# UML软件设计课程论文

燕山学院2021-2022-1学期

班级：2019信息管理与信息系统

学号：20190501002

姓名：尹勇胜

# 摘要

本次论文主要介绍了计算机语言的发展历程和提出的各种概念，以及他们的作用，在软件设计的发展过程中，形成了面向过程和面向对像的软件设计方法，本次课程论文主演介绍了面向对象的软件设计方法的设计过程和优势。

面向对象的软件工程，同传统的面向过程的软件工程相比，在需求的获取、系统分析、设计和实现方面都有着很大的区别。UML是00A和00D的常用工具。使用UML来构建软件的面向对象的软件工程的过程，就是一个对系统进行不断精化的建模的过程。这些模型包括用例模型、分析模型、设计模型，然后，我们需要使用具体的计算机语言来建立系统的实现模型。当然，在整个软件工程中，我们还需要建立系统的测试模型，以保证软件产品的质量。

在发展过程中，为了代码的重用，更有效率的开发，逐渐形成了更加规范的设计要求。

应用软件系统,就其本质来说，是使用计算机对现实世界进行的数字化模拟。应用软件的制造过程，按照UML的方法，就是建立这一些列模型的过程。

关于这个网上订餐系统，其功能要求比较负责，真实的过程以我的水平可能没有办法去做深入的了解，所以我将一些集成的系统，划分成八个子系统来进行建模

# 一丶软件编程和设计简述

## 软件编程与设计的发展

#### 机器语言

计算机使用的是由“0”和“1”组成的二进制数，二进制编码方式是计算机语言的基础。计算机发明之初，科学家只能用二进制数编制的指令控制计算机运行。每一条计算机指令均由一组“0”、“1”数字，按一定的规则排列组成，若要计算机执行一项简单的任务，需要编写大量的这种指令。这种有规则的二进制数组成的指令集，就是机器语言也称为指令系统。不同系列的CPU，具有不同的机器语言，如目前个人计算机中常用AMD公司的系列CPU和Intel公司的系列CPU，具有不同的机器语言。机器语言是计算机唯一能识别并直接执行的语言，与汇编语言或高级语言相比，其执行效率高。但其可读性差，不易记忆;编写程序既难又繁，容易出错;程序调试和修改难度巨大，不容易掌握和使用。此外，因为机器语言直接依赖于中央处理器，所以用某种机器语言编写的程序只能在相应的计算机上执行，无法在其他型号的计算机上执行，也就是说，可移植性差。

#### 汇编语言

为了减轻使用机器语言编程的痛苦，20世纪50年代初，出现了汇编语言。汇编语言用比较容易识别、记忆的助记符替代特定的二进制串。下面是几条Intel80x86的汇编指令:ADDAX，BX;表示将寄存器AX和BX中的内容相加，结果保存在寄存器AX中。SUB AX，NUM;表示将寄存器AX中的内容减去NUM，结果保存在寄存器AX中。MOVAX，NUM;表示把数NUM保存在寄存器AX中。通过这种助记符，人们就能较容易地读懂程序，调试和维护也更方便了。但这些助记符号计算机无法识别，需要一个专门的程序将其翻译成机器语言，这种翻译程序被称为汇编程序。

汇编语言的一条汇编指令对应一条机器指令，与机器语言性质上是一样的，只是表示方式做了改进，其可移植性与机器语言一样不好。总之，汇编语言是符号化的机器语言，执行效率仍接近于机器语言，因此，汇编语言至今仍是一种常用的软件开发工具。

* 1. **C++语言**

C++是一种高级语言，是面向对象的语言，它作为一种高级语言在形式商接近于算术语言和自然语言，程序员编写方法，使用该语言编写的程序有易读而且通用性强

它是C语言的继承，尽量**兼容C语言**，既保持了C语言的简洁和高效，可以像C语言那样进行结构化程序设计，同时也增强了C语言对类型的处理。

加入了**面向对象**的特征，可以进行以抽象数据类型为特点的基于对象的程序设计，还可以进行以继承和多态为特点的面向对象的程序设计。

**与C语言相比，C++语言的优点**：从程序运行的稳定性来说，C++语言比C语言更安全，它支持过程化编程、面向对象编程和泛型编程。因为能够支持面向对象的开发方式，所以C++语言的应用领域更加广泛。C++语言可以运行于多种平台上，如Windows、MAC操作系统及UNIX等多种版本。C++语言中加入了**面向对象**的概念，虽然C语言的语法绝大多数被保留在C++语言中，但C++的程序结构和C语言的程序机构存在很大差别。C++语言对C语言做了很多改进，C++语言相对于C语言最根本的变化是引进了**类和对象的概念**。

#### 变量，数组，结构（数据组织）

**变量：**由于变量让你能够把程序中准备使用的每一段数据都赋给一个简短、易于记忆的名字，因此它们十分有用。变量可以保存程序运行时用户输入的数据（如使用InputBox函数在屏幕上显示一个对话框，然后把用户键入的文本保存到变量中）、特定运算的结果以及要在窗体上显示的一段数据等。

简而言之，变量是用于跟踪几乎所有类型信息的简单工具。变量声明后没有赋值的话，编译器会自动提示并赋予默认值。变量是一种使用方便的占位符，用于引用计算机内存地址，该地址可以存储Script运行时可更改的程序信息。例如，可以创建一个名为Click Count的变量来存储用户单击Web页面上某个对象的次数。

使用变量并不需要了解变量在计算机内存中的地址，只要通过变量名引用变量就可以查看或更改变量的值。

**数组：**数组（Array）是有序的元素序列。 [1] 若将有限个类型相同的变量的集合命名，那么这个名称为数组名。组成数组的各个变量称为数组的分量，也称为数组的元素，有时也称为下标变量。用于区分数组的各个元素的数字编号称为下标。数组是在程序设计中，为了处理方便， 把具有相同类型的若干元素按有序的形式组织起来的一种形式。 [1] 这些有序排列的同类数据元素的集合称为数组。

数组是用于储存多个相同类型数据的集合。

**数据结构：**数据结构(data structure)是带有结构特性的数据元素的集合，它研究的是数据的逻辑结构和数据的物理结构以及它们之间的相互关系，并对这种结构定义相适应的运算，设计出相应的算法，并确保经过这些运算以后所得到的新结构仍保持原来的结构类型。简而言之，数据结构是相互之间存在一种或多种特定关系的数据元素的集合，即带“结构”的数据元素的集合。“结构”就是指数据元素之间存在的关系，分为逻辑结构和存储结构。 [2]

数据的逻辑结构和物理结构是数据结构的两个密切相关的方面，同一逻辑结构可以对应不同的存储结构。算法的设计取决于数据的逻辑结构，而算法的实现依赖于指定的存储结构。 [2]

数据结构的研究内容是构造复杂软件系统的基础，它的核心技术是分解与抽象。通过分解可以划分出数据的3个层次；再通过抽象，舍弃数据元素的具体内容，就得到逻辑结构。类似地，通过分解将处理要求划分成各种功能，再通过抽象舍弃实现细节，就得到运算的定义。上述两个方面的结合可以将问题变换为数据结构。这是一个从具体（即具体问题）到抽象（即数据结构）的过程。然后，通过增加对实现细节的考虑进一步得到存储结构和实现运算，从而完成设计任务。这是一个从抽象（即数据结构）到具体（即具体实现）的过程

#### 子过程，函数（代码重用和维护）

**子过程：**在一个加工程序中，如果其中有些加工内容完全相同或相似，为了简化程序，可以把这些重复的程序段单独列出，并按一定的格式编写成子程序。主程序在执行过程中如果需要某一子程序，通过调用指令来调用该子程序，子程序执行完后又返回到主程序，继续执行后面的程序段。

子过程（subroutine）是一个概括性的术语，任何高级程序所调用的程序，都被称为子程序。它经常被使用在汇编语言层级上。子程序的主体（body）是一个代码区块，当它被调用时就会进入运行。

能被其他程序调用，在实现某种功能后能自动返回到调用程序去的程序。其最后一条指令一定是返回指令，故能保证重新返回到调用它的程序中去。也可调用其他子程序，甚至可自身调用

**函数：**函数是指一段可以直接被另一段程序或代码引用的程序或代码。也叫做子程序、（OOP中）方法。

一个较大的程序一般应分为若干个程序块，每一个模块用来实现一个特定的功能。所有的高级语言中都有子程序这个概念，用子程序实现模块的功能。在C语言中，子程序是由一个主函数和若干个函数构成的。由主函数调用其他函数，其他函数也可以互相调用。同一个函数可以被一个或多个函数调用任意多次。

在程序设计中，常将一些常用的功能模块编写成函数，放在函数库中供公共选用。要善于利用函数，以减少重复编写程序段的工作量。

#### 1.6类：属性和方法的组合

类是面向对象语言的程序设计中的概念，是面向对象编程的基础。

类的实质是一种引用数据类型，类似于 byte、short、int（char）、long、float、double 等基本数据类型，不同的是它是一种复杂的数据类型。因为它的本质是数据类型，而不是数据，所以不存在于内存中，不能被直接操作，只有被实例化为对象时，才会变得可操作。

类是对现实生活中一类具有共同特征的事物的抽象。如果一个程序里提供的数据类型与应用中的概念有直接的对应，这个程序就会更容易理解，也更容易修改。一组经过很好选择的用户定义的类会使程序更简洁。此外，它还能使各种形式的代码分析更容易进行。特别地，它还会使编译器有可能检查对象的非法使用。 [2]

类的内部封装了属性和方法，用于操作自身的成员。类是对某种对象的定义，具有行为（behavior），它描述一个对象能够做什么以及做的方法（method），它们是可以对这个对象进行操作的程序和过程。它包含有关对象行为方式的信息，包括它的名称、属性、方法和事件。

类的构成包括成员属性和成员方法（数据成员和成员函数）。数据成员对应类的属性，类的数据成员也是一种数据类型，并不需要分配内存。成员函数则用于操作类的各项属性，是一个类具有的特有的操作，比如“学生”可以“上课”，而“水果”则不能。类和外界发生交互的操作称为接口。

#### 1.7复杂大系统软件设计的难点和问题

**体系结构设计**

体系结构是指软件系统的基本和主体的形态，也就是软件系统中最根本的东西。

**合适性**

系统设计之源是需求，这是由商业目标决定的。高水平的设计高就高在“设计出恰好满足客户需求的软件，并且使开发方和客户方获取最大的利益，而不是不惜代价设计出最先进的软件。评估体系结构好不好的第—个指标就是“合适性”，即体系结构是否适合于软件的“功能性需求”和“非功能性需求”。大部分软件开发人员天生有使用新技术的倾向，而这种倾向对开发商业产品而言可能是不利的。

**结构稳定性**

体系结构是系统设计的第一要素，详细设计阶段的工作如用户界面设计、数据库设计、模块设计、数据结构与算法设计，等等，都是在体系结构确定之后开展的。高水平的设计应当能够分析需求文档，判断出哪些需求是稳定不变的，哪些需求是可能变动的。于是根据那些稳定不变的需求设计体系结构，而根据那些可变的需求设计软件的“可扩展性”。

**可扩展性**

可扩展性是指软件扩展新功能的容易程度。可扩展性越好，表示软件适应“变化”的能力越强。由于软件是“软”的，是否它天生就容易扩展新功能？这要视软件的规模和复杂性而定，但是并不是任何软件在设计的时候都要考虑可扩展性的。

**可复用性**

# 二丶面向对象的软件设计

## 简介

### 面向对象方法是一种把面向对象的思想应用于软件开发过程中，指导开发活动的系统方法是建立在“对象”概念基础上的方法学。对象是由数据和容许的操作组成的封装体，与客观实体有直接对应关系，一个对象类定义了具有相似性质的一组对象。继承性是对具有层次关系的类的属性和操作进行共享的一种方式。所谓面向对象，就是基于对象概念，以对象为中心,以类和继承为构造机制,来认识、理解、刻画客观世界和设计、构建相应的软件系统。

UML统一了符号体系，并从其他方法和工程实践中吸收了许多经过际检验的概念和技术。这些方法的提出，标志着面向对象方法逐步发展成为一类完整的法学和系统化的技术体系。有关抽象数据类型的基础研究，为面向对象开发方法提供了步的理论。面向对象方法作为一种独具优越性的新方法,引起计算机界广泛的关注和高度重视

## 优点

1. 强调从现实世界中客观存在的事物(对象)出发,来认识问题域和构造系统，使系销

的事物些类和能更准确地反映问题域。

1. 运用人类日常的思维方法和原则(体现在面向对象方法的抽象、分类、继承、封装

、消息等基本原则)进行系统开发,有利于发挥人类的思维能力,有效控制系统复杂性

(3) 对象的概念贯穿于开发全过程,使各个开发阶段的系统成分具有良好的对应关系即所无显著提高系统的开发效率与质量,并大大降低系统维护的难度。

(4) 对象概念的一致性使参与系统开发的各类人员在开发的各阶段具有共同语言，有效地改善了人员之间的交流和协作。

(5) 对象的相对稳定性和对易变因素隔离,增强了系统对环境的适应能力。

(6) 对象、类之间的继承关系和对象的相对独立性,对软件复用提供了强有力的支持。

## 面向对象设计的过程

设计OOD模型的五个部分为问题域部分的设计，人机交互部分的设计，控制驱动部分的设计，数据管理部分的设计，构建部署的设计

#### 问题域部分的设计

开始进行设计工作之前，设计者应该连接奔项目预计要使用的编程语言，可用的软构件库，以及程序员的编程经验

通常ODD仅需从实例的角度对问题模型做一些补充和修改，当问题域子系统过于复杂，应该进一步分解成若干个更小的子系统

可能对模型做的补充和修改：

1. 调整需求
2. 重用已有的类
3. 把问题域类组合在一起
4. 调整继承层次

#### 人机交互的设计

在OOD过程中，则应该对系统的人机交互的子系统进行详细的设计，以确定人机交互的细节。

人机交互子系统的策略：

1. 分类用户
2. 描述用户
3. 设计命令层次

#### 控制驱动部分的设计

确定哪些是必须同时动作的对象，哪些是相互排斥的对象，人后进一步设计控制驱动子系统。

1. 分析并发性
2. 实际控制驱动子系统

设计控制驱动子系统，包括确定各类控制列兵吧控制刘分配给适当硬件去执行

1. 确定时间驱动型控制流
2. 确定始终驱动型控制流
3. 确定优先控制流
4. 确定关键控制流
5. 确定协调控制流
6. 尽量减少控制流数
7. 确定资源需求

#### 数据管理部分的设计

数据管理子系统是系统存储或检索对象的基本设施，建立在某种数据存储管理子系统之上，并且隔离了数据存储管理模式的影响

1. 选择数据存储管理模式
2. 设计数据管理子系统
3. 构件部署设计

部署设计用来设计软件开发过程中形成的软件制品，软件运行平台中的物理: 卡的谈 信方式以及软件制品到相应硬件节点的部署或映射。一般需要考虑以下问题：

(1)最终开发完成的软件包括哪些制品形式? 人借阅

(2)软件运行环境存在哪些类型的物理节点?

(3)不同节点之间的连接和通信形式是什么? 后，输 图书，

(4)软件制品应该如何在物理节点上进行部署,即部署映射关系是什么?

# 三丶UML软件设计的规范和主要组成

## 软件设计规范的提出

20世纪70年代，Frederick P. Books.,Jr,Dijkstra等软件工程提出了概念完整性，结构化程序设计，模块化，信息隐藏，封装等软件结构相关的重要原则。

软件设计是从软件需求规格说明出发，形成软件的具体设计方案的过程，也就是说在需求分析阶段明确软件是“做什么”的基础上，解决软件“怎么做”的问题。结构化设计方将把软件设计分为概要设计和详细设计两个阶段。概要设计的主要任务是，通过分析需求规格说明对软件进行功能分解，确定软件的总体结构；详细设计阶段确定每个模块功能所需要的算法和数据结构，设计出每个模块的逻辑结构。软件设计阶段结束时的工作成果是软件设计说明书，它描述软件系统的组成模块结、模块间的调用关系，以及每个模块的输入、输出和详细的过程描述

## 软件外包对设计规范的需求

合适性

即体系结构是否适合于软件的功能性需求”和“非功能性需求”。高水平的设计师高就高

在“设计出恰好满足客户需求的软件，并且使开发方和客户方获取最大的利益。 而不是不惜

代价设计出最先进的软件。

·结构稳定性

详细设计阶段的工作如用户界面设计、数据库设计、模块设计、数据结构与算法设计等

等，都是在体系结构确定之后开展的，而编程和测试则是更后面的工作，因此体系结构应在

一定的时间内保持稳定。

软件开发最怕的就是需求变化，但“需求会发生变化”是个无法逃避的现实。人们希望在

需求发生变化时，最好只对软件做些皮皮毛毛的修改，可千万别改动软件的体系结构。如果

当需求发生变化时，程序员不得不去修改软件的体系结构，那么这个软件的系统设计是失败

的。

高水平的设计师应当能够分析需求文档，判断出哪些需求是稳定不变的，哪些需求是可能变

动的。于是根据那些稳定不变的需求设计体系结构，而根据那些可变的需求设计软件的“可扩展性"

可扩展性

可扩展性是指软件扩展新功能的容易程度。可扩展性越好，表示软件适应变化”的能力

越强。

可复用性

由经验可知，通常在一个新系统中，大部分的内容是成熟的，只有小部分内容是创新的。

般地可以相信成熟的东西总是比较可靠的(即具有高质量)，而大量成熟的工作可以通过

复用来快速实现(即具有高生产率)。

## 设计转换为代码的需求

为了解决某种需求/需要，我们计划设计一个软件系统。通过与利益相关者进行交流之后，确认了新的系统是有必要存在的。于是，我们产生了一系列的概念和想法，并通过诸如愿景梳理、用户分析等一系列的想法，我们将这些想法明确下来。

在开始软件开发前，我们定义好了产品是什么，随后梳理出了用户故事地图。我们定义了在什么场景下，需要哪些用户，在哪里做些什么事情，并对这些行为做出响应。有了这些定义之后，我们作为这个系统的架构设计师，便开始思考需要保存、显示哪些数据，才能完成这个业务目标。

在进入开发之前，这些想法设计等，都被明确为软件需求，简称为需求。

需求代码化，即将软件开发需求抽象为特定的领域语言，并使用管理代码一样的方式来管理需求，追踪需求的变化 。同时，为通过新的 API 来对接版本管理系统，以可视化需求，演变为看板代码化。

它具备这么一些特征：

使用标记语言编写内容。如 Cucumber

可通过版本控制系统进行版本控制。如 git

与编程一致的编程体验，还可以作为测试代码的一部分

支持集成到现有的看板系统中

可集成到 IDE 中协作

支持 Git 转换为 CRUD 接口

为了进一步实现万物即代码，它还具备这么一些特征：

可对需求进行重构

可转化为设计语言

# 四丶UML软件设计案例

## UML设计软件与StarUML简介

### UML设计软件简介

外卖订单系统是服务于餐馆外卖活动的一个简单的信息系统，开发该系统主要希望实现扩大本餐馆宣传、缩短顾客订餐时间、减少订餐错误、便于订单统计分析等，最终达到扩大餐馆影响力、提高餐馆外卖业务效率、实现一定程度的决策支持的目的。

该系统主要有八个子系统，分别是:订餐者管理系统、商家管理系统，订单管理系统，账务管理系统，收银系统，店面管理系统，配送员管理系统，信息管理系统

订餐者系统：面向订餐者，主要实现用户登录，注册，选餐，付款，确认收餐，评论等功能

商家管理系统：面向商家，主要实现登录，注册，更新菜品信息，接收等功能

订单管理系统：面向，顾客，商家，配送员，主要实现订单的状态和信息的更新，历史订单浏览与查询，订单统计分析等功能

账务管理系统：面向账务人员，实现登录，验证，生成账单，核实收支功能

收银系统：面向支付平台，主要实现收取用户支付，确认支付，判断支付状态等功能

店面管理系统：面向店铺管理员，实现店铺的录入，更新已有店铺信息，建立店铺评价档案，监督店铺的合法性等功能

配送员管理系统：面向配送员，主要实现接受订单，确认取餐，确认配送等功能

信息管理系统：面向管理员，主要实现对用户信息的增删查和商家的增删查

### StarUML简介

在该项目中用到的工具StarUML，是一种创建[UML类图](https://baike.baidu.com/item/UML%E7%B1%BB%E5%9B%BE/6842152" \t "https://baike.baidu.com/item/staruml/_blank)，生成类图和其他类型的[统一建模语言](https://baike.baidu.com/item/%E7%BB%9F%E4%B8%80%E5%BB%BA%E6%A8%A1%E8%AF%AD%E8%A8%80)(UML)图表的工具。StarUML是一个开源项目之一发展快、灵活、可扩展性强。

## 软件项目简介

#### 背景介绍

随着网络技术的飞速发展，人们的生活也越来越追求方便化。经过观察，发现整个大学城的学生对平常订餐需求很大，但他们订餐的方式都是比较原始的申话订餐。而各个餐饮店也是各自为战，自己接电话，记录订单需求，自己配送。这样做效率很低，利润薄，而且信息不流畅。所以我决定为大学生提供一个平台 ---网上订餐系统。在网上给申请的商家一个虚拟店面，可以在上面挂上该商家的名称，饭菜的图片和价格等信息，让订餐者可以方便地订餐，还可以对商家的睿饮进行评价，由系统生成评价档案以供其他人参考等，而商家后期只负责做饭菜并安排人配送。此外，需要定期对商家进行卫生安全监察，生成商家监察档案，并以此为依据来决定商家的去留等。

## 系统分析

2.1获取需求

大学生网上订餐系统主要有以下几方面需求:

**1)订餐者**

订餐者首先需要注册一个账号用于系统登录，登录后可以查看店铺信息，并选中某一店铺后进入其餐饮信息界面，最终选中所需餐饮，下订单。当然用餐后还可以对此餐饮进行评价。

**2）商家**

商家首先需要申请一个网上店铺，当申请通过后，登录到系统中，可以核实订单并安排配送，然后对本店的餐饮信息进行更新。

**3）订单管理员**

当订餐者下订单后，订单管理员需及时生成订单，如果订餐者对订单有所更改时，订单管理员也要及时对数据进行更新。

**4）店铺管理员**

当商家申请通过时，店铺管理员需要及时录入店铺信息，并为其设立店面、建立客户评价档案、商家监察档案。当商家增加、修改、删除其餐饮信息时，店铺管理员需及时对数据进行更新，以便其他人订餐。如果订餐者对某餐饮店的某餐饮进行评价后，店铺管理员需及时更新评价档案。

**5）信息管理员**

系统管理员主要完成对商家和订餐者信息的管理

**6）收银员**

主要完成收取用户支付，和财务管理，确认未支付的功能

1. **配送员**

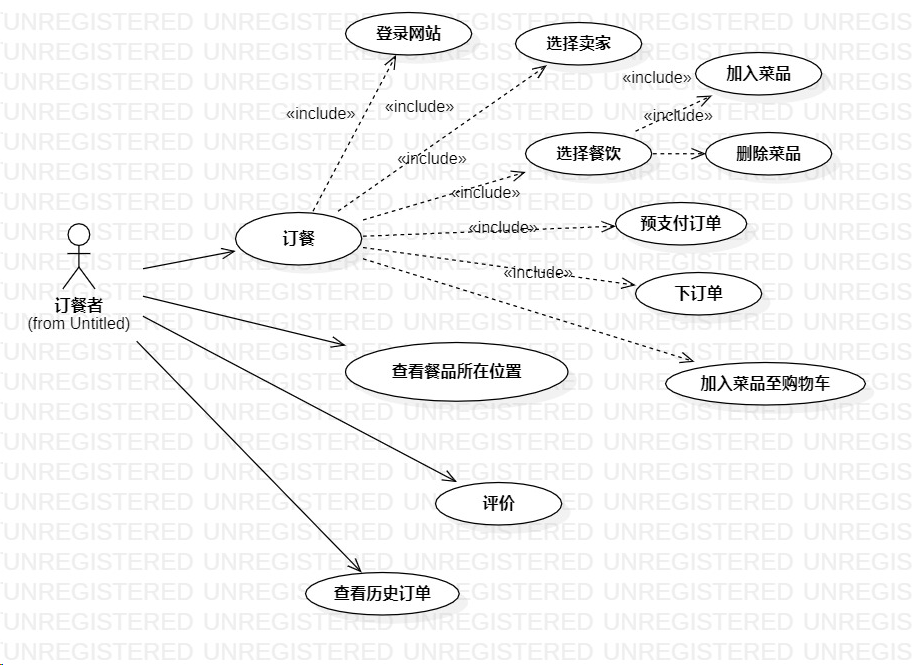
当产生新订单，配送员选择接受时，对餐的状态进行更新

1. **账务管理员**

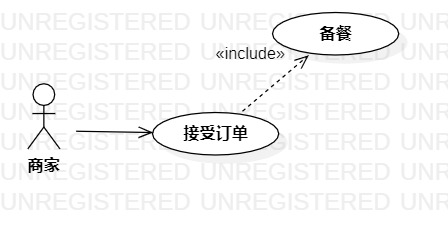
提取所有订单的信息，核实收支

## 2.2 各个子系统的用例图

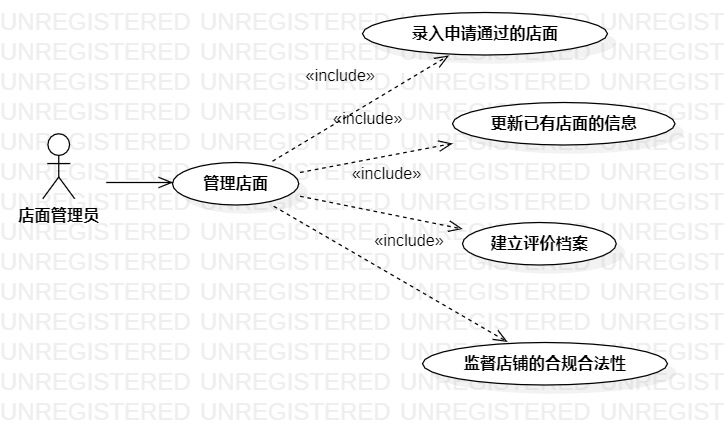
#### 订餐者用例图：



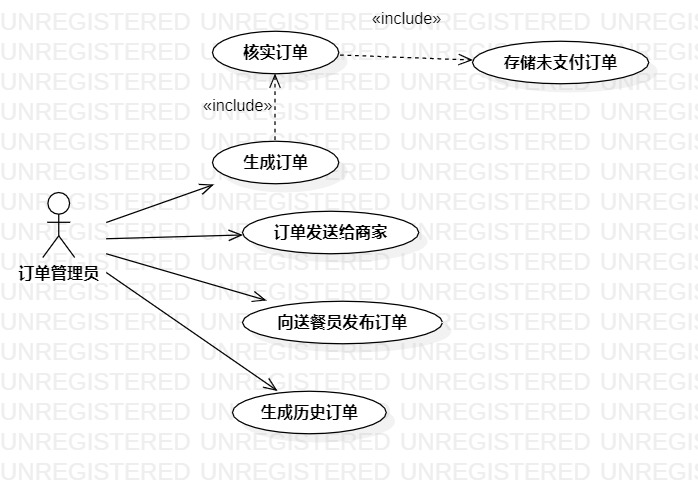
#### 商家用例图：



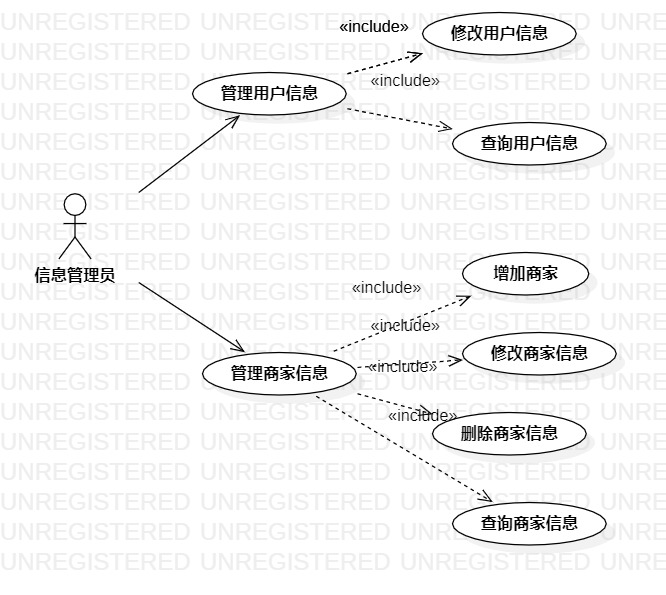
店铺管理用例图：



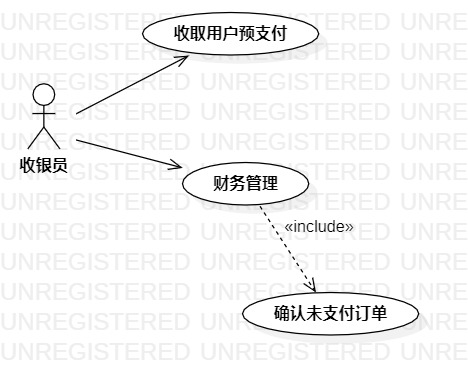
#### 订单管理用例图：



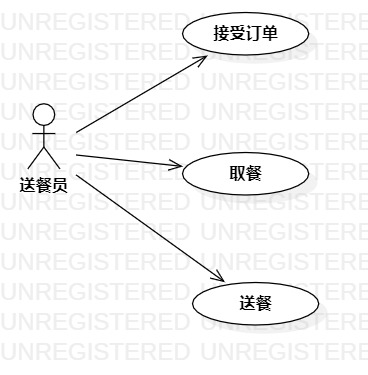
信息管理用例图：



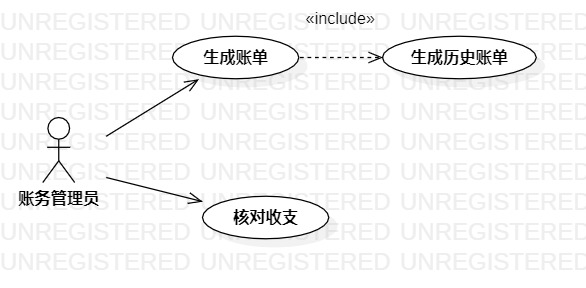
#### 收银员用例图：



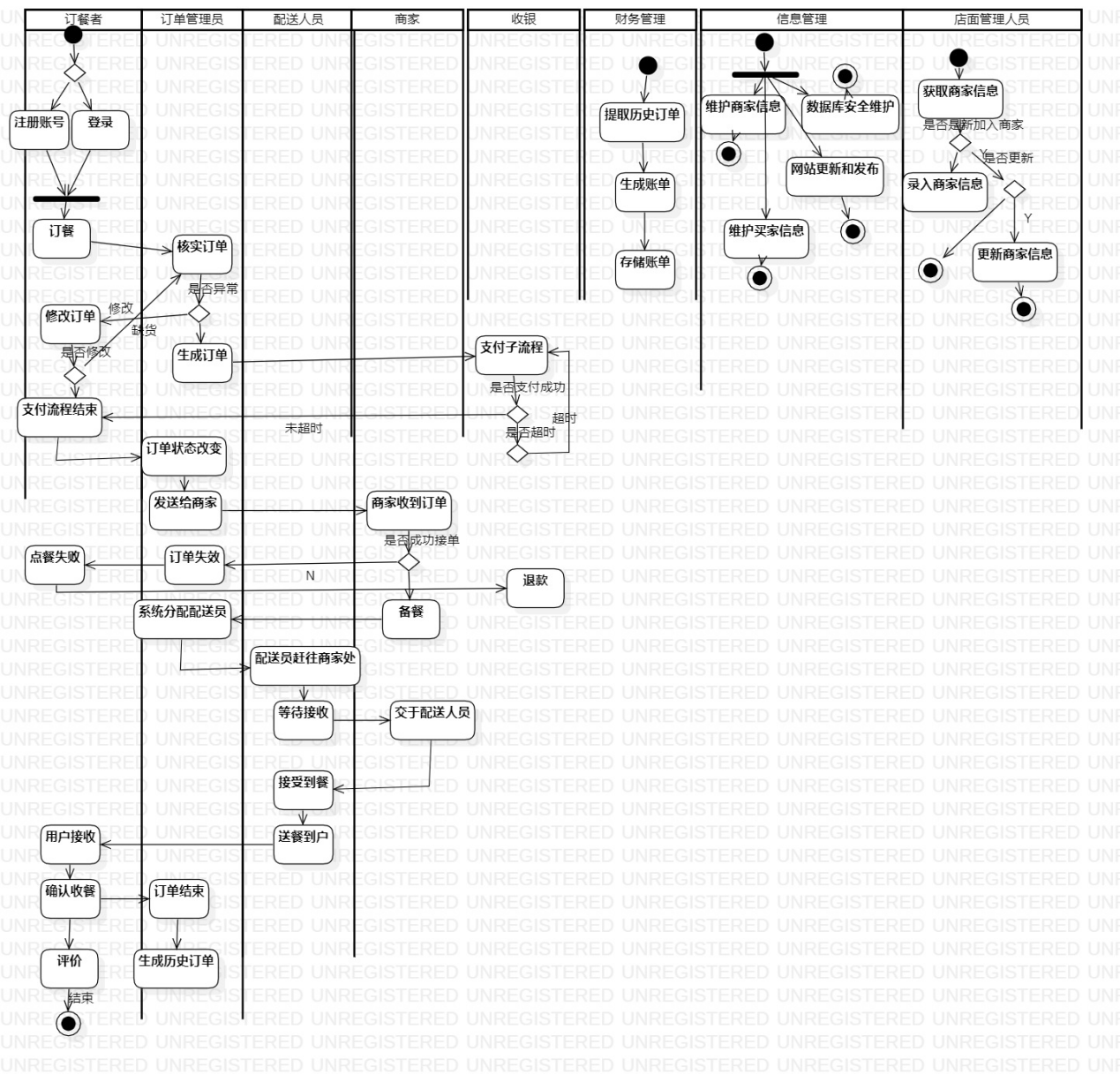
配送员用例图：



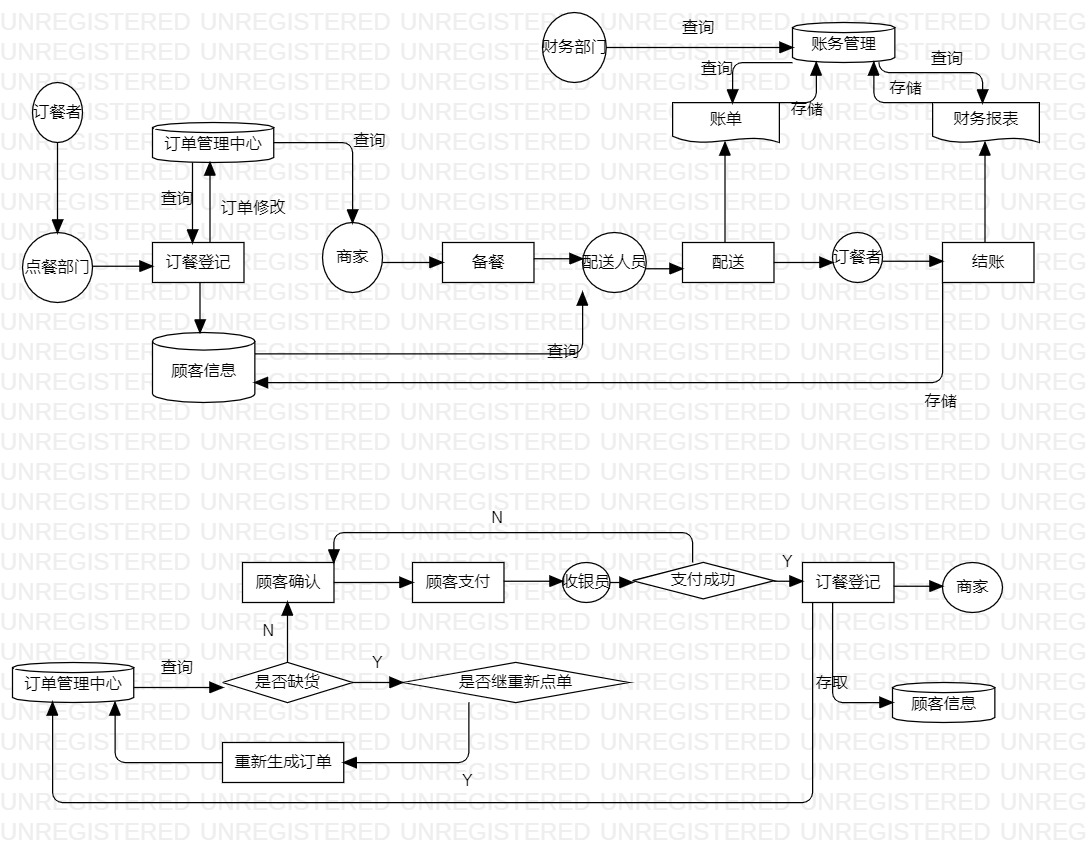
#### 账务管理用例图：



#### 2.3 系统活动图



#### 2.4 系统业务流程图



## 2.5顺序图

#### 2.5.1 订餐者订餐时序图

(1)订餐者注册一个系统登录账号;

(2)在登录界面输入相应的用户名和密码进行登录

(3)系统展示有关店铺信息;

(4)订餐者选择店铺;

(5)系统展示该店铺的有关餐饮信息;

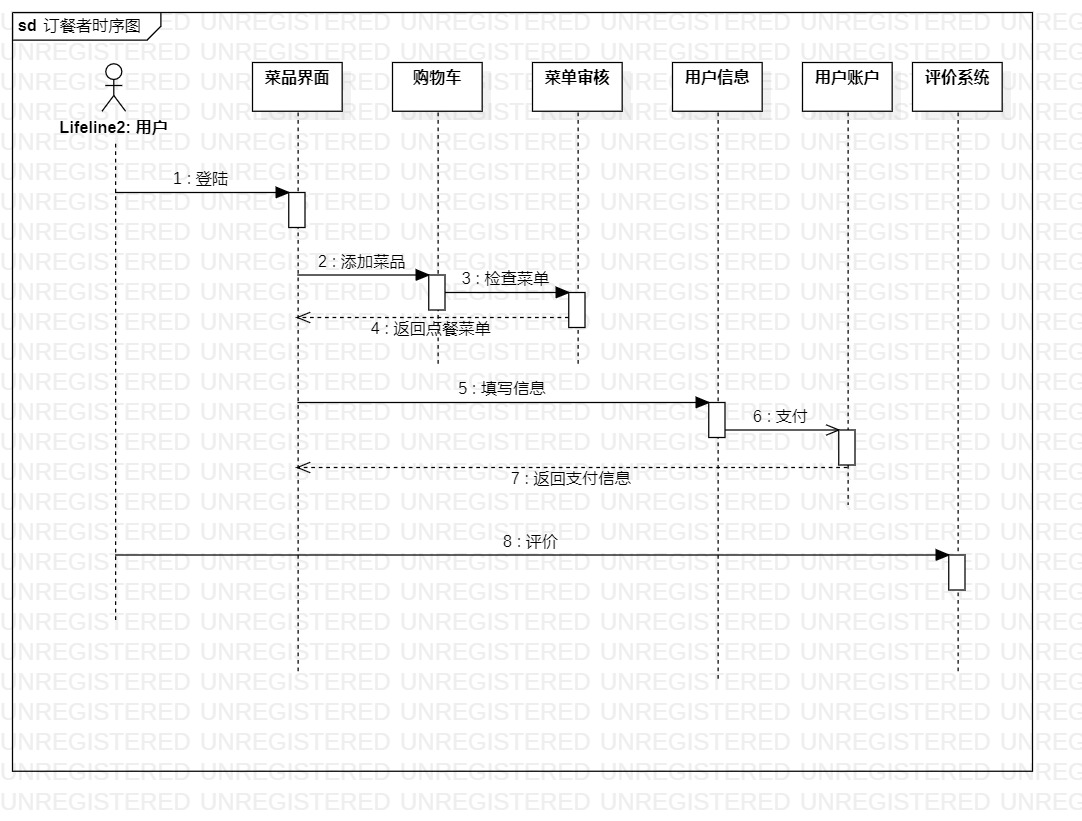
(6)订餐者浏览信息并选择所需餐饮;

(7)订餐者下订单;

(8)下订单成功后，系统更新餐饮信息;

(9)退出系统;

(10)用餐后对餐饮进行评价。



#### 2.5.2商家管理店铺

(1)商家申请一个网上虚拟店铺;

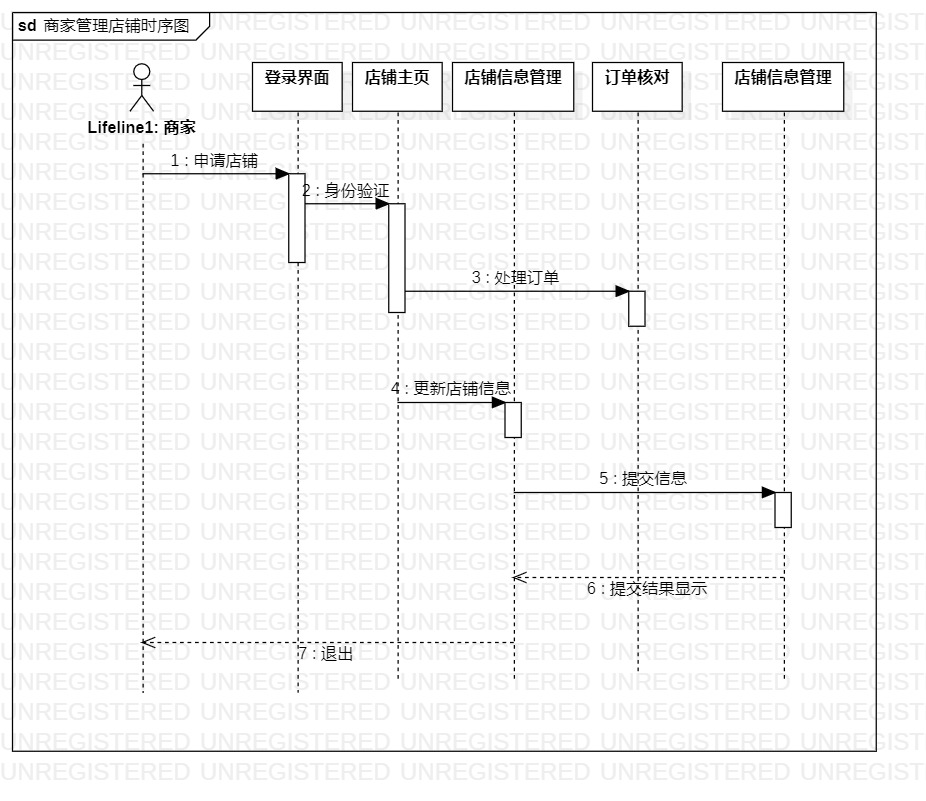
(2)申请成功后，在登录界面输入自己的用户名和密码进行身份验证;

(3)商家处理订单;

(4)更新店铺相关餐饮信息;

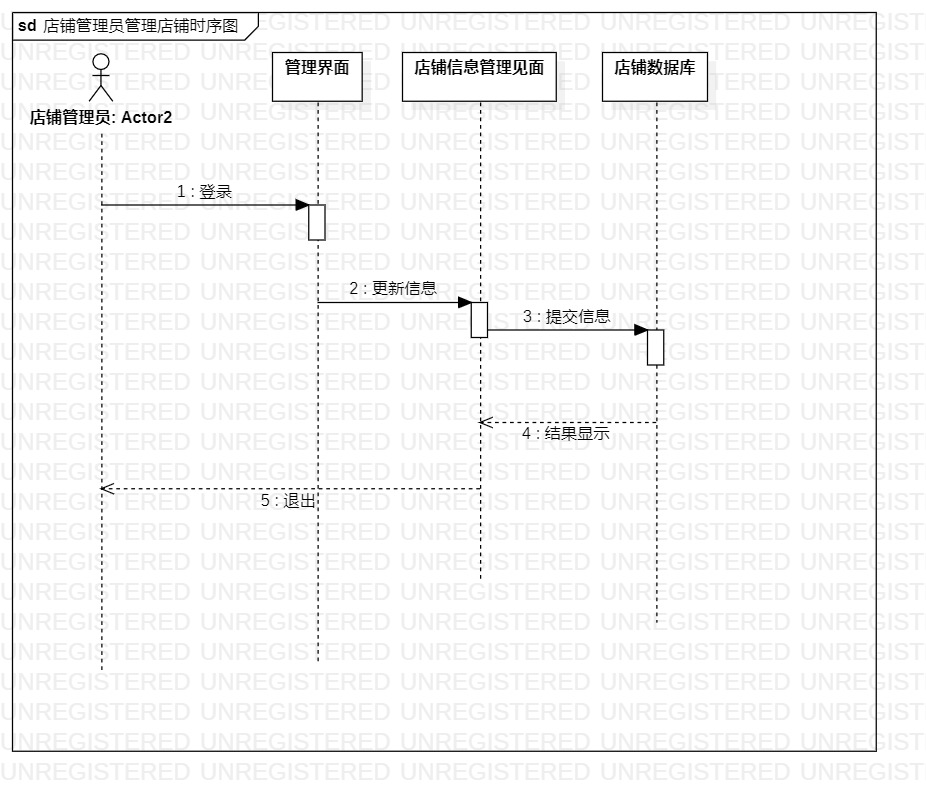
(5)提交更新数据;

(6)系统将提交的结果显示给商家;(8)提交成功，退出系统。



#### 2.5.3店铺管理员管理店铺

1. 店铺管理员登录系统进入管理界面;
2. 对店铺的信息进行更新;
3. 提交更新数据:
4. 系统将提交结果显示给店铺管理员:
5. 提交成功，退出系统。



#### 2.5.4 店铺管理员管理店铺信息

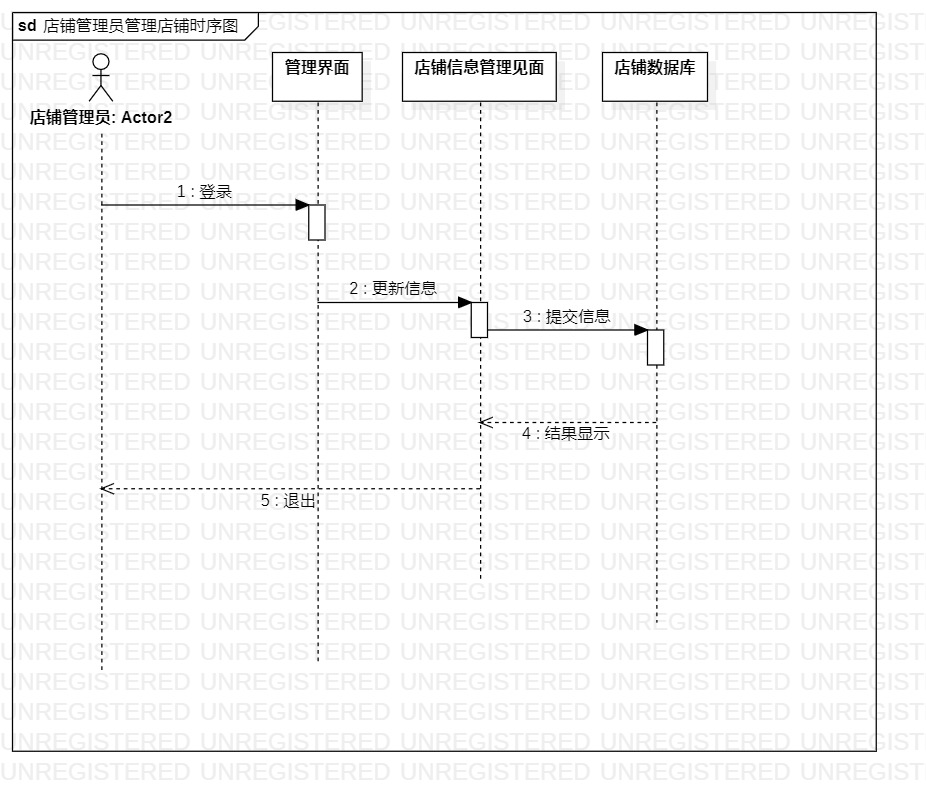
1） 店铺管理员登录系统进入管理界面;

2）对店铺的信息进行更新;

3）提交更新数据:

4）系统将提交结果显示给店铺管理员:

5）提交成功，退出系统



#### 2.5.5 店铺管理员建立评价档案

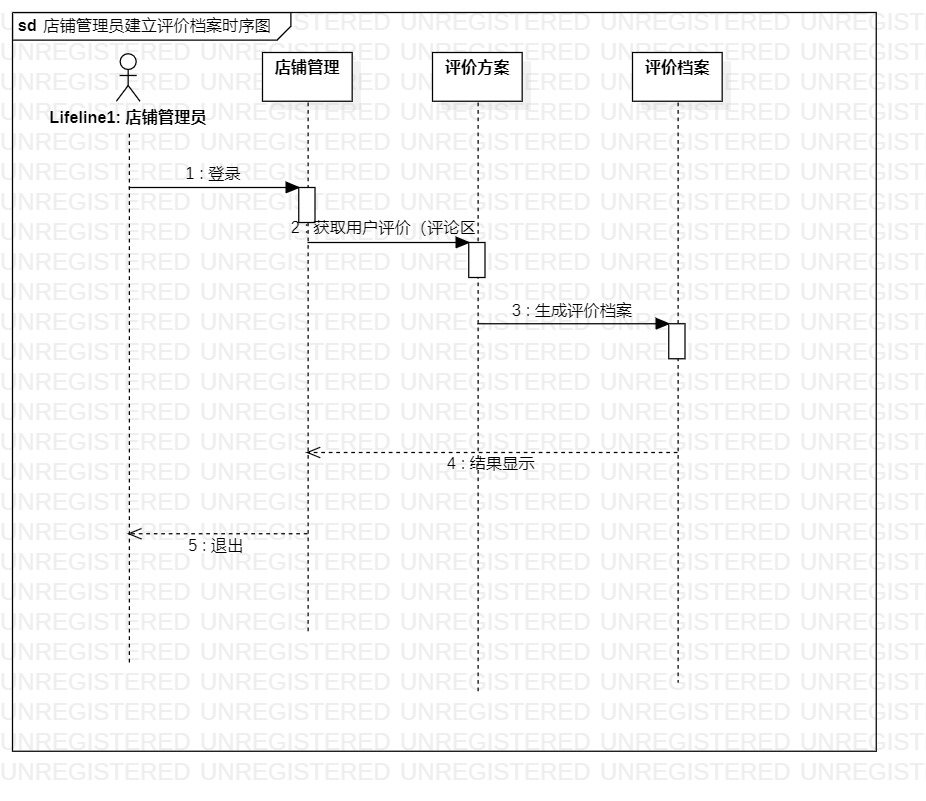
(1)店铺管理员登录到系统;

(2)对订餐者的评价进行核实并筛选;

(3)生成最终的评价档案;

(4)系统将结果显示给店铺管理员;

(5)退出系统。



#### 2.5.6 店铺管理员建立商家监察档案

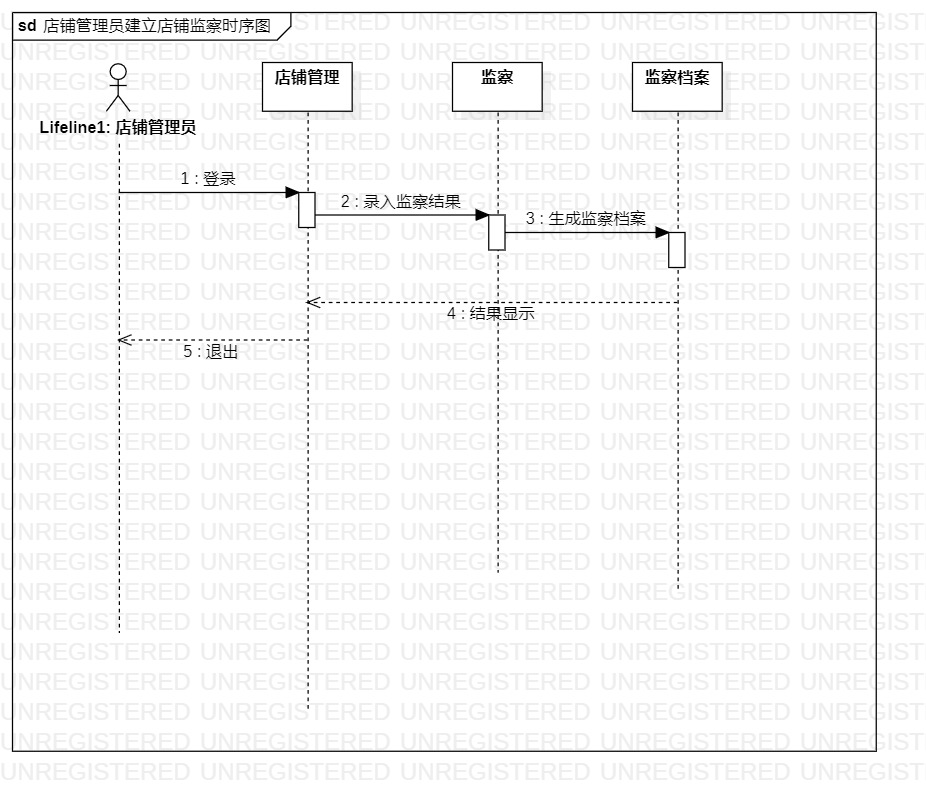
(1)店铺管理员登录到系统:

(2)录入通过检查得出的监察结果;

(3)生成监察档案;

(4)系统将结果显示给店铺管理员;

(5)退出系统



#### 2.5.7 账务管理员管理店铺

（1）账务物管理员登录验证账务管理

（2）验证

（3）查看账务信息

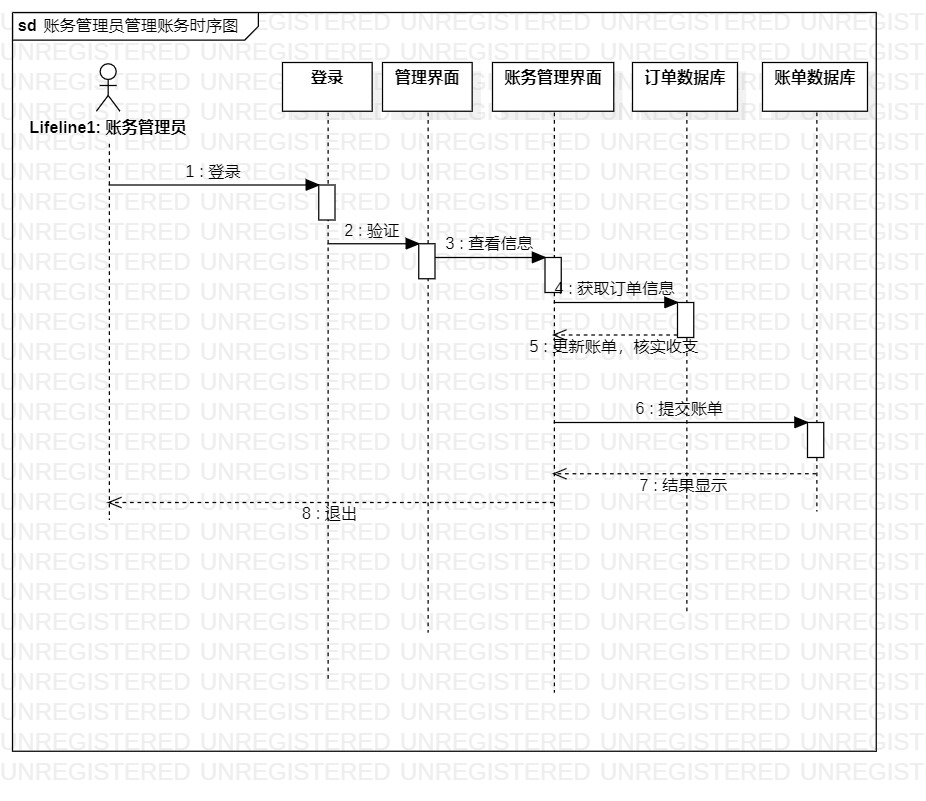
（4）获取订单信息

（5）更新账单，核实收支

（6）提交账单

（7）结果显示

（8）退出



#### 2.5.8 信息管理员管理订餐者信息

(1)系统管理员注册一个自己的登录账号;

(2)输入用户名和密码进行身份验证;

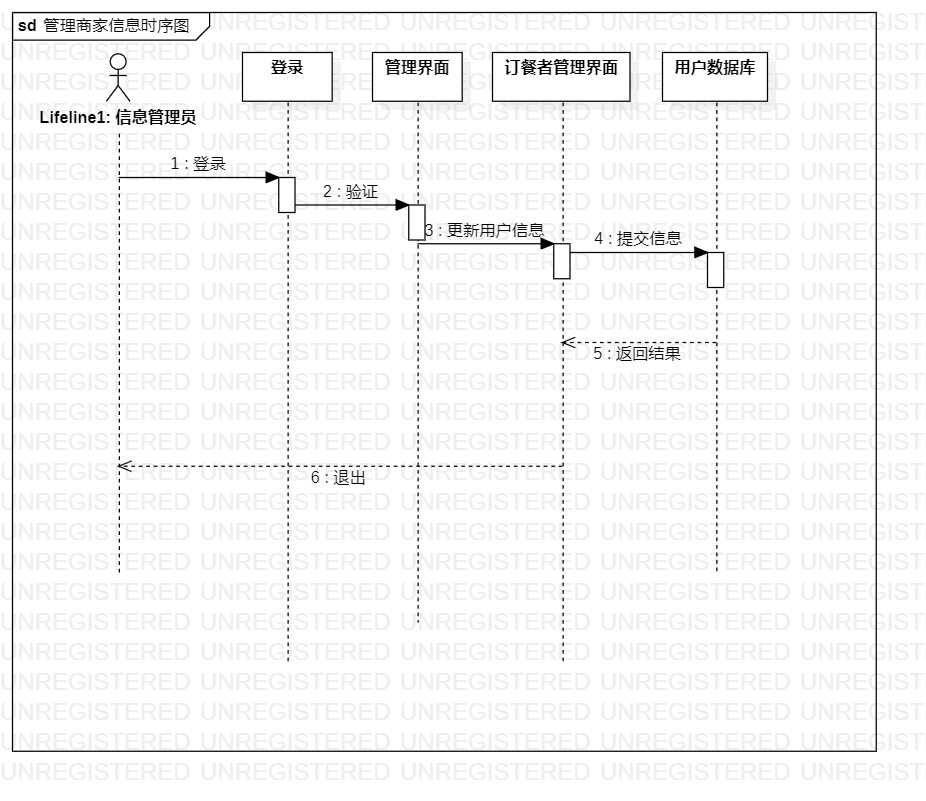
(3)系统将验证结果显示结果返回给系统管理员;

(4)系统管理员在订餐者信息管理界面对订餐者信息进行更新;

(5)提交更新数据;

(6)系统将提交结果返回;

(7)。提交成功，退出系统。



##### 2.5.9 信息管理员管理商家信息

(1)系统管理员注册一个自己的登录账号:

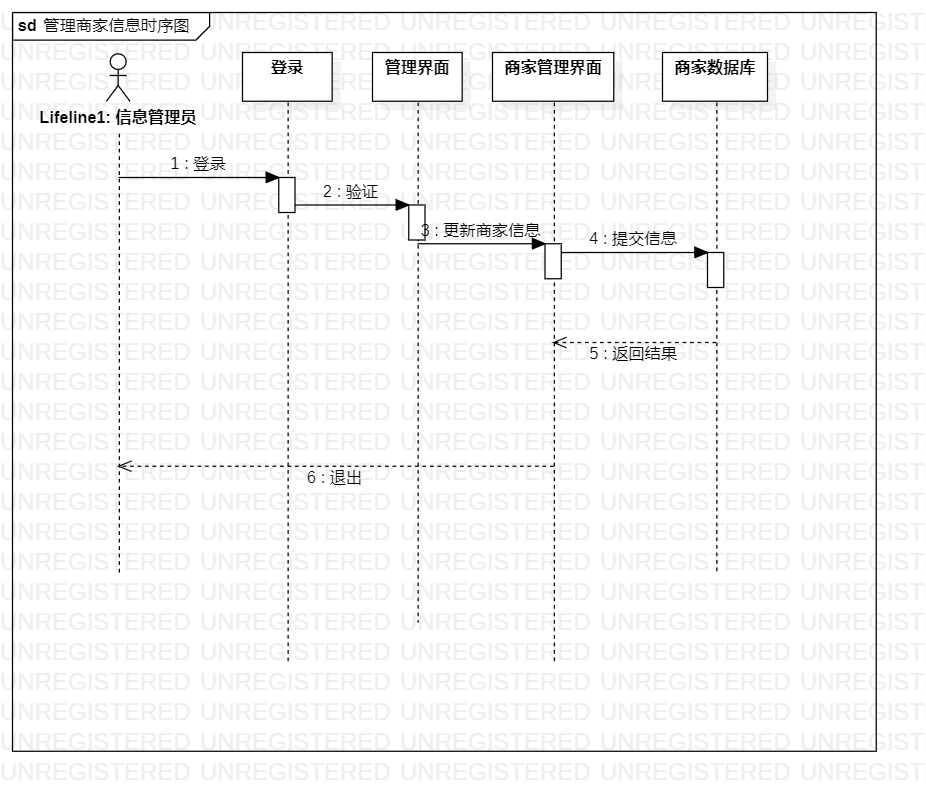
(2)输入用户名和密码进行身份验证;

(3)系统将验证结果显示结果返回给系统管理员;

(4)系统管理员在商家信息管理界面对商家信息进行更新:

(5)提交更新数据;

(6)系统将提交结果返回;(7)提交成功，退出系统



#### 2.5.10 收银员收取支付

（1）登录支付系统

（2）收取支付

（3）生成账单

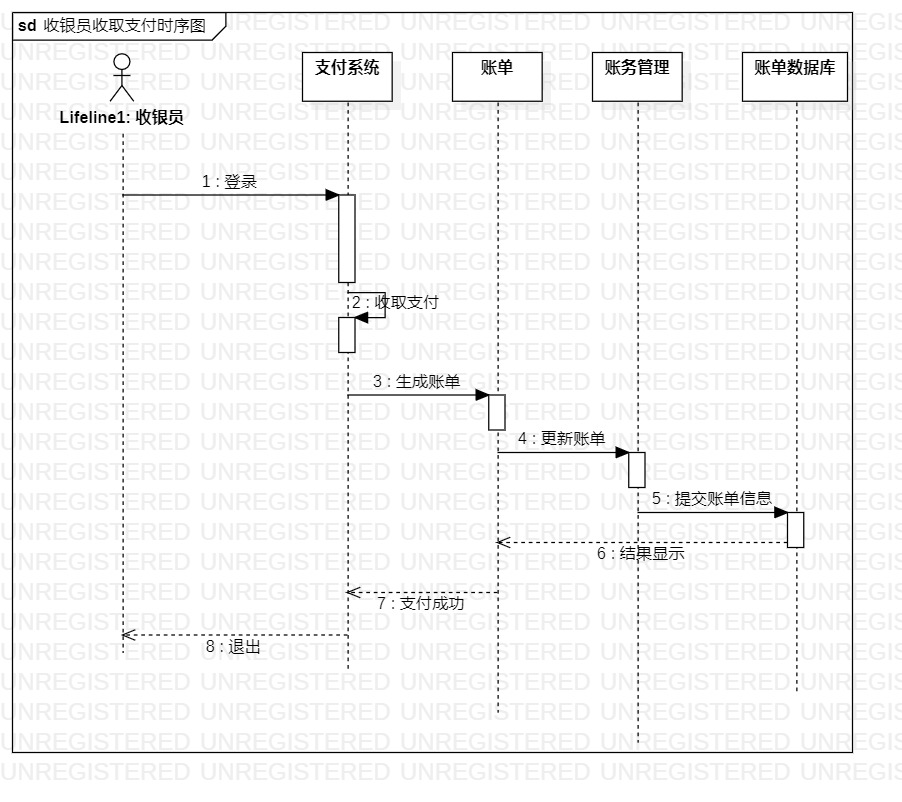
（4）更新账单

（5）提交账单信息至账单数据库

（6）结果显示

（7）支付成功

（8）退出



#### 2.5.11 订单管理员管理订单

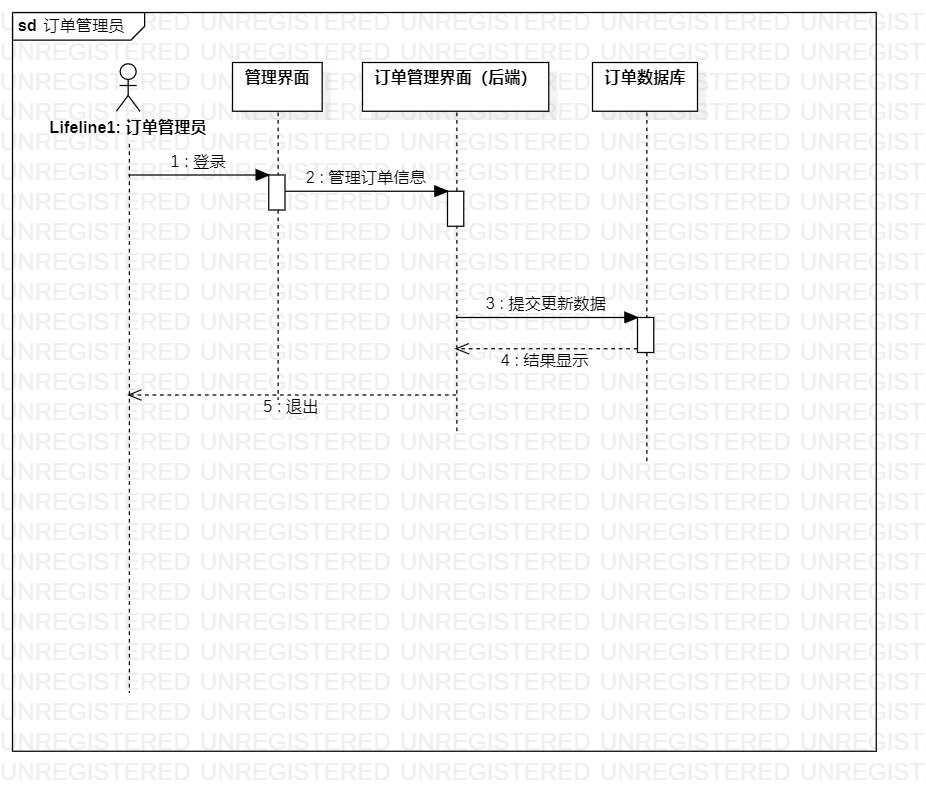
(1)订单管理员登录到系统中;

(2)管理订单信息;

(3)提交更新数据;

(4)系统将更新结果显示给定单管理员;

(5)提交成功，退出系统。



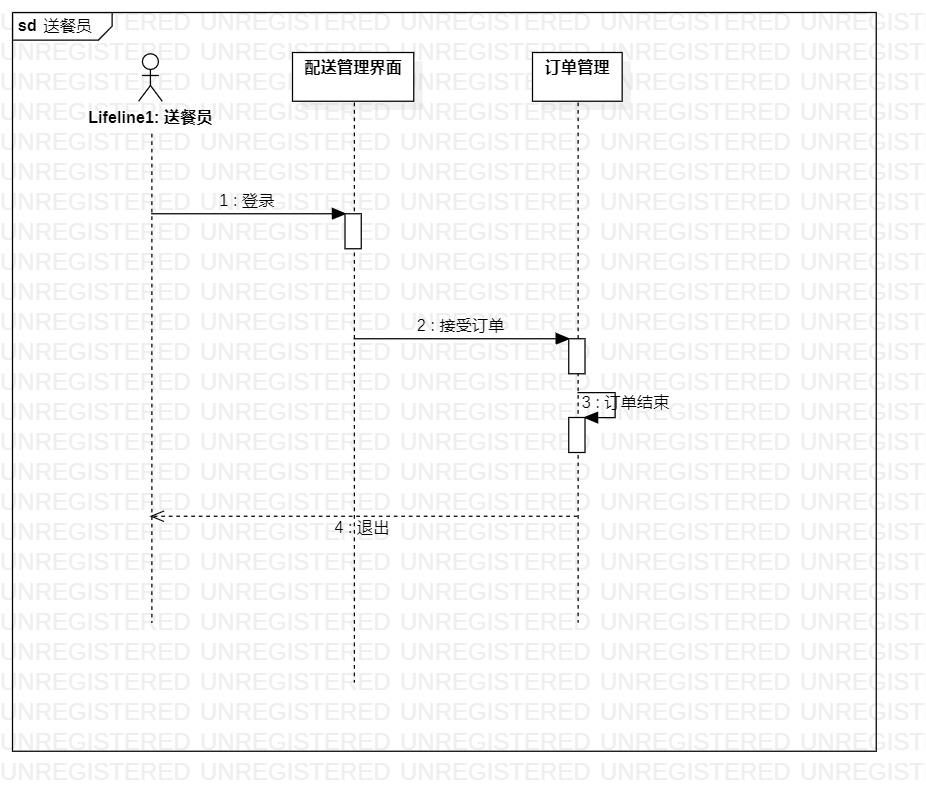
#### 2.5.12 送餐员送餐

(1)登录到配送管理界面

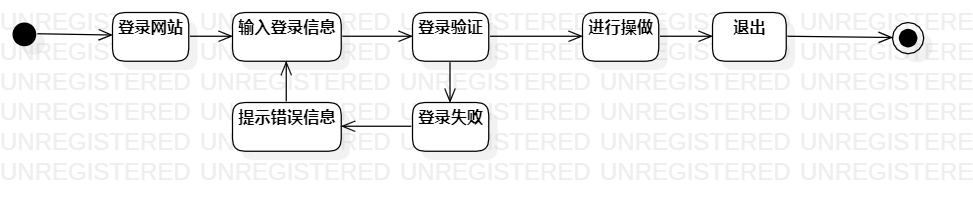
(2)接受订单中心订单

(3)订单结束

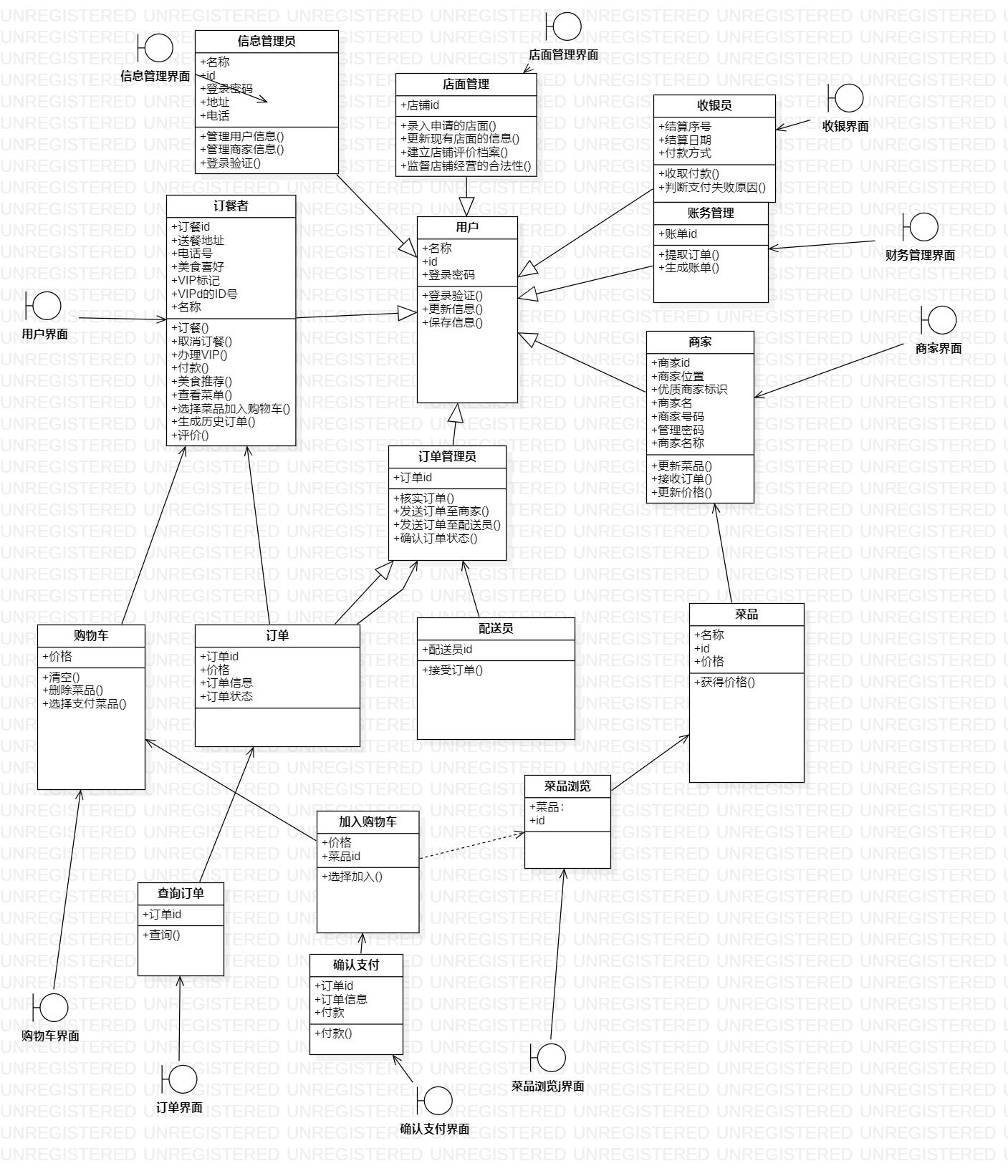
(4)退出



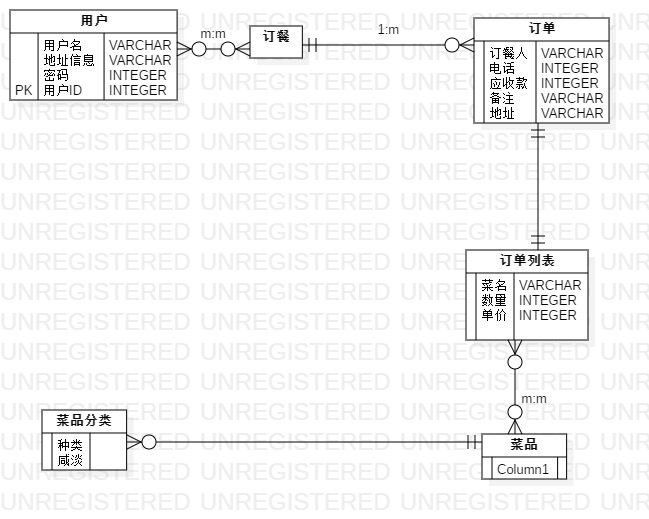
## 状态图



#### 4.类图



#### E-R图



## 总结

经过认真地分析我得出了系统的需求分析，确定了系统的主要参与者以及各自的主要活动。通过学习UML建模的有关知识和RationalRose工具，我亲自动手练习，最终画出了系统的系统用例模型(各自用例的用例图)、系统静态模型(系统类图)、系统的动态模型(系统时序图、系统活动图和系统状态图)以及系统部署模型(系统构件图和系统部署图)。通过自己的亲自动手操作，使我进一步了解并掌握了UML的建模过程和RationalRose工具的使用。同时，我也发现了自己思考问题不全面等一系列不足，促使自己不断改正、不断进步。