

考情分析



UML ( Unified Modeling Language , 统一建模语言 ) 是用来对软件系统进行可视化建模的一种语言。UML是面向对象方法开发系统的产品进行说明、可视化和编制文档的一种标准语言。1994年10月 , Grady Booch和Jim Rumbaugh首先将Booch 93和OMT-2 统一起来 , 并于1995年10月发布了第一个公开版本 , 称之为统一方法UM 0.8 ( Unitted Method ) 。1995年秋 , OOSE 的创始人Ivar Jacobson加盟到这一工作。经过Booch、Rumbaugh和Jacobson三人的共同努力 , 于1996年6月和10月分别发布了两个新的版本 , 即UML 0.9和UML 0.91 , 并将UM重新命名为UML , 1997年1月发布了UML 1.0。

UML是一种定义良好、易于表达、功能强大且普遍适用的建模语言 , 它融入了软件工程领域的新思想、新方法和新技术 , 它的作用域不限于支持OOA ( 面向对象分析 ) 和OOD ( 面向对象设计 ) , 还支持从需求分析开始的软件开发的全过程。

UML是一个标准的图形表示法 , 在软件设计师考试中 , 主要考查UML的图形 , 特别是考查用例图、类图、顺序图、状态图和活动图。

版权方授权希赛网发布，侵权必究

[上一节](#)      [本书简介](#)      [下一节](#)

考试大纲要求分析

根据考试大纲和培训指南 , 在UML方面 , 要求考生使用面向对象分析方法定义软件需求 , 能够根据具体问题 , 创建符合UML标准的类图、用例图 , 并准确识别类中的关键属性和方法 , 以及类之间的关联。具体包括以下考点 :

- ( 1 ) UML的基本概念与作用。
- ( 2 ) 用例图的表示与应用。
- ( 3 ) 类图与对象图的表示与应用。
- ( 4 ) 顺序图的表示与应用。
- ( 5 ) 活动图的表示与应用。
- ( 6 ) 通信图的表示与应用。
- ( 7 ) 构件图的表示与应用。
- ( 8 ) 部署图的表示与应用。
- ( 9 ) 状态图的表示与应用。

版权方授权希赛网发布，侵权必究

## 命题特点与趋势分析

---

软件设计师考试的试题主要集中在以下几个方面：

- (1) 根据试题的描述填写用例图，主要是填写所缺的用例。
- (2) 根据试题的描述填写类图，主要是填写所缺的各种类，找出类的属性和方法。
- (3) 根据试题的描述填写顺序图，主要是填写所缺的各种消息。
- (4) 根据试题的描述填写状态图，主要是填写所缺的各种状态。
- (5) 根据试题的描述填写活动图，主要是填写所缺的各种符号。
- (6) 识别关联的多重度（0、1、0..1、0..\*、1..\*、0..N、1..N等，其中，“N”代表某个具体的整数，“\*”代表不确定的某个整数）。
- (7) 识别用例之间的关系（包含关系、扩展关系和泛化关系）。

版权方授权希赛网发布，侵权必究

## 考点精讲

---

本节根据2.1节的分析，对考试的重点内容进行精讲，具体内容包括UML的概述、用例图、类图、顺序图、活动图和其他一些图形。

版权方授权希赛网发布，侵权必究

## UML概述

---

从总体上来看，UML的结构包括构造块、规则和公共机制三个部分。

(1) 构造块。UML有三种基本的构造块，分别是事物（thing）、关系（relationship）和图（diagram）。事物是UML的重要组成部分，关系把事物紧密联系在一起，图是多个相互关联的事物的集合。

(2) 公共机制。公共机制是指达到特定目标的公共UML方法，主要包括规格说明（详细说明）、修饰、公共分类（通用划分）和扩展机制四种。规格说明是事物语义的细节描述，它是模型真正的核心；UML为每个事物设置了一个简单的记号，还可以通过修饰来表达更多的信息；UML包括两组公共分类，分别是类与对象（类表示概念，而对象表示具体的实体）、接口与实现（接口用

来定义契约，而实现就是具体的内容）；扩展机制包括约束（扩展了UML构造块的语义，允许增加新的规则或修改现有的规则）、构造型（扩展UML的词汇，用于定义新的构造块）和标记值（扩展了UML构造块的特性，允许创建新的特殊信息来扩展事物的规格说明）。

（3）规则。规则是构造块如何放在一起的规定，包括为构造块命名；给一个名字以特定含义的语境，即范围；怎样使用或看见名字，即可见性；事物如何正确、一致地相互联系，即完整性；运行或模拟动态模型的含义是什么，即执行。

UML对系统架构的定义是系统的组织结构，包括系统分解的组成部分，以及它们的关联性、交互机制和指导原则等提供系统设计的信息。具体来说，就是指以下5个系统视图：

（1）逻辑视图。逻辑视图也称为设计视图，它表示了设计模型中在架构方面具有重要意义的一部分，即类、子系统、包和用例实现的子集。

（2）进程视图。进程视图是可执行线程和进程作为活动类的建模，它是逻辑视图的一次执行实例，描述了并发与同步结构。

（3）实现视图。实现视图对组成基于系统的物理代码的文件和构件进行建模。

（4）部署视图。部署视图把构件部署到一组物理节点上，表示软件到硬件的映射和分布结构。

（5）用例视图。用例视图是最基本的需求分析模型。

另外，UML还允许在一定的阶段隐藏模型的某些元素、遗漏某些元素，以及不保证模型的完整性，但模型逐步地要达到完整和一致。

## 1. 事物

UML中的事物也称为建模元素，包括结构事物（structural things）、行为事物（behavioral things，动作事物）、分组事物（grouping things）和注释事物（annotational things，注解事物）。这些事物是UML模型中最基本的构造块。

（1）结构事物。结构事物在模型中属于最静态的部分，代表概念上或物理上的元素。UML有七种结构事物，分别是类、接口、协作、用例、活动类、构件和节点。类是描述具有相同属性、方法、关系和语义的对象的集合，一个类实现一个或多个接口；接口是指类或构件提供特定服务的一组操作的集合，接口描述了类或构件的对外的可见的动作；协作定义了交互的操作，是一些角色和其它事物一起工作，提供一些合作的动作，这些动作比事物的总和要大；用例是描述一系列的动作，产生有价值的结果。在模型中用例通常用来组织行为事物。用例是通过协作来实现的；活动类的对象有一个或多个进程或线程。活动类和类很相似，只是它的对象代表的事物的行为和其他事物是同时存在的；构件是物理上或可替换的系统部分，它实现了一个接口集合；节点是一个物理元素，它在运行时存在，代表一个可计算的资源，通常占用一些内存和具有处理能力。一个构件集合一般来说位于一个节点，但有可能从一个节点转到另一个节点。

（2）行为事物：行为事物是UML模型中的动态部分，代表时间和空间上的动作。UML有两种主要的行为事物。第一种是交互（内部活动），交互是由一组对象之间在特定上下文中，为达到特定目的而进行的一系列消息交换而组成的动作。交互中组成动作的对象的每个操作都要详细列出，包括消息、动作次序（消息产生的动作）、连接（对象之间的连接）；第二种是状态机，状态机由一系列对象的状态组成。

（3）分组事物。分组事物是UML模型中组织的部分，可以把它们看成是个盒子，模型可以在其中进行分解。UML只有一种分组事物，称为包。包是一种将有组织的元素分组的机制。与构件不同的是，包纯粹是一种概念上的事物，只存在于开发阶段，而构件可以存在于系统运行阶段。

(4) 注释事物。注释事物是UML模型的解释部分。

## 2. 关系

UML 用关系把事物结合在一起，主要有下列四种关系：

(1) 依赖 (dependency)。依赖是两个事物之间的语义关系，其中一个事物发生变化会影响另一个事物的语义。

(2) 关联 (association)。关联描述一组对象之间连接的结构关系。

(3) 泛化 (generalization)。泛化是一般化和特殊化的关系，描述特殊元素的对象可替换一般元素的对象。

(4) 实现 (realization)。实现是类之间的语义关系，其中的一个类指定了由另一个类保证执行的契约。

## 3. 图

UML 2.0包括14种图，分别列举如下：

(1) 类图 (class diagram)。类图描述一组类、接口、协作和它们之间的关系。在OO系统的建模中，最常见的图就是类图。类图给出了系统的静态设计视图，活动类的类图给出了系统的静态进程视图。

(2) 对象图 (object diagram)。对象图描述一组对象及它们之间的关系。对象图描述了在类图中所建立的事物实例的静态快照。和类图一样，这些图给出系统的静态设计视图或静态进程视图，但它们是从真实案例或原型案例的角度建立的。

(3) 构件图 (component diagram)。构件图描述一个封装的类和它的接口、端口，以及由内嵌的构件和连接件构成的内部结构。构件图用于表示系统的静态设计实现视图。对于由小的部件构建大的系统来说，构件图是很重要的。构件图是类图的变体。

(4) 组合结构图 (composite structure diagram)。组合结构图描述结构化类（例如，构件或类）的内部结构，包括结构化类与系统其余部分的交互点。组合结构图用于画出结构化类的内部内容。

(5) 用例图 (use case diagram)。用例图描述一组用例、参与者及它们之间的关系。用例图给出系统的静态用例视图。这些图在对系统的行为进行组织和建模时是非常重要的。

(6) 顺序图 (sequence diagram，序列图)。顺序图是一种交互图 (interaction diagram)，交互图展现了一种交互，它由一组对象或参与者以及它们之间可能发送的消息构成。交互图专注于系统的动态视图。顺序图是强调消息的时间次序的交互图。

(7) 通信图 (communication diagram)。通信图也是一种交互图，它强调收发消息的对象或参与者的结构组织。顺序图和通信图表达了类似的基本概念，但它们所强调的概念不同，顺序图强调的是时序，通信图强调的是对象之间的组织结构（关系）。在UML 1.X版本中，通信图称为协作图 (collaboration diagram)。

(8) 定时图 (timing diagram，计时图)。定时图也是一种交互图，它强调消息跨越不同对象或参与者的实际时间，而不仅仅只是关心消息的相对顺序。

(9) 状态图 (state diagram)。状态图描述一个状态机，它由状态、转移、事件和活动组成。状态图给出了对象的动态视图。它对于接口、类或协作的行为建模尤为重要，而且它强调事件导致的对象行为，这非常有助于对反应式系统建模。

(10) 活动图 (activity diagram)。活动图将进程或其他计算结构展示为计算内部一步步的控制

制流和数据流。活动图专注于系统的动态视图。它对系统的功能建模和业务流程建模特别重要，并强调对象间的控制流程。

(11) 部署图 ( deployment diagram )。部署图描述对运行时的处理节点及在其中生存的构件的配置。部署图给出了架构的静态部署视图，通常一个节点包含一个或多个部署图。

(12) 制品图 ( artifact diagram )。制品图描述计算机中一个系统的物理结构。制品包括文件、数据库和类似的物理比特集合。制品图通常与部署图一起使用。制品也给出了它们实现的类和构件。

(13) 包图 ( package diagram )。包图描述由模型本身分解而成的组织单元，以及它们之间的依赖关系。

(14) 交互概览图 ( interaction overview diagram )。交互概览图是活动图和顺序图的混合物。

版权方授权希赛网发布，侵权必究

[上一节](#)      [本书简介](#)      [下一节](#)

## 用例图

结构化分析 ( SA ) 方法采用功能分解的方式来描述系统功能，在这种表达方式中，系统功能被分解到各个功能模块中，通过描述细分的系统模块的功能来达到描述整个系统功能的目的。采用SA方法来描述系统需求，很容易混淆需求和设计的界限，这样的描述实际上已经包含了部分的设计在内。因此，系统分析师常常感到迷惑，不知道系统需求应该详细到何种程度。一个极端的做法就是将需求详细到概要设计，因为这样的需求描述既包含了外部需求也包含了内部设计。SA方法的另一个缺点是分割了各项系统功能的应用环境，从各项功能项入手，很难了解到这些功能项如何相互关联来实现一个完整的系统服务的。

从用户的角度来看，他们并不想了解系统的内部结构和设计，他们所关心的是系统所能提供的服务，这就是用例方法的基本思想。用例方法是一种需求合成技术，采用各种需求获取方法获取用户需求，记录下来，然后从这些零散的要求和期望中进行整理与提炼，从而建立用例模型。在OOA方法中，构建用例模型一般需要经历四个阶段，分别是识别参与者、合并需求获得用例、细化用例描述和调整用例模型，其中前三个阶段是必需的。

### 1. 用例图的元素

用例是一种描述系统需求的方法，使用用例的方法来描述系统需求的过程就是用例建模。在用例图中，主要包括参与者、用例和通信关联三种元素，如图2-1所示。

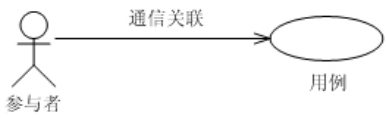


图2-1 用例图中的基本元素

(1) 参与者。参与者 ( 角色、动作者、执行者 ) 是指存在于系统外部并与系统进行交互的任何事物，既可以是使用系统的用户，也可以是其他外部系统和设备等外部实体。

(2) 用例。用例是在系统中执行的一系列动作，这些动作将生成特定参与者可见的价值结果。也就是说，用例表示系统所提供的服务，它定义了系统是如何被参与者所使用的，它描述的是参与者为了使用系统所提供的某一完整功能而与系统之间发生的一段对话。

(3) 通信关联。通信关联表示的是参与者和用例之间的关系，或用例与用例之间的关系。箭头表示在这一关系中哪一方是对话的主动发起者，箭头所指方是对话的被动接受者，箭尾所指方是对话的主动发起者。如果不想强调对话中的主动与被动关系，可以使用不带箭头的关联实线。在用例模型中，信息流不是由通信关联来表示的，信息流是默认存在的，并且是双向的，它与箭头所指的方向没有关系。

## 2. 识别参与者

参与者是与系统交互的所有事物，该角色不仅可以由人承担，还可以是其他系统和硬件设备，甚至是系统时钟。

(1) 其他系统：当系统需要与其他系统交互时，其他系统就是一个参与者。例如，对某企业的在线教育平台系统而言，该企业的OA（办公自动化）系统就是一个参与者。

(2) 硬件设备：如果系统需要与硬件设备交互，硬件设备就是一个参与者。例如，在开发IC（Integrated Circuit，集成电路）卡门禁系统时，IC卡读写器就是一个参与者。

(3) 时钟：当系统需要定时触发时，时钟就是一个参与者。例如，开发在线测试系统中的“定时交卷”功能时，就需要引入时钟作为参与者。

要注意的是，参与者一定在系统之外，不是系统的一部分。可以通过下列问题来帮助系统分析师发现系统的参与者：谁使用这个系统？谁安装这个系统？谁启动这个系统？谁维护这个系统？谁关闭这个系统？哪些（其他的）系统使用这个系统？谁从这个系统获取信息？谁为这个系统提供信息？是否有事情自动在预计的时间发生？

执行系统某项功能的参与者可能有多个，但这多个参与者在系统使用时会有不同的职责划分，根据职责的重要程度不同，有主要参与者和次要参与者之分。主要参与者是从系统中直接获得可度量价值的参与者，次要参与者的需求驱动了用例所表示的行为或功能，在用例中起支持作用，帮助主要参与者完成他们的工作，次要参与者不能脱离主要参与者而存在。开发用例的重点是要找到主要参与者。

## 3. 合并需求获得用例

将参与者都找到之后，接下来就是仔细地检查参与者，为每一个参与者确定用例。首先，要将获取到的需求分配给与其相关的参与者，以便可以针对每个参与者进行工作，而无遗漏；其次，进行合并操作。在合并之前，要明确为什么要合并，知道了合并的目的，才可能选择正确的合并操作。合并后，将产生用例。将识别到的参与者和合并生成的用例，通过用例图的形式整理出来，以获得用例模型的框架，如图2-2所示。

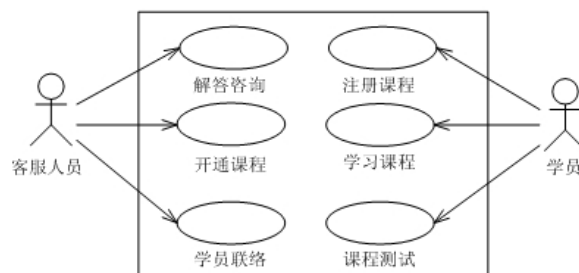


图2-2 用例图示例

在确定用例的过程中，需要注意以下问题：

(1) 用例命名。用例的命名应该注意采用“动词(短语)+名词(短语)”的形式,例如,“开通课程”和“课程测试”等。而且,最好能够对用例进行编号,这也是实现需求跟踪管理的重要技巧,通过编号可以将用户的需求落实到特定的用例中去。

(2) 不能混淆用例和用例所包含的步骤。例如,“开通课程”功能要经过验证学员信息、检查学员权限、保存开通记录、修改课程选修人数等步骤才能完成,在系统中这些步骤不能作为单独的功能对外提供,它们只是一个用例所包含的事件流,或是用例的子功能。

(3) 注意区分业务用例和系统用例。当针对整个业务领域建模时,需要使用业务用例,其中会涉及大量的人工活动,例如,在线教育平台系统中有一项重要工作是“编写教材”,这就是业务用例,是信息系统无法完成的。信息系统作为整个业务系统的一部分,只负责实现系统的部分功能,因此,只需要识别出系统用例,而不需考虑业务用例。

#### 4. 细化用例描述

用例建模的主要工作是书写用例规约(use case specification),而不是画图。用例模板为一个给定项目的所有人员定义了用例规约的结果,其内容至少包括用例名、参与者、目标、前置条件、事件流(基本事件流和扩展事件流)和后置条件等,其他的还可以包括非功能需求和用例优先级等。

一个较为复杂的系统会有较多的用例,为便于理解,可以为它们建立多张用例图。更为复杂的情况将导致所有用例难以维持一种平面结构,这时可以对用例进行分组。UML使用用例主题划分用例图,一组用例放置在以主题命名的方框中(类似于系统边界),每个主题中可以包含多个用例图。

用例的描述可以迭代完成,先对一些重要的用例编制相对细致的用例描述,对于一些不重要的用例,可以留待以后再补充完成。用例描述通常包括以下几个部分:

(1) 用例名称。用例名称应该与用例图相符,并写上其相应的编号。

(2) 简要说明。对用例为参与者所传递的价值结果进行描述,应注意语言简要,使用用户能够阅读的自然语言。

(3) 事件流。事件流是指当参与者和系统试图达到一个目标时所发生的一系列活动,也就是用例所完成的工作步骤。在编写时应注意使用简单的语法,主语明确,语义易于理解;明确写出“谁控制球”,也就是在事件流描述中,让读者直观地了解是参与者在控制还是系统在控制;从俯视的角度来编写,指出参与者的动作,以及系统的响应;描述用户意图和系统职责,而不叙述具体的行为和技术细节,特别是有关用户界面的细节。

执行一个用例的事件流有多种可能的路线,其中主事件流(基本事件流)是指能够满足目标的典型的成功路径,主事件流通常不包括任何条件和分支,符合大多数人的期望,从而更容易理解和扩展;备选事件流(扩展事件流)也称为备选路径,是完成用例可能出现失败的情况、分支路径或扩展路径,为了不影响用例活动清晰的主线,将这些分支处理全部抽取出来作为备选事件流。例如,在“开通课程”用例执行的过程中,如果学员所交的费用多于所选修课程规定的费用,则需要把多余的费用转换为学习币;如果学员选修课程数量超出最大限额,则用例未达到期望目标而终止。在事件流的描述中,主事件流使用“确认”和“验证”等确定性语句,而不是“检查是否……”和“如果……,那么……,否则……”等条件语句,这些分支情况利用备选事件流来说明。

另外,事件流的编写过程也是可以分阶段迭代进行的,对于优先级高的用例花更多的时间,更加的细化;对优先级低的使用例可以先简略地将主事件流描述清楚,备选事件流留待以后处理。对于

一些事件流较为复杂的，可以在用例描述中引用顺序图、状态图和通信图等手段进行描述。

（4）非功能需求。因为用例所涉及的非功能需求通常很难在事件流中进行表达，因此单列为一小节进行描述。在非功能需求的描述方面，一定要注意使其可度量和可验证。否则，就容易流于形式，形同摆设。

（5）前置条件和后置条件。前置条件是执行用例之前必须存在的系统状态，如果前置条件不满足，则用例无法启动。例如，“开通课程”用例的前置条件是客服人员已正确登录到系统中；后置条件是用例执行完毕系统可能处于的一组状态。一旦用例成功执行，可能会导致系统内部某些状态的变化，例如，成功地“开通课程”会使该课程的选修人数增加，会使学员的权限发生变化等。而某些用例也可能没有前置条件或后置条件，例如，“学员联络”用例没有后置条件，因为该用例执行后不会改变系统状态。如果在当前阶段不容易确定前置条件或后置条件，则可以在以后再细化。

（6）扩展点。如果包括扩展（或包含）用例，则写出扩展（或包含）用例名，并说明在什么情况下使用。

（7）优先级。说明用户对该用例的期望值，为以后的开发工作确定先后顺序。可以采用满意度/不满意度指标进行说明，例如，设置为1~5的数值。

表2-2是图2-2中“开通课程”的用例描述，这些内容不一定要一次完成。如果需要调整用例模型，则在调整用例模型之后，还可以修改和细化用例描述。

**表2-2 用例描述示例**

1.用例名称：

开通课程（UC02）

2.简要说明：

为用户开通学习课程的权限，将其标记为“学员”，同时修改所选修课程的选修人数。

3.事件流：

3.1 主事件流

1) 客服人员向系统发出“开通课程”请求。

2) 系统要求客服人员选择开通课程的类型（软考、考研、专业课程、自学考试、Java课程）。

3) 客服人员做出选择后，系统显示相应界面，让客服人员输入信息，并自动根据权限规则生成权限。

4) 客服人员输入学员的相关信息，包括学员用户名、所交费用、交费时间、选修课程名称。

5) 系统确认学员所交费用和所选修课程的规定费用一致。

6) 系统将所输入的信息存储建档，开通学员课程权限。

3.2 备选事件流

5a) 如果学员所交费用少于所选修课程的规定费用，则显示所选修课程的规定费用，并要求客服人员选择修改或取消输入。

5a1) 客服人员选择取消输入，则结束用例，不做存储建档工作。

5a2) 客服人员选择修改用户所交费用后，转到5)。

5b) 如果学员所交费用多于所选修课程的规定费用，则显示多余的费用数量，并要



求客服人员选择是转换为学习币还是退还给学员。

5b1) 客服人员选择转换为学习币，则把多余的费用转换为学习币，记入学员账户中，转到5)。

5b2) 客服人员选择退还给学员，转到5)。

5c) 如果学员所选修的课程超出了系统规定的选修人数，则提示客服人员，结束用例。

4.非功能需求

无特殊要求。

5.前置条件

客服人员登录在线教育平台系统。

6.后置条件

修改学员权限，修改课程选修人数。

7.扩展点

无。

8.优先级

最高(满意度5，不满意度5)。

5 . 调整用例模型

在建立了初步的用例模型后，还可以利用用例之间的关系来调整用例模型。用例之间的关系主要有包含、扩展和泛化，利用这些关系，把一些公共的信息抽取出来，以便于复用，使得用例模型更易于维护。

(1) 包含关系。当可以从两个或两个以上的用例中提取公共行为时，应该使用包含关系来表示它们。其中这个提取出来的公用用例称为抽象用例，而把原始用例称为基本用例或基础用例。例如，图2-2中的“学习课程”和“课程测试”两个用例都需要检查学员的权限，为此，可以定义一个抽象用例“检查权限”。用例“学习课程”和“课程测试”与用例“检查权限”之间的关系就是包含关系，如图2-3所示。其中“<<include>>”是包含关系的构造型，箭头指向抽象用例。

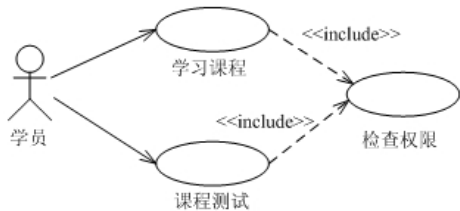


图2-3 包含关系的例子

当多个用例需要使用同一段事件流时，抽象成为公用用例，可以避免在多个用例中重复地描述这段事件流，也可以防止这段事件流在不同用例中的描述出现不一致。当需要修改这段公共的需求时，也只要修改一个用例，避免同时修改多个用例而产生的不一致性和重复性工作。另外，当某个用例的事件流过于复杂时，为了简化用例的描述，也可以将某一段事件流抽象成为一个被包含的用例。

(2) 扩展关系。如果一个用例明显地混合了两种或两种以上的不同场景，即根据情况可能发生多种分支，则可以将这个用例分为一个基本用例和一个或多个扩展用例，这样使描述可能更加清晰。例如，图2-2中的学员进行“课程测试”时，其测试的次数可能已超出系统规定的限额，这时就需要学员“充入学习币”。用例“课程测试”和“充入学习币”之间的关系就是扩展关系，如图2-4

所示。其中 “<<extend>>” 是扩展关系的构造型，箭头指向基本用例。

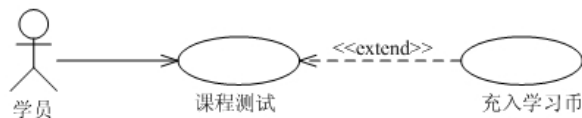


图2-4 扩展关系的例子

（3）泛化关系。当多个用例共同拥有一种类似的结构和行为的时候，可以将它们的共性抽象成为父用例，其他的用例作为泛化关系中的子用例。在用例的泛化关系中，子用例是父用例的一种特殊形式，子用例继承了父用例所有的结构、行为和关系。例如，图2-2中学员进行课程注册时，假设既可以通过电话注册，也可以通过网上注册，则“注册课程”用例就是“电话注册”用例和“网上注册”用例的泛化，如图2-5所示。其中三角箭头指向父用例。

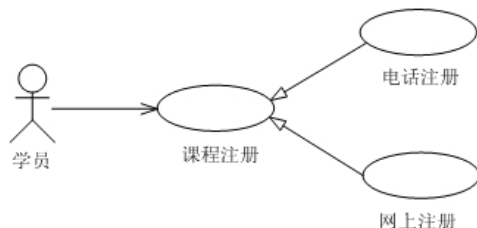


图2-5 泛化关系的例子

从UML事物关系的本质上来看，包含关系和扩展关系都属于依赖关系。对包含关系而言，抽象用例中的事件流是一定插入到基本用例中去的，并且插入点只有一个。扩展用例的事件流往往可以抽象为基本用例的备选事件流，在扩展关系中，可以根据一定的条件来决定是否将扩展用例的事件流插入到基本用例的事件流中，并且插入点可以有多个。在实际应用中，很少使用泛化关系，子用例的特殊行为都可以作为父用例中的备选事件流而存在。

在实际工作中，系统分析师要谨慎选用这些关系。从上面的介绍可以看出，包含、扩展和泛化关系都会增加用例的个数，从而增加用例模型的复杂度。另外，一般都是在用例模型完成之后才对它进行调整，在用例模型建立之初不必急于抽象用例之间的关系。

版权方授权希赛网发布，侵权必究

[上一节](#)      [本书简介](#)      [下一节](#)

## 类图与对象图

2.2.2节从用户的观点对系统进行了用例建模，但捕获了用例并不意味着分析的结束，还要对需求进行深入分析，获取关于问题域本质内容的分析模型。分析模型描述系统的基本逻辑结构，展示对象和类如何组成系统（静态模型），以及它们如何保持通信，实现系统行为（动态模型）。

为了使模型独立于具体的开发语言，系统分析师需要把注意力集中在概念性问题上而不是软件技术问题上，这些技术的起点就是领域模型。领域模型又称为概念模型或简称为域模型，也就是找到那些代表事物与概念的对象，即概念类。概念类可以从用例模型中获得灵感，经过完善将形成分析模型中的分析类。每一个用例对应一个类图，描述参与这个用例实现的所有概念类，而用例的实现主要通过交互图来表示。例如，用例的事件流会对应产生一个顺序图，描述相关对象如何通过合作来完成整个事件流，复杂的备选事件流也可以产生一个或多个顺序图。所有这些图的集合就构成

了系统的分析模型。

建立分析模型的过程大致包括定义概念类、确定类之间的关系、为类添加职责、建立交互图等，其中有学者将前三个步骤统称为CRC（Class-Responsibility-Collaborator，类-责任-协作者）建模。

1. 定义概念类

OOA的中心任务就是要找到系统中的对象或类，这些类将反映到系统设计中的软件类和系统实现中某个OOP（面向对象程序设计）语言声明的类。例如，在领域模型中，学员学习某门课程是一个事件，该事件记录了某个学员和某门课程在一定时期内的责任关系，表达的是领域概念；在系统设计模型中，学习记录就是一个软件类。虽然它们是不同的事物，但领域模型中的命名启发了后者的命名和定义，从而缩小了表示的差距。在整个系统开发过程中，要尽量使这些类或对象在不同阶段保持相同的名称。

发现类的方法有很多种，其中广泛应用的是名词短语法。它的主要规则是先识别有关问题域文本描述中的名词或名词短语，然后将它们作为候选的概念类或属性，其具体步骤如下：

- （1）阅读和理解需求文档或用例描述。
- （2）筛选出名词或名词短语，建立初始类清单（候选类）。
- （3）将候选类分成三类，分别是显而易见的类、明显无意义的类和不确定类别的类。
- （4）舍弃明显无意义的类。
- （5）小组讨论不确定类别的类，直到将它们都合并或调整到其他两个类别，并进行相应的操作。

例如，根据表2-2所描述的“开通课程”用例的事件流，可以获得候选概念类的清单，如表2-3所示。

表2-3 候选概念类清单

名词类别	概念类列表
显而易见的类	课程、学习币、学员
明显无意义的类	请求、界面、信息
不确定类别的类	学员账户、课程类型、权限、用户名、费用、课程名称

经过简单分析可以得出，“学员账户”、“用户名”和“权限”可以归结到“学员”类，作为“学员”类的属性；“课程类型”和“课程名称”可以归结到“课程”类，作为“课程”类的属性；“费用”可以单独列为一个类，称为“费用清单”。这样，针对“开通课程”这个用例，就确定了四个类，分别是课程、学习币、学员和费用清单。

另外，也可以根据描述中的名词类别来发现候选类，例如，划分为人员、组织、物品、设备、事件、规格说明、业务规则或政策等。要注意的是，不是所有的名词或名词短语都是系统中的一个合适的候选类，因为有的在系统之外，有的与系统不相关，有的名词概念较小，只适合于作为某个候选类的属性。因此，必须对其进行一番筛选，把不合适的过滤掉。

2. 确定类之间的关系

当完成了类的寻找工作之后，就需要理清这些类之间的关系，类之间的主要关系有关联、依赖、泛化、聚合、组合和实现等，它们在UML中的表示方式如图2-6所示。

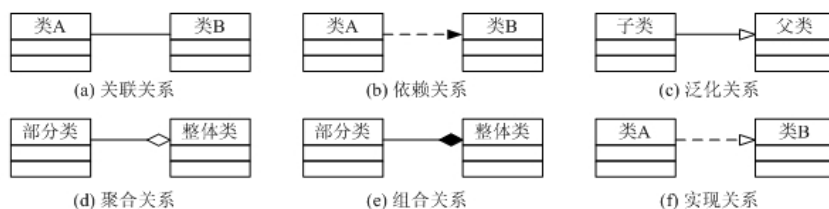


图2-6 类之间的关系表示

(1) 关联关系。关联提供了不同类的对象之间的结构关系，它在一段时间内将多个类的实例连接在一起。关联体现的是对象实例之间的关系，而不表示两个类之间的关系。其余的关系涉及类元自身的描述，而不是它们的实例。对于关联关系的描述，可以使用关联名称、角色、多重性和导向性来说明，如图2-7所示。

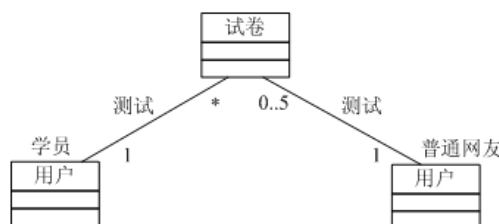


图2-7 关联关系的表示

关联名称反映该关系的目的，并且应该是一个动词，例如，图2-7中的“测试”。如果某种关联的含义对于开发人员和用户都是非常明确的，则可以省略关联名称；关联路径的两端为角色，角色规定了类在关联中所起的作用，例如，图2-7中的“学员”和“普通网友”。一般情况下，只有在关联名称不能明确表述时，才使用角色名称；多重性指定所在类可以实例化的对象数量（重数），即该类的多少个对象在一段特定的时间内可以与另一个类的一个对象相关联。例如，图2-7中的数字和“\*”都表示关联，其中“\*”等价于“0..\*”；导向性表示可以通过关联从源类导向到目标类，也就是说，给定关联一端的对象就能够容易并直接地得到另一端的对象。导向性用一个箭头表示，如果没有箭头，就认为是一个双向关联或是一个未定义的关联。关联关系的表示方式，同样也适合其他关系的表示，只是箭线符号不同而已。

(2) 依赖关系。两个类A和B，如果B的变化可能会引起A的变化，则称类A依赖于类B。依赖可以由各种原因引起，例如，一个类向另一个类发送消息、一个类是另一个类的数据成员、一个类是另一个类的某个操作参数等。

(3) 泛化关系。泛化关系描述了一般事物与该事物中的特殊种类之间的关系，也就是父类与子类之间的关系。继承关系是泛化关系的反关系，也就是说，子类继承了父类，而父类则是子类的泛化。

(4) 共享聚集。共享聚集关系通常简称为聚合关系，它表示类之间的整体与部分的关系，其含义是“部分”可能同时属于多个“整体”，“部分”与“整体”的生命周期可以不相同。例如，汽车和车轮就是聚合关系，车子坏了，车轮还可以用；车轮坏了，可以再换一个。

(5) 组合聚集。组合聚集关系通常简称为组合关系，它也是表示类之间的整体与部分的关系。与聚合关系的区别在于，组合关系中的“部分”只能属于一个“整体”，“部分”与“整体”的生命周期相同，“部分”随着“整体”的创建而创建，也随着“整体”的消亡而消亡。例如，一个公司包含多个部门，它们之间的关系就是组合关系。公司一旦倒闭，也就无所谓部门了。

(6) 实现关系。实现关系将说明和实现联系起来。接口是对行为而非实现的说明，而类中则包含了实现的结构。一个或多个类可以实现一个接口，而每个类分别实现接口中的操作。

确定了类之间的关系之后，通过UML的类图将这些关系记录下来，形成领域模型。例如，图2-8

就是与在线教育平台系统相关的一个简单的领域模型。

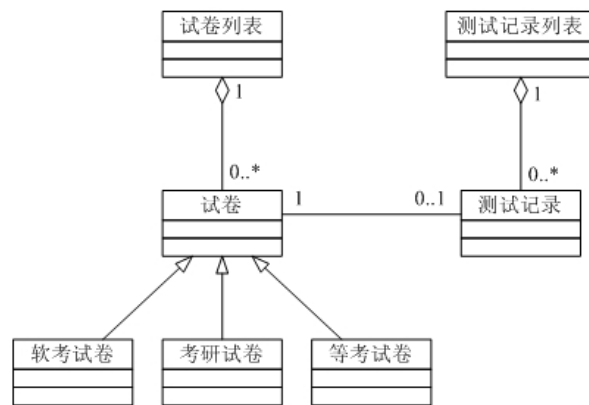


图2-8 领域模型示例

3 . 为类添加职责

找到了反映问题域本质的主要概念类，而且还理清了它们之间的协作关系之后，系统分析师就可以为这些类添加其相应的职责。类的职责包括两个方面的内容，一个是类所维护的知识，即成员变量或属性；另一个是类能够执行的行为，即成员方法或责任。

属性是描述类静态特征的一个数据项。系统分析师可以与用户进行交谈，提出问题来帮助寻找类的属性。从概念建模的角度来看，属性越简单越好。要保持属性的简单性，应该做到只定义与系统责任和系统目标有关的属性；使用简单数据类型来定义属性，不使用可由其他属性导出的属性（冗余属性）；不为类关联定义属性。最后，要对属性加以说明，包括名称、解释和数据类型，以及其他的一些要求。

对于类的责任的确定，可以根据用例描述中的动词来进行判断，然后再进行筛选。这个过程与类的识别过程是类似的，此处不再赘述。系统分析师应该要通用性地描述类的成员方法，例如，“交费”和“组卷”等。另外，根据封装性原则，信息和与其相关的行为应该存在同一个类中。关于一个事物的信息应该包含在单个类中，而不是分布在多个类中。但是，在适当的时候，可以在相关的类之间分享责任。

要注意的是，为类添加职责与找到类之间的关系一样，这个阶段也只能找到那些主要的、明显的、与业务规则相关的部分。切忌在这个阶段不断地细化，甚至引入一些与具体实现相关的技术内容（例如，数据库和分布式对象之类的东西）。

4 . 对象图

UML中对象图与类图具有相同的表示形式。对象图可以看做是类图的一个实例。对象是类的实例；对象之间的链（Link）是类之间的关联的实例。对象与类的图形表示相似，均划分成三个格子的长方形（下面的两个格子可省略）。最上面的格子是对象名，对象名带有下列线；中间的格子记录属性值；下面的格子记录操作方法。链的图形表示与关联相似，对象图常用于表示复杂类图的一个实例。

版权方授权希赛网发布，侵权必究

[上一节](#)      [本书简介](#)      [下一节](#)

顺序图用来描述对象之间动态的交互关系，着重体现对象间消息传递的时间顺序。顺序图允许直观地表示出对象的生存期，在生存期内，对象可以对输入消息做出响应，并且可以发送信息。如图2-9就是一个顺序图，图中标识出了每个部分的名称。

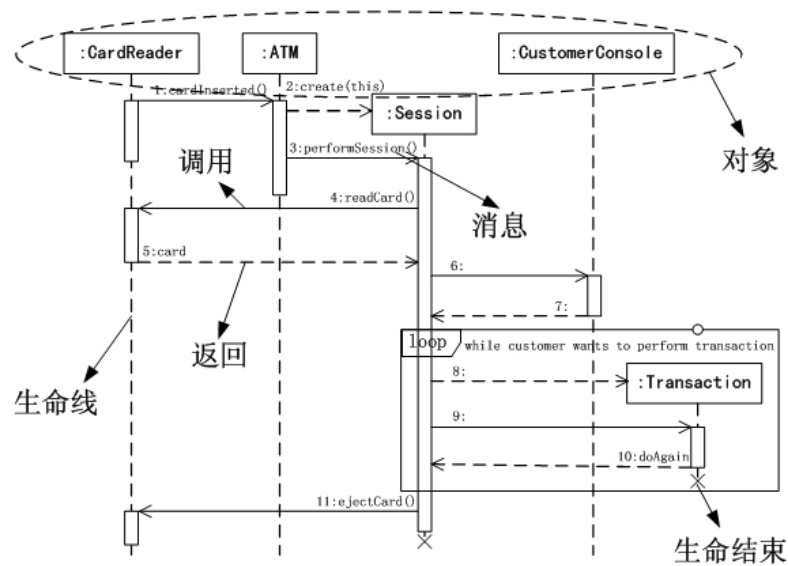


图2-9 顺序图示意图

版权方授权希赛网发布，侵权必究

[上一节](#)   [本书简介](#)   [下一节](#)

## 活动图

类模型体现了系统的静态结构，用例模型则从用户的角度对系统的动态行为进行了宏观建模，并通过交互模型将对象与消息有机地结合在一起。但有些时候，我们还需要更好地表示行为的细节，这就可以借助于活动图 and 状态图来实现。

### 1. 简单活动图

图2-10展示了一个用户订单处理过程的流程图，接下来就结合这个基本的活动图来学会正确的阅读方法。

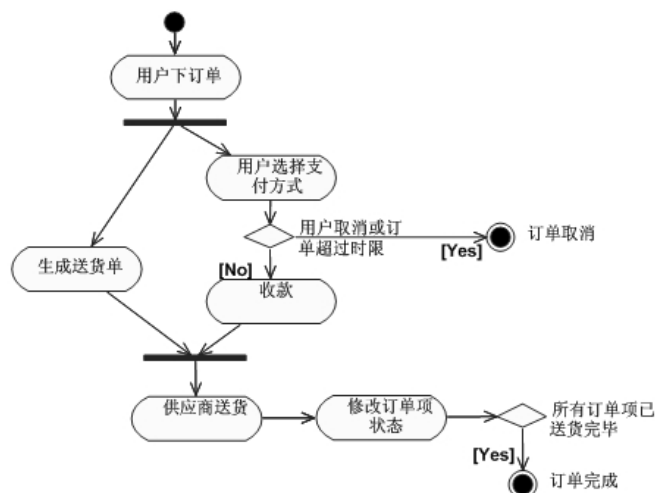


图2-10 用户订单处理简单活动图

(1) 初始节点和活动终点。在活动图中有两个特殊的节点，一个用来表示活动的初始节点，它用一个实心圆表示，在一张不包括子图的活动图中有且只有一个初始节点。而另一个则是表示活动处理完成的活动终点，它用一个圆圈内加一个实心圆来表示，在活动图中可能包含多个活动终点。例如，在本例中，用户取消和订单完成就是两个可能的活动终点。

(2) 活动节点。活动节点是活动图中最主要的元素之一，它用来表示一个活动，例如图2-10中的“用户下订单”、“用户选择支付方式”、“生成送货单”等都是活动节点。在UML中，活动节点所描述的活动可以是原子的动作，也可以是能进一步分解的一系列操作；它可以是文字描述、表达式、事件等。在图2-11中列出的就是一些可能的活动节点描述。



图2-11 活动节点

(3) 转换。当一个活动结束时，控制流就会马上传递给下一个活动节点，在活动图中称之为转换，用一条带箭头的直线来表示。如果需要对这些转换设置一些条件，使其在满足特定的条件时才触发，则可以借助监护条件来完成。

(4) 分支与监护条件。对于任何一个控制流而言，都一定会存在分支、循环等形式的控制流。在活动图中，分支用一个菱形表示，它有一个进入转换（箭头从外指向分支符号），一个或多个离开转换（箭头从分支符号指向外）。而每个离开转换上都会有一个监护条件，用来表示满足什么条件的时候执行该转换。但要注意，在多个离开转换上的监护条件不能有矛盾，否则就会使得流程产生混乱。

虽然在活动图中，没有直接提供表示循环的建模元素，但可以利用分支来实现“循环”控制流的表示。例如，在图2-10所示的例子中，一个订单可能对应多个供应商，如果订单没有完成的话，说明还有供应商没有完成送货任务，因此可以在分支“所有订单项已送货完毕”中，增加一个离开转换，指向“供应商送货”活动节点来表示这种循环，修改后的活动图如图2-12所示。

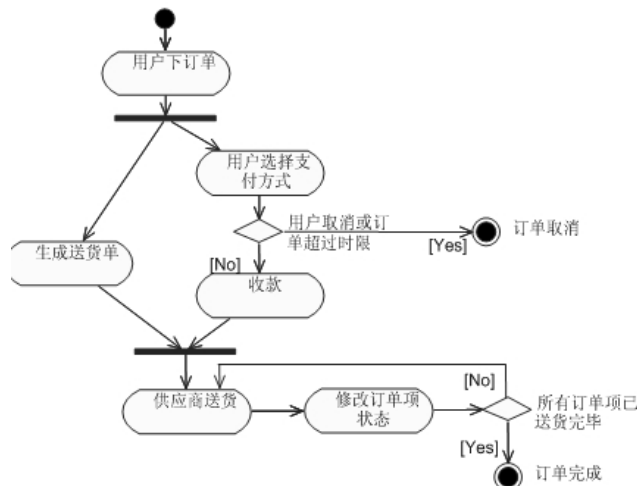


图2-12 修改后的简单活动图

(5) 分岔与汇合。在实际的控制流中，除了顺序结构、分支结构和循环结构之外，还可能存在并发的控制流。在UML中，可以采用一个同步线来说明这些并行控制流的分岔和汇合。如图2-13所示，同步线是一条水平或垂直的粗线段。如图2-13所示，分岔是有一个进入转换，两个或多个离开转换；而汇合则是两个或多个进入转换，一个离开转换。例如，在本例中，当“用户下订单”之后，系统将并行处理两方面事务：一是根据订单所涉及的产品生成送货单；二是处理用户的支付。

这两类事件是并发处理。当这两个并发处理都完成时，这时控制流汇合，转到“供应商送货”活动中。

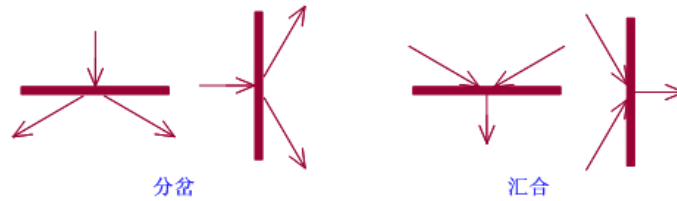


图2-13 分岔与汇合图示

## 2. 带泳道的活动图

简单活动图虽然明确地说明了整个控制流的过程，但是却没有说明每个活动是由谁做的。对应到编程而言，就是没有明确地表示出每个活动是由什么类来负责的；对应到业务建模，就是没有明确地表示出机构中的哪一个部门负责实施什么操作。

为了在简单活动图的基础上，有效地表示各个活动由谁负责的信息，可以通过泳道（Swim Lane）来实现。例如针对图2-12所示的活动图，活动的主要负责人包括客户、系统、供应商，因此可以将其分成三个泳道，绘制出如图2-14所示的活动图。

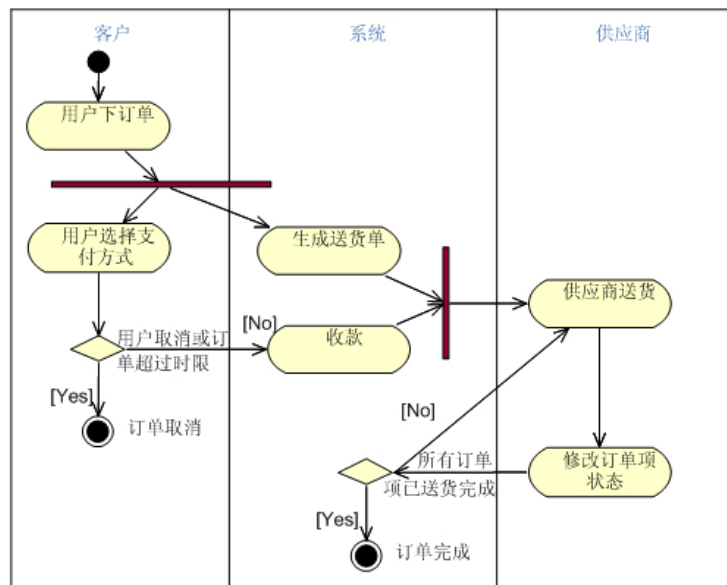


图2-14 带泳道的活动图

在图2-14中，泳道将活动图中的活动节点分成了几个小组，每个小组都显示出了负责实施这些操作的角色。在本图中，这些都是一些现实世界中的实体，而同样，也可以用来表示不同的类。

每个泳道在视觉上是用一条垂直的线将它们分开，并且每个泳道都必须有一个唯一的名称，例如本图中的客户、系统、供应商。从图中也可以看出，每个活动节点、分支是必须只属于一个泳道的，而转换、分岔与汇合是可以跨泳道的。通过泳道，我们不仅体现了整个活动控制流，还体现出了每个活动的实施者。

版权方授权希赛网发布，侵权必究

[上一节](#)      [本书简介](#)      [下一节](#)



状态图用来描述一个特定对象的所有可能状态及其引起状态转移的事件。大多数面向对象技术都用状态图表示单个对象在其生命周期中的行为。一个状态图包括一系列的状态及状态之间的转移。

例如，可以采用如图2-15所示的状态机图，来描述一个烧水器在工作时的详细行为细节。

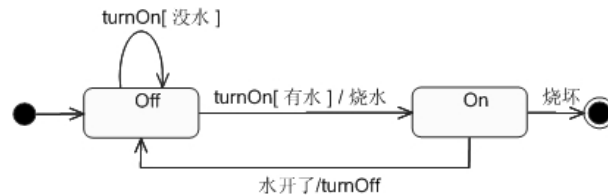


图2-15 简单状态机图

从图2-15中不难看出，在一张状态机图中，最为核心的元素无外乎是两个：一个是用圆角矩形表示的状态（初态和终态例外）；另一个则是在状态之间的、包含一些文字描述的有向箭头线，这些箭头线称为转换。在前面我们已经说过了状态的含义和表示法，在此我们重点在于理解“转换”的含义和表示法。一个转换是两个状态之间的一种关系，表示对象将在第一个状态中执行一定的动作，并在某个特定事件发生时、且满足特定条件时进入第二个状态。

如图2-16所示，转换涉及的内容包括源状态、触发事件、监护条件、动作和目标状态五个方面的内容。



图2-16 转换的五要素

### 1. 源状态和目标状态

当一个转换发生时，就称为转换被激活。在转换激活前，处于源状态，在转换激活后就进入目标状态。

（1）源状态：即受转换影响的状态，如果一个对象处于源状态，那么当该对象收到转换的触发事件并满足监护条件（如果有）时，就会激活转换。

（2）目标状态：即转换完成后对象的状态。

### 2. 触发事件

它用来为转换定义一个事件，当源状态中的对象接收到这个事件时，就会使转换合法的激活（如果有定义监护条件，则还需满足监护条件）。在UML中，事件主要可以分为调用事件、改变事件、信号事件和时间事件四种。

（1）调用事件：用某对象的成员方法就称之为调用事件，它是一种同步的机制。例如在图2-15中，turnOn就是一种调用事件，用来将开关置于“on”状态。

（2）改变事件：改变事件是指依赖于指定属性值的布尔表达式得到满足。这是一种一直等待到特定条件被满足的声明方式。它并不常使用。它和监护条件不同，改变事件中的条件是一直计算，直到布尔表达式为真为止。

（3）信号事件：信号是两个对象之间通信的媒介，信号是一种异步机制。在计算机中，鼠标和键盘的操作均是属于此类事件。对于一个信号而言，对象一般都有相应的事件处理器，例如onMouseClicked（）等。例如，图2-15中“水开了”就是一个信号。

(4) 时间事件：时间事件代表时间的流逝。它可以指定为绝对形式（每天的某时，例如after（11:00）），也可以指定为相对形式（从某一指定事件发生开始所经过的时间，例如after（2 seconds））。不过值得注意的是，对于前一种形式，也可以使用变化事件来描述：when（11:00）。

### 3. 监护条件

大家对于监护条件应该不会陌生了，在前面的许多地方都提到过。在状态机图中，它的语义仍然没变。它通常是一个布尔表达式，当对象接收到触发事件时，就将对该布尔表达式求值；如果表达式取值为真，则激活转换；如果取值为假，则不激活转换。例如，在图2-15中，当收到turnOn事件时，还将判断壶中“有水”否；如果有水，激活转换；否则不激活转换，因此状态就不会发生改变了。

再次重申监护条件和改变事件的区别，监护条件只在对象收到触发事件时才会计算一次，而改变事件是一直计算的。

### 4. 动作

当转换被激活后，如果定义了相应的动作，那么就将执行这个动作。它可以是一个赋值语句、简单的算术运行、发送信号、调用操作、创建和销毁对象、读取和设置属性的值，甚至是一个包含多个动作的活动。例如，在图2-15中，当turnOn后，就将执行“烧水”的动作。

认识了转换的五个要素之后，就不难正确地理解整个状态机图中所蕴含的含义了。由于初态到off状态之间是没有任何描述的，因此说明初态就是off。

(1) 与状态off相关的转换有两个，其触发事件都是turnOn，只不过其监护条件不同。如果对象收到事件turnOn，那么将判断壶中是否有水；如果[没水]，则仍然处于off状态；如果[有水]，则转为on状态，并执行“烧水”动作。值得说明的是，其实从off状态到on状态的转换是可以不必绘出的，在此给出只是为了例子的需要。

(2) 与状态on相关的转换也有两个，如果“水开了”就执行turnOff，关掉开关；如果烧坏了，就进入终态。

版权方授权希赛网发布，侵权必究

[上一节](#)

[本书简介](#)

[下一节](#)

## 其他图形

考试大纲中要求考生掌握通信图、构件图（组件图）和部署图，但从历年试题来看，并没有出现这方面的试题，因此，本节只作简单介绍。

### 1. 通信图

通信图和顺序图同属于交互图，它们之间的语义是等价的，只不过它们关注点有所不同而已。也就是说，可以很容易地完成从顺序图到通信图的转换。图2-17是一个通信图的示例。

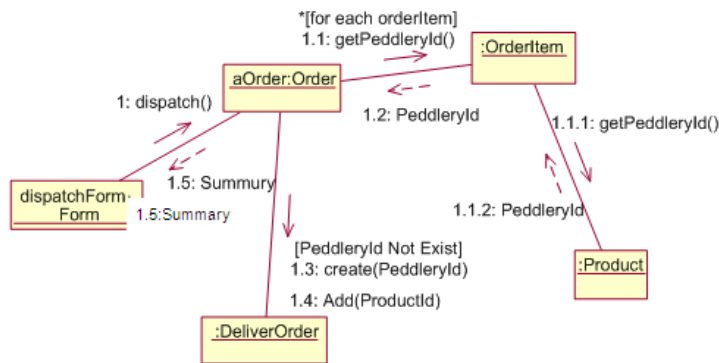


图2-17 通信图示例

## 2. 构件图

构件图是面向对象系统的物理方面进行建模时要用的两种图之一。它可以有效地显示一组构件及它们之间的关系。构件图中通常包括构件、接口及各种关系。图2-18就是一个构件图的例子。

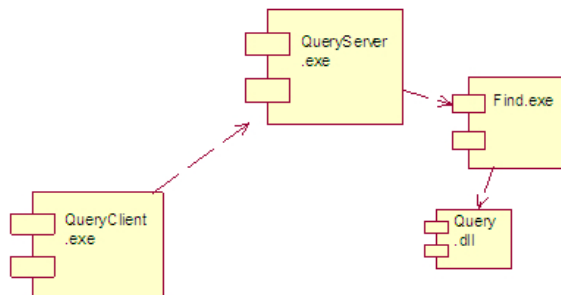


图2-18 构件图示例

构件指的是源代码文件、二进制代码文件和可执行文件等。而构件图就是用来显示编译、链接或执行时构件之间的依赖关系。例如，图2-18说明QueryClient.exe将通过调用QueryServer.exe来完成相应的功能，而QueryServer.exe则需要Find.exe的支持，Find.exe在实现时调用了Query.dll。

通常来说，可以使用构件图完成以下工作：

对源代码进行建模：这样可以清晰地表示出各个不同源程序文件之间的关系。

对可执行体的发布建模：如上图所示，将清晰地表示出各个可执行文件、DLL（动态链接库）文件之间的关系。

对物理数据库建模：用来表示各种类型的数据库、表之间的关系。

对可调整的系统建模：例如对于应用了负载均衡、故障恢复等系统的建模。

在绘制构件图时，应该注意侧重于描述系统的静态实现视图的一个方面，图形不要过于简化，应该为构件图取一个直观的名称，在绘制时避免产生线的交叉。

## 3. 部署图

通过构件图将能够理解系统的物理组成结构，但是它并没有办法体现出这些物理组成部分是如何反映在计算机硬件系统之上的。而部署图正是用来弥补这个不足的，它的关注点就在于系统如何部署。图2-19是一个部署图的例子。

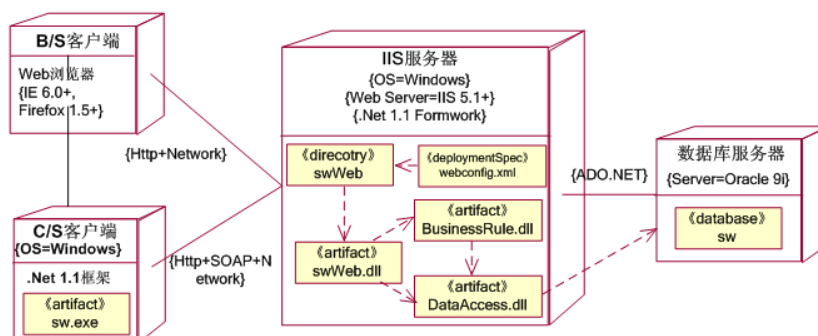


图2-19 部署图示例

部署图也称为实施图，和构件图一样，是面向对象系统的物理方面建模的两种图之一。构件图相对来说，是说明构件之间的逻辑关系，而部署图则是在此基础上更进一步，描述系统硬件的物理拓扑结构以及在此结构上执行的软件。部署图可以显示计算结点的拓扑结构和通信路径、结点上运行的软件构件，常常用于帮助理解分布式系统。

开发团队通过构建和维护部署图，将可以为维护人员提供足够的技术信息支持，以保证部署、安装、维护工作的顺利实施。

版权方授权希赛网发布，侵权必究

[上一节](#)    [本书简介](#)    [下一节](#)

本节从历年考试真题中，精选出6道典型的试题进行分析，这6道试题所考查的知识点基本上覆盖了本章的所有内容，非常具有代表性。

版权方授权希赛网发布，侵权必究

[上一节](#)    [本书简介](#)    [下一节](#)

希赛教育公司决定开发一个管理所有客户信息的交互式网络系统。系统的功能如下：

（1）浏览客户信息：任何使用Internet的网络用户都可以浏览电话公司所有的客户信息（包括姓名、住址、电话号码等）。

（2）登录：电话公司授予每个客户一个帐号。拥有授权帐号的客户，可以使用系统提供的页面设置个人密码，并使用该帐号和密码向系统注册。

（3）修改个人信息：客户向系统注册后，可以发送电子邮件或者使用系统提供的页面，对个人信息进行修改。

（4）删除客户信息：只有公司的管理人员才能删除不再接受公司服务的客户的信息。

系统采用面向对象方法进行开发，在开发过程中认定出的类如表2-4所示。

表2-4 开发过程中认定出的类

编号	类名	描述
1	InternetClient	网络用户
2	CustomerList	客户信息表，记录公司所有客户的信息
3	Customer	客户信息，记录单个客户的信息
4	CompanyCustomer	公司客户
5	InternalClient	公司的管理人员

【问题1】

在需求分析阶段，采用UML的用例图描述系统功能需求，如图2-20所示。请指出图中的A、B、C和D分别是哪个用例？

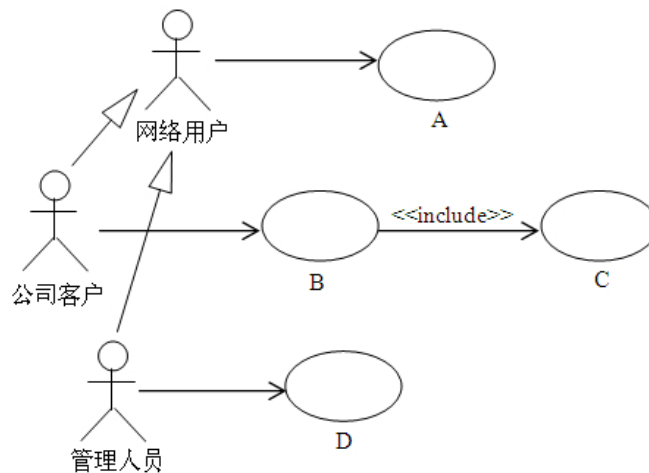


图2-20 用例图

【问题2】

在UML中，重复度定义了某个类的一个实例可以与另一个类的多少个实例相关联。通常把它写成一个表示取值范围的表达式或者一个具体的值。例如，图2-21中的类InternetClient和CustomerList，InternetClient端的“0..\*”表示：一个CustomerList的实例可以与0个或多个InternetClient的实例相关联；CustomerList端的“1”表示：一个InternetClient的实例只能与一个CustomerList的实例相关。

请指出图2-21中（1）到（4）处的重复度分别为多少？

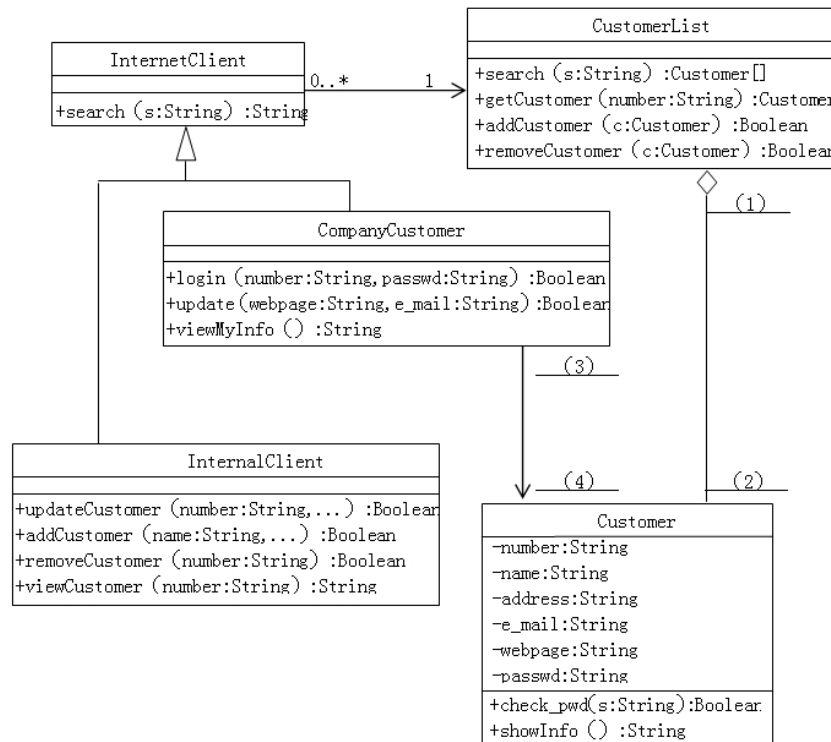


图2-21 类图

【问题3】

类通常不会单独存在，因此当对系统建模时，不仅要识别出类，还必须对类之间的相互关系建模。在面向对象建模中，提供了四种关系：依赖（dependency）、概括（generalization）、关联（association）和聚集（aggregation）。请分别说明这四种关系的含义，并说明关联和聚集之间

的主要区别。

### 例题1分析

本题涉及到用例之间的关系、类的重复度，以及类之间的关系等问题。

#### 【问题1】

图2-20中的网络用户、公司客户、管理人员都是参与者。题目说明中提到了系统有4个功能：浏览客户信息、登录、修改个人信息、删除客户信息。这也就是4个用例，现在只需把它们对号入座即可。根据题目说明可以知道，任何使用Internet的网络用户都可以浏览电话公司所有的客户信息，在图2-20中符合这一条件的只有A了，所以A应填“浏览客户信息”。又因为只有公司的管理人员才能删除不再接受公司服务的客户的信息，所以D应填“删除客户信息”。

剩下就只有“登录”和“修改个人信息”两个用例了，B和C考查的是这2个用例之间的关系。根据“<<include>>”可知，这里是包含关系。根据常识我们知道，在修改个人信息之前需要登录，因此，“修改个人信息”包含“登录”表，即B应填“修改个人信息”，C应填“登录”。

#### 【问题2】

在UML中，重复度又称多重性，多重性表示为一个整数范围n..m，整数n定义所连接的最少对象的数目，而m则为最多对象数（当不知道确切的最大数时，最大数用\*号表示）。最常见的多重性有0..1、0..\*、1..1和1..\*。

因为一个CustomerList的实例可以与0个或多个Customer的实例相关联；而一个Customer的实例只能与一个CustomerList的实例相关。所以（1）应填“1”，（2）应填“0..\*”。因为Customer是CompanyCustomer相应的详细信息，所以（3）和（4）都应该填写“0..1”。

#### 【问题3】

类之间的各种关系，请参考2.2.3节点内容，此处不再赘述。

### 例题1参考答案

#### 【问题1】

A：浏览客户信息 B：修改个人信息

C：登录 D：删除客户信息

#### 【问题2】

（1）1 （2）0..\* （3）0..1 （4）0..1

#### 【问题3】

（1）4种关系的含义

依赖表示类之间的使用关系。

概括表示一般类和特殊类之间的关系。

关联和聚集都表示实例之间的结构关系。

（2）关联和聚集的区别

关联指明一个类的对象与另一个类的对象间的联系；两个类之间的关联表示了两个同等地位类之间的结构关系，这两个类在概念上是同级别的。

聚集是一种特殊的关联，它表示整体/部分关系。

## 例题2

某公司的主要业务是出租图书和唱碟。由于业务需求，该公司委托希赛公司开发一套信息管理系统。该系统将记录所有的图书信息、唱碟信息、用户信息、用户租借信息等。希赛公司决定采用面向对象的分析和设计方法开发此系统。图2-22所示为某类图书或唱碟被借阅时应记录的信息，图2-23描述了系统定义的两个类Book和CD，分别表示图书和唱碟的信息。



图2-22 借阅时应记录的信息

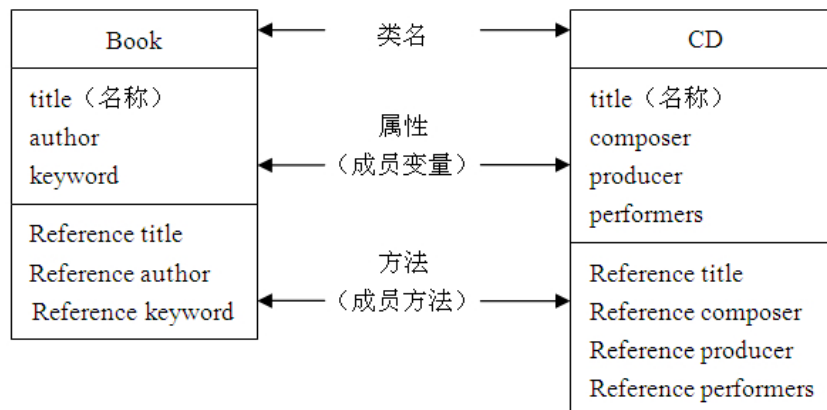


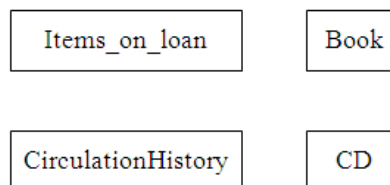
图2-23 借阅时应记录的信息

### 【问题1】

经过进一步分析，设计人员决定定义一个类Items\_on\_loan，以表示类Book和CD的共有属性和方法。请采用图2-23中属性和方法的名称给出类Items\_on\_loan应该具有的属性和方法。（注意：不同名称的属性和方法表示不同的含义，如CD中的composer与Book中的author无任何关系）

### 【问题2】

为了记录每种图书或唱碟租借的历史记录，引入类CirculationHistory，类中存储的信息是图2-22中所表示的内容。请采用UML表示法将下列四个类之间的关系表示出来。



### 【问题3】

现需了解十大最畅销（借出次数最多）图书或唱碟。为此，引入TenPopulate类以存储所有十大畅销图书或CD的名称及其被借出的次数。图2-24的顺序图描述了某类图书或唱碟被借出后成为十



大畅销图书或唱碟时对象间的消息交互。系统在一次运行过程中，应有 (1) 个TenPopulate实例对象最合适，一个TenPopulate类实例对象最多需要和 (2) 个Items\_on\_loan实例对象交互。

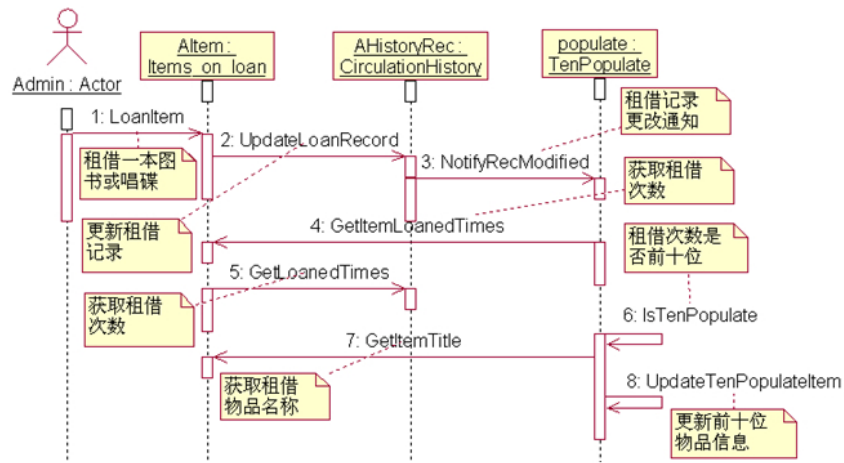


图2-24 顺序图

### 例题2分析

本题主要涉及类的设计、类之间的关系和顺序图。

在面向对象的程序设计当中，类的设计是非常重要的，类设计的合理性直接影响到整个系统的性能。

#### 【问题1】

问题1要求考生写出类Items\_on\_loan的属性和方法，由于题目已经说明此类的属性和方法是Book类和CD类的公共属性和方法；又因为Book类和CD类中，不同名的属性、方法表示的含义不同，所以公共属性和方法就是同名属性和方法，因此，Items\_on\_loan的属性有title，方法有Reference title。

#### 【问题2】

问题2引入了CirculationHistory类，此类用于记录每种图书或者光碟的租借记录。现要求CirculationHistory类、Book类、CD类及Items\_on\_loan类之间的关系，根据【问题1】可以知道，Items\_on\_loan是类Book和CD的公共部分，用面向对象的术语来说，类Items\_on\_loan是类Book和CD的父类，所以它们之间存在继承关系。

再看CirculationHistory类和其他类的关系，CirculationHistory类只需要记录图书或唱碟的名称及借阅记录，而不需要其他详细资料，这样，CirculationHistory不必和Book与CD产生关系，只需要与Items\_on\_loan产生关系即可。由于CirculationHistory中除记录图书或唱碟名称以外，还需要记录借出时间、归还时间及用户名，这些数据无法从Items\_on\_loan中获取。一个CirculationHistory只包含一个Items\_on\_loan，存在1:1的关系，这说明Items\_on\_loan其实只是CirculationHistory的组成部分，但Items\_on\_loan可脱离CirculationHistory而独立存在，也就是说，一本图书或一张CD可以没有记录其借阅历史的CirculationHistory，但有记录其基本信息的Items\_on\_loan，所以它们之间又存在聚集关系（而不是那种部分随整体销毁而销毁的组合关系）。综上所述，4个类的关系如图2-25所示。

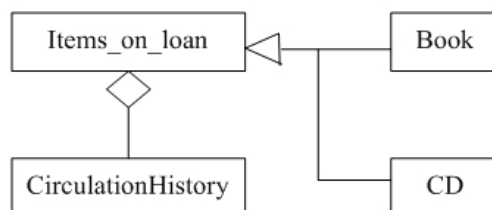




图2-25 4个类之间的关系

聚合关联中涉及到重复度，当没有指定重复度时，默认重复度为1，那么，图2-25中两个类CirculationHistory、Items\_on\_loan所在端的重复度都为1。

【问题3】

题目中说“引入TenPopulate类以存储所有十大畅销图书或CD的名称及其被借出的次数”，可见TenPopulate类的功能是存储所有十大畅销图书或CD的名称及其被借出的次数。既然如此，系统在一次运行中只需要1个TenPopulate实例对象就可以了，因为它存储所有十大畅销图书或CD的名称及其被借出的次数。每当有图书或唱碟被借出时，都需要和TenPopulate类的对象发生交互，因此，当所有图书或CD都被借阅时，TenPopulate类实例对象需要跟所有这些Items\_on\_loan实例对象交互更新借出次数以评出十大最畅销图书或CD，一个TenPopulate类实例对象最多需要和“图书和唱碟种类总数”个Items\_on\_loan实例对象交互。

例题2参考答案

【问题1】

属性：title

方法：Reference Title

【问题2】

如图2-25所示。

【问题3】

(1) 1

(2) 图书和唱碟种类数

版权方授权希赛网发布，侵权必究

上一节

本书简介

下一节

### 例题3

希赛客户信息管理系统中保存着两类客户的信息：

(1) 个人客户。对于这类客户，系统保存了其客户标识（由系统生成）和基本信息（包括姓名、住宅电话和E-mail）。

(2) 集团客户。集团客户可以创建和管理自己的若干名联系人。对于这类客户，系统除了保存其客户标识（由系统生成）之外，也保存了其联系人的信息。联系人的信息包括姓名、住宅电话、E-mail、办公电话以及职位。

该系统除了可以保存客户信息之外，还具有以下功能：

- (1) 向系统中添加客户（addCustomer）；
- (2) 根据给定的客户标识，在系统中查找该客户（getCustomer）；
- (3) 根据给定的客户标识，从系统中删除该客户（removeCustomer）；
- (4) 创建新的联系人（addContact）；
- (5) 在系统中查找指定的联系人（getContact）；

( 6 ) 从系统中删除指定的联系人 ( removeContact ) 。

该系统采用面向对象方法进行开发。在面向对象分析阶段，根据上述描述，得到如表2-5所示的类。

表2-5 得到的各种类

类名	说明
CustomerInformationSystem	客户信息管理系统
IndividualCustomer	个人客户
InstitutionalCustomer	集团客户
Contact	联系人

类名说明

CustomerInformationSystem客户信息管理系统

IndividualCustomer个人客户

InstitutionalCustomer集团客户

Contact联系人

描述该客户信息管理系统的UML类图如图2-26所示。

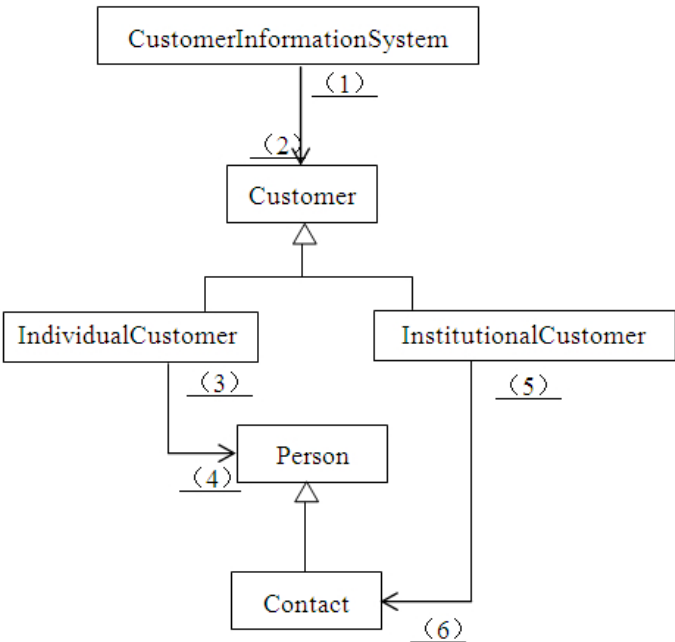


图2-26 客户信息管理系统的UML类图

【问题1】

请使用说明中的术语，给出图2-26中类Customer和类Person的属性。

【问题2】

识别关联的多重度是面向对象建模过程中的一个重要步骤。根据说明中给出的描述，完成图中的( 1 ) ~ ( 6 )。

【问题3】

根据说明中的叙述，抽象出如表2-6所示的方法，请指出图2-26中的类CustomerInformationSystem和InstitutionalCustomer应分别具有其中的哪些方法。

表2-6 抽象出的方法

功能描述	方法名
向系统中添加客户	addCustomer
根据给定的客户标识，在系统中查找该客户	getCustomer
根据给定的客户标识，从系统中删除该客户	removeCustomer
创建新的联系人	addContact
在系统中查找指定的联系人	getContact
从系统中删除指定的联系人	removeContact

### 例题3分析

根据题目描述得知，客户信息管理系统的功能是管理个人用户和集团用户，个人用户和集团用户的区别在于集团用户有自己的联系人，而个人用户没有。

#### 【问题1】

问题1要求考生给出类Customer和Person的属性。在图2-26中，Customer表示客户类，IndividualCustomer和InstitutionalCustomer都是Customer的子类，IndividualCustomer是个人客户，InstitutionalCustomer是集团客户。既然Customer是个人客户类和集团客户类的父类，则它必有两者的共同点。由于集团客户和个人客户都有客户标识，所以Customer的属性为“客户标识”。Contact是联系人，只有集团客户才有联系人。Person类是Contact类的父类，同时又与IndividualCustomer有关联，所以Person类应具有Contact与IndividualCustomer的共同点，它们的共同点就是联系人与个人客户都有自己的基本信息：姓名、住宅电话和E-mail。因此，Person的属性有：姓名、住宅电话和E-mail。

#### 【问题2】

类CustomerInformationSystem表示的是客户信息管理系统，Customer表示的是客户类，一个客户信息管理系统中应有多个客户。所以CustomerInformationSystem与Customer之间的关系应是1:\*. IndividualCustomer与Person其实是个人客户与该客户的基本信息之间的关系，显然一个客户只有一个基本信息，所以它们也是1:1的关系。最后是InstitutionalCustomer与Contact之间的关系，这也就是集团客户与联系人之间的关系，由于一个集团客户可有多个联系人，同时至少需要有一个联系人（如果联系人数量为0，则此客户为个人客户）。所以InstitutionalCustomer与Contact之间的关系应为1:1..\*。

#### 【问题3】

一个原则就可以解决这个问题：与客户操作相关的为CustomerInformationSystem的方法，而与联系人相关的操作是InstitutionalCustomer的方法。

### 例题3参考答案

#### 【问题1】

Customer的属性：客户标识。

Person的属性：姓名、住宅电话、E-mail。

#### 【问题2】

(1) 1 (2) 0..\* (3) 1

(4) 1 (5) 1 (6) 1..\*

#### 【问题3】

CustomerInformationSystem的方法：addCustomer，getCustomer，removeCustomer。

InstitutionalCustomer的方法：addContact，getContact，removeContact。

例题4

希赛教育图书管理系统的主要功能如下：

- （1）图书管理系统的资源目录中记录着所有可供读者借阅的资源，每项资源都有一个唯一的索引号。系统需登记每项资源的名称、出版时间和资源状态（可借阅或已借出）。
  - （2）资源可以分为两类：图书和唱片。对于图书，系统还需登记作者和页数；对于唱片，还需登记演唱者和介质类型（CD或者磁带）。
  - （3）读者信息保存在图书管理系统的读者信息数据库中，记录的信息包括：读者的识别码和读者姓名。系统为每个读者创建了一个借书记录文件，用来保存读者所借资源的相关信息。
- 现采用面向对象方法开发该图书管理系统。识别类是面向对象分析的第一步。比较常用的识别类的方法是寻找问题描述中的名词，再根据相关规则从这些名词中删除不可能成为类的名词，最终得到构成该系统的类。

表2-7给出了上述描述中的所有名词。

表2-7 所有名词

图书管理系统	资源目录	读者	资源
索引号	系统	名称	出版时间
资源状态	图书	唱片	作者
页数	演唱者	介质类型	CD
磁带	读者信息	读者信息数据库	识别码
姓名	借书记录文件	信息	

通过对表2-7中的名词进行分析，最终得到了图2-27所示的UML类图（类的说明如表2-8所示）。

表2-8 类的说明

类 名	说 明
LibrarySystem	图书管理系统
BorrowerDB	保存读者信息的数据库
CatalogItem	资源目录中保存的每项资源
Borrower	读者
BorrowerItems	为每个读者创建的借书记录文件

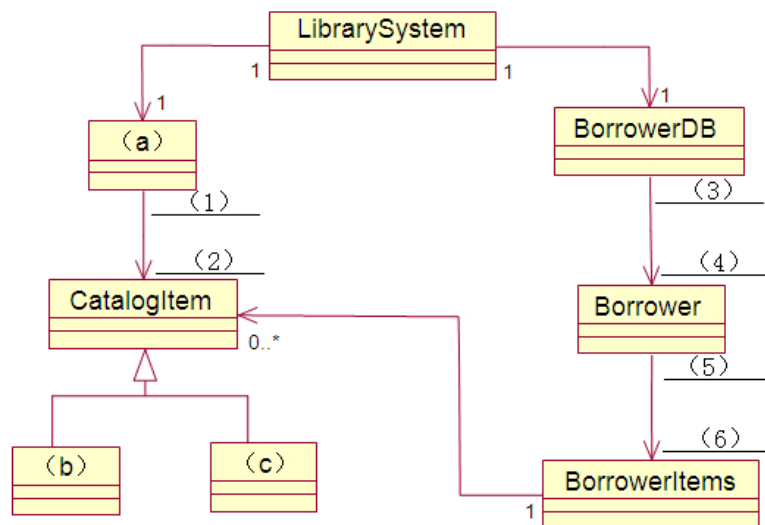


图2-27 UML类图

【问题1】

表2-8所给出的类并不完整，根据题目说明和表2-7，将图2-27中的（a）~（c）处补充完整。

【问题2】

根据题目中的描述，给出图2-27中的类CatalogItem以及（b）、（c）处所对应的类的关键属性（使用表2-7中给出的词汇），其中，CatalogItem有4个关键属性；（b）、（c）处对应的类各有2个关键属性。

【问题3】

识别关联的多重度是面向对象建模过程中的一个重要步骤。根据题目中给出的描述，完成图2-27中的（1）~（6）。

例题4分析

本题主要考查UML中的类图设计，3个问题都是对类图的元素进行补充。类图的设计是根据系统的功能需求而来的，所以解题的关键在于对系统功能说明的理解。

（1）从系统功能说明中的“图书管理系统的资源目录中记录着所有可供读者借阅的资源”和“资源可以分为两类：图书和唱片”，可以得知1个资源目录中对应着多个可供读者借阅的资源，同时图书类与唱片类是资源类的子类。所以（a）为资源目录，（b）和（c）分别为图书和唱片，同时（1）处应填“1”，（2）处应填“0..\*”（所有的可供读者借阅资源数有可能为0，即还未录入任何资源的状态）。

（2）从“每项资源都有一个唯一的索引号。系统需登记每项资源的名称、出版时间和资源状态”，可以得知，资源目录中的每项资源，即类图中的CatalogItem，有索引号、名称、出版时间和资源状态4个关键属性。

（3）从“对于图书，系统还需登记作者和页数；对于唱片，还需登记演唱者和介质类型（CD或者磁带）”可以得知，图书有作者和页数2个关键属性，唱片有演唱者和介质类型2个关键属性。

（4）BorrowerDB代表保存读者信息的数据库，而Borrower代表读者，所以它们之间的关系毫无疑问是1对多，但具体是0..\*还是1..\*呢？应是0..\*，因为读者信息数据库可以为空，即还没有任何读者。即第（3）空填“1”，第（4）空填“0..\*”。

（5）由于Borrower代表读者，而BorrowerItems为借书记录文件，同时系统功能说明中有“系统为每个读者创建了一个借书记录文件，用来保存读者所借资源的相关信息”，所以它们之

间的关系应为1对1，即第（5）空和第（6）空均填“1”。

#### 例题4参考答案

##### 【问题1】

（a）资源目录（b）图书（c）唱片

注：（b）和（c）的答案可以互换

##### 【问题2】

CatalogItem的属性：索引号、名称、出版时间、资源状态。

图书的属性：作者、页数。

唱片的属性：演唱者、介质类型。

##### 【问题3】

（1）1（2）0..\*（3）1

（4）0..\*（5）1（6）1 或者 0..1

版权方授权希赛网发布，侵权必究

上一节

本书简介

下一节

### 例题5

某汽车停车场欲建立一个信息系统，已经调查到的需求如下：

（1）在停车场的入口和出口分别安装一个自动栏杆、一台停车卡打印机、一台读卡器和一个车辆通过传感器，示意图如图2-28所示。

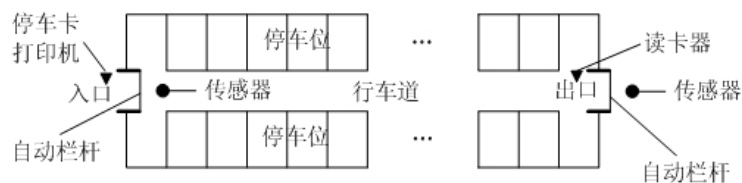


图2-28 停车示意图

（2）当汽车到达入口时，驾驶员按下停车卡打印机的按钮获取停车卡。当驾驶员拿走停车卡后，系统命令栏杆自动抬起；汽车通过入口后，入口处的传感器通知系统发出命令，栏杆自动放下。

（3）在停车场内分布着若干个付款机器。驾驶员将在入口处获取的停车卡插入付款机器，并缴纳停车费。付清停车费之后，将获得一张出场卡，用于离开停车场。

（4）当汽车到达出口时，驾驶员将出场卡插入出口处的读卡器。如果这张卡是有效的，系统命令栏杆自动抬起；汽车通过出口后，出口传感器通知系统发出命令，栏杆自动放下。若这张卡是无效的，系统不发出栏杆抬起命令而发出告警信号。

（5）系统自动记录停车场内空闲的停车位的数量。若停车场当前没有车位，系统将在入口处显示“车位已满”信息。这时，停车卡打印机将不再出卡，只允许场内汽车出场。

根据上述描述，采用面向对象方法对其进行分析与设计，得到了表2-9所示的类/用例/状态列表、图2-29所示的用例图、图2-30所示的初始类图以及图2-31所示的描述入口自动栏杆行为的UML

状态图。

表2-9 类/用例/状态列表

用例名	说明	类名	说明	状态名	说明
Car entry	汽车进入停车场	CentralComputer	停车场信息系统	Idle	空闲状态, 汽车可以进入停车场
Car exit	汽车离开停车场	PaymentMachine	付款机器	Disable	没有车位
Report Statistics	记录停车场的相关信息	CarPark	停车场, 保存车位信息	Await Entry	等待汽车进入
		Barrier	自动护栏	Await Ticket Take	等待打印停车卡
Car entry when full	没有车位时, 汽车请求进入停车场	EntryBarrier	入口的护栏	Await Enable	等待停车场内有空闲车位
		ExitBarrier	出口的护栏		

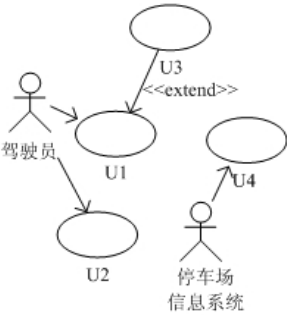


图2-29 用例图

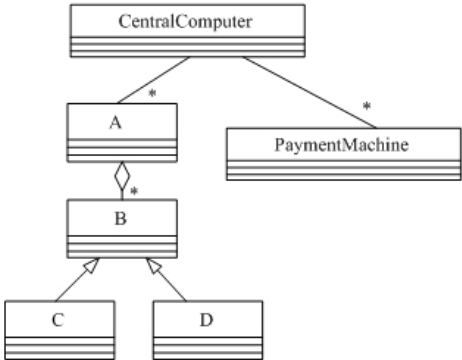


图2-30 初始类图

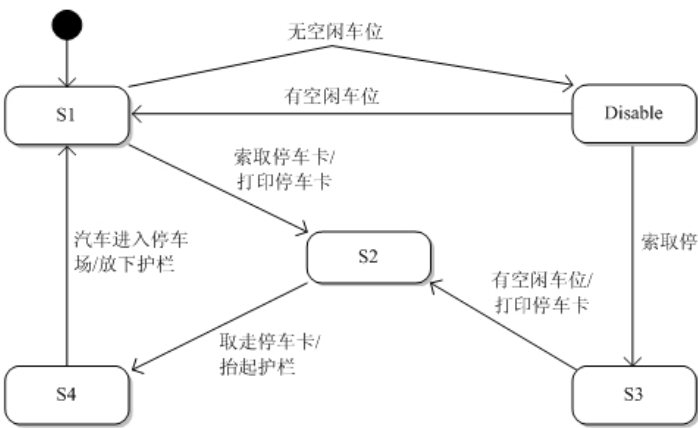


图2-31 入口护栏的状态图

【问题1】

根据说明中的描述，使用表2-9给出的用例名称，给出图2-29中U1、U2和U3所对应的用例。

【问题2】

根据说明中的描述，使用表2-9给出的类的名称，给出图2-30中的A~D所对应的类。

【问题3】

根据说明中的描述，使用表2-9给出的状态名称，给出图2-31中S1~S4所对应的状态。

#### 【问题4】

简要解释图2-29中用例U1和U3之间的extend关系的内涵。

#### 例题5分析

本题考查的面比较广，涉及到用例图、类图 and 状态图。

#### 【问题1】

题目中车辆入场和出场，而入场时分有空位和无空位的情形，当无车位时显示“车位已满”信息。这时，停车卡打印机将不再出卡，只允许场内汽车出场。说明入场时，没有车位入场是一种扩展关系。根据图2-29和表2-9可以得出U1为Car entry，U2为Car exit，U3为Car entry when full。

#### 【问题2】

根据题目的描述及表2-9中的内容，可以先来确定类B。汽车出入口，当卡有效时，系统自动抬起栏杆；当卡无效时，则系统不抬栏杆，且发出警告。所以自动护栏类（Barrier）有两种子类：一个是入口的护栏类（EntryBarrier），另一个就是出口的护栏类（ExitBarrier）。构成了这种父子关系的类在图2-30中表示为：类B为护栏类（Barrier），类C为入口护栏类（EntryBarrier），类D为出口护栏类（ExitBarrier）。

再确定类A，由于停车场管理系统管理着多张卡，从图2-30中可以看出类Centralcomputer与类A之间有1..\*的关系；而且类A与类B（Barrier）之间存在聚集关系；题目的描述中有：当有车位时允许入场，无车位时停车卡打印机将不再出卡，只允许场内汽车出场。所以一张卡片可以确定多个护栏抬起或不发卡入场，由表2-9可以得出类A为停车场保存卡位信息类（CarPark）。

#### 【问题3】

根据题目的描述和表2-9，黑点表示开始状态，到达S1，很容易确定S1为状态：Idle（空闲状态，汽车可以进入停车场）。又因为状态Disable（没有车位）到S3有事件“索取停车卡”，而从S3到S2有事件“有空闲车位/打印停车卡”，由题目的第（4）、（5）点可知，车位满了后，若有车辆出去，则释放一个车位；若没有，则等待打印停车卡。所以可以确定S3的状态为Await Ticket Take（等待打印停车卡）。

S1到S2有事件“索取停车卡/打印停车卡”，S2到S4有事件“取走停车卡/抬起护栏”，包括S3到S2有事件“有空闲车位/打印停车卡”，则说明S2这个状态都与“有车位，才发卡”有关，要等待有车位才发卡，或取卡放行后进入等待。所以S2为状态Await Enable（等待停车场内有空位）。

最后确定S4。由于S2到S4有事件“取走停车卡/抬起护栏”，S4到S1有事件“汽车进入停车场/放下护栏”。很显然，当取走停车卡/抬起护栏将车子放行后，管理系统将停车位的空闲车位数加1；当汽车进入停车场/放下护栏后，管理系统将停车位的空闲车位数减1。因此状态S4为Await Entry（等待汽车进入）。

#### 【问题4】

题目中汽车的入场，通常是指有空位入场；但也有要入场但没有空位的情况，这要等待。而这种关系就是扩展了的入场关系。

#### 例题5参考答案

#### 【问题1】

U1：Car entry U2：Car exit U3：Car entry when full

#### 【问题2】

A：CarPark B：Barrier C：EntryBarrier D：ExitBarrier



其中，C、D的答案可以互换

【问题3】

S1：IdleS2：Await Ticket TakeS3：Await EnableS4：Await Entry

【问题4】

用例之间的延伸关系用于对被用户看作是可选系统行为的用例的一部分建模。通过这种方式，可以把可选行为从必需的行为中分离出来。

版权方授权希赛网发布，侵权必究

上一节      本书简介      下一节

例题6

某银行计划开发一个自动存提款机模拟系统（ATM System）。系统通过读卡器（CardReader）读取ATM卡；系统与客户（Customer）的交互由客户控制台（CustomerConsole）实现；银行操作员（Operator）可控制系统的启动（System Startup）和停止（System Shutdown）；系统通过网络和银行系统（Bank）实现通信。

当读卡器判断用户已将ATM卡插入后，创建会话（Session）。会话开始后，读卡器进行读卡，并要求客户输入个人验证码（PIN）。系统将卡号和个人验证码信息送到银行系统进行验证。验证通过后，客户可从菜单选择如下事务（Transaction）：

- 从ATM卡账户取款（Withdraw）；
- 向ATM卡账户存款（Deposit）；
- 进行转账（Transfer）；
- 查询（Inquire）ATM卡账户信息。

一次会话可以包含多个事务，每个事务处理也会将卡号和个人验证码信息送到银行系统进行验证。若个人验证码错误，则转个人验证码错误处理（Invalid PIN Process）。每个事务完成后，客户可选择继续上述事务或退卡。选择退卡时，系统弹出ATM卡，会话结束。

系统采用面向对象方法开发，使用UML进行建模。系统的顶层用例图如图2-32所示，一次会话的顺序图（不考虑验证）如图2-33所示。消息名称参见表2-10。

表2-10 可能的消息名称列表

名称	说明	名称	说明
cardInserted（）	ATM 卡已插入	performTransaction（）	执行事务
performSession（）	执行会话	readCard（）	读卡
readPIN（）	读取个人验证码	PIN	个人验证码信息
creat（atm, this, card, pin）	为当前会话创建事务	create（this）	为当前 ATM 创建会话
card	ATM 卡信息	doAgain	执行下一个事务
ejectCard（）	弹出 ATM 卡		

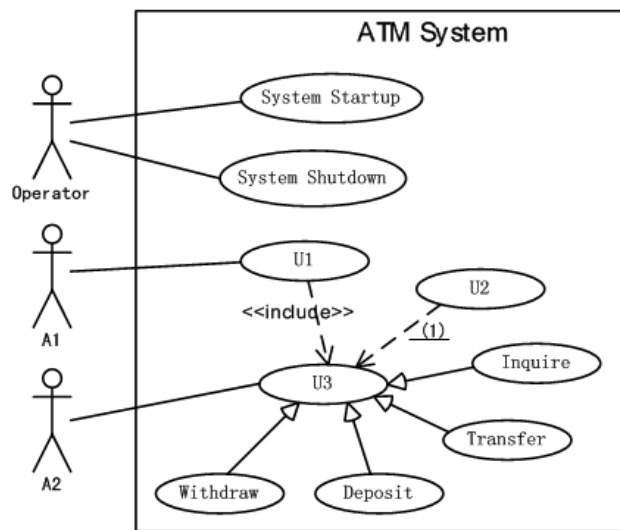


图2-32 ATM系统顶层用例图

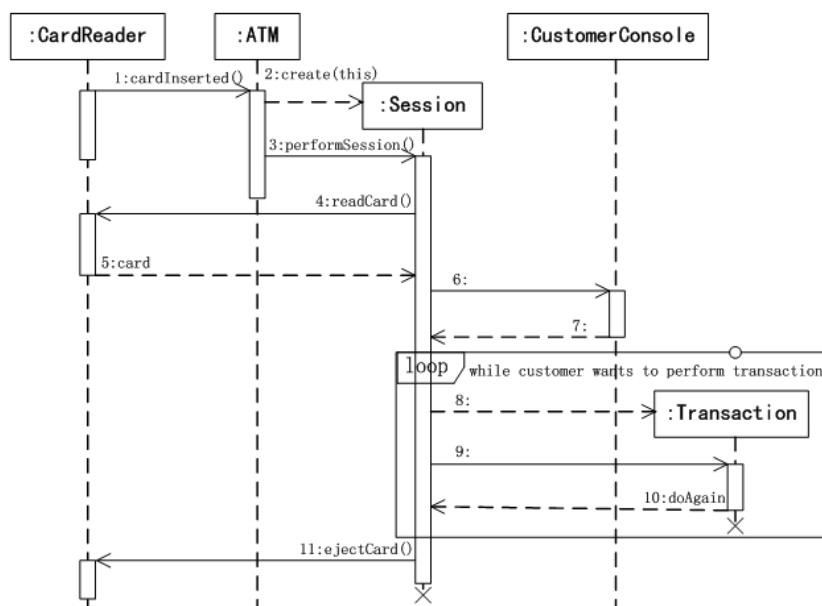


图2-33 一次会话的顺序图（无验证消息）

#### 【问题1】

根据题目中的描述，给出图2-32中A1和A2所对应的参与者，U1至U3所对应的用例，以及该图中空（1）所对应的关系。（U1至U3的可选用例包括：Session、Transaction、Insert Card、Invalid PIN Process和Transfer）

#### 【问题2】

根据题目中的描述，使用表2-10中的英文名称，给出图2-33中6~9对应的消息。

#### 【问题3】

解释图2-32中用例U3和用例Withdraw、Deposit等四个用例之间的关系及其内涵。

#### 例题6分析

本题涉及用例图和顺序图，以及用例之间的关系。

#### 【问题1】

构建用例图时，常用的方式是先识别参与者，然后确定用例以及用例之间的关系。识别参与者时，考查和系统交互的人员和外部系统。在本题中，与系统交互的人员包括客户（Customer）和银行操作员（Operator），与本模拟系统交互的外部系统包括银行（Bank）。

客户一旦插卡成功，系统就创建会话（Session），会话中可以执行用户从菜单选择的

Withdraw、Deposit、Transfer和Inquire等事务（Transaction）。由图2-32中U3和Withdraw之间的扩展关系，可知U3为Transaction；又由U1和U3之间的<<include>>关系，得知U1为Session，进而判定图中A1为Customer，A2为Bank。每个事务处理也会将卡号和个人验证码信息送到银行系统进行验证，若个人验证码错误，则转个人验证码错误处理（Invalid PIN Process，图中U2），所以（1）处应填“<<extend>>”。

#### 【问题2】

根据题目中的描述，从ATM机判断卡已插入（cardInserted（））开始会话，即为当前ATM创建会话（create（this））并开始执行会话（performSession（））；读卡器读片（readCard（））获得ATM卡信息（card），然后从控制台读取个人验证码输入（readPIN（），图2-33中标号6处）并获得个人验证码信息（PIN，图2-33中标号7处）；然后根据用户选择启动并执行事务，即为当前会话创建事务（creat（atm, this, card, pin），图2-33中标号8处）和执行事务（performTransaction（），图2-33中标号9处）；可以选择继续执行某个事务（doAgain）循环，或者选择退卡（ejectCard（））。

#### 【问题3】

显然，Transaction和Withdraw、Deposit等4个用例之间的关系为泛化关系。

#### 例题6参考答案

##### 【问题1】

A1：Customer A2：Bank U1：Session

U2：Invalid PIN Process U3：Transaction（1）：<<extend>>

##### 【问题2】

6：readPIN（） 7：PIN

8：creat（atm, this, card, pin） 9：performTransaction（）

##### 【问题3】

Transaction是一个抽象泛化用例，具有其它事务类型共有的属性和行为，每个具体的事务类型继承它，并实现适合自己的特定的操作。

版权方授权希赛网发布，侵权必究

[上一节](#)    [本书简介](#)    [下一节](#)

## 考前必练

为了帮助读者进行考前训练，本节给出5道典型的试题，以及这些试题的分析与解答。请读者独立完成这些练习题，然后再去阅读试题分析与解答。根据自己所做试题的情况，查漏补缺。

版权方授权希赛网发布，侵权必究

[上一节](#)    [本书简介](#)    [下一节](#)

试题1

考前必做的练习题

某指纹门禁系统的体系结构如图2-34所示，其主要部件有：主机（MainFrame）、锁控器（LockController）、指纹采集器（FingerReader）和电控锁（Lock）。

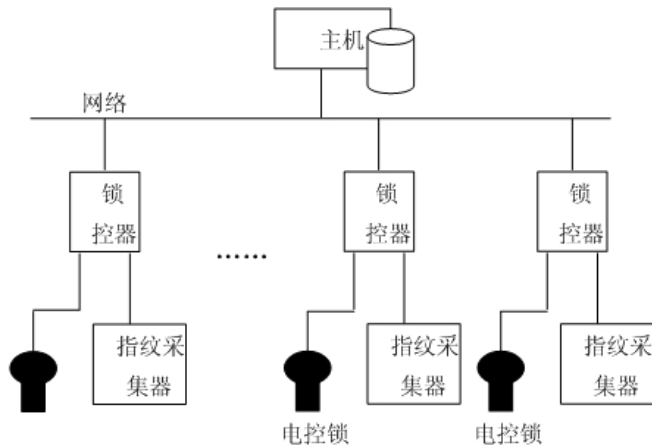


图2-34 某指纹门禁系统的体系结构

- （1）系统中的每个电控锁都有一个惟一的编号。锁的状态有两种：“已锁住”和“未锁住”。
- （2）在主机上可以设置每把锁的安全级别以及用户的开锁权限。只有当用户的开锁权限大于或等于锁的安全级别并且锁处于“已锁住”状态时，才能将锁打开。
- （3）用户的指纹信息、开锁权限以及锁的安全级别都保存在主机上的数据库中。
- （4）用户开锁时，只需按一下指纹采集器。指纹采集器将发送一个中断事件给锁控器，锁控器从指纹采集器读取用户的指纹并将指纹信息发送到主机，主机根据数据库中存储的信息来判断用户是否具有开锁权限，若有且锁当前处于“已锁住”状态，则将锁打开；否则系统报警。

该系统采用面向对象方法开发，系统中的类以及类之间的关系用UML类图表示，图2-35是该系统类图的一部分；系统的动态行为采用UML顺序图表示，图2-36是用户成功开锁的顺序图。

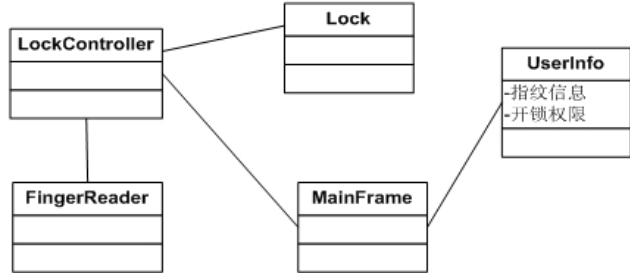


图2-35 系统类图的一部分

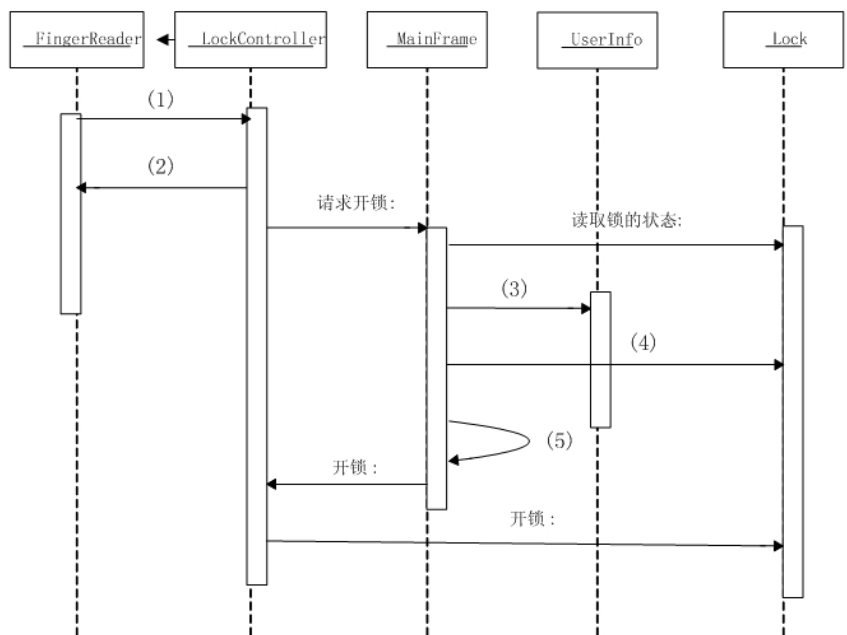


图2-36 顺序图

【问题1】

图2-35是该系统类图的一部分，依据上述说明中给出的术语，给出类Lock的主要属性。

【问题2】

依据上述说明中给出的词语，将图2-36中的（1）~（5）处补充完整。

【问题3】

组装（Composition）和聚集（Aggregation）是UML中两种非常重要的关系。请说明组装和聚集分别表示什么含义？两者的区别是什么？

版权方授权希赛网发布，侵权必究

上一节      本书简介      下一节

试题2

希赛公司开办了在线电子商务网站，主要为各注册的商家提供在线商品销售功能。为更好地吸引用户，希赛公司计划为注册的商家提供商品（Commodity）促销（Promotion）功能。商品的分类（Category）不同，促销的方式和内容会有所不同。

注册商家可发布促销信息。商家首先要在自己所销售的商品的分类中，选择促销涉及的某一具体分类，然后选出该分类的一个或多个商品（一种商品仅仅属于一种分类），接着制定出一个比较优惠的折扣政策和促销活动的优惠时间，最后由系统生成促销信息并将该促销信息公布在网站上。

商家发布促销信息后，网站的注册用户便可通过网站购买促销商品。用户可选择参与某一个促销（Promotion）活动，并选择具体的促销商品（Commodity），输入购买数量等购买信息。系统生成相应的一份促销订单（POrder）。只要用户在优惠活动的时间范围内，通过网站提供的在线支付系统，确认在线支付该促销订单（即完成支付），就可以优惠的价格完成商品的购买活动，否则该促销订单失效。

系统采用面向对象方法开发，系统中的类以及类之间的关系用UML类图表示，图2-37是该系统类图中的一部分；系统的动态行为采用UML顺序图表示，图2-38是发布促销的顺序图。

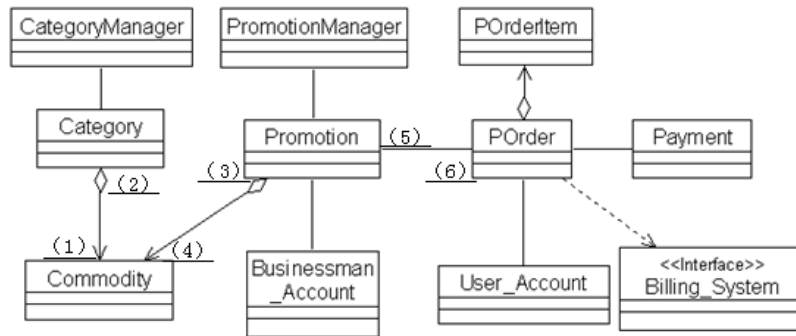


图2-37 在线促销系统部分类图

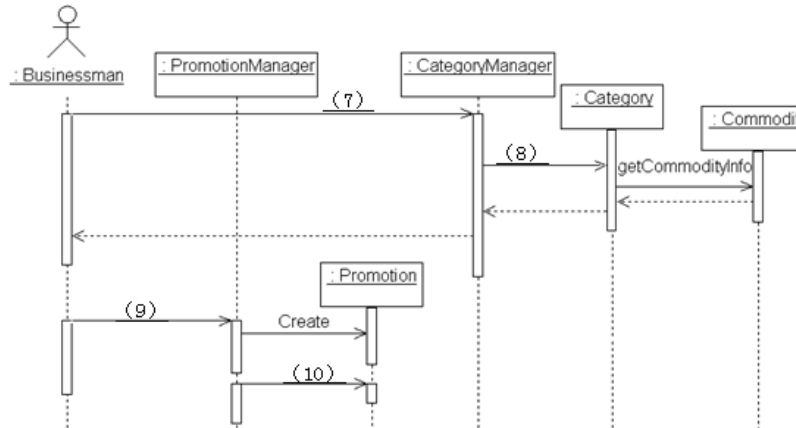


图2-38 发布促销顺序图

【问题1】

识别关联的多重度是面向对象建模过程中的一个重要步骤。根据说明中给出的描述，完成图2-37中的（1）～（6）。

【问题2】

请从表2-11中选择方法，完成图2-38中的（7）～（10）。

表2-11 可选消息列表

功能描述	方法名
向促销订单中添加所选的商品	<u>buyCommodities</u>
向促销中添加要促销的商品	<u>addCommodities</u>
查找某个促销的所有促销订单信息列表	<u>getPromotionOrders</u>
生成商品信息	<u>createCommodity</u>
查找某个分类中某商家的所有商品信息列表	<u>getCommodities</u>
生成促销信息	<u>createPromotion</u>
生成促销订单信息	<u>createPOrder</u>
查找某个分类的所有促销信息列表	<u>getCategoryPromotion</u>
查找某商家所销售的所有分类列表	<u>getCategories</u>
查找某个促销所涉及的所有商品信息列表	<u>getPromotionCommodities</u>

功能描述方法名

向促销订单中添加所选的商品buyCommodities

向促销中添加要促销的商品addCommodities

查找某个促销的所有促销订单信息列表getPromotionOrders

生成商品信息createCommodity

查找某个分类中某商家的所有商品信息列表getCommodities

生成促销信息createPromotion

生成促销订单信息createPOrder  
查找某个分类的所有促销信息列表getCategoryPromotion  
查找某商家所销售的所有分类列表getCategories  
查找某个促销所涉及的所有商品信息列表getPromotionCommodities

【问题3】

关联（Association）和聚集（Aggregation）是UML中两种非常重要的关系。请说明关联和聚集的关系，并说明其不同点。

版权方授权希赛网发布，侵权必究

上一节      本书简介      下一节

试题3

已知某唱片播放器不仅可以播放唱片，而且可以连接电脑并把电脑中的歌曲刻录到唱片上（同步歌曲）。连接电脑的过程中还可自动完成充电。

关于唱片，还有以下描述信息：

（1）每首歌曲的描述信息包括：歌曲的名字、谱写这首歌曲的艺术家以及演奏这首歌曲的艺术家。只有两首歌曲的这三部分信息完全相同时，才认为它们是同一首歌曲。艺术家可能是一名歌手或一支由2名或2名以上的歌手所组成的乐队。一名歌手可以不属于任何乐队，也可以属于一个或多个乐队。

（2）每张唱片由多条音轨构成；一条音轨中只包含一首歌曲或为空，一首歌曲可分布在多条音轨上；同一首歌曲在一张唱片中最多只能出现一次。

（3）每条音轨都有一个开始位置和持续时间。一张唱片上音轨的次序是非常重要的，因此对于任意一条音轨，播放器需要准确地知道，它的下一条音轨和上一条音轨是什么（如果存在的话）。

根据上述描述，采用面向对象方法对其进行分析与设计，得到了如表2-12所示的类列表、如图2-39所示的初始类图以及如图2-40所示的描述播放器行为的UML状态图。

表2-12 类列表

类 名	说 明
Artist	艺术家
Song	歌曲
Band	乐队
Musician	歌手
Track	音轨
Album	唱片

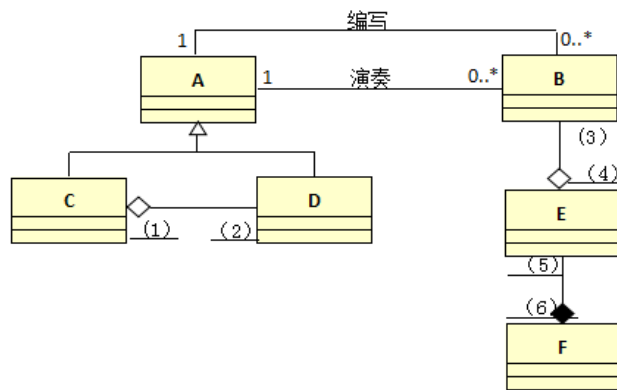


图2-39 初始类图

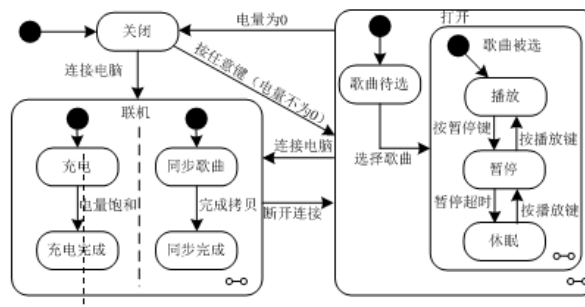


图2-40 播放器行为UML状态图

【问题1】

根据题目中的描述，使用表2-12给出的类的名称，给出图2-39中的A~F所对应的类。

【问题2】

根据题目中的描述，给出图2-39中（1）~（6）处的多重度。

【问题3】

图2-39中缺少了一条关联，请指出这条关联两端所对应的类以及每一端的多重度。

类	多重度

类多重度

【问题4】

根据图2-40所示的播放器行为UML状态图，给出从“关闭”状态到“播放”状态所经过的最短事件序列（假设电池一开始就是有电的）。

版权方授权希赛网发布，侵权必究

上一节 本书简介 下一节

## 试题4

希赛在线会议审稿系统（CSAI Online Reviewing System，CORS）主要处理会议前期的投稿和审稿事务，其功能描述如下：

- （1）用户在初始使用系统时，必须在系统中注册（register）成为作者或审稿人。
- （2）作者登录（login）后提交稿件和浏览稿件审阅结果。提交稿件必须在规定提交时间范围



内，其过程为先输入标题和摘要、选择稿件所属主题类型、选择稿件所在位置（存储位置）。上述几步若未完成，则重复；若完成，则上传稿件至数据库中，系统发送通知。

（3）审稿人登录后可设置兴趣领域、审阅稿件给出意见以及罗列录用和（或）拒绝的稿件。

（4）会议委员会主席是一个特殊审稿人，可以浏览提交的稿件、给审稿人分配稿件、罗列录用和（或）拒绝的稿件以及关闭审稿过程。其中关闭审稿过程须包括罗列录用和（或）拒绝的稿件。

系统采用面向对象方法开发，使用 UML 进行建模。在建模用例图时，常用的方式是先识别参与者，然后确定参与者如何使用系统来确定用例，每个用例可以构造一个活动图。参与者名称、用例和活动名称分别参见表2-13、表2-14和表2-15。系统的部分用例图和提交稿件的活动图分别如图2-41和图2-42所示。

表2-13 参与者列表

名称	说明	名称	说明
User	用户	Author	作者
Reviewer	审稿人	PCChair	委员会主席

表2-14 用例名称列表

名称	说明	名称	说明
login	登录系统	register	注册
submit paper	提交稿件	browse review results	浏览稿件审阅结果
close reviewing process	关闭审稿过程	assign paper to reviewer	分配稿件给审稿人
set preferences	设定兴趣领域	enter review	审阅稿件给出意见
list accepted/rejected papers	罗列录用或/和拒绝的稿件	browse submitted papers	浏览提交的稿件

表2-15 活动名称列表

名称	说明	名称	说明
select paper location	选择稿件位置	upload paper	上传稿件
select subject group	选择主题类型	send notification	发送通知
enter title and abstract	输入标题和摘要		

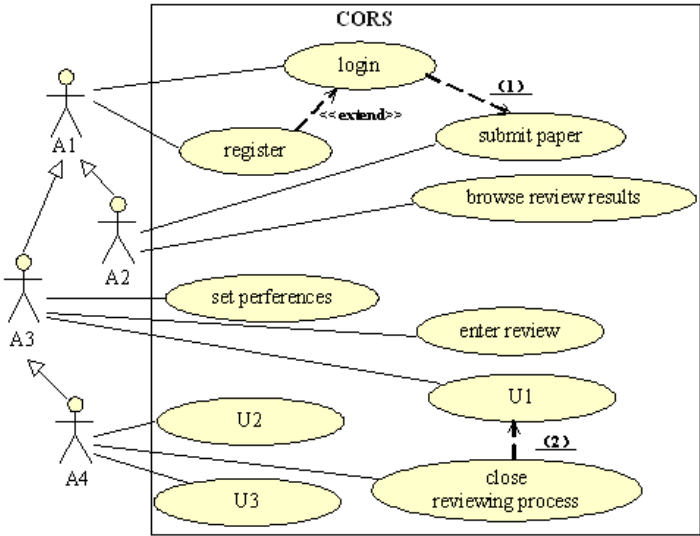


图2-41 CORS 用例图

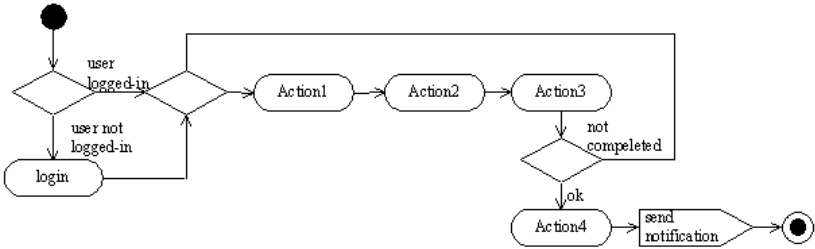


图 2-42 提交稿件过程的活动图

【问题1】

根据题目中的描述，使用表2-13中的英文名称，给出图2-41中A1 ~ A4所对应的参与者。

【问题2】

根据题目中的描述，使用表2-14中的英文名称，给出图2-41中U1 ~ U3所对应的用例。

【问题3】

根据题目中的描述，给出图2-41中（1）和（2）所对应的关系。

【问题4】

根据题目中的描述，使用表2-14和表2-15中的英文名称，给出图2-42中Action1 ~ Action4对应的活动。

版权方授权希赛网发布，侵权必究

[上一节](#)

[本书简介](#)

[下一节](#)

## 试题5

希赛公司为了方便员工用餐，餐厅开发了一个订餐系统（CSAI Cafeteria Ordering System，CCOS），员工可通过企业内联网使用该系统。

企业的任何员工都可以查看菜单和今日特价。

系统的顾客是注册到系统的员工，可以订餐（如果未登录，需先登录）、注册工资支付、预约规律的订餐，在特殊情况下可以覆盖预订。

餐厅员工是特殊顾客，可以进行备餐、生成付费请求和请求送餐，其中对于注册工资支付的顾客生成付费请求并发送给工资系统。

菜单管理员是餐厅特定员工，可以管理菜单。

送餐员可以打印送餐说明，记录送餐信息（如送餐时间）以及记录收费（对于没有注册工资支付的顾客，由送餐员收取现金后记录）。

顾客订餐过程如下：

- （1）顾客请求查看菜单；
- （2）系统显示菜单和今日特价；
- （3）顾客选菜；
- （4）系统显示订单和价格；
- （5）顾客确认订单；
- （6）系统显示可送餐时间；
- （7）顾客指定送餐时间、地点和支付方式；
- （8）系统确认接受订单，然后发送Email给顾客以确认订餐，同时发送相关订餐信息通知给餐厅员工。

系统采用面向对象方法开发，使用UML进行建模。系统的顶层用例图和一次订餐的活动图初稿分别如图2-43和图2-44所示。

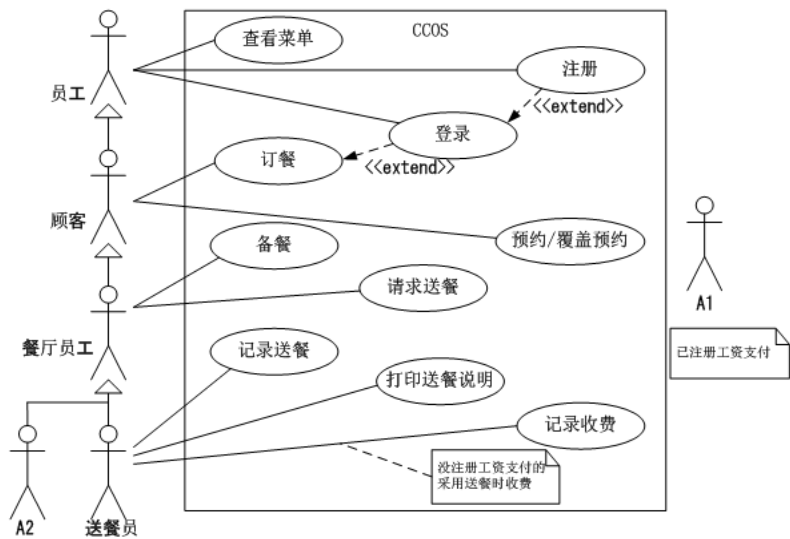


图2-43 CCOS系统顶层用例图

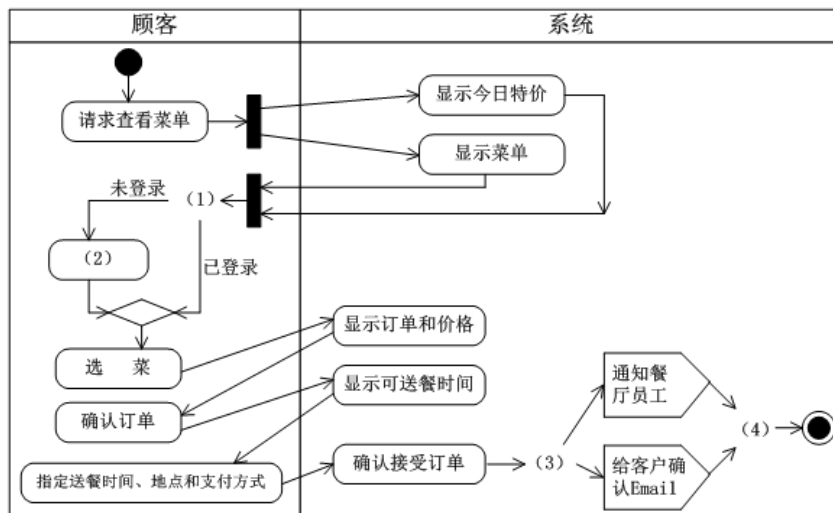


图2-44 一次订餐的活动图

【问题1】

根据试题的描述，给出图2-43中A1和A2所对应的参与者。

【问题2】

根据试题的描述，给出图2-43中缺少的四个用例及其所对应的参与者。

【问题3】

根据试题的描述，给出图2-44中（1）～（4）处对应的活动名称或图形符号。

【问题4】

指出图2-43中员工和顾客之间是什么关系，并解释该关系的内涵。

版权方授权希赛网发布，侵权必究

上一节

本书简介

下一节

本题主要考查类图和顺序图。

**【问题1】**

需要分析此门禁系统的体系结构，根据体系结构的描述来看什么数据放在什么类中最为合适。

题目中提到：系统中的每个电控锁都有一个唯一的编号。锁的状态有两种：“已锁住”和“未锁住”。所以Lock中含有锁编号和锁状态这两个属性。又因为题中有：在主机上可以设置每把锁的安全级别以及用户的开锁权限。只有当用户的开锁权限大于或等于锁的安全级并且锁处于“已锁住”状态时，才能将锁打开。因此，Lock中还有锁的安全级别。

**【问题2】**

首先，（1）、（2）是FingerReader和LockController之间的交互。所以我们看题目中是如何描述他们的交互的。题目中有“指纹采集器将发送一个中断事件给锁控器，锁控器从指纹采集器读取用户的指纹”，所以（1）应填“中断事件”，（2）应填“读取指纹”。（3）是主机与UserInfo的交互，从图2-35中可以看出，UserInfo中存储了用户的指纹信息和开锁权限，所以（3）应是从UserInfo读取用户的指纹信息和开锁权限。（4）空应填“读取锁的安全级别”。（5）是MainFrame向自己发送的一条消息，从题目中的“主机根据数据库中存储的信息来判断用户是否具有开锁权限，若有且锁当前处于已锁住状态，则将锁打开；否则系统报警”可以看出，主机在得到所有信息后要判断用户是否能开锁，所以（5）应填“判断用户是否能开锁”。

**【问题3】**

这是纯理论题，请直接阅读参考答案。

**试题1参考答案**

**【问题1】**

锁的编号、安全级别、锁的当前状态。

**【问题2】**

- （1）中断事件（2）读取用户指纹  
（3）读取用户开锁权限（4）读取锁的安全级别  
（5）判断用户是否有权限开锁，或用户是否可以开锁

**【问题3】**

组装和聚集都表示实例之间的整体/部分关系。组装是聚集的一种形式。

聚集是概念性的，只是区分整体与部分。

组装具有很强的归属关系，而且整体与部分的对象生存周期是一致的。

**试题2分析**

本题考查的是UML类图和顺序图的基本知识。

**【问题1】**

由于一个商品分类中可以有多个商品，而一个商品仅仅对应一个商品分类，所以商品分类与商品之间的关系是1：0..\*，即第（1）空填“0..\*”，第（2）空填“1”。

促销活动与商品之间的关系是这样的：一个促销活动至少得有一种促销商品，否则就无法成为促销活动；而一种商品可以参与多个促销活动，所以促销活动与商品之间的关系有些特别，应是0..\*：1..\*，故第（3）空填“0..\*”，第（4）空填“1..\*”。

再看订单与促销活动之间的关系：由于题目中说明“用户可选择参与某一个促销（Promotion）活动”，同时对于一个促销活动可以有多个客户下订单，所以它们之间的关系为1：

0..\*，所以第（5）空填“1”，第（6）空填“0..\*”。

#### 【问题2】

在顺序图中，消息的执行顺序为：在垂直方向自上至下地执行，其中的虚线表示消息结果的返回。在图2-38中，包含着两个操作，第一个操作是得到某个商品的信息，其流程是：先在商品分类列表中找到相应的分类，再从分类中找到具体的商品，从此商品对应的类中得到相应信息。所以第（7）空应填“getCategories”，第（8）空应填“getCommodities”。第二个操作是创建一次促销活动，并为其指定促销品，所以第（9）空应填“createPromotion”，第（10）空应填“addCommodities”。

#### 【问题3】

这是一个纯理论问题，请直接阅读参考答案。

#### 试题2参考答案

##### 【问题1】

（1）0..n或1..n（2）1（3）0..n

（4）1..n（5）1（6）0..n

##### 【问题2】

（7）getCategories（8）getCommodities

（9）createPromotion（10）addCommodities

##### 【问题3】

关系：聚集（聚合）是关联的特例。（聚集是关联的一种）

不同点：聚集表示部分与整体关系的关联。

#### 试题3分析

本题考查类图和状态图。

##### 【问题1】

根据“每首歌曲的描述信息包括：歌曲的名字、谱写这首歌曲的艺术家及演奏这首歌曲的艺术家”和图2-39中类A与类B之间约束为“编写”、“演奏”，所以类A与类B只能是艺术家和歌曲，又根据图上标示的关联关系（1,0..\*），可以确定类A为艺术家（Artist）；类B为歌曲（Song）。类B与类E之间是聚集关系，根据题中“一条音轨中只包含一首歌曲或为空，一首歌曲可分布在多条音轨上”，可以得到类E为音轨（Track）。

接下来看类E与类F之间存在组成的关系，根据“每张唱片由多条音轨构成”得到，类F为唱片（Album）。再来看类C和类D，它们与类A存在泛化关系，根据“艺术家可能是一名歌手或一支由2名或2名以上的歌手所组成的乐队”可知，类C与类D为歌手和乐队，又因为类C与类D存在聚集关系，根据题中“一名歌手可以不属于任何乐队，也可以属于一个或多个乐队”可知，类C为乐队（Band），类D为歌手（Musician）。

##### 【问题2】

类C为乐队，类D为歌手，题中“一支由2名或2名以上的歌手所组成的乐队。一名歌手可以不属于任何乐队，也可以属于一个或多个乐队”，则第（1）空应填“0..\*” ，第（2）空应填“2..\*” 。类B与类E存在聚集关系，题中“一条音轨中只包含一首歌曲或为空，一首歌曲可分布在多条音轨上”，所以第（3）空应填“0..1” ，第（4）空应填“1..\*” 。

类E与类F存在泛化关系，题中“每张唱片由多条音轨构成”，所以第（5）空应填“1..\*” ，第

(6) 空应填“1”。

特别要说明一下，是“0..\*”还是“1..\*”，要看表述和实际情况，比如第(5)空，一张唱片至少有几条音轨，当然至少有一条，否则就不是唱片了，故是从1开始的。

【问题3】

本问题考查的是类/对象关联中的一种特殊关联：递归关联，它描述的是同一个类的不同实例之间的关系。而类Track的不同实例之间恰好具有这种关系（因此对于任意一条音轨，播放器需要准确地知道，它的下一条音轨和上一条音轨是什么）。所以缺少的那条联系的两端都是类Track，其多重度都为0..1。下限为0，是对应不存在上一条或下一条音轨的情况。

【问题4】

问题4给定了两个状态“关闭”和“播放”，要求找出从“关闭”到“播放”的最短事件序列。这就要求我们能够在状态图上找到连接这两个状态的最短迁移，然后将迁移上的事件记录下来就可以了。

从“关闭”状态到“播放”状态可以选择经过迁移“连接电脑”到达“联机”状态，再经过迁移“断开连接”到达状态“打开”，再从“打开”状态的初始状态“歌曲待选”，经过迁移“选择歌曲”到达“播放状态”。这样经过的事件序列为：连接电脑电量饱和/完成复制断开连接选择歌曲。显然这样的事件序列远比“关闭”经过“按任意键”直接到达“打开”状态要长得多。所以从“关闭”到“播放”的最短事件序列是：按任意键，选择歌曲。

试题3参考答案

【问题1】

A : Artist B : Song C : Band

D : Musician E : Track F : Album

【问题2】

(1) 0..\* (2) 2..\* (3) 0..1

(4) 1..\* (5) 1..\* (6) 1

【问题3】

类	多重度
Track 或 E	0..1
Track 或 E	0..1

【问题4】

按任意键，选择歌曲。

试题4分析

题目以希赛公司在线会议审稿系统为例，考查考生对UML用例图与活动图的掌握情况。

【问题1】

题目已经给出了4类参与者：用户、作者、审稿人、委员会主席，关键在于弄清楚各个参与者之间的关系，这些关系是通过题目中的系统功能描述来获得的。

(1) “用户在初始使用系统时，必须在系统中注册（register）成为作者或审稿人”，从此处可以得知系统中的用户分成了两类：作者和审稿人。

(2) “会议委员会主席是一个特殊审稿人”。

从上面两个条件得知：A1对应用户，A2对应作者，A3对应审稿人，A4对应会议委员会主席。同时由于UML图中不允许出现中文，且题目明确要求用英文名称给出A1~A4所对应的参与者，所以

A1~A4处应分别填写User、Author、Reviewer和PCChair。

【问题2】

由“会议委员会主席是一个特殊审稿人，可以浏览提交的稿件，给审稿人分配稿件，罗列录用和（或）拒绝的稿件，以及关闭审稿过程”结合“用例名称列表”可以得知：会议委员会主席能操作的功能有浏览提交的稿件、分配稿件给审稿人、罗列录用或/和拒绝的稿件、关闭审稿过程。而从“其中关闭审稿过程须包括罗列录用和（或）拒绝的稿件”可以看出，用例“关闭审稿过程”与“罗列录用或/和拒绝的稿件”之间有包含关系。从这个关系可以得知，U1对应的用例为：罗列录用或/和拒绝的稿件。同时（2）对应的关系为包含关系，即U1应填“list accepted/rejected papers”，（2）应填“<<include>>”。这样，剩余的两项功能“浏览提交的稿件”和“分配稿件给审稿人”对应的为U2与U3，所以U2和U3分别应填“browse submitted papers”和“assign paper to reviewer”。

【问题3】

该小题考查考生对包含与扩展关系的理解。在对问题2的分析中，已经得出（2）填“<<include>>”。现在来看（1），该空是填“登录”与“提交稿件”之间的关系，在提交稿件时，若用户已经登录，则可直接提交；但如果用户没有登录，则需要先登录再提交，所以它们之间的关系应是扩展关系，即（1）应填“<<extend>>”。

【问题4】

该活动图所描述的是作者提交稿件的过程，对此过程题目有详细的描述：“作者登录（login）后提交稿件和浏览稿件审阅结果。提交稿件必须在规定提交时间范围内，其过程为先输入标题和摘要、选择稿件所属主题类型、选择稿件所在位置（存储位置）。上述几步若未完成，则重复；若完成，则上传稿件至数据库中，系统发送通知。”，所以Action1~Action4分别对应：输入标题和摘要、选择稿件所属主题类型、选择稿件所在位置、上传稿件。所以Action1~Action4分别填：enter title and abstract、select subject group、select paper location和upload paper。

**试题4参考答案**

【问题1】

A1 : User A2 : Author A3 : Reviewer A4 : PCChair

【问题2】

U1 : list accepted/rejected papers U2 : browse submitted papers

U3 : assign paper to reviewer

注：U2和U3的答案可互换

【问题3】

（1）<<extend>> （2）<<include>>

【问题4】

Action1 : enter title and abstract

Action2 : select subject group

Action3 : select paper location

Action4 : upload paper

**试题5分析**

本题考查面向对象系统开发时，采用UML模型进行建模的方法。

【问题1】

识别参与者时，考查和系统交互的人员和外部系统。在本题中，与系统交互的人员包括员工、注册到系统的员工（顾客）、餐厅员工、菜单管理员、送餐员以及工资系统。



由“菜单管理员是餐厅特定员工”以及图2-43中A2和图中餐厅员工之间的“是一种”关系可知，A2为菜单管理员；图2-43中还缺少描述中与工资系统的交互，由“.....并发送给工资系统”可知，A1为工资系统。

【问题2】

在本题中，由“任何员工都可以查看菜单和今日特价”可知，图2-43中缺少用例查看今日特价，对应参与者是员工；由“系统的顾客是.....，注册工资支付、.....”可知，图中缺少用例注册工资支付，对应参与者是顾客和工资系统；由“餐厅员工是.....，可以进行备餐、生成付费请求.....发送给工资系统”可知，图2-43中缺少用例“生成付费请求”，对应的参与者是餐厅员工和工资系统；由“菜单管理员是餐厅特定员工，可以管理菜单”可知，图2-43中缺少用例管理菜单，对应的参与者是菜单管理员。

需要注意的是，在注册工资支付所对应的参与者中，虽然没有明确说明要和工资系统交互，但是由“对于注册工资支付的顾客生成付费请求并发送给工资系统”可知，工资支付是由工资系统控制，所以注册也需要和工资系统交互。

【问题3】

在顾客订餐过程的描述中，在“顾客选菜”之前，图中缺少符号和活动。由说明中顾客“可以订餐（如果未登录，需先登录）”可以判断，在系统“显示菜单和今日特价”之后“顾客选菜”之前，需要判断（判定符号 ）当前用户身份是否为顾客，如果不是，需先登录；由“.....发送E-mail给顾客以确认订餐，同时发送相关订餐信息通知给餐于员工”可知，发送E-mail和通知餐厅员工为并行活动，需要在前后有同步条（或纵向 ）。

【问题4】

参与者之间的关系表示子类型“是一种”父类型，即泛化关系。其中父类型通常是一个抽象泛化的参与者，可以完成子类型可完成的共同行为，每个具体的子类型继承它，可以完成父类型参与者同样的任务，并可以补充额外的角色功能。

试题5参考答案

【问题1】

A1：工资系统A2：菜单管理员

【问题2】

用例名	参与者
查看当日特价	员工
注册工资支付	顾客和工资系统（或顾客和 A1）
生成付费请求	餐厅员工和工资系统（或餐厅员工和 A1）
管理菜单	菜单管理员（或 A2）

【问题3】

- (1)  (2) 登录
- (3)  或  (4)  或 

【问题4】

泛化关系（一般/特殊关系、继承关系）。泛化关系描述了一个参与者可以完成另一个参与者同



样的任务，并可补充额外的角色功能。

版权方授权希赛网发布，侵权必究

[上一节](#)    [本书简介](#)    [下一节](#)

第 2 章：UML 建模技术

作者：希赛教育软考学院    来源：希赛网    2014年05月05日

## 试题解答方法

UML 建模技术类题目要求考生认真阅读题目说明中对现实问题的描述，使用 UML 建模的原则，从中确定用例图、类图、顺序图、状态图和活动图的各种元素，以及图中的各种元素之间的关系。一般情况下，题目会给出未完成的图形，需要考生根据描述给出参与者、用例、类、状态、活动和符号、类之间的重复度，以及参与者之间的关系及内涵。

### 1. 用例图

用例图是用例建模的一个重要产物，它以图形化的方式将系统描述成用例、参与者及其之间的关系。用例图在高层交流了系统必须处理的业务事件的范围，是描述系统与其他外部系统以及用户之间交互的图形。发起或者触发用例的外部用户称为参与者。为了完成某些业务任务，参与者发起系统活动，即用例。

简单地理解，用例就是系统的功能分解，因此，在填写所缺少的用例时，需要从试题描述中去找系统的功能，将系统功能与用例图中的用例进行一一对应。在构建用例图时，常用的方式是先识别参与者，然后确定用例以及用例之间的关系。考查用例时，通过判断哪一个特定参与者发起或者触发了与系统的哪些交互，来识别用例并建立和参与者之间的关联。

考查用例之间的关系时，<<include>> 定义了用例之间的包含关系，用于一个用例包含另一个用例的行为的建模；如果可以从一个用例的执行中，在需要时转向执行另一个用例，执行完返回之前的用例继续执行，用例即存在<<extend>> 关系。另外，用例之间也存在泛化关系，这跟类之间的继承关系是类似的。

### 2. 类图

类图是描述类之间关系的一种图形，在考试中，对类图的考查，主要集中在填写缺少的类、填写类的属性和方法、填写类之间的重复度。

简单地理解，可以从试题描述中去找名词，将所有名词列出来，这些名词可能是类，也可能是类的属性。然后，再根据题目描述和上下文关系，来确定某个名词究竟是类还是类的属性。类的方法其实就是类的功能，即这个类应该要做的事情。这也只需要到题目描述中去找功能性的描述。

类之间的重复度是几乎每次考试都要考的内容，它是指类之间的对应关系，即 1 个 A 类的实例对应 N 个 B 类的实例。根据 N 的取值就可以得到 0、1、0..\*、1..\*、0..1、1..M 等关系，其中 M 是某个确定的整数；反过来，1 个 B 类的实例对应 X 个 A 类的实例。根据 X 的取值也可以得到 0、1、0..\*、1..\*、0..1、1..Y 等关系，其中 Y 是某个确定的整数。

### 3. 顺序图

对顺序图的考查形式主要是给出系统描述，要求考生根据描述填充消息名称。解答这类试题，需要以试题说明为主线来分析操作步骤然后按时间顺序与试题给出的图进行匹配，对号入座。构造顺序图时遵循如下指导原则：

确定顺序图的范围，描述这个用例场景或一个步骤；

绘制参与者和接口类，如果范围包括这些内容的话；

沿左边列出用例步骤；

对控制器类及必须在顺序中协作的每个实体类，基于它拥有的属性或已经分配给它的行为绘制框；

为持续类和系统类绘制框；

绘制所需消息，并把每条消息指到将实现响应消息的责任的类上；

添加活动条指示每个对象实例的生命期；

为清晰起见，添加所需的返回消息；

如果需要，为循环、可选步骤和替代步骤等添加框架。

#### 4. 状态图

状态图用于描述对象在某个时刻所处的状态，以及根据某些事件进行状态的转化。在考试中，通常会要求考生填写所缺的状态，这只需要根据试题的描述去查找相关的名词就可以了。

绘制状态图的理想步骤是：寻找主要的状态，确定状态之间的转换，细化状态内的活动与转换，用复合状态来展开细节。

寻找主要状态。在绘制状态机图时，最重要的一个活动就是寻找出主要的状态。

确定状态间转换。在确定了主要的状态之后，就要分析状态之间的转换，在此基础上就可以绘制出相应的状态图。

细化状态内的活动与转换。当明确了各个状态之间的转换（外部转换）之后，可以根据需要添加内部转换、进入和退出转换，以及相关的活动等。

复合状态的使用。如果可以将某些状态归结为1个状态，就可以采用一个复合状态来表示。

#### 5. 活动图

活动图用于建模系统的过程步骤或活动。构造活动图通常先为用例添加开始和结束点，为用例的主要步骤添加一个活动，从每个活动到其他活动、决策点和终点添加转换，在并行活动的地方添加同步条。绘制活动图的主要思路如下：

首先，决定是否采用泳道，主要根据活动图中是否要体现出活动的不同实施者。

然后，尽量使用分支、分岔和汇合等基本的建模元素来描述活动控制流程。

如果需要，加入对象流以及对象的状态变化。

如果需要，利用一些高级的建模元素（如辅助活动图、汇合描述、发送信号与接收信号、引脚、扩展区）来表示更多的信息。

活动图的建模关键是表示出控制流，其它的建模元素都是围绕这一宗旨所进行的补充。

版权方授权希赛网发布，侵权必究

[上一节](#)      [本书简介](#)      [下一节](#)

### 考情分析

数据库设计的任务是针对一个给定的应用环境，在给定的（或选择的）硬件环境和操作系统及