统进行连接，获得导航信息并发送到客户端的导航模块。
- 数据管理层中包含 3 个包，分别为司机信息管理模块，乘客信息管理模块和面向系统管理员进行系统信息管理的用户信息审核管理模块。

5.2.3 架构概述-模块中类的分布

本节讲述系统体系中主要的各个类在 5.2.2 节中 11 个包内的分布。

5.2.3.1 表示层 • 人机交互模块

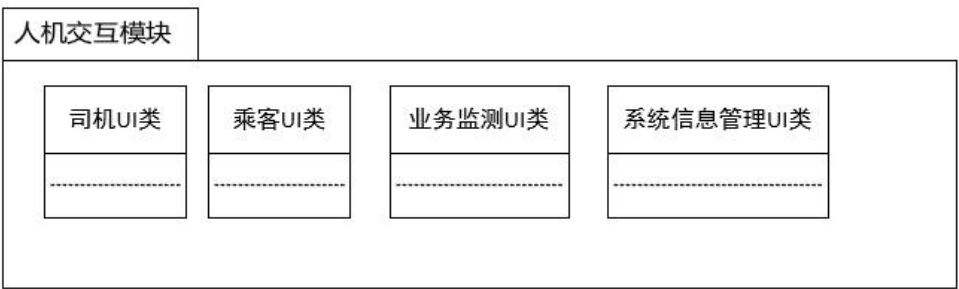


Figure 5.3—人机交互模块中类的分布

实现人机交互模块的包中包含 4 个主要类，分别为

- 司机 UI 类：负责部署司机对应的客户端用户界面。
- 乘客 UI 类：负责部署乘客对应的客户端用户界面。
- 业务监测 UI 类：负责部署业务监测人员对应业务监测工作的业务监测网站界面。
- 系统信息管理 UI 类：负责部署系统管理人员对应系统信息权限管理工作的系统管理网站界面。

5.2.3.2 客户端层 • 打车操作模块

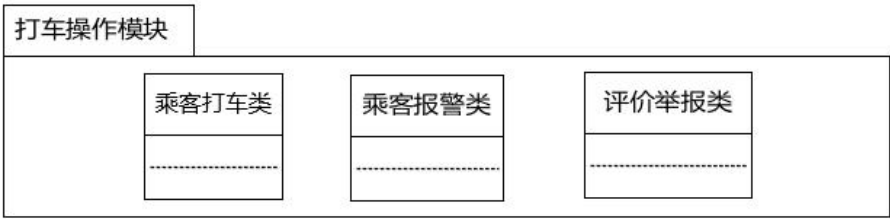


Figure 5.4—打车操作模块中类的分布

打车操作模块部署在客户端上，实现客户端层打车操作相关功能。该包中包含 3 个主要类，分别为：

乘客打车类：负责在客户端记录、处理、传递用户发出的打车请求，在订单的不同状态调用乘客 UI 类为乘客给出相应反馈。将相应信息发送给服务器。

乘客报警类：负责在客户端处理、传递乘客在交易过程中可能发出的警报信息。将相应信息发送给服务器。

举报评价类：负责在客户端处理用户对特定交易的评价信息和举报信息。将相应信息发送给服务器。

5.2.3.3 客户端层 • 接单操作模块

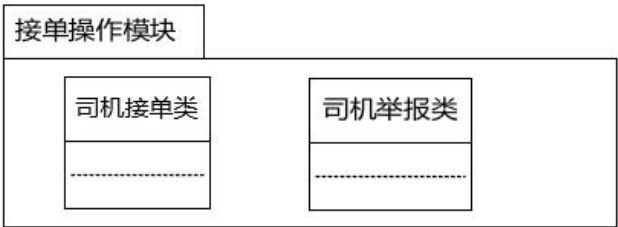


Figure 5.5—接单操作模块中类的分布

接单操作模块部署在客户端上，实现客户端层接单操作相关功能。该包中包含 2 个主要类，分别为：

- 司机接单类：负责在客户端向司机广播可能的订单信息，并记录、处理、传递司机的接单信息，在订单的不同状态调用司机 UI 类为司机给出相应反馈。将相应信息发送给服务器。
- 司机举报类：负责在客户端处理司机对特定交易的举报信息。将相应信息发送给服务器。

5.2.3.4 客户端层 • 付款模块



Figure5.6—付款模块中类的分布

付款模块部署在客户端上，实现客户端收付款功能，与第三方收付款系统相连。该包中包含 2 个主要类，分别为：

- 司机收款类：为司机提供设置修改默认收款方式和收款金额功能的类。
- 乘客付款类：为乘客提供设置修改默认付款方式功能的类。

5.2.3.5 业务逻辑层 • 业务监测模块

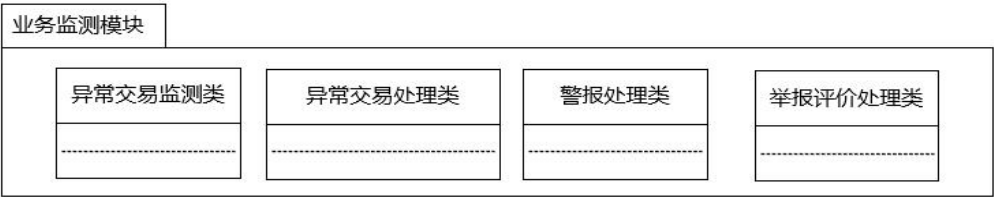


Figure 5.7—业务监测模块中类的分布

业务监测模块部署在服务器上,实现业务监测人员执行的业务监测工作所需的功能。该包中包含 4 个主要类，分别为：

- 异常交易监测类：实时监测所有正在进行的交易信息，发现异常及时提示业务监测人员的类。
- 异常交易处理类：为业务监测人员提供异常交易反馈处理功能的类。
- 警报处理类：为业务监测人员提供接收并处理警报信息功能的类。
- 举报处理类：为业务监测人员提供接收并处理反馈用户举报信息功能的类。

5.2.3.6 业务逻辑层 • 打车处理模块

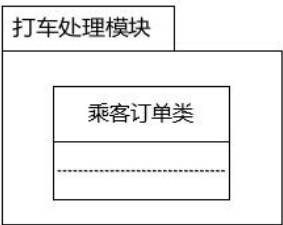


Figure5.8—打车处理模块中类的分布

打车处理模块部署在服务器上，实现乘客打车时订单信息的存储和传递，对用户请求进行处理和反馈。该包中包含 1 个主要类，为：

- 乘客订单类：拥有当前订单的所有信息，拥有系统处理打车功能时需要调用的函数。

5.2.3.7 业务逻辑层 • 接单处理模块

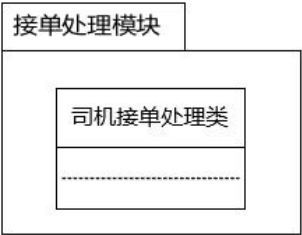


Figure 5.9—接单处理模块中类的分布

接单处理模块部署在服务器上，实现司机接单时订单信息的广播，存储和传递。对司机接单操作进行处理和反馈。该包中包含 1 个主要类，为：

- 司机接单处理类：拥有司机接受的当前订单的所有信息，拥有系统处理司机接单功能时需要调用的函数。

5.2.3.8 业务逻辑层 • 导航模块

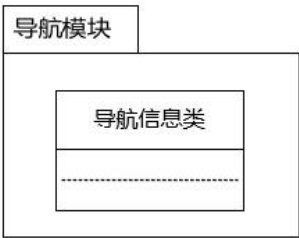


Figure 5.10—业务逻辑层中类的分布

导航模块部署在服务器上，实现导航信息的存储，查找，更新和对客户端的同步。该包中包含 1 个主要类，为：

- 导航信息类：实现导航信息的存储，查找，更新和导航信息在各个客户端上的广播与同步。

5.2.3.9 数据管理层 • 司机信息管理模块

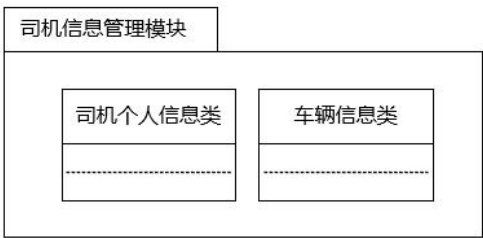


Figure 5.11—司机管理模块模块中类的分布

司机信息管理模块实现司机本人和系统管理员对司机信息和司机对应车辆信息的录入、更新和管理功能。该包中包含 2 个主要类，分别为：

- 司机个人信息类：存储司机本人的详细信息，提供司机本人和系统管理员管理司机信息所需方法。
- 车辆信息类：存储车辆的详细信息，提供车主和系统管理员管理车辆信息所需方法。

5.2.3.10 数据管理层 • 乘客信息管理模块



Figure 5.12—乘客信息管理模块中类的分布

乘客信息管理模块实现乘客本人和系统管理员对乘客信息的录入、更新和管理功能。该包中包含 1 个主要类，为：

- 乘客个人信息类：存储乘客本人的详细信息，提供乘客本人和系统管理员管理乘客信息所需方法。

5.2.3.11 数据管理层 • 用户信息审核管理模块

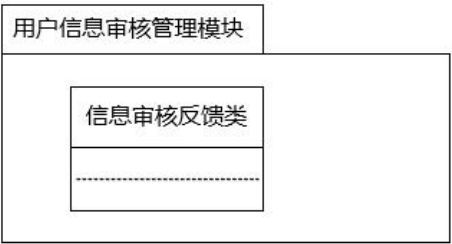


Figure 5.13—用户信息审核管理模块中类的分布

用户信息审核管理模块实现系统管理员对申请注册本软件的司机信息的审核，并给出审核反馈。该包中包含 1 个主要类，为：

- 信息审核反馈类：将注册司机提交的审核信息传送给系统管理网站，为系统管理人员提供审核并反馈注册信息的功能。

六、进程视角

本章使用 SDL 语言对针对进程视角进行描述。主要描述在概念级的软件体系结构下，系统运行态的情况。描述系统在执行时，包括哪些进程（包括线程、进程、进程组），以及它们之间是如何进行通信的、如何进行消息传递、接口如何。并且来说明如何进行组织。

本章将从乘客打车进程，司机接单进程，业务监测人员工作进程和系统管理人员工作进程共 4 个进程的进程流程和每个进程涉及的主要类之间的关联对本软件打车系统的体系结构进行说明，最后给出系统的整体类图。

6.1 进程流程图

6.1.1 乘客打车进程流程图

乘客使用本软件打车的流程对应流程图如下图 6.1:

未注册的乘客首先注册后登陆，已注册的用户直接登陆。之后进入乘客 UI 界面，可以发出打车请求或对个人信息进行管理。

若用户选择打车，则用户输入打车的需求，包括选择的车型，打车的出发地，目的地，是否拼车等，之后发出订单等待接单。用户在交易过程中可以发送警报。在交易结束后可以对交易进行评价和举报。

若用户选择管理个人信息，则允许用户对个人信息进行查看和更改，对默认付款信息进行更改和设置。

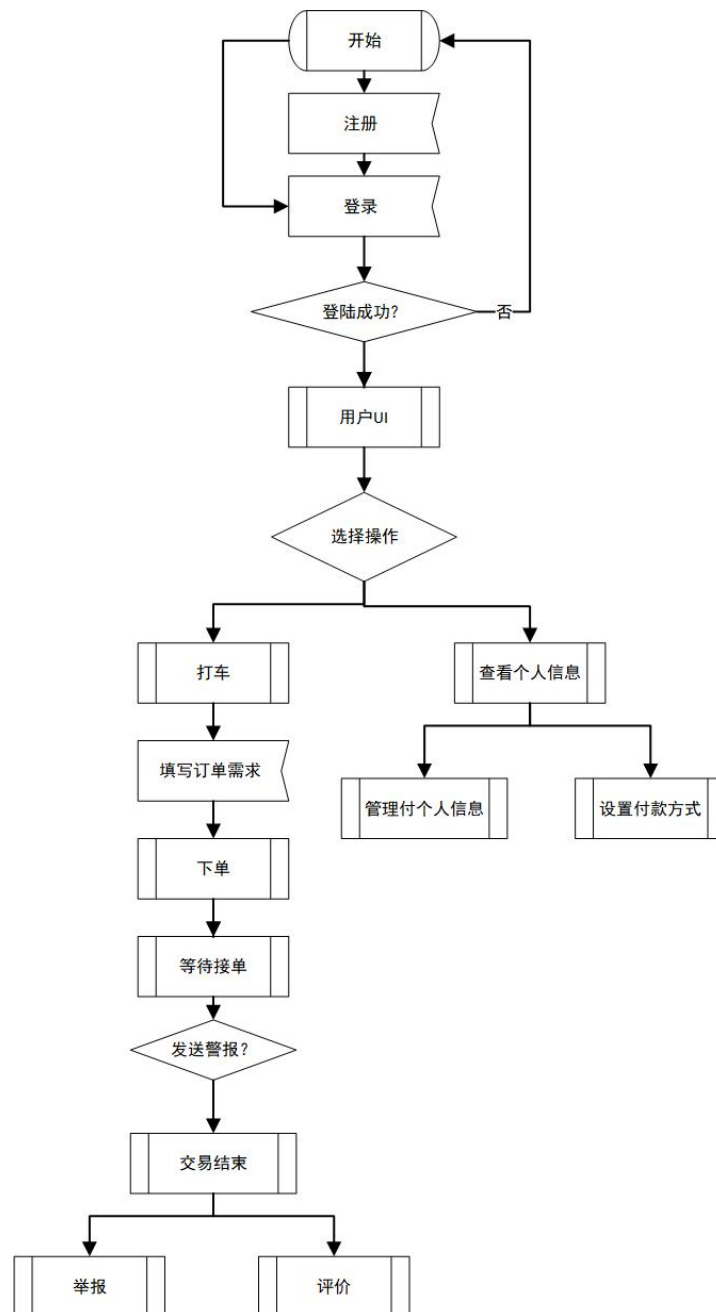


Figure6.1—乘客打车进程流程图

6.1.2 司机接单进程流程图

司机使用本软件打车的流程对应流程图如下图 6.2:

未注册的司机首先注册后登陆，已注册的司机直接登陆。之后进入司机 UI 界面，可以接收订单信息选择接收的订单或对司机个人信息进行管理。

若司机选择接单，则在交易结束之后司机可以设置收款金额，对乘客进行收款。司机在交易结束后可以对乘客的不正当行为进行举报。

若司机选择管理个人信息，则允许司机对个人信息，车辆信息进行查看和更改，对默认收款信息进行更改和设置。

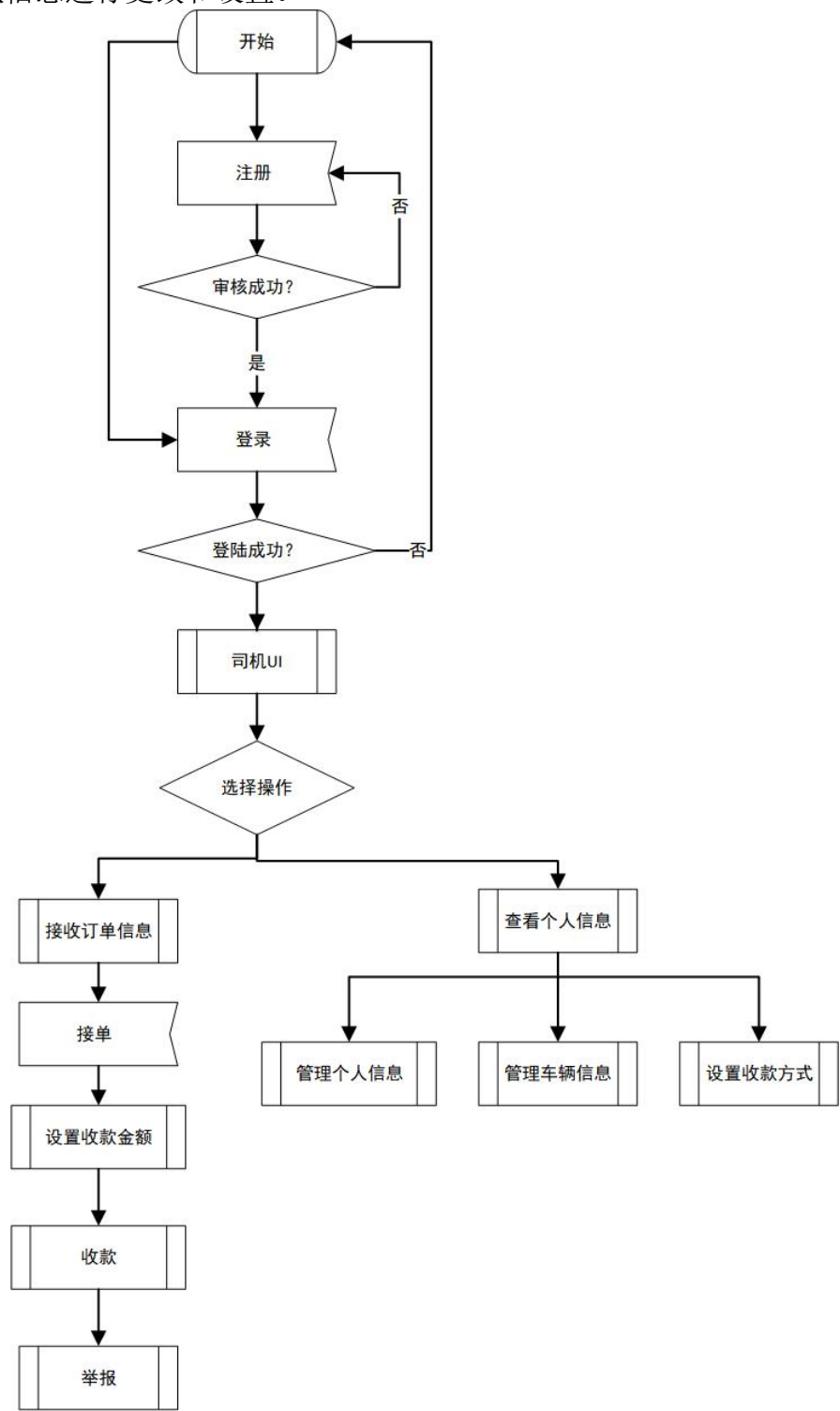


Figure6.2—司机接单进程流程图

6.1.3 业务监测人员工作流程图

业务监测人员使用本软件打车的流程对应流程图如下图 6.3:

业务监测人员首先登陆业务监测系统，登陆成功之后进入业务监测人员对应 UI 界面。业务监测人员可以监测异常交易，接收处理警报信息，查看所有正在进行的交易信息，接收处理用户的举报信息。

业务监测人员可以通过业务监测系统实时监测并及时发现异常交易，可以获得异常交易的详细信息并及时进行相应核实和处理。

业务监测人员可以及时接收到乘客的警报信息，并获得发出警报信息的用户正在进行的交易的详细信息，及时处理。

业务监测人员可以通过业务监测系统实时查询每一单交易的详细信息，包括每一辆正在搭载乘客的出租车的详细位置，

业务监测系统可以接受到所有用户发送的举报信息，业务监测人员负责对举报信息进行审核并反馈相应的处理结果。

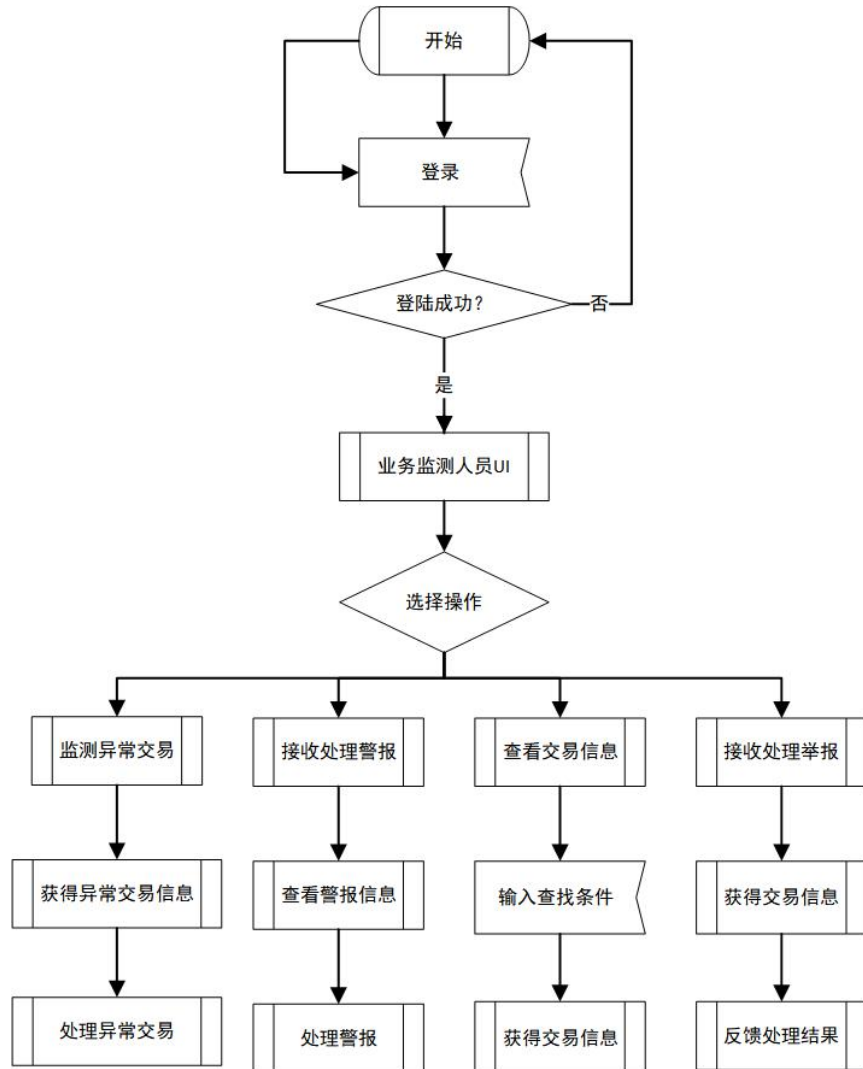


Figure 6.3—业务监测人员工作流程图

6.1.4 系统管理人员工作流程图

系统管理人员使用本软件打车的流程对应流程图如下图 6.4:

系统管理人员首先登陆系统管理页面，登陆成功之后进入系统管理人员对应 UI 界面。系统管理人员可以管理司机信息，管理乘客信息，管理车辆信息，并审核注册的司机提交的信息。

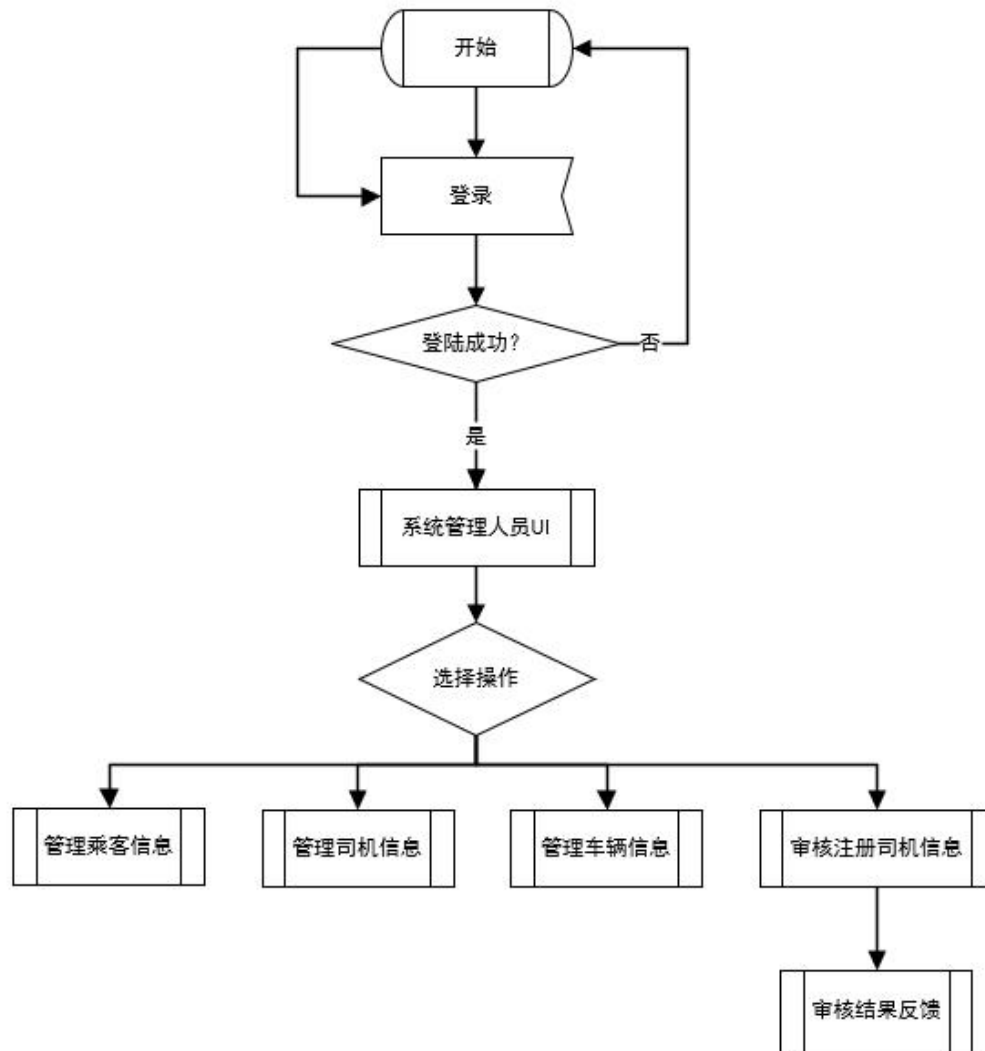


Figure 6.4—系统管理人员工作流程图

6.2 进程有关类图

6.2.1 乘客打车进程有关类图

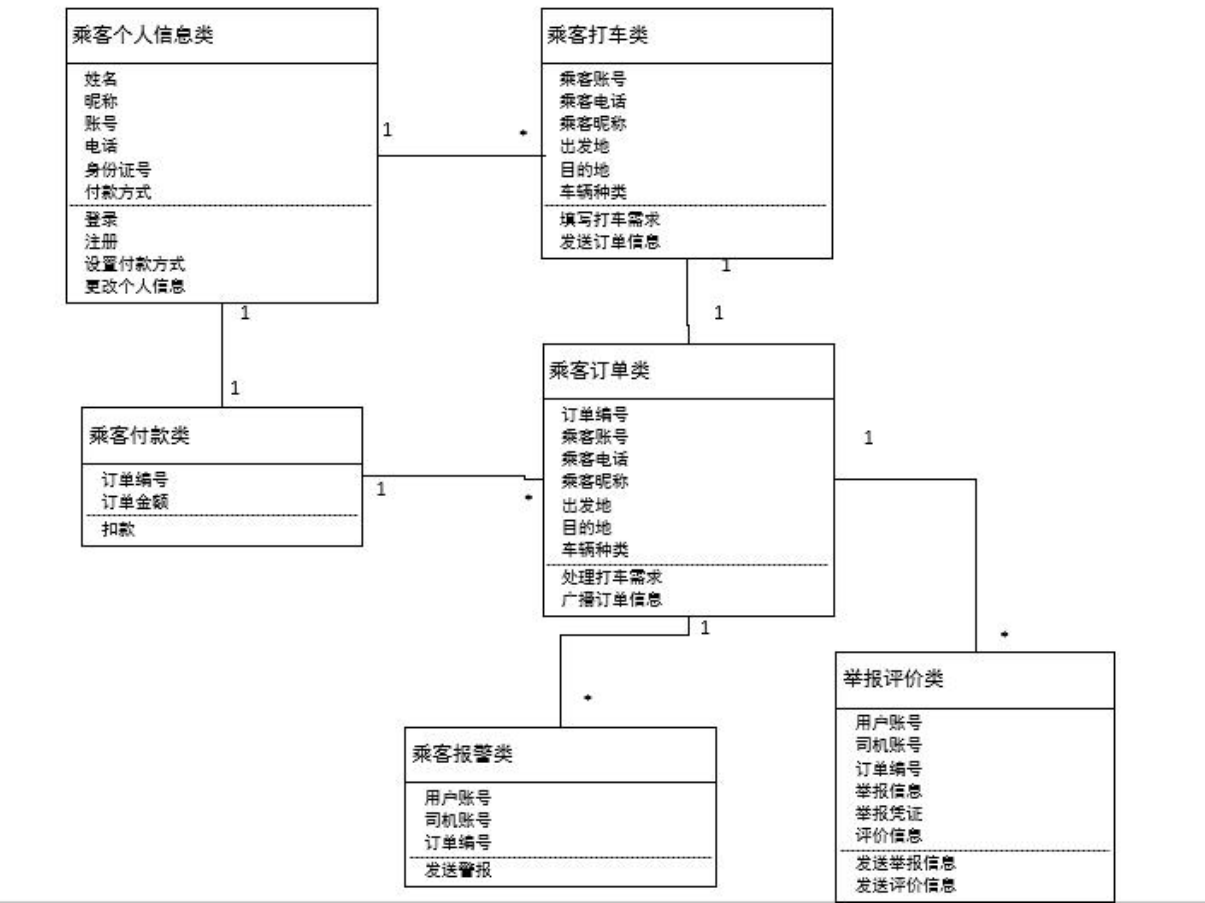


Figure 6.5—乘客打车进程有关类图

乘客打车进程涉及的主要类有：

1. 表现层——人机交互模块——乘客 UI 类
包含展现打车请求发送页面，等待接单页面，交易进行页面，交易完成页面和个人信息管理页面的方法
2. 客户端层——打车操作模块——乘客打车类
属性：包含乘客账号，乘客电话，乘客昵称，乘客选择的出发地，目的地以及乘客接收的车辆种类等属性。
方法：包含填写打车需求，发送订单信息等方法。
3. 客户端层——打车操作模块——乘客警报类
属性：包含用户账号，司机账号，订单编号等属性、
方法：包含发送警报等方法。
4. 客户端层——打车操作模块——评价举报类

属性：包含用户账号，司机账号，订单编号，举报信息，举报凭证，评价信息等属性

6.2.2 司机接单进程有关类图

Figure 6.6—司机接单进程有关类图

包含展现司机选择接单页面，司机交易进行页面，司机交易完成页面，司机个人信息管理页面的方法

2. 客户端层——接单操作模块——司机接单类

属性：包含司机账号，司机姓名，车牌号，出发地，目的地，车辆型号，车辆颜色等属性。

方法：包含填写打车需求，发送订单信息接单信息处理，发送接单信息等方法。

3. 客户端层——接单操作模块——司机举报类

属性：包含司机账号，订单编号，举报信息，举报凭证等属性。

方法：包含发送举报信息等方法。

4. 客户端层——付款模块——司机收款类

属性：包含订单编号，订单金额等属性

方法：包含设置收款金额，收款等方法。

5. 业务逻辑层——接单处理模块——司机接单处理类

属性：包含订单编号，司机账号，司机电话，司机姓名等属性。

方法：包含处理接单操作，回馈接单信息等方法。

6. 数据管理层——司机信息管理模块——司机个人信息类

属性：包含司机姓名，司机性别，司机账号，司机电话，身份证号，收款方式等属性。

方法：包含登录，注册，设置收款方式，更改司机个人信息等方法。

7. 数据管理层——司机信息管理模块——车辆信息类

属性：包含车牌号，发动机号，型号，车辆颜色等属性。

方法：包含更改车辆信息等方法。

6.2.3 业务监测人员进程有关类图

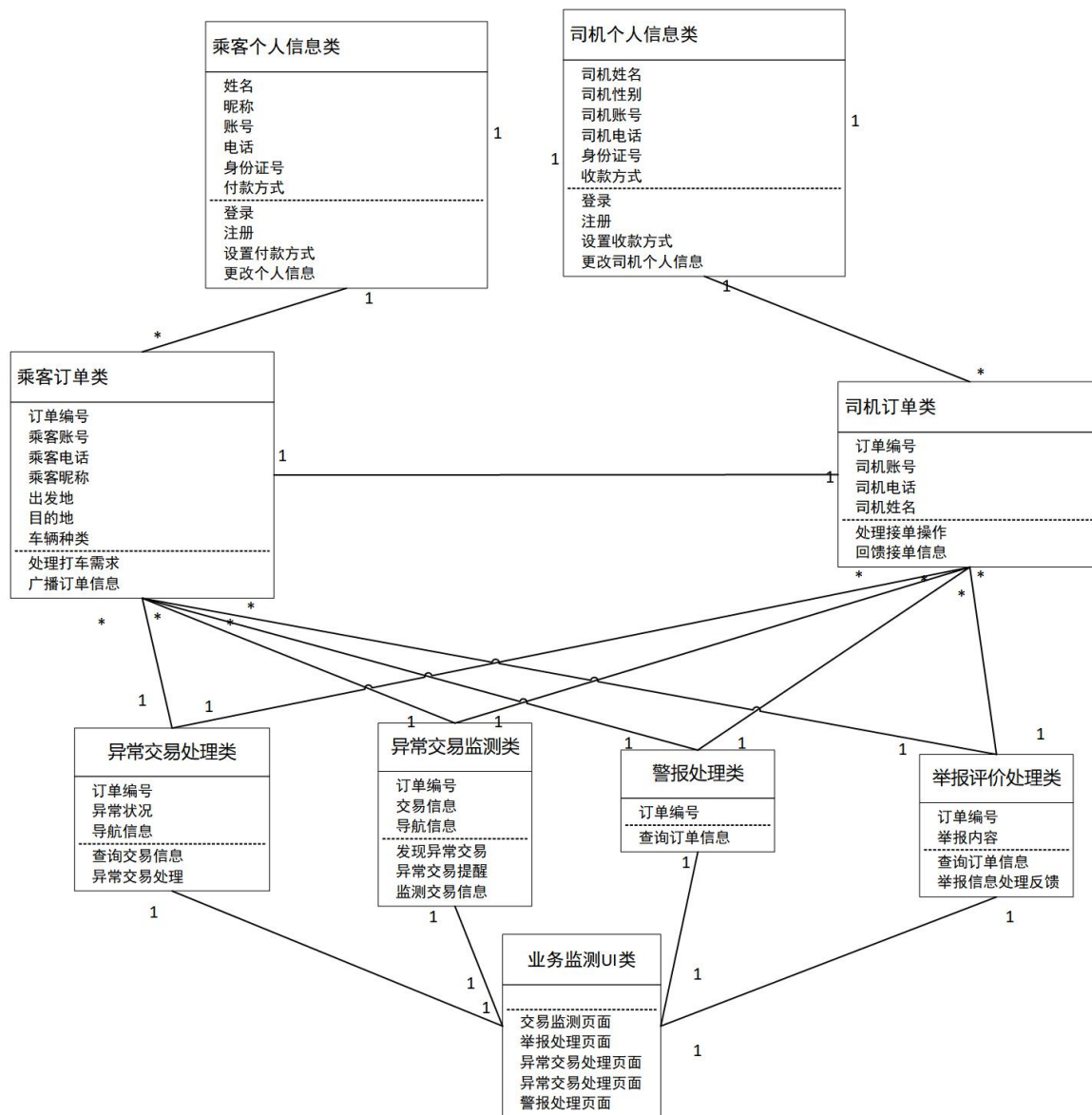


Figure 6.7—业务监测人员进程有关类图

业务监测人员工作进程涉及的主要类有：

1. 表现层——人机交互模块——业务监测 UI 类

负责为业务监测人员提供进行业务监测工作，警报处理工作，举报处理工作的页面。
方法：包含交易监测页面，举报处理页面，异常交易处理页面，异常交易处理页面等方法。

2. 业务逻辑层——业务监测模块——异常交易处理类

为业务监测人员提供异常交处理反馈的方法。

属性：包含订单编号，异常状况，导航信息等属性。

方法：包含查询交易信息，异常交易处理等方法。

3. 业务逻辑层——业务监测模块——异常交易监测类

实时监控正在进行的交易信息，找到异常交易并提示业务监测人员。为业务监测人员提供异常交处理反馈的方法。

属性：包含订单编号，交易信息，导航信息等属性。

方法：包含发现异常交易，异常交易提醒，监测交易信息等方法。

4. 业务逻辑层——业务监测模块——警报处理类

接收用户在交易过程中发出的警报信息，并能查询当前订单信息进行一定的处理。

属性：包含订单编号等属性。

方法：包含查询订单信息等方法。

5. 业务逻辑层——业务监测模块——举报评价处理类

为业务监测人员提供对用户的举报信息进行审核处理并反馈审核结果的方法。

属性：包含订单编号，举报内容等属性。

方法：包含查询订单信息，举报信息处理反馈等方法。

6.2.4 系统管理人员进程有关类图

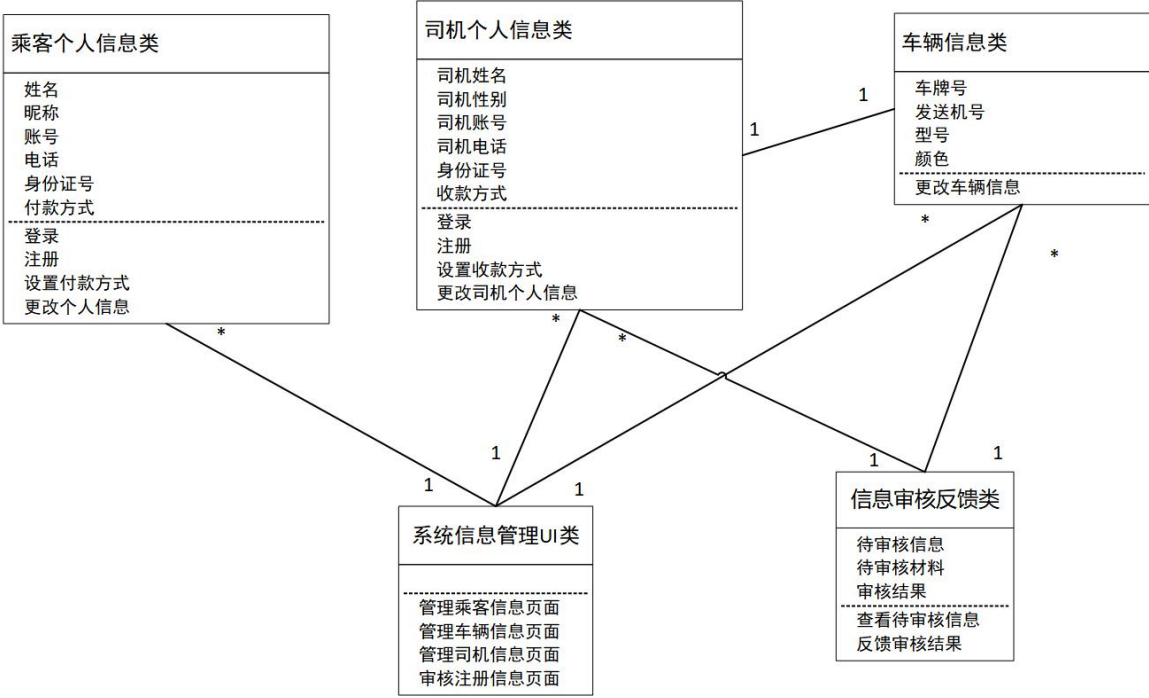


Figure 6.8—系统管理人员进程有关类图

系统管理人员进程涉及的类有：

1. 表现层——人机交互模块——系统信息管理系统 UI 类

为系统管理人员提供管理用户信息，车辆信息和提交注册审核的司机对应信息的 UI 界面。

方法：包含管理乘客信息页面，管理车辆信息页面，管理司机信息页面，审核注册信息页面等方法。

2. 数据管理层——用户信息审核管理模块——信息审核反馈类

属性：包含待审核信息，待审核材料，审核结果等属性。

方法：包含查看待审核信息，反馈审核结果等方法。

七、部署视角

从系统软硬件物理配置的角度，描述系统的网络逻辑拓扑结构。模型包括各个物理节点的硬件与软件配置，网络的逻辑拓扑结构，节点间的交互和通讯关系等。同时还表达了进程视图中的各个进程具体分配到物理节点的映射关系。由于本系统采用的是 C/S 架构，结合现实的网络拓扑结构，本文设计部署视角如下图：

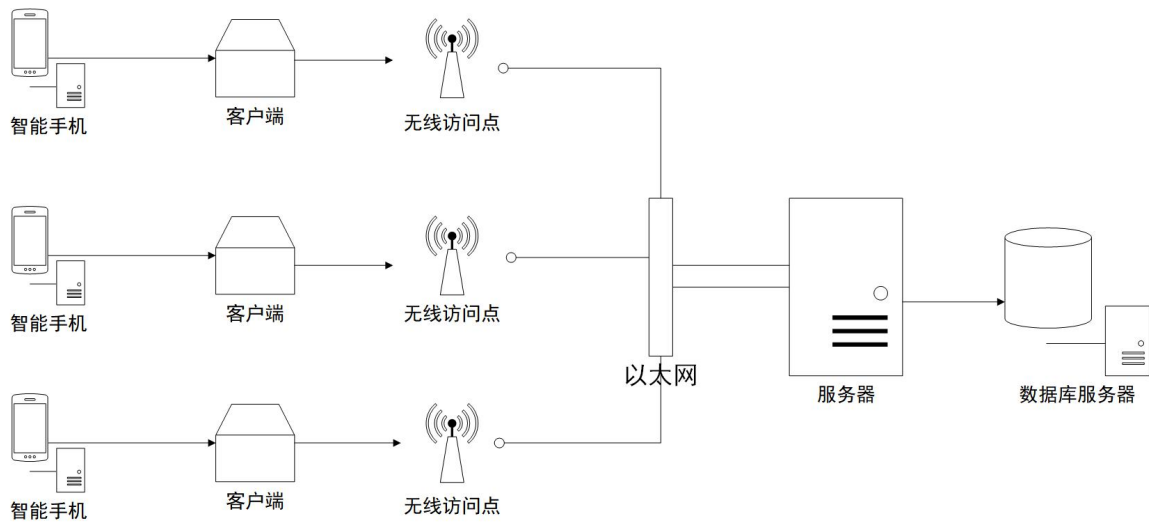


Figure 7.1—设计部署图

7.1 智能手机

安装本软件的还能手机终端。要求本该系统必须能够在安卓系统和 IOS 系统的智能手机上安装并运行。且系统应当能够和其他软件共存于一个平台上，不与其他任何用户超过 5000 万人的软件发生冲突。

7.2 客户端

即安装在智能手机上的打车软件，在客户端方面对用户的操作和服务器的返回信息进行传递和处理。

7.3 无线访问点

即安装本打车软件的智能手机所在的无线网络环境

7.4 以太网

将客户端信息传递到服务器，并将服务器的信息返回客户端以来的网络环境。

7.5 服务器

即实现本系统主要业务逻辑和信息同步功能的远程服务器系统。

7.6 数据库

即为存储本系统用户信息，订单信息和导航信息的数据库。

八、规格和性能

8.1 性能与效率

8.1.1 用户总量

系统必须能存储 7000 万用户的注册信息，系统最好能够存储 1 亿用户的注册信息。

8.1.2 用户同时使用该软件时

系统必须支持 3000 万用户同时使用该软件，且 95%以上的用户操作的系统响应时间不超过 1s。

系统必须在 5000 万用户同时使用该软件时仍能正常使用，且 90%以上的用户的执行所有操作的系统响应时间不超过 1.5s。

8.1.3 乘客同时发出打车需求时

系统必须在 500 万乘客同时发出打车请求的情况下仍然能够成功且正确地处理打车请求给出相应反馈，并且能够将 90%以上乘客的打车有关操作的反应时间控制在 1.5s 以内。

系统最好能够在 1000 万乘客同时发出打车请求的情况下仍然能够成功且正确地处理打车请求给出相应反馈。

8.1.4 司机接收订单时

系统必须能够在系统同时处理订单数量为 500 万条时，将 97%的用户的订单信息能够在 2s 内发布给符合范围条件的司机。

系统必须能够在 200 万位司机同时接单时，能够做到在 1s 内将 97%的接单成功的信息返回给司机和下单乘客。

系统应当能够同时处理 1500 万条订单信息并给出正确反馈，并且 90%以上的订单能够在 4s 内得到反馈。

8.1.5 业务监测人员处理警报时

系统必须能够将 1000 条不同乘客同时发布的警报信息在 1s 内传递到业务监测相关系统并给出明显的提示，且确保 100%的警报信息不出现错误或缺漏。

业务监测系统必须能够在 0.5s 内根据警报信息给出相应订单信息和车辆位置，且确保其中 90%的车辆位置的偏差范围在 1 千米以内。

九、质量需求

9.1 可靠性

9.1.1 导航信息的可靠性

要求导航系统提供的位置信息误差在 300 内。

要求导航系统提供的路径信息准确率最低达到 95%.

要求在车辆行驶过程中，车辆位置在导航系统中的延时不超过 1s。

9.1.2 付款系统的可靠性

关于乘客：要求所有的乘客实名制注册并选择一种默认的免密的付款方式。当该付款方式余额不足时，需要乘客更换付款方式由司机重新收款。

关于司机：要求司机指定默认收款方式，出租车司机严格按照打表计价的价格填写收款金额，快车和拼车由系统按照距离和路况计算价格，不交由司机填写金额。

关于发票：出租车由出租车司机提供发票，快车拼车由软件系统提供电子发票凭证。

关于故障：若付款时发生故障导致付款失败，由司机重新提交收款请求，重新收款。系统需要保证在付款操作失败时，不存在错误扣款，重复扣款或漏扣款的情况。

9.1.3 系统可靠性

系统必须保证在系统出现繁忙，系统故障或网络故障时，若所有的订单信息在传送和处理过程中不出现信息错误。若信息传送或处理失败，应当为用户提供反馈信息。系统对数据库中的信息应当定期备份，从而进一步提高系统数据抵抗外界破坏的能力。

9.2 易用性

该系统的主要用户是司机和乘客，司机会在驾驶时使用本软件，乘客大多在室外使用本软件。为安全计，手机端软件的界面设计应当尽量简洁，界面交互应当尽量简单。

司机

司机接单交互应当简洁，要求司机交互少，司机接受一个订单在屏幕上的有效点击不应超过 3 下。

司机接单后，软件应给与准确的导航信息，导航路径图最少占页面的 60%。

当司机需要输入文字时，应提供语音输入功能。

乘客

乘客下单的交互操作应当简洁，尽量用选择代替输入。选择地点是也应提供自动填充功能。

乘客的报警按钮应当明显。易于寻找且报警不需要任何填写信息的操作，按钮采取长按报警的方式，避免错误触碰发出不必要的报警信息。

9.3 安全性

该系统的安全性主要体现在用户信息的管理和保密，以及乘客信息对司机的部分可见性。

要求系统具有安全可靠的用户实名制认证机制，防止用户通过伪造身份的手段伪造他人的身份获取到其不该知道的信息或进行非法交易。

要求具有可靠的权限管理机制，以保证不同角色的用户之间不能使用未被授权的功能。只有系统管理员具备修改所有用户信息的权限。

数据库中的信息必须要得到良好的保护，可以采取加密存储，并严格限制修改数据库的权限。数据库应当定时备份，具有较强的抗干扰能力。

9.4 可维护性

开发人员在实现此系统时必须根据此系统地业务模块和运行周境，设计各个模块之间以及系统与第三方系统之间的有效接口和连接实现方法。

开发方必须提供系统日常维护说明手册以及系统错误诊断手册，以保证维护人员能够方便容易地理解、改正、改动和改进此系统，保证系统长期运行的可靠性。

本系统使用的操作系统接口采用 IEEE 规定的操作系统的接口标准，数据库接口采用 IEEE 规定的 SQL 标准。

9.5 可移植性

- 1) 易安装性：该系统必须能够在安卓系统和 IOS 系统的智能手机上安装并运行。
- 2) 共存性：系统应当能够和其他软件共存于一个平台上，不与其他任何用户超过 5000 万人的软件发生冲突。
- 3) 易替换性：系统可被容易地卸载，也易被高版本的系统替换。

9.6 可用性：

注册系统应 24 小时、每周 7 天提供。停机时间不得超过 2%。

平均无故障时间应超过 400 小时。

9.7 工程需求:

要求该软件使用统一的，一致性高的编程语言开发。
要求系统部署于 Linux 服务器上，以保证提供稳定可靠的服务。
为了系统将来的可扩展性，系统不应需要具有针对性的硬件或网络环境。整个系统也应尽量减少各模块间的调用，尽量做到低耦合。

9.8 数据库需求:

- 1) 统使用开源，费用较低，使用量较大、 可靠性较高的 MySQL 数据库。
- 2) 数据库的字段定义设计应具有一定的灵活性，保证一定的可修改性。
- 3) 数据库表的设计应保证一致性、完整性，避免出现数据过度冗余，出现数据不一致现象应进行及时的调整。
- 4) 数据库数据存储应尽量节省空间。
- 5) 从未来可扩展的角度上也要求数据库可以移植到 oracle 上。

A • 附录

A.1 随文所附图

Figure 2.1—三层体系结构示意图..... 9

Figure 2.2—代理器模式示例图..... 12

Figure 2.3—MVC 模式实例图..... 13

Figure 2.4—视图模型..... 13

Figure 4.1—整体用例图..... 19

Figure 4.2—乘客用例图..... 20

Figure 4.3—司机用例图..... 21

Figure 4.4—业务监测人员用例图..... 21

Figure 4.5—系统管理人员用例图..... 22

Figure 5.1—系统分层图..... 25

Figure 5.2—系统包图..... 27

Figure 5.3—人机交互模块中类的分布..... 28

Figure 5.4—打车操作模块中类的分布..... 28

Figure 5.5—接单操作模块中类的分布..... 29

Figure 5.6—付款模块中类的分布..... 29

Figure 5.7—业务监测模块中类的分布..... 30

Figure 5.8—打车处理模块中类的分布..... 30

Figure 5.9—接单处理模块中类的分布..... 30

Figure 5.10—业务逻辑层中类的分布..... 31

Figure 5.11—司机管理模块模块中类的分布..... 31

Figure 5.12—乘客信息管理模块中类的分布..... 32

Figure 5.13—用户信息审核管理模块中类的分布..... 32

Figure 6.1—乘客打车进程流程图.....	34
Figure 6.2—司机接单进程流程图.....	35
Figure 6.3—业务监测人员工作流程图.....	36
Figure 6.4—系统管理人员工作流程图.....	37
Figure 6.5—乘客打车进程有关类图.....	38
Figure 6.6—司机接单进程有关类图.....	39
Figure 6.7—业务监测人员进程有关类图.....	41
Figure 6.8—系统管理人员进程有关类图.....	42
Figure 7.1—设计部署图.....	43

A.2 随文所附表

Table 1.1—术语解释表.....	7
Table 1.2—缩写解释表.....	7
Table 2.1—C/S 模式对质量属性的影响.....	10
Table 3.1—体系约束表.....	15