

## 第1章 复习思考题

### 一、选择题

1. 封装是指把对象的（ A ）结合在一起，组成独立的对象。  
A. 属性和服务      B. 信息流      C. 消息和事件      D. 数据的集合
2. （ A ）模型的缺点是缺乏灵活性，特别是无法解决软件需求不明确或不准确的问题。  
A. 瀑布模型      B. 原型模型      C. 增量模型      D. 螺旋模型
3. （ A ）不是面向对象程序的特征。  
A. 消息      B. 继承      C. 封装      D. 多态性
4. 在面向对象的方法学中，对象可看成是属性及对于这些属性的专用服务的封装体。封装是一种（ D ）技术。  
A. 组装      B. 产品化      C. 固化      D. 信息隐藏
5. 封装的目的是使对象的（ A ）分离。  
A. 定义和实现      B. 设计和测试  
C. 设计和实现      D. 分析和定义

### 二、填空题

1. 软件开发模型主要有瀑布模型、快速原型模型、增量模型、螺旋模型、喷泉模型、智能模型、V模型和RUP模型。
2. RUP核心过程工作流有业务建模、需求、分析与设计、实现、测试和部署。
3. RUP核心支持工作流有配置与变更管理、项目管理和环境。
4. 软件工程知识体系分为软件工程教育基础和软件工程实践两大类。
5. 软件工程教育需求包括工程经济基础学、计算基础、数学基础和工程基础知识域。
6. 软件工程实践包括软件需求、软件设计、软件构造、软件测试、软件维护、软件配置管理、软件工程管理、软件工程模型与方法、软件工程过程、软件质量和软件工程职业实践。
7. 软件工程知识体系的辅助学科是计算机工程、计算机科学、管理学、数学、项目管理、质量管理和系统工程。
8. 软件工程的框架由软件工程目标、软件工程活动和软件工程原则内容构成。

### 三、名词解释

【答】软件开发模型：软件开发模型是指软件开发中的所有过程、活动和人物的结构框架，它能清晰、明确地表达软件开发的全过程。对于不同的软件系统，可以采用不同的软件开发模型和方法、不同的软件工具和软件工程环境、不同的管理方法和手段来实现软件项目的跟踪和把控。【课本 P9】

封装：封装就是把对象的属性和服务结合成一个独立的相同单位，并尽可能隐蔽对象的内部细节，也叫信息隐蔽。【课本 P19】

信息隐蔽：信息隐蔽，即尽可能隐蔽对象的内部细节，对外形成一个边界（或者说形成一道屏障），只保留优先的对外接口，使之与外部发生联系。【课本 P19】

继承：继承是指一个对象直接使用另一个对象的属性和方法，是表示相似性质的机制。继承是指类之间由继承关系，子类有条件地继承父类的特征。【课本 P20】

对象：对象是问题域中某个实体的抽象，反映该事物在系统中需要保存的信息和发挥的作用。对象=属性+操作。【课本 P19】

类：类是具有相似结构、行为和关系的一组对象的描述符，类包括属性和操作。【课本 P18】

消息：消息是面向对象方法中对象之间相互联系的方法，是对传送消息的对象之间所进行的通信的规约，其中含有将要发生的活动的期望。【课本 P21】

#### 四、综合题

##### 1. 什么是软件危机？【课本 P3】

【答】软件危机是计算机软件在它的开发和维护过程中所遇到的一系列严重问题，主要包括：(1)如何开发软件，以满足不断增长，日趋复杂的需求；(2)如何维护数量不断膨胀的软件产品。软件危机主要的表现：对软件①对软件开发成本和进度的估计常常不准确②用户对“已完成”系统不满意的现象经常发生③软件产品的质量往往靠不住，可维护程度低④软件通常没有适当的文档资料⑤软件的成本不断提高⑥软件开发生产率的提高赶不上硬件的发展和人们需求的增长。

##### 2. 什么是软件工程？软件工程的原理（基本原则）有哪些？【课本 P4，P7】

【答】软件工程是采用工程的概念、原理、技术和方法老计划、开发与维护软件，把经过时间考验而证明正确的管理技术和当前能够得到的最好的技术方法结合起来；以较经济的手段获得能在实际机器上运行的可靠软件的一系列方法。简而言之：软件工程=工程方法+管理技术+技术方法。

原理：七条原理相互独立，是缺一不可的最小集合：(1)用分阶段的生命周期计划严格管理。(2)坚持进行阶段评审。(3)实行严格的产品控制。(4)采用现代程序设计技术。(5)结果应能清楚地审查。(6)开发小组的人员应该少而精。(7)承认不断改进软件工程实践的必要性。【课本 P7】

##### 3. 软件生命周期模型有哪些？各有哪些优缺点？【课本 P15】

【答】软件生命周期模型由：瀑布模型、快速原型模型、增量模型、螺旋模型、喷泉模型、V 模型、智能模型。

(1) 瀑布模型：优点：①严格规定应提交的文档，为项目提供检查点；②利于大型软件项目中人员的组织与管理；③利于开发方法工具的研究和使用；④提高大型项目直的开发效率和质量。 缺点：①文档驱动，可能不能完全满足用户需求；②呈线性，成果未经测试时，用户看不到结果，增加了风险；③前阶段未发现的错误传到后阶段甚至扩散，可导致项目失败；④需求阶段，完全确定需求比较困难。

(2) 快速原型模型：优点：①需要尽可能尽快建造软件原型，可能会限制开发人员创新；②更好和客户沟通，提高客户对软件的满意度；③减少技术，应用风险，缩小成本，提高产品质量。 缺点：①需要尽可能尽快建造软件原型，可能会限制开发人员创新②所选技术工具不一定是主流，效率低；③原型快速简历的内部结构及连续修改可能导致产品设计差；④客户确定真正需求，原型可能被弃。

(3) 增量模型：优点①反馈及时，较好适应变化②客户看到不断变化的软件，降低开发风险；③鼓舞团队的士气。 缺点：①容易退化成边做边改，失去对软件过程的整体控制，效率低；②不破坏现有架构；③产品、架构不是开放的，维护难度加大。

(4) 螺旋模型：优点：①强调可选方案及约束条件，支持软件重用；②有助于提升软件质量。 缺点：①要求客户接收其风险分析，并作出反应不容易；②风险分析成本>项目利润，项目风险分析无意义；③要善于识别风险，且准确，否则将带来更大风险。

(5) 喷泉模型：优点：①各阶段无明显界限，开发人员可同步开发；②提高开发效率，节省开发时间；缺点：①各阶段重叠，需大量开发人员；②不利于项目管理；③需更严格管理文档及文档变更。

##### 4. 软件开发过程与模型有哪些？【课本 P15-16】

【答】常见的软件过程模型有：瀑布模型、快速原型模型、增量模型、喷泉模型等。瀑布模型：过程划分为计划、需求、设计、编码、测试和维护阶段；各阶段自上而下，相互衔接固定次序，如同瀑布流水；各阶段评审确认通过，前阶段输出时后阶段的输入；文档驱动。快速原型模型：快速建造初步、非完整软件原型，实现客户与系统交互；客户对原型评价，细化需求；逐步调整原型满足要求，原型内部结构不重要；原型驱动。增量模型：分阶段实现，软件增量设计，实现，计程，测试；整个产品拆成多个构建，分次逐个构建，交付可运行产品；功能驱动。螺旋模型：分成计划、评估、设计、实时、用户反馈四个象限，沿着螺旋线进行若干次迭代；关注风险，风险分析之后决策项目是否继续；风险驱动。以需求为动力，以对象为驱动，支持面向对象开发来开发过程自下而上，各阶段之间时相互迭代和无间隙；对象驱动。

##### 5. 典型的面向对象开发方法有哪些？【课本 P23-31】

【答】有 Coad Yourdon 方法、Booch 方法、OMT 方法、OOSE 方法、Rational 软件统一开发过程。

(1) Coad Yourdon 方法的开发步骤也是由面向对象分析、面向对象设计和面向对象实现所组成。这

种方法严格区分了面向对象分析和面向对象设计。OOA 完成系统分析，包括以下五个步骤：① 确定类及对象。从应用领域开始识别类及对象，形成整个应用的基础，然后，据此分析系统的责任。② 标识结构。该阶段分为两个步骤。第一，识别一般—特殊结构，该结构捕获了识别出的类的层次结构；第二，识别整体—部分结构，该结构用来表示一个对象如何成为另一个对象的一部分及多个对象如何组装成更大的对象。③ 定义主题。主题由一组类及对象组成，用于将类及对象模型划分为更大的单位，便于理解。④ 定义属性。其中包括定义类的实例（对象）之间的连接。⑤ 定义服务。其中包括定义对象之间的消息连接。OOD 负责系统设计，包括以下四个步骤：①设计问题域部分（PDC）。细化分析结果，面向对象分析的结果直接放入该部分。②设计人机交互部分（HIC）。设计用户界面的活动包括对用户分类，描述人机交互的脚本，设计命令层次结构，设计详细的交互，生成用户界面的原型，定义 HIC 类。③设计任务管理部分（TMC）。确定系统资源的分配，这部分的的活动包括识别任务（进程），任务所提供的服务，任务的优先级，进程是事件驱动还是时钟驱动、任务与其他进程及外界如何通信。④设计数据管理部分（DMC）。确定持久对象的存储，这一部分依赖于存储技术。数据管理是采用文件系统，关系数据库管理系统，还是面向对象数据库管理系统。

（2）Booch 方法包含的概念非常丰富，如类、对象、继承、元类、消息、域、操作、机制、模块、子系统和进程等。Booch 在其 OOAda 中提出了面向对象的四个模型：逻辑模型、物理模型、静态模型和动态模型，其模型主要包括：逻辑静态视图（类图和对象图）、逻辑动态视图（状态变迁图和交互图）、物理静态视图（模块图、进程图）和物理动态视图。Booch 方法的开发包括以下步骤：①在给定的抽象层次上识别类和对象。②识别这些对象和类的语义。③识别这些类和对象之间的关系。④实现类和对象。要考虑怎样定义属性和提供服务，这涉及到选择结构和算法。

（3）OMT 方法包含系统分析、系统设计、对象设计和实现四个步骤，它定义了三种模型。这些模型贯穿于每个步骤，在每个步骤中被不断地精化和扩充。① OMT 方法的系统分析。用 OMT 方法进行系统分析需要建立对象模型、动态模型和功能模型。对象模型描述系统中的对象、对象之间的关系，标识类中的属性和操作，反映系统的静态结构。动态模型表述系统与时间的变化有关的性质。功能模型由多张数据流程图组成，用来描述系统中所有的计算。②OMT 方法的系统设计。用 OMT 方法进行系统设计的主要内容有：把系统分解成子系统；识别问题中固有的并发性；把子系统分配给处理器和任务；选择数据存储管理的方法；处理访问全局资源；选择软件中的控制实现；处理边界条件及设置综合优先权。③OMT 方法的对象设计。用 OMT 方法进行对象设计的开发步骤是组合三种模型来获得类上的操作；实现操作的算法设计；优化数据的访问路径；实现外部交互式的控制；调整类结构，提高继承性；设计关联；确定对象表示；把类和关联封装成模块。④ OMT 方法的实现。用 OMT 方法进行实现，可以使用面向对象语言、非面向对象语言等程序设计语言，也可以使用数据库管理系统实现。

（4）OOSE（Object-Oriented Software Engineering）方法是 Ivar Jacobson 在 1992 年提出的一种使用事例驱动的面向对象开发方法。其最大特点是面向用例（use case），并在用例的描述中引入了外部角色的概念。OOSE 过程分为分析、构造和测试 3 个阶段。Jacobson 将用例模型与其他五种系统模型关联：（1）领域对象模型。用例模型根据领域来表示。②分析模型。用例模型通过分析来构造。③设计模型。用例模型通过设计来具体化。④实现模型。该模型依据具体化的设计来实现用例模型。⑤测试模型。用来测试具体化的用例模型。OOSE 方法对以用例作为一种途径来驱动需求获取、分析和高层设计提供了很好的支持。OOSE 是一种在 OMT 基础上用于对功能模型进行补充以指导系统开发活动的系统方法。

（6）RUP 用二维坐标来描述。横轴通过时间来组织，是过程展开的生命周期特征，体现开发过程的动态结构。纵轴以内容来组织，体现开发过程的静态结构。RUP 中有 9 个核心 workflow，分为 6 个核心过程 workflow（Core Process Workflows）和 3 个核心支持 workflow（Core Supporting Workflows）。

#### 6. 简述 RUP 软件开发过程包括哪些开发阶段？【课本 P27-28】

【答】（1）业务建模：workflow 描述了如何为新的目标组织开发一个构想，并基于这个构想在商业用例模型和商业对象模型中定义组织的过程、角色和责任。（2）需求：workflow 的目标是描述系统应该做什么，并使开发人员和用户就这一描述达成共识。为达到该目标，要对需要的功能和约束进行提取、组织、文档化；最重要的时理解系统所解决问题的定义和范围。（3）分析和设计：workflow 将需求转化成未来系统的设计，为系统开发一个健壮的结构并调整设计使其与显示环境相匹配，优化器性能。分析设计的结果是一个

设计模型和一个可选的分析模型。(4) 实现：工作流的目的包括以层次化的子系统形式定义代码的组织结构；以组件的形式（源文件、二进制文件、可执行文件）实现类和对象；将开发出的组件作为单元进行测试以及集成由单个开发者（小组）所产生的结果，使其成为可执行的系统。(5) 测试：工作流要验证对象间的交互作用，验证软件中所有组件的正确计程，检验所有的需求已被正确实现，识别并确认缺陷在软件部署之前被提出并处理。(6) 部署：工作流的目的时成功地生成版本并将软件分发给最终用户。(7) 配置和变更管理：工作流描绘了如何在多个成员组成的项目中控制大量地产物，并提供了准则来管理演化系统中的多个变体，跟踪软件创建过程中的版本。(8) 项目管理：平衡各种可能产生冲突的目标和管理风险，克服各种约束并成功交付使用户满意的产品。(9) 环境：工作流的目的是向软件开发组织提供软件开发环境，包括过程和工具。

#### 7. 面向对象方法有哪些特点？【课本 37】

【答】面向对象方法具有封装、抽象、继承和多态的先进机制，使得面向对象方法成为当前软件工程中 最有效、最流行的方法。面向对象开发方法的主要特点如下：(1) 与人类习惯的思维方法一致。(2) 稳定性好。(3) 可复用性好。(4) 比较容易开发大型产品。(5) 易于维护。

目前，运用面向对象方法进行系统开发的项目很多。面向对象方法具有如下的优越性：(1) 面向对象的方法更接近于人类的自然思维。(2) 系统分析、系统设计及实现从同样的角度看待问题，甚至采用同样的表示方法来描述问题，它们之间的连接是自然的无缝连接。(3) 面向对象的方法将对象的属性及服务视为一个整体。这更符合客观世界的规律，从而使其理解与实现起来更加容易。(4) 采用继承的方法。继承一方面符合客观世界的规律，一方面加强代码重用的可能性，可以提高软件的开发效率。(5) 信息隐蔽原理使系统在变化的环境中有良好的适应性，从而使整个系统更加稳定和易于维护。总之，面向对象的方法可以采用正向工程和逆向工程进行映射，可以提高软件开发效率、可靠性及可维护性。

## 第 2 章 复习思考题

### 一、选择题

1. UML 具有扩展性，常见的扩展机制有（ BCD ）。  
A. 修饰                  B. 版类                  C. 加标签值                  D. 约束
2. UML 语言支持的建模方式有（ ABD ）。  
A. 静态建模              B. 动态建模              C. 模块化建模              D. 功能建模
3. 下列各种图可用于动态建模的有（ ACD ）。  
A. 状态机图              B. 类图                  C. 顺序图                  D. 活动图
4. 下列属于状态的组成部分的有（ AB ）。  
A. 名称                  B. 活动                  C. 条件                  D. 事件
5. 属性的可见性有（ ABD ）。  
A. 公有的                  B. 私有的                  C. 私有保护的                  D. 保护的
6. UML 体系包括三个部分：UML 基本构造块、（ A ）和 UML 公共机制。  
A. UML 规则              B. UML 命名              C. UML 模型              D. UML 约束
7. UML 图不包括（ D ）。  
A. 用例图                  B. 类图                  C. 状态图                  D. 流程图
8. 在 UML 的最上一层，视图被划分为（ ACD ）视图域。  
A. 模型管理              B. 扩展机制              C. 动态行为                  D. 结构分类
9. UML2.0 在 1.0 的基础上，对如下（ ABCD ）的建模能力进行了增强。  
A. 活动                  B. 交互                  C. 复杂结构                  D. 状态机

10. UML 的（ B ）是由建模者设计的新的建模元素，但是这个模型元素的设计要建立在 UML 已定义的模型元素基础上。

- A. 标记值      B. 构造型      C. 注释      D. 约束
11. UML 中的事物包括结构事物、分组事物、注释事物和 ( D )。
- A. 实体事物      B. 边界事物      C. 控制事物      D. 动作事物

12. UML 中的四种关系是依赖、泛化、关联和 ( C )。

- A. 继承      B. 合作      C. 实现      D. 抽象

13. 用例用来描述系统在事件做出响应时所采取的行动。用例之间具有相关性。在一个“订单输入子系统”中，创建新的订单和更新订单都需要检查用户账号是否正确。那么，用例“创建新订单”、“更新订单”与用例“检查用户账号”之间是 ( A ) 关系。

- A. 包含      B. 拓展      C. 分离      D. 聚集

14. UML 的全称是 ( B )。

- A. Unify Modeling language      B. Unified Modeling language  
C. Unified Modem language      D. Unified Making language

15. UML 的最终产物是最后提交软件系统和 ( D )。

- A. 用户手册      B. 类图      C. 动态图      D. 相应的软件文档资料

## 二、判断题

- ( ✓ ) 1. UML 是一种建模语言，是一种标准的表示，是一种方法。
- ( × ) 2. UML 支持面向对象的主要概念，并与具体的开发过程有关。
- ( ✓ ) 3. 1997 年 11 月，UML1.1 规范被 OMG 组织确认，成为正式的规范
- ( × ) 4. UML 既是一门建模语言，又是一门编程语言。
- ( × ) 5. UML2.0 彻底推翻了 UML1.x 中的核心概念，发展成为一门与之前截然不同的建模语言。
- ( ✓ ) 6. UML 的公共机制有规约、修饰、通用划分和扩展机制。
- ( ✓ ) 7. UML 构造块由事物、关系和图组成。
- ( × ) 8. 用例之间有扩展、使用、组合等几种关系。
- ( ✓ ) 9. 常见的结构性事物有类、接口、用例、协作、构件和结点等。
- ( ✓ ) 10. 协作定义了一个交互，它是为实现某个目标而共同工作、相互配合的多个元素之间的交互动作。

## 三、填空题

1. UML 是面向对象技术领域内占主导地位的标准建模语言，它统一了过去相互独立的数十种面向对象的建模语言共同存在的局面，形成了一个统一的、公共的、具有广泛适应性的建模语言。
2. UML2.0 将图分为结构图和行为图两类。
3. UML2.0 包括 13 种图，分别是类图、对象图、构件图、部署图、组合结构图、包图、用例图、顺序图、通信图、交互概览图、定时图、状态机图和活动图。
4. UML2.0 结构图包括类图、对象图、构件图、部署图、组合结构图和包图。
5. UML2.0 行为图包括用例图、交互图、状态机图和活动图。
6. UML 中主要包括四种关系，分别是关联、泛化、依赖和实现。
7. UML 的通用机制分别是规约、修饰和通用划分。
8. 常用的 UML 扩展机制分别是衍型、标记值和约束。
9. Rational 统一过程的五种视图结构，分别是用例视图、设计视图、交互视图、实现视图和部署视图。
10. UML 的事物包括结构事物、行为事物、分组事物和注释事物。

## 四、综合题

1. 为什么要使用 UML? 【课本 P46, 9】

【答】UML 是一种标准的图形化建模语言，它是面向对象分析与设计的一种标准表示，它统一了

Booch、OMT 和 OOSE 等方法中的基本概念，易于使用，表达能力强，便于进行可视化建模，它与具体的实现无关，可用于任何语言平台和工具平台，可用于任何软件开发的过程，简单，可扩展，可与最好的软件工程实践经验集成，具有广阔的适用性和实用性。使用 UML 进行系统分析和设计，可加速开发进程，提高代码质量，支持动态的业务需求。UML 能促进软件复用，方便集成已有的系统，并能有效处理开发中的各种风险。随着 UML 工作进一步展开，必将有助于实现软件自动化。UML 作为一种先进实用的标准建模语言，其某些概念尚待实践来验证，UML 也必然存在一个进化的过程。

2. UML 有哪些特点？【课本 P48】

【答】①统一标准②提出了许多新的概念③独立于开发过程④可视化，表达能力强⑤独立于程序设计语言⑥应用领域广泛。

3. UML2.0 包括哪些图？【课本 P61】

【答】UML2.0 包括 14 种图，分别是类图、对象图、构件图、部署图、组合结构图、包图、外廊图、用例图、顺序图、通信图、交互概览图、定时图、状态机图和活动图。

4. 简述视图和图的关系。【课本 P68-69】

【答】视图是由多个图构成，它不是一组图表，而是在某个抽象层次上对系统的抽象表示。某些图可能同时从属于多个视图，体现出视图之间的重叠。如果要为系统建立一个完整的模型图，需要定义多个反映系统不同方面的视图

5. 简述类图与定时图之间的关系。

【答】类图展现了一组类、接口、协作和他们之间的关系。在面向对象系统的建模中，所建立的最常用的图就是类图。类图用于对系统静态设计视图建模。定时图描述某一个时间段内的对象行为，特别展现对象响应外部事件而发生的状态变化，一般用于描述嵌入式系统。

6. UML 建模由哪些视图组成？每个视图包括哪些模型图？【课本 P68-69】

【答】UML 建模由用例视图、设计视图、交互视图、实现视图和部署视图组成。

用例视图包括用例图、交互图、状态机图和活动图。设计视图包括类图、对象图、交互图、状态机图和活动图。交互视图包括类图、对象图、交互图、状态机图和活动图。实现视图包括构件图、交互图、状态机图和活动图。部署视图包括部署图、交互图、状态机图和活动图。

## 第 3 章 复习思考题

### 一、选择题

- 用例之间的关系主要有（BCD）。  
A. 聚合                  B. 继承                  C. 扩展                  D. 包含
- 基于用例图的需求捕获的第一步就是确定系统的参与者，在寻找系统参与者时，可以根据以下（ACD）等问题来确定。  
A. 系统同环境如何进行交互                  B. 由谁安装系统  
C. 系统为哪些对象提供信息、服务                  D. 系统的使用者是谁
- 如果用例 B 是用例 A 的某项子功能，并且建模者确切地知道在 A 所对应的动作序列中何时将调用 B，则称（A）。  
A. 用例 A 扩展用例 B                  B. 用例 A 继承用例 B  
C. 用例 A 包括用例 B                  D. 用例 A 实现用例 B
- 如果用例 A 与用例 B 相似，但 A 的动作序列是通过改写 B 的部分或者扩展 B 的动作而获得的，则称（B）。  
A. 用例 A 实现用例 B                  B. 用例 A 继承用例 B  
C. 用例 A 扩展用例 B                  D. 用例 A 包括用例 B
- 如果用例 A 与用例 B 相似，但 A 的功能较 B 多，A 的动作序列是通过在 B 的动作序列中的某些

执行点上插入附加的动作序列而构成的，则称（ C ）。

- A. 用例 A 扩展用例 B
  - B. 用例 A 包含用例 B
  - C. 用例 A 继承用例 B
  - D. 用例 A 实现用例 B
6. 在 UML 中，（ A ）表示使用软件系统的功能，与软件系统交换信息的外部实体。
- A. 参与者
  - B. 类
  - C. 用例
  - D. 用例图
7. 用例之间的关系主要有（ ABC ）。
- A. 包含
  - B. 继承
  - C. 扩展
  - D. 聚合
8. 消息的类型有（ ABC ）。
- A. 同步
  - B. 异步
  - C. 简单
  - D. 复杂
9. UML 图不包括（ D ）。
- A. 用例图
  - B. 类图
  - C. 状态图
  - D. 流程图
- 10.（ ACD ）是构成用例图的基本元素。
- A. 参与者
  - B. 泳道
  - C. 系统边界
  - D. 用例
11. 下面不是用例间主要关系的是（ C ）。
- A. 扩展
  - B. 包含
  - C. 依赖
  - D. 泛化
12. 下列关于状态机图的说法中，正确的是（ C ）。
- A. 状态机图是 UML 中对系统的静态方面进行建模的五种图之一
  - B. 状态机图是活动图的一个特例，状态机图中的多数状态是活动状态
  - C. 活动图和状态机图是对一个对象的生命周期进行建模，描述对象随时间变化的行为
  - D. 状态机图主要用于多个对象参与的活动过程建模，而活动图更强调对单个对象建模
13. 对反应型对象建模一般使用（ A ）。
- A. 状态机图
  - B. 顺序图
  - C. 活动图
  - D. 类图
14. 顺序图由类角色、生命线、激活期和（ B ）组成。
- A. 关系
  - B. 消息
  - C. 用例
  - D. 实体
15. 在 UML 的状态机图中，转换通常由以下哪些部分构成？（ ABCD ）
- A. 动作
  - B. 触发事件
  - C. 源状态
  - D. 目标状态
16. UML 的类图包含哪些抽象的层次？（ ABC ）
- A. 概念层
  - B. 说明层
  - C. 实现层
  - D. 业务层

## 二、判断题

- （×）1. UML 建模语言是由事物、关系和公共机制构成的层次关系来描述的。
- （×）2. 状态机图是 UML 中对系统进行静态建模的 5 种图之一。
- （×）3. 泳道是一种分组机制，它描述了状态机图中对象所执行的活动。
- （×）4. 同步消息和异步消息的主要区别是：同步消息的发送对象在消息发送后，不必等待消息处理，可立即继续执行，而异步消息的发送对象则必须等待接收对象完成消息处理后，才能继续执行。
- （×）5. 类图中的角色是用于描述该类在关联中所扮演的角色和职责的。
- （×）6. 类图用来表示系统中类和类与类之间的关系，它是对系统动态结构的描述。
- （×）7. 用例模型的基本组成部件是用例、角色和用例之间的联系。
- （×）8. 类之间有扩展、使用、组合等几种关系。
- （×）9. 顺序图描述对象之间的交互关系，重点描述对象间消息传递的时间顺序。
- （✓）10. 活动图显示动作及其结果。着重描述操作实现中所完成的工作以及用例实例或类中的活动。

## 三、填空题

- 1. 在 UML 提供的图中，用例图用于描述系统与外部系统及用户之间的交互。
- 2. 用例图两个最核心的元素是参与者与用例。
- 3. 用例的组成要素是参与者、用例和关系。

4. 用例中的主要关系有包含关系、扩展关系和泛化关系。

5. 用例图中以实线方框表示系统的范围和边界，在系统边界内描述的是系统的组成部分，在边界外描述的是系统环境部分。

#### 四、综合题

1. 在 UML 中的状态机图、通信图、活动图、顺序图在系统分析中各起到了什么作用？【课本 74】

【答】(1) 状态机图用来描述一个特定对象在其生存周期或在某段时间内的所有可能的状态及其引起状态转移的事件。状态机图可以用于需求分析，更好地了解用户的需求。(2) 通信图从另一个角度来更好地描述相互协作的对象间的交互关系和链接关系。着重体现交互对象间的静态链接关系和协作关系。用于需求分析中获取对象的交互关系和链接关系。(3) 活动图可以有效地描述整个系统的流程，描述了系统的全局的动态行为，可以很好地获取需求信息。(4) 顺序图清晰地描述一组对象之间动态的交互关系、时间的约束关系，着重描述对象间消息传递的时间顺序。顺序图也用于需求分析，获取对象之间的交互关系。

2. UML2.0 包括哪些图？【课本 73】

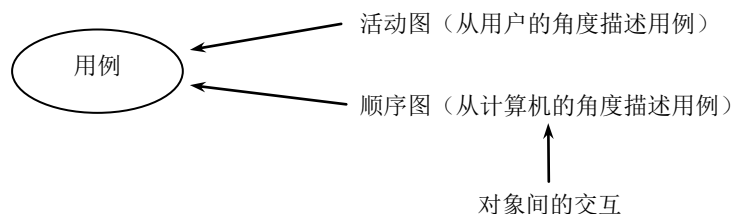
【答】UML2.0 语言定义了两类 14 种不同的图，即类图、对象图、构件图、组合结构图、包图、外廊图、用例图、顺序图、通信图、状态机图、活动图、交互概览图、定时图和部署图。

3. 类图与定时图之间的关系。

【答】类图是面向对象系统建模中最常用和最重要的图，是定义其他图的基础。类图由类和类间关系组成。定时图描述对象状态随着时间改变的情况，很像示波器，适合分析周期和非周期性任务。它展现了消息跨越不同对象或角色的实际时间，而不仅仅是关心消息的相对顺序。

4. 用例图、活动图、顺序图之间的关系。【课本 135】

【答】用例图的主要目的是帮助开发团队以一种可视化的方式来理解系统的功能需求，包括基于基本流程的“角色”之间的关系，以及系统内用例之间的关系。活动图用来描述一组顺序的或并发的活动，着重表现从一个活动到另一个活动的控制流，是内部处理驱动的流程。顺序图通过描述对象之间的交互来表达被描述对象的行为。顺序图显示多个对象间的动作协作，它以图形化方式描述了在一个用例或操作的执行过程中对象如何通过消息互相交互，重点是显示对象之间发送消息的时间顺序。用例图、顺序图和活动图的关系，如下图所示。



5. 什么是包？【课本 86】

【答】包是一种对模型元素进行成组组织的通用机制，将许多类集成成一个更高层次的单位，形成一个高内聚/低耦合的类的集合。包可以控制这些元素的可见性，使一些元素在包外是可见的，另一些元素要隐藏在包内。包也可以用来表示系统体系结构的不同视图，但是包不能执行，不能被实例化。

6. 什么是包的泛化、包的依赖？【课本 88】

【答】包的泛化关系：使用继承中通用和特例的概念来说明通用包和专用包的关系。

依赖关系其实是耦合的一种体现，如果两个包中的类之间存在依赖关系，那么这两个包之间也就有了依赖关系，也就存在了耦合关系。好的设计要求体现高内聚、低耦合的特性。

7. 包和类有何区别？【课本 75, 86】

【答】类是任何面向对象系统中最重要构造块，是对一组具有相同属性、操作、关系和语义对象的描述。包是把 UML 中的建模元素组织成组的通用机制。UML 把概念上相似的、有关联的、会一起产生变化的模型元素组织在一个包中。包就像一个“容器”，可用于组织模型中的相关元素以便控制模型的复杂度，使得开发人员更容易理解模型。如果直接使用类图进行整个系统的表示，将使类图变得混乱不堪、无法理解，所以有必要把这些元素组织成较大的组块，即包。



8. 哪些模型元素可以组成包？【课本 86】

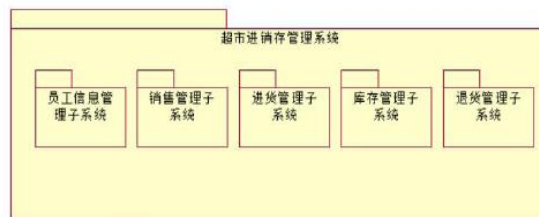
【答】一个包可以包括类、接口、构件、结点、协作、用例和图等，包可以嵌套在另一个包中。

9. 当把模型元素组成一个包时应该考虑哪些问题？【课本 89】

【答】当把模型元素组成一个包时，要考虑模型元素之间的关系，并按照以下原则来组织包：重用发布等价原则（Release Reuse Equivalency Principle, REP）；共同闭包原则（Common Closure Principle, CCP）；共同重用原则（Common Reuse Principle, CRP）；非循环依赖原则（Acyclic Dependencies Principle, ADP）；稳定抽象等价原则（Stable Abstractions Principle, SAP）和包的稳定依赖原则（Stable Dependence Principle, SDP）。

10. 对超市管理系统绘制相应的包图。

超市进销存管理系统包含了员工管理、销售管理、进货管理、库存管理和退货管理几部分，可画出超市进销存管理系统包图，如下图所示



## 第 4 章 复习思考题

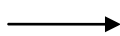
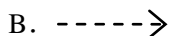
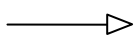
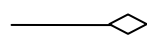
### 一、选择题

- 在 UML 活动图中，（ B ）表示一个操作完成后对其后续操作的触发。  
A. 信息流      B. 控制流      C. 初始活动      D. 活动
- 对于活动图，以下说法正确的有（ ABD ）。  
A. 活动图适用于精确地描述单个用例中的处理流程，也可用来描述多个用例联合起来形成的处理流程，表达相对复杂的业务操作或软件处理过程，有时甚至可以针对类中某个复杂的操作使用活动图给出实现细节  
B. 活动图中包含控制流和信息流，控制流表示一个操作完成后对其后续操作的触发，信息流则刻画操作期间的信息交换  
C. 活动图的基本建模机制包括结点、边及泳道  
D. 活动图描述实体为完成某项功能而执行的操作序列，其中的某些操作或者操作的子序列可以并发和同步
- 下列对系统边界的描述不正确的是（ D ）。  
A. 系统边界是指系统与系统之间的界限  
B. 用例图中的系统边界是用来表示正在建模的系统的边界  
C. 边界内表示系统的组成部分，边界外表示系统外部  
D. 我们可以使用 Rose 绘制用例中的系统边界
- UML 中关联的多重度是指（ B ）。  
A. 一个类有多个方法被另一个类调用  
B. 一个类的实例能够与另一个类的多个实例相关联  
C. 一个类的某个方法被另一个类调用的次数  
D. 两个类所具有的相同的方法和属性
- 在某个信息系统中，存在如下的业务陈述：①一个客户提交 0 个或多个订单；②一个订单由一个

且仅由一个客户提交。系统中存在两个类：“客户”类和“订单”类。对应每个“订单”类的实例，存在（1）“客户”类的实例；对应每个“客户”类的实例，存在（2）个“订单”类的实例。供选择的答案：

- B(1) A. 0 个                      B. 1 个                      C. 1 个或多个                      D. 0 个或多个  
D(2) A. 0 个                      B. 1 个                      C. 1 个或多个                      D. 0 个或多个

6. 在类图中，下面哪个符号表示继承关系（ C ）。

- A.       B.       C.       D. 

7. 在类图中，“#”表示的可见性是（ B ）。

- A. Public                      B. Protected                      C. Private                      D. Package

8. 消息的组成不包括（ C ）。

- A. 接口                      B. 活动                      C. 发送者                      D. 接收者

9. 类之间的关系不包括（ D ）。

- A. 依赖关系                      B. 泛化关系                      C. 实现关系                      D. 分解关系

10. 类，其属性的可见性表示对类的外部世界的可见性，它有以下（ ABCD ）选项。

- A. 公开                      B. 包内公开                      C. 保护                      D. 私有

11. 两个类之间的关联表示他们之间存在一种不适于继承的逻辑关系。在关联关系的表示图元的两端，可以表示参与关联的（ABCD）特性。

- A. 约束                      B. 可见性                      C. 角色名                      D. 多重性

12. （ D ）。

- A. 概念类                      B. 分析类                      C. 实现类                      D. 接口

13. 泛化使得（A）操作成为可能，即操作的实现是由它们所使用的对象的类，而不是由调用确定的。

- A. 多态                      B. 多重                      C. 传参                      D. 传值

14. （ A ）关系是类元的一般描述和具体描述之间的关系，具体描述建立在一般描述的基础之上，并对其进行了扩展，具体描述具有与一般描述完全一致的所有特性、成员和关系，并且包含补充的信息，它用从子指向父的箭头表示，指向父的是一个空三角形。

- A. 泛化                      B. 继承                      C. 组成                      D. 聚集

15. （ D ）是对象与其外界相互关联的唯一途径。

- A. 函数调用                      B. 接口                      C. 状态转换                      D. 消息传递

16. UML 的（ A ）表示消息源发出消息后不必等待消息处理过程的返回，即可继续执行自己的后续操作。

- A. 异步消息                      B. 返回消息                      C. 同步消息                      D. 简单消息

17. 如果对一个类的意义进行描述，那么应该采用（ C ）。

- A. 标记值                      B. 规格描述                      C. 注释                      D. 构造型

## 二、填空题

- 在类图中一共包含了以下几种元素，分别是：类、接口、关系、协作、注释、约束以及包。
- 类的定义要包含类名、属性和操作要素。
- 对象图中的对象是类的特定实例，链是类之间关系的实例，表示对象之间的特定关系。
- 类之间的关系包括关联关系、依赖关系、泛化关系和实现关系。
- 在 UML 的图形表示中，类的表示法是一个矩形，这个矩形由三个部分构成。
- UML 中类元的类型有类、用例、构件和接口。
- 类中方法的可见性包含三种，分别是公共的、受保护的和私有的。

## 三、名词解释

【答】（1）需求分析：在软件工程中，需求分析指的是在建立一个新的或改变一个现存的系统时描写新系统的目的、范围、定义和功能所要做的所有工作。

（2）活动图：活动图用来描述一组顺序的或并发的活动，着重表现从一个活动到另一个活动的控制

流，是内部处理驱动的流程。活动图是通过一系列活动描述对象的行为，对象可以是程序、模块、子系统、系统。

(3) 动作状态：是指原子的、不可中断的动作，并在此动作完成后通过完成转换转向另一个状态。它与活动状态有些相似，但是它们是原子活动，并且当它们处于活动状态时不允许发生转换。

(4) 活动：活动状态表示过程中命令的执行或工作流程中活动的进行。与等待某一个事件发生的一般等待状态不同，活动状态等待计算处理工作的完成。

(5) 泳道：将活动用线条分成一些纵向的矩形，这些矩形称为泳道。泳道将 UML 活动图中的活动划分为若干组，并把每一组指定给负责这组活动的业务组织，即对象。每个矩形属于一个特定的对象或部门（子系统）责任区。属于同一个对象的活动都放在同一个泳道内，对象或子系统的名字放在泳道的顶部。

(6) 可见性：怎样让其他人使用或看见名称。

(7) 类：类是具有相似结构、行为和关系的一组对象的描述符，类包括属性和操作。类的属性是对象的状态的抽象，用数据结构来描述类的属性。类的操作是对象的行为的抽象，用操作名和实现该操作的方法来描述。

(8) 对象：对象是类的一个实例。对象的抽象是类，类的具体化就是对象，也可以说类的实例是对象。对象是一个封装了状态和行为的具有良好边界和标识符的离散实体。

(9) 系统用例：系统用例是以系统内某一具体功能来研究的，它的发起者可能是业务工人，或者业务实体。

(10) 关联：关联是类与类，类与接口之间的一种比较强烈的关系，它使得一个类知道另一个类的属性和方法。它不是临时性的，而是长期性的。

(11) 依赖关系：表示类与类之间的连接，表示一个类依赖于另外一个类的定义，方向是单向的；它具有偶然性，临时性，关系较弱。

(12) 泛化关系：类的泛化关系是一般类目（称为超类或父类）和较特殊的类目（称为子类或孩子类）之间的关系。通过子类到父类的泛化关系，子类继承父类的所有结构和行为。在子类中可以增加新的属性和操作，也可以覆写父类的操作。

(13) 实现关系：表示一个模型元素实现了另一个模型元素定义的操作，一般是指一个类实现了一个接口定义的方法。

#### 四、综合题

##### 1. 简述扩展、包含和泛化三种 UML 依赖关系的异同。【课本 P105-107】

【答】包含关系（include）是把几个用例的公共行为分离成一个单独的用例，使这几个用例与该单独的用例之间建立的关系。被抽取出来的单独的用例叫做被包含用例（Inclusion），而抽取出公共用例的几个用例称为基本用例（Base）。包含用例是被封装的，代表在各种不同基本用例中复用的行为。

扩展关系（extend）是将基本用例中一段相对独立并且可选的动作，用扩展用例加以封装，再让它从基本用例中声明的扩展点上进行扩展，从而使基本用例行为更简练、目标更集中。扩展用例为基本用例添加新的行为，扩展用例可以访问基本用例的属性，因此，它能根据基本用例中扩展点的当前状态来判断是否执行自己。对于包含关系而言，子用例中的业务过程是一定要插入到基础用例中去的，并且插入点只有一个。而扩展关系可以根据一定的条件来决定是否将扩展用例的业务过程插入基础用例业务过程，并且插入点可以有多个。但是扩展用例对基本用例不可见。对于一个扩展用例，可以在基本用例上有几个扩展点。

泛化关系（generalization）描述用例之间的一般和特殊的关系，特殊用例是在继承了一般用例的特性的基础上添加了新的特性。子用例将继承父用例的所有结构、行为和关系。子用例可以使用父用例的一段行为，也可以重载它。父用例通常是抽象的。

共性：都是从现有的用例中抽取出公共的那部分信息，作为一个单独的用例，然后通过不同的方法来重用这个公共的用例，以减少模型维护的工作量。

三者对用例关系的优化侧重点是不同的。泛化侧重表示子用例间的互斥性；包含侧重表示被包含用例对 Actor 提供服务的间接性；扩展侧重表示扩展用例的触发不定性。

##### 2. 什么是用例？什么是用例图？两者之间有什么不同？【课本 P103-109】

【答】用例是系统执行的一组动作序列，并为参与者产生一个可供观察的结果，这个结果对系统的一个或多个参与者是有价值的。用例描述一个系统做什么，而不是怎么做。用例有系统用例和业务用例，用例习惯上也叫系统用例，是一种软件需求定义的方法或形式。

用例图描述系统的功能需求、使用者作用于系统边界的方法以及系统的反应。用例图展现了一组用例、参与者及它们之间的关系，它以图形化的方式描述系统与外部系统和用户的交互，强调从系统的外部参与者（主要是用户）的角度看到的或需要的系统功能。

用例图模型元素有：用例（use case）、参与者（actor）和关系（relation）。用例是用例图的组成部分。

### 3. 简述用例模型的组成元素以及建模步骤。【课本 P151】

【答】用例模型描述系统外部的参与者所理解的系统功能，它由用例图和用例描述组成，用例描述可以用用例规约描述或活动图描述。业务用例建模步骤如下：

- (1) 确定系统的业务主角和业务工人，确定将要设计的系统范围和它的边界。
- (2) 确定每一个参与者所期望的系统行为，即参与者对系统的基本业务需求。
- (3) 把这些系统行为作为基本用例。在建立业务模型、查找业务用例时必须使用业务主角，而不是普通的参与者。在做寻找业务主角的时候要抛开计算机，就算没有计算机系统这些业务人员存在。
- (4) 对复杂的用例做进一步分解，并确定底层用例以及用例间的关系。
- (5) 对每一用例做进一步细化。利用包含关系和扩展关系分解公共行为，并区分异常的行为。
- (6) 寻找每一个用例发生的前提条件和发生后对系统产生的结果。
- (7) 寻找每一个用例在正常条件下的执行过程。
- (8) 寻找每一个用例在非正常条件下的执行过程。
- (9) 用 UML 建模工具画出分层的用例模型图，并编写出每个业务用例的用例描述。
- (10) 审核业务用例模型。
- (11) 编写业务用例模型图的补充说明文档。

### 4. 用例图在系统中有什么作用？【课本 P103】

【答】用例图用于对系统、子系统或类的行为进行可视化，使用户能够理解如何使用这些元素，并使开发者能够实现这些元素，它将系统功能划分成对参与者（即系统的理想用户）有用的需求，而交互功能部分被称作用例。用例图的主要目的是帮助开发团队以一种可视化的方式来理解系统的功能需求，包括基于基本流程的“角色”之间的关系，以及系统内用例之间的关系。

### 5. 试述如何识别用例？【课本 P154】

【答】寻找业务用例的方法：(1) 从原始需求中寻找所包含的功能。(2) 未来的系统，需要和哪些系统发生信息交互？(3) 哪些人将操作未来的软件系统？(4) 系统有哪些受限的条件？(5) 未来的软件界面怎样组织？(6) 系统的响应有哪些去向？

一个用例应有明确有效的目标，一个真实的目标应完备地表达主角的期望，一个有效的目标应当在系统边界之内，由主角发动，具有明确后果。

### 6. 用例之间的三种关系各使用在什么场合？【课本 P107】

【答】用例和用例之间的关系，主要分为包含关系、扩展关系和泛化关系。运用包含关系，可以把公共的行为放到它自己的一个用例中，避免多次描述相同的事件流。扩展用例为基本用例添加新的行为，扩展用例可以访问基本用例的属性，因此，它可以根据基本用例中扩展点的当前状态来判断是否执行自己。对于包含关系而言，子用例中的业务过程是一定要插入到基础用例中去的，并且插入点只有一个。而扩展关系可以根据一定的条件来决定是否将扩展用例的业务过程插入基础用例业务过程，并且插入点可以有多个。但是扩展用例对基本用例不可见。对于一个扩展用例，可以在基本用例上有几个扩展点。扩展用例带有抽象性质，表示用例场景中的某个“支流”，由特定的扩展点触发而被启动。与包含关系不同，扩展表示“可选”，而不是“必需”，这意味即使没有扩展用例，基本用例也是完整的，如果没有基本用例，扩展用例不能单独存在。泛化关系和扩展关系表示的是用例间的“is a”关系，包含关系表示的是用例间的“has a”关系。泛化关系和扩展关系有相似之处，不同的是扩展关系需要明确标明被扩展用例的扩展点，也就是说，一个扩展用例只能在基本用例的扩展点上扩展。在扩展关系中，基本用例一定是一个真实存在的用例，一个基本用例执行时，可以执行，也可以不执行扩展用例。在包含关系中，基本用例可能是，也

可能不是一个真实存在的用例，一定会执行包含用例部分。如果需要重复处理两个或多个用例时，可以考虑使用包含关系。

7. 请问在设计系统时，绘制的用例图是多一些好，还是少一些好，为什么？

答：【答】用例是有一定粒度的，开发者往往难于确定用例粒度。如果用例粒度很大，那么得到的用例数就会很少，如果用例粒度很小，那么得到的用例数就会很多。如果用例数目过多，会造成用例模型过大，设计难度会加强。如果用例数目过少，会造成用例粒度太大，不利于进一步的分析。一个基本用例可以分解出许多更小的关键精化用例，这些小的精化用例展示了基本用例的核心业务。

8. 请简述为何在系统设计时要使用用例图。它对我们有什么帮助？

【答】用例图是从软件需求分析到最终实现的第一步，它显示了系统的用户和用户希望提供的功能，有利于用户和软件开发人员之间的沟通。借助于用例图，系统用户、系统分析人员、系统设计人员、领域专家能够以可视化的方式对问题进行探讨，减少了大量交流上的障碍，便于对问题达成共识。

9. 类图的组成元素有哪些？对象图有哪些组成部分？

【答】类的组成元素有类的名称、类的属性、类的操作、类的职责、类的约束和类的注释。对象图是由对象和链组成的。

10. 为什么要使用类图 and 对象图？请简要说明类图 and 对象图的关系和异同。

【答】类图（Class Diagram）是面向对象系统建模中最常用和最重要的图，是定义其他图的基础。类图由类和类间关系组成，对象图是类图的变体，它使用与类图相类似的符号描述。

类图和对象图的关系：对象图是类图的实例，几乎使用与类图完全相同的标识。对象图的两个基本元素是对象和它们之间的关系。

异同：在类中包含三个部分，分别是类名、类的属性和类的操作。类的名称栏只包含类名。类的属性栏定义了所有属性的特征。类中列出了操作类中使用了关联连接，关联中使用名称、角色以及约束等特征定义。类是一类的对象的抽象，类不存在多重性。与类图相似，水平线将图标内的文字分成了两部分，上边代表对象的名称，下边代表对象的属性和值。对象的名称栏包含“对象名：类名”。对象的属性栏定义了属性的当前值。对象图中不包含操作内容，因为对属于同一个类的对象，其操作是相同的。对象使用链进行连接，链中包含名称、角色。对象可以具有多重性。

## 第 5 章 复习思考题

### 一、选择题

- 顺序图的用途包括（ ABCD）。
  - 显示并发进程和激活
  - 当不同的类之间存在多个简短的方法时，描述控制流的整体序列
  - 显示在协作图中难于描述的事件序列
  - 显示涉及类交互而与对象无关的一般形式
- 下面哪些图形可以清楚地表达并发行为（ CD ）。
  - 类图
  - 状态机图
  - 活动图
  - 顺序图
- 下面哪个视图属于 UML 语言的交互图？（ D ）
  - 行为图
  - 状态机图
  - 实现图
  - 顺序图
- 在 UML 中，通信图的组成不包括？（ C ）种。
  - 对象
  - 消息
  - 发送者
  - 链
- 在 UML 中，接口有几种表达方式（ 2 ）。
  - 2
  - 4
  - 6
  - 8
- 下面哪个 UML 视图是描述一个对象的生命周期的？（ B ）
  - 类图
  - 状态机图
  - 协作图
  - 顺序
- 在 UML 顺序图中，（ A ）是对消息传递的目标对象的销毁。

- A. 销毁消息      B. 创建消息      C. 返回消息      D. 自返消息
8. ( D ) 用于描述相互合作的对象间的交互关系的图。
- A. 类图      B. 顺序图      C. 用例图      D. 通信图

## 二、判断题

- (✓) 1. 通信图作为一种交互图，强调的是参加交互的对象的组织。
- (✗) 2. 通信图是顺序图的一种特例。
- (✓) 3. 通信图中有消息流的顺序号。
- (✗) 4. 状态机图通过建立类对象的生命周期模型来描述对象随时间变化的动态行为。
- (✗) 5. 状态机图适用于描述状态和动作的顺序，不仅可以展现一个对象拥有的状态，还可以说明事件如何随着时间的推移来影响这些状态。
- (✗) 6. 状态机图的主要目的是描述对象创建和撤销的过程中资源的不同状态，有利于开发人员提高开发效率。
- (✗) 7. 状态机图描述了一个实体基于事件反应的动态行为，显示了该实体如何根据当前所处状态对不同的事件做出反应。
- (✗) 8. 顺序图由对象、生命线、控制焦点和实体组成。

## 三、填空题

- 在通信图中通过顺序号表示出消息的时间顺序。
- 顺序图是由对象类角色、生命线、激活期和消息等构成的。
- 在 UML 的表示中顺序图将交互关系表示为一张二维图，其中纵向是时间轴，时间沿竖线向下延伸。横向代表了在协作中各独立对象的角色。
- 状态机图描述从状态到状态的控制流程，常用来对系统的动态特征进行建模。
- 在 UML 中，状态机由对象的各个状态和连接这些状态的转换组成，是展示状态与状态转换的图。

## 四、名称解释

【答】系统用例：系统用例是以系统内某一具体功能来研究的，它的发起者可能是业务工人，或者业务实体。

状态机图：状态机图展示了一个由状态、转换、事件和活动组成的状态机，用来说明系统的动态视图。

子状态机：在一个状态机中可以引用另一个状态机，被引用的状态机称为子状态机。

顺序图：顺序图显示多个对象间的动作协作，它以图形化方式描述了在一个用例或操作的执行过程中对象如何通过消息互相交互，重点是显示对象之间发送消息的时间顺序。

通信图：通信图对在交互中有意义的对象和对象间的链建模。除了显示消息的交互以外，通信图也显示对象及其关系。通信图强调收发消息的对象或角色的结构组织。

同步消息：同步消息=调用消息。消息的发送者把控制传递给消息的接收者，然后停止活动，等待消息的接收者放弃或者返回控制。

异步消息：消息发送者通过消息把信号传递给消息的接收者，然后继续自己的活动，不等待接收者返回消息或者控制。异步消息的接收者和发送者是并发工作的。

生命线：生命线（Lifeline）是一条垂直的虚线，表示顺序图中的对象在一段时间内的存在。每个对象的底部中心的位置都带有生命线，生命线是一个时间线，从顺序图的顶部一直延伸到底部，所用的时间取决于交互持续的时间。

## 五、综合题

1. 业务用例模型和系统用例模型有哪些联系与区别？【课本 P192-193】

【答】业务用例着重于业务操作。它们表示实现业务目标的业务中的具体 workflows。业务过程可能涉及手工和自动过程，并且在一段长期的时间内进行。系统用例着重于要设计的软件系统。参与者如何与软件

系统进行交互？在系统用例说明中书写的事件流应该足够详细，便于用作编写系统测试脚本的出发点。系统用例的参与者为操作人员所代表的岗位角色，可以是实际与系统交互的操作人员、外部衔接系统、自动服务、定时器等。系统用例不能是一个步骤，如输入用户名，或者验证用户名等。业务用例和系统用例在用例技术的使用上没什么差别，如用例的关系、用例的描述等。在业务模型中还有一个概念，即“业务工人（Business Worker）”。业务工人表示实现业务的人、软件或硬件等角色。比如银行的“开户”业务用例中，银行柜员、软件系统、打印存折的打印机等都可看作是“业务工人”。业务用例模型与系统用例模型有很多相似之处。两个模型都有用例说明。如果对业务用例模型以及系统用例模型的 RUP 模板进行检查，就会发现它们的格式十分相似。两者都包含先决条件、后置条件以及特殊需求等。业务用例说明有基本的工作流和可选择的工作流，从而取代了基本的事件流和可选流。系统用例的设计范围就是这个计算机系统设计的范围。它是一个系统参与者，与计算机系统共同实现一个目标。系统用例就是参与者如何与计算机技术相联系，而不是业务过程。业务用例模型与系统用例模型在设计范围、系统测试以及业务角色方面有所不同。

## 2. 简述活动图 and 状态机图的区别？【课本 P202】

【答】虽然 UML 活动图与状态机图都是状态机的表现形式，两者很相似（所用的图形符号的画法），但是两者还是有本质区别的。（1）两者的描述重点不同。状态机图描述的是一个对象的生命周期内的状态机状态之间的转移，以及引擎状态转移的时间和对象在状态中的动作等。而活动图描述的是从一个活动到另一个活动的控制流，用于描述多个对象在交互时所采取的活动，它关注对象如何相互活动以完成一个事务，是内部处理驱动的流程。（2）两者的使用的场合不同。如果要表示一个对象在其生命周期内的行为，则使用状态机图。如果是为了分析用例，或理解设计多个用例的工作流程，或处理多线程应用等，则使用活动图。（3）两者的用途不同。状态机图的主要用途是为一个对象在其生命周期中的一组属性值对所发生的时间的反应的建模。活动图的主要用途有 2 种，一是为业务流程建模；二是为对象的特定操作建模。在实际的项目中，活动图不是必需的。主要用活动图描述并行的过程或者行为；描述一个算法；描述一个跨越多个用例的活动。状态机图描述了一个具体对象的可能状态以及它们之间的转换。（4）两者包含的动作性质不同。状态机图中的动作是原子计算的额，不能被分解。活动图中的动作有活动和动作之分，是非原子的，可以被分解。状态是行为的结果，活动是行为的动作。（5）两者包含的动作的场所不同。状态机图中的动作发生在状态中或转移中，活动图中的动作可以放在泳道中。（6）两者包含的动作的执行条件不同。状态机图中的动作的执行需要事件的触发，活动图中的动作执行不需要事件的触发。（7）两者与对象的关系不同。状态机图中可以表示对象的属性值，活动图中既可以表示对象的值流，也可以表示动作的控制流。其实活动图是用来建模不同区域的工作如何彼此交互的，而状态机图用来表示单个对象以及对象的行为如何改变其状态。（8）两者在软件生命周期中的阶段位置不同。状态机图只在系统设计阶段使用，活动图可以在需求分析阶段，也可以在设计阶段使用。活动图可以使用泳道把活动进行分组，目的是用来描述对象间的合作关系。状态机图中某些标记符与活动图的标记符非常相似，有时候会让人混淆。

## 3. 交互图有哪些类型？【课本 P121】

【答】交互图包括通信图、顺序图、交互概览图和定时图。

## 4. 如何在顺序图中表示消息的循环发送和条件发送？【课本 P125】

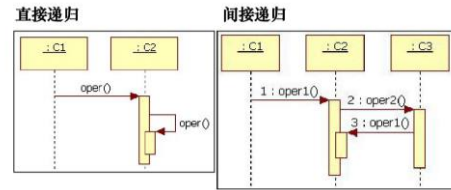
【答】循环（迭代）执行的标签是 loop。。在顺序图中的消息名前加循环条件。条件执行的标签是 alt。控制操作符的主体用水平虚线分割成几个分区，每个分区表示一个条件并有一个监护条件。表示消息的条件发送有以下几种方法：（1）在消息上加警戒条件；（2）在消息名字前加条件子句；（3）使用文字说明；（4）分成多个顺序图。

## 5. 如何在顺序图中表示时间约束？【课本 P124】

【答】在顺序图上，如果消息的箭头和生命线垂直，那么表明立即发送该消息。如果箭头是倾斜的，那么表明该消息的传送有一定的时间延迟，其间可以传送其他消息。建模人员可在顺序图的时间轴附近用标签来定义消息上的时间约束。

## 6. 如何在顺序图中表示方法的递归？





【答】利用嵌套的 FOC（focus of control 控制焦点）表示。

7.如果对象具有多态性，发送对象，不可能事先知道目标对象属于哪个类，因此在交互图中如何确定目标对象所属的类？

【答】多态性属于运行时的问题，这个类是目标对象有可能所属类的超类（一般类或父类）。

8.请分析顺序图和通信图的主要差别和各自的优缺点？【课本 P198】

【答】顺序图描述了消息的时间顺序，适用于描述实时系统和复杂的脚本；通信图描述了对象间的关系。顺序图和通信图都属于交互图，用于描述单个用例中多个对象的交互行为，它们用于描述简单的行为，行为复杂时将失去清晰性。顺序图和通信图从不同的角度表达了系统中的交互和系统的行为。顺序图着重描述对象按照时间顺序的消息交换，并且把用例行为分配给类，各角色之间的关系是隐含的，并且没有明确地表达对象间的关系。通信图用各个角色排列来表示角色之间的关系，并用消息类说明这些关系。通信图着重描述系统成分如何协同工作，强调对象间的结构关系，时间顺序必须从顺序号获得。顺序图和通信图在语义上是等价的，两者之间可以相互转换，但两者并不能完全相互代替。顺序图可以表示某些通信图无法表示的信息，同样，通信图也可以表示某些顺序图无法表示的信息。例如，在顺序图中不能表示对象与对象之间的链，对于多对象和主动对象也不能直接显示出来，在通信图中则可以表示；通信图不能表示生命线的分叉，顺序图中则可以表示。顺序图有两个不同于通信图的特征：

第一，顺序图有对象生命线。对象生命线是一条垂直的虚线，表示一个对象在一段时间内存在。在交互中可以创建对象，它们的生命线从接收到 **create** 消息时开始。在交互中也可以撤销对象，它们的生命线在接收到 **destroy** 消息时结束，用一个大的标记标明对象生命的结束。通信图中尽管可以展示 **create** 和 **destroy** 消息，但是不能显式地展示对象的生命线。

第二，顺序图有控制焦点。控制焦点是一个瘦高的矩形，表示对象执行一个动作所经历的时间段，既可以是直接执行，也可以是通过下级过程执行。矩形的顶部表示动作的开始，底部表示动作的结束。还可以通过将一个控制焦点放在它的父控制焦点的右边表示控制焦点的嵌套。在通信图中，尽管各消息的序号可以表示嵌套，但是不能显式地展示控制焦点。

通信图有两个不同于顺序图的特征：

第一，通信图有路径。可以根据关联画一个路径，也可以根据本地变量、参数、全局变量和自访问呈现路径。路径表示一个对象的知识源。

第二，通信图中有消息顺序号。为表示消息的时间顺序，可以给消息加一个数字前缀，在控制流中，每个新消息的序号单调增加。为了显示嵌套，可以使用杜威十进分类号。需要注意的是，沿同一个链，可以显示许多的消息（可能发自不同的方向），并且每个消息都各有唯一的序号。

顺序图在表示算法、对象的生命期、具有多线程特征的对象等方面相对来说更容易一些，但在表示并发控制流方面会困难一些。

9.创建的新对象在顺序图中如何表示？

【答】顺序图是显示对象之间交互的图，这些对象是按时间顺序排列的。特别地，顺序图中显示的是参与交互的对象及对象之间消息交互的顺序。UML 动态模型图中顺序图用来显示对象之间的关系，并强调对象之间消息的时间顺序，同时显示了对象之间的交互。显示的对象沿 X 轴排列，在传送消息时，对消息的接收通常会产生一个动作。这个动作可能引发目标对象以及该对象可以访问的其他对象的状态改变。在 UML 的顺序图中可以创建一个新对象，创建的对象在创建的位置，用对象符号和生命线表示。

10.在通信图中如何表示出消息的时间顺序？

【答】通信图中的消息类型与顺序图中的相同，只不过为了说明交互过程中消息的时间顺序，需要给消息添加顺序号。顺序号是消息的一个数字前缀，是一个整数，由 1 开始递增，每个消息都必须有唯一的顺序号。



### 11.消息有几种类型？它们之间的区别是什么？

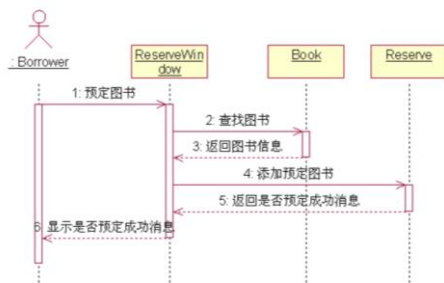
【答】消息有简单消息、同步消息、异步消息、返回消息、自关联消息、阻止消息等。（1）简单消息，指不考虑同步或异步问题的一种消息；（2）同步消息=调用消息，消息的发送者把控制传递给消息的接收者，然后停止活动，等待消息的接收者放弃或者返回控制，这是最常用的消息，用来表示同步的意思，由发送消息的来源对象指向负责执行的目标对象；（3）异步消息，消息发送者通过消息把信号传递给消息的接收者，然后继续自己的活动，不等待接收者返回消息或者控制；（4）返回消息，返回消息表示从过程调用返回；（5）自关联消息，表示方法的自身调用以及一个对象内的一个方法调用另一个方法；（6）阻止消息，Rose 的扩充消息类型，指消息的发送者发送消息给消息的接收者，如果接收者无法立即接收消息，则发送者放弃这个消息；（7）超时消息，指消息的发送者发送消息给消息的接收者并按指定的时间等待，如果接收者无法立即接收消息，则发送者放弃这个消息；（8）返身消息（自返消息），指消息的发送者发送消息给自身。

### 12.用例图与顺序图之间的区别是什么？

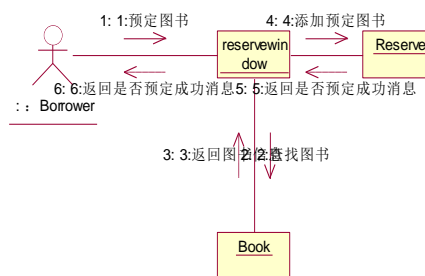
【答】（1）用例图。它展现了一组用例、用户以及它们之间的关系，即从用户角度描述系统功能，并指出各功能的操作者。用于收集用户实际需求所采用的一些方法中。用例图描述系统的功能及外部的使用者，即确定谁来使用系统，使用系统做什么。用例就是系统提供的功能的一种描述，参与者是那些可能使用这些用例的人或者外部系统，二者之间的联系描述了“谁使用哪个用例”。用例图着重于从系统外部参与者的角度描述系统需要提供哪些功能，并且指明这个系统的使用者是谁。

（2）顺序图。它展现了一组对象和由这组对象收发的消息。用于按时间顺序对控制流建模，说明系统的动态视图，强调时间和顺序。顺序图描述几个对象之间的动态协作关系。顺序图的重点在于它非常直观地展示了对对象之间传递消息的时间顺序，反映了对象之间的一个特定的交互过程，如在系统执行过程某个特定时刻发生的事情。用例图侧重描述用户需求，顺序图描述的是系统的行为。

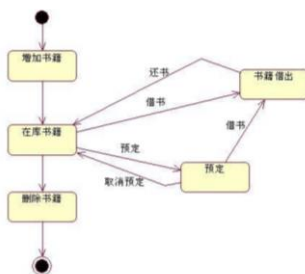
13. 请您根据分析结果，运用 UML 绘制出图书馆管理系统中“借阅者预定图书”的顺序图、通信图和还书用例的状态机图。



借阅者预定图书顺序图



借阅者预定的通信图



图书馆图书状态机图

## 第6章 复习思考题

### 一、选择题

1. 面向对象系统中功能复用的两种最常用技术是（ AB ）。  
A. 对象组合（优先使用）      B. 类继承（限制使用类继承）  
C. 多态      D. 组合
2. 信息系统体系结构的要素主要包括（ ABC ）。  
A. 拓扑结构      B. 层次结构      C. 计算模式设计      D. C/S 模式
3. 系统体系结构模型可以用（ AB ）表示。  
A. 构件图      B. 部署图      C. 结构图      D. 类图
4. 构件图构成元素有（ BC ）。  
A. 结点      B. 构件      C. 接口      D. 连接线
5. 部署图包括的基本元素有（ ABCD ）。  
A. 接口      B. 工件      C. 结点      D. 结点间的连接

### 二、名称解释

【答】软件系统体系结构：软件体系结构描述软件的组织模式。软件体系结构又被称为软件构件结构（Software Component Infrastructure, SCI）或软件架构，是对软件系统整体结构的刻画。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。

硬件系统体系结构：硬件体系结构指系统的硬件组织模式。硬件系统体系结构模型用部署图表示。部署图是 UML 用来描述系统的硬件配置、硬件部署以及软件构件和模块在不同结点上分布的模型图。部署图显示了系统中的硬件，安装在硬件上的软件，以及用于连接异构的机器之间的中间件。

构件：也称为组件，它是软件系统中具有独立功能的部分，也是软件系统结构中重要的组成要素，它在功能和数据上构成了一个软件系统的基础。

结点：是运行时硬件资源的通用名称，可以用来表示各种资源的类型，如 CPU、设备和内存等；还可以包含对象和构件的实例。

接口：接口表示了构件间的交互，接口也叫连接件。接口由一组角色组成，连接件的每一个角色定义了该连接件表示的交互的参与者，二元连接件有两个角色。

依赖：构件与接口之间有依赖关系。构件间的依赖关系是提供服务的构件称为提供者，使用服务的构件称为客户，依赖关系用带箭头的虚线表示。

### 三、填空题

1. 系统体系结构建模可分为软件系统体系结构建模和硬件系统体系结构建模。
2. 软件构件分为源代码构件（编译时构件）、二进制代码（连接时构件）和可执行代码（运行时构件）。
3. 构件图主要用于软件系统体系结构建模。
4. 部署图由结点和结点之间的联系组成，描述了处理器、设备和软件构件运行时的体系结构。
5. 结点之间、结点与构件之间的联系包括通信关联、依赖关联。
6. 面向对象系统中的“黑盒复用”是指对象组合。
7. 设计模式中应优先使用的复用技术是对象组合。

### 四、综合题

1. 构件之间有什么联系？【课本 94】

【答】构件之间、构件与接口之间有依赖关系。构件间的依赖关系是：提供服务的构件称为提供者，使用服务的构件称为客户，依赖关系用带箭头的虚线表示。

## 2. 如何进行构件图建模？【课本 220】

【答】(1) 确定构件。首先分析系统，然后从系统组成结构、软件复用、物理结点配置、系统归并、确定构件成分等几个方面寻找并确定构件。根据系统的功能和结构模块，从方便管理出发确定构件；考虑在同领域不同系统或更大范围内进行软件复用，据此来确定构件；根据网络计算机（结点）分布的配置来确定构件；将关系密切的可执行程序及与之有关的持久对象库归并为一个构件；为每个构件找出并确定相关的对象类及各种借口等。(2) 说明构件。接着使用构造型说明构件的性质，并为构件命名，构件的命名应有意义。UML 的标准构造型有<<file>>、<<page>>、<<document>>、<<library>>、<<application>>和<<table>>等。开发者还可以自定义新的构造型。(3) 标识构件之间的联系。一个构件使用了另一个构件所提供的接口，则说该构件和另一个构件之间存在依赖关系。依赖关系用带箭头的虚线表示。分析完构件就需要标示构件之间的依赖关系，对于接口应注意的是判断该接口是输出接口还是输入接口。构件实现接口，即构件被使用的接口就是输出接口。一个构件可以有多个输出接口。构件使用的接口，即使用其他构件的接口就是输入接口。一个构件可以有多个输入接口。另外，一个构件既可以是输入接口，又可以是输出接口。(4) 组织构件。最后进行构件的组织，对于复杂的软件系统，应使用“包”组织构件，形成清晰的结构层次图。

## 3. 如何进行部署图建模？【课本 222】

【答】(1) 确定结点。根据硬件设备配置和软件体系结构功能确定结点。一个结点可以部署一个或多个构件，一个构件可以部署在一个或多个结点上。(2) 确定驻留构件。确定驻留在结点内的构件和对象，并标明构件之间以及构件内对象之间的依赖关系。(3) 注明结点性质。用构造型注明结点的性质。(4) 确定结点之间的联系。确定结点之间的通信联系。(5) 绘制部署图。对结点进行统一组织和分配，绘制结构清晰并具有层次的部署图。

## 4. 构件有几类？【课本 92】

【答】在 UML 中将软件构件分为源代码构件（编译时构件）、二进制代码（连接时构件）和可执行代码（运行时构件）。源代码构件是软件开发过程中产生的，是实现一个或多个类的源代码文件，用于产生可执行系统。二进制代码构件是源代码构件经过编译后产生的目标代码文件或静态、动态库文件。可执行代码构件是系统执行时使用的构件，表示在处理器上运行的可执行单元。

## 5. 如何确定构件？【课本 220】

【答】首先分析系统，然后从系统组成结构、软件复用、物理结点配置、系统归并、确定构件成分等几个方面寻找并确定构件。根据系统的功能和结构模块，从方便管理出发确定构件；考虑在同领域不同系统或更大范围内进行软件复用，据此来确定构件；根据网络计算机（结点）分布的配置来确定构件；将关系密切的可执行程序及与之有关的持久对象库归并为一个构件；为每个构件找出并确定相关的对象类及各种借口等。

## 6. 简述构件图和部署图的区别。【课本 220-227】

【答】构件图描述了软件构件及各构件之间的关系。在构件图部分需要理解构件的概念，了解构件的分类；理解接口的概念和作用。构件图的使用有利于了解系统的功能和结构，以及软件的复用。部署图描述了系统运行时进行处理的结点和在结点上活动的构件的配置。部署图主要用来对系统的静态部署进行建模。在使用部署图时不一定要使用 UML 中的图符，也可以根据自己的习惯来绘制部署图，只要保证绘制的部署图能够被所有的开发人员认可和理解即可。在绘制部署图时，只需要描述那些对系统的实现至关重要的构件。

# 第 7 章 复习思考题

## 1. 可行性分析包括哪些内容？

【答】可行性分析首先需要进行现行系统的需求分析。对系统进行调查，了解系统的需求，进行系统的需求描述，对系统进行需求分析，并用适当的工具进行描述。了解系统的费用、计算机及软件应用情况及现行系统存在的主要问题和薄弱环节等，提出新系统开发方案。新系统开发方案包括拟建系统的目标、系统规划及初步开发方案、计算机逻辑配置方案、系统的实施方案、投资方案、人员培训及补充方案等。

然后对新系统方案进行可行性研究。可行性研究从技术上的可行性、经济上的可行性、系统运行可行性、进度的可行性等方面进行分析，得出分析的结论。

2. 请选择一个实例使用面向对象的方法进行开发。

【答】需求管理系统的需求陈述是：物流运输管理系统是现代信息化时代各种网络物资交换的重要环节。物流运输管理系统的主要目标是实现物资运输的合理化和信息化，确保企业在物流运输系统中成本少获益高同时也为用户提供更加全面更加便捷的服务。物流运输管理系统的目的也为了解决现有的物流运输管理系统中存在的货运信息不对称，物资流通过程中信息不明确出现物资丢失，漏件等现象。此系统运用了信息化的手段更好的解决了这一问题。为了更好的满足企业和用户的需求就需要建立一个流程化、智能化、信息化和系统化的物流运输管理系统。物流运输是双向的，而且包含了许多重要的中间环节，这就需要建立一个更加完备的物流。运输管理系统，这不仅是现代物流运输行业适应信息化时代发展的必要要求，也是现代物流运输业发展的必然趋势。物流运输管理系统需要的核心功能包括对物流运输过程的实时监控，对在途货物的实时跟踪，对订单管理的实时监测，当物流运输过程发生故障时系统进行及时的维护使其物流运输系统能够快速运转并且及时有效地恢复正常状态。同时将用户和企业对物流运输的需求相结合，最大程度地满足了用户和企业的需求。

(1) 系统需求建模

1) 确定参与者。通过对系统的需求陈述进行分析，确定了物流运输管理系统的参与者主要为系统维护员、货物管理员和用户。2) 确定用例。系统管理、订单管理、货物管理、仓储管理、财务管理、运输管理等。3) 绘制用例图。物流运输管理用例图如图1所示。

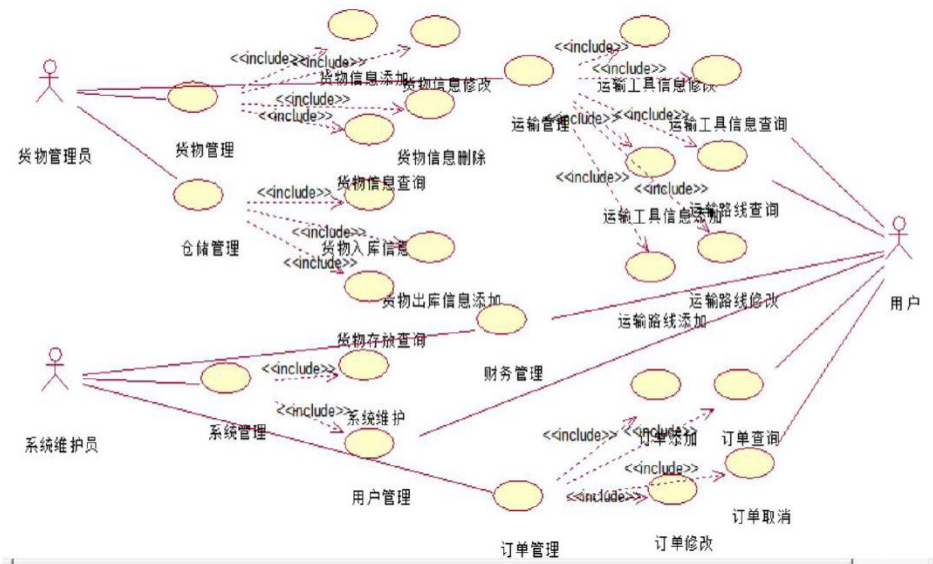


图1 物流运输管理用例图

4) 用例描述。绘制完用例图，需要对用例进行描述。用例描述可以采用规约描述，也可以使用活动图来表示。订单管理的规约描述如表1所示。

表1 订单管理的规约描述

用例描述项目	内容
用例编号	UC2
用例名称	订单管理
用例描述	订单管理
用户角色	系统维护员、用户
用例发生背景	当用户生成新的订单时
前置条件	用户可以进入订单管理界面对订单进行查询和取消
主事件流	用户进入订单管理中心，选择“订单管理”； 生成新的订单信息，并保存订单信息到数据库中； 选择要查询的订单记录，并提交“订单查询”按钮； 选择要取消的订单记录，并提交“订单取消”按钮。

订单管理的活动图表示如图2所示。

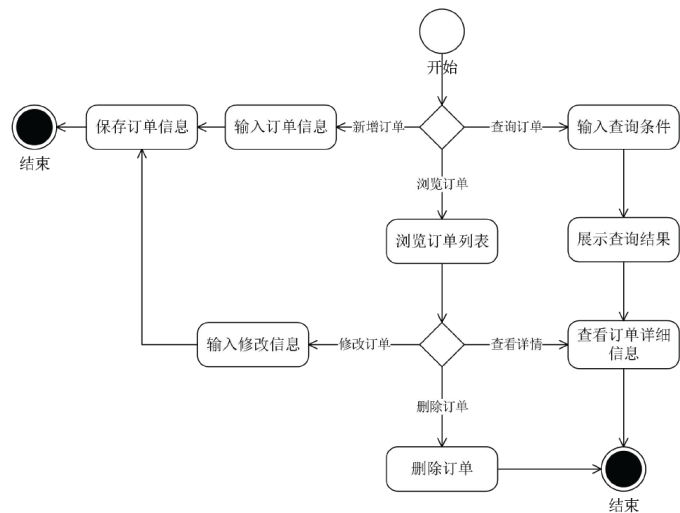


图2 订单管理的活动图

(2) 类图的构建

1) 类的识别。本系统中有订单类、货物类、运输类、仓储类和用户类等。

2) 类图的绘制。本系统的类图如图3所示。

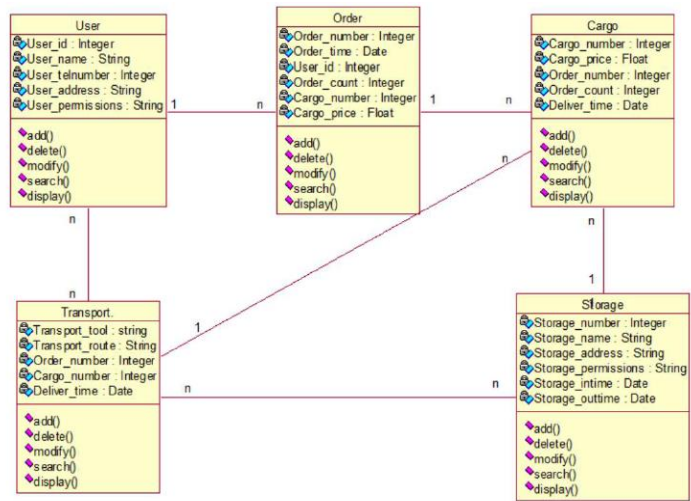


图3 系统类图

(3) 系统执行顺序分析

1) 系统管理。它主要实现管理员对用户信息进行增删改查。系统管理顺序图如图4所示。

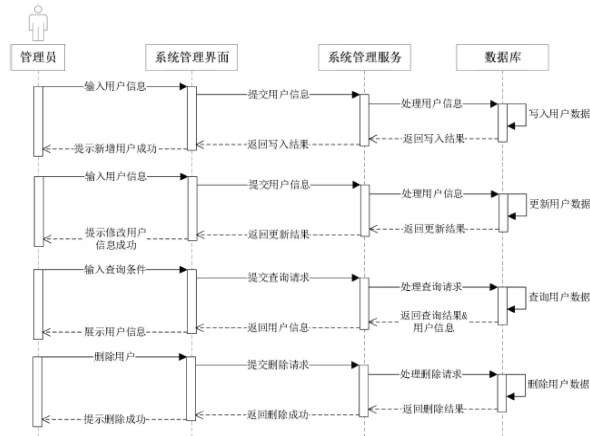


图4 系统管理顺序图

2) 订单管理。主要是管理员对订单信息处理的过程。订单管理顺序图如图4所示。

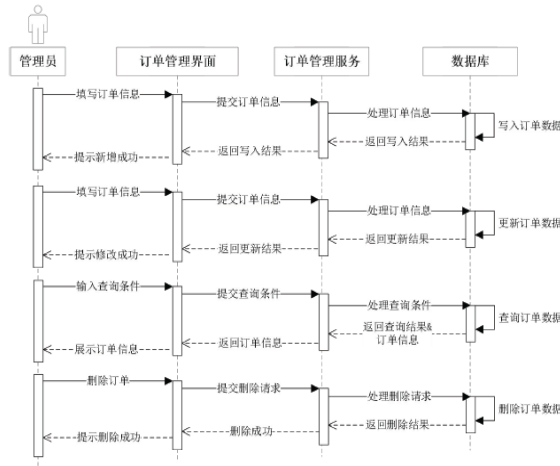


图5 订单管理顺序图

#### (4) 系统协作分析

1) 系统管理。系统管理分为用户管理和系统维护。其中用户管理通信图如图6所示。

2) 订单管理。它主要完成用户对订单的增删改活动。订单管理通信图如图7所示。

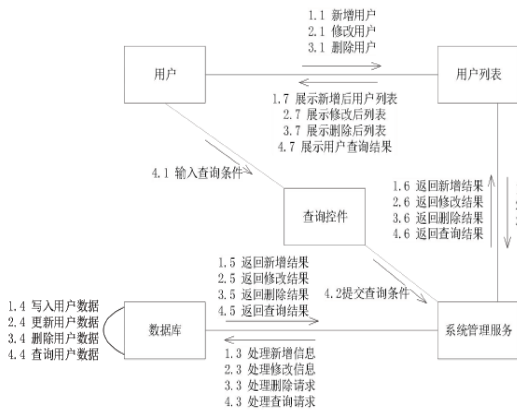


图6 用户管理通信图

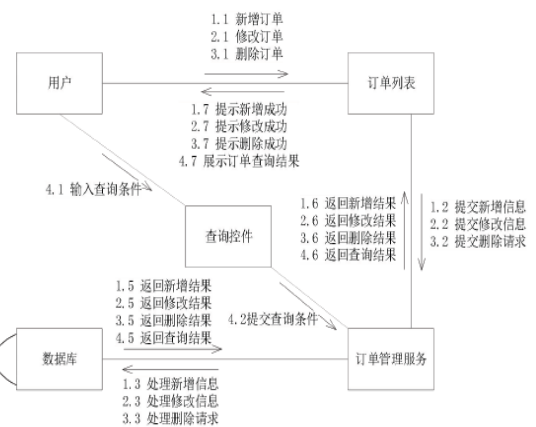


图7 订单管理的通信图

#### (5) 系统的状态分析

1) 财务管理。它主要完成对订单中货物的运费进行记录的过程，财务的状态机图如图8所示。

2) 货订单管理主要实现对订单进行增删改查信息记录的过程，订单管理的状态机图如图9所示。

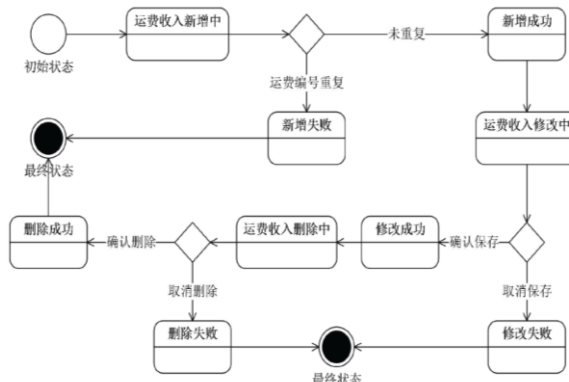


图8 财务管理的状态机图



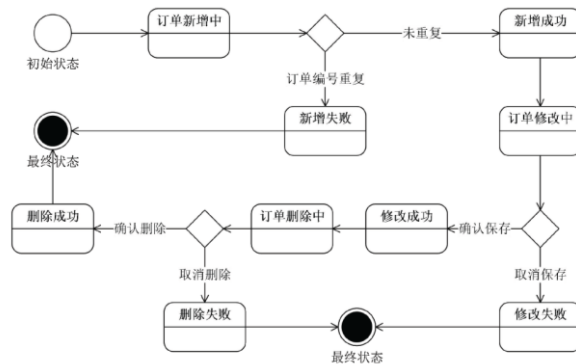


图9 订单管理的状态机图

6. 数据库设计（略）

7. 软件系统体系结构建模

物流运输管理系统的构件图如图10所示。

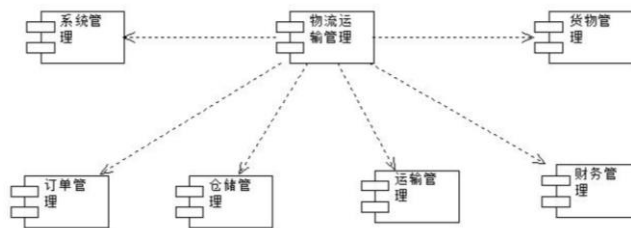


图10 物流运输管理系统的构件图

8. 硬件系统体系结构建模

物流运输管理系统的部署图如图11所示。

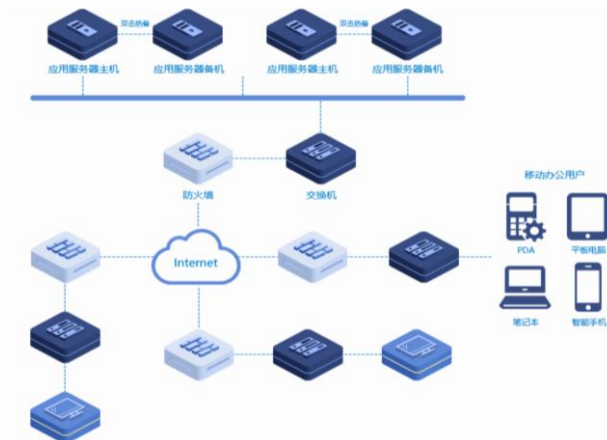


图11 物流运输管理系统的部署图

## 第8章 复习思考题

### 一、选择题

- Rational Rose 中 Rose 模型的视图包括（ ABD ）。  
A. 用例视图      B. 部署视图      C. 数据视图      D. 逻辑视图
- 在用例视图下可以创建（ ACD ）。  
A. 类图      B. 构件图      C. 包      D. 活动图
- Rational Rose 建模工具可以执行的任务有（ ABCD ）。  
A. 非一致性检查      B. 生成 C++语言代码





体实现方法。(2) 逻辑视图 (Logical View): 逻辑视图关注系统如何实现用例中提出的功能, 它提供系统的详细图形, 描述组件间如何关联。(3) 构件视图 (Component View): 构件视图用来描述系统中的各个实现模块以及它们之间的依赖关系。(4) 部署视图 (Deployment View): 部署视图建模系统的各个硬件结点及其相互间的通信方式。在系统中, 只包含有一个部署视图, 用来说明各种处理活动在系统各结点的分布。

2. 试述如何使用 Rational Rose 生成代码。【课本 293-297】

【答】使用 Rational Rose 生成代码有通过以下四个步骤进行: 选择待转换的目标模型、检查语言的语法错误、设置代码生成属性、最后生成代码。

3. 请简要说明使用逆向工程的步骤。【课本 297-298】

【答】在 Rational Rose 中, 可以通过收集有关类、类的属性、类的操作、类与类之间的关系以及包和构件等静态信息, 将这些信息转化成为对应的模型, 在相应的图中显示出来。

4. 请阐述用例视图和逻辑视图的区别以及各自的使用场合。【课本 262-263】

【答】用例视图关注的是系统功能的高层抽象, 适合于对系统进行分析和获取需求, 而不关注于系统的具体实现方法。逻辑视图关注系统如何实现用例中所描述的功能, 主要是对系统功能性需求提供支持, 即在为用户提供服务方面, 系统所应该提供的功能。

5. 简单描述 Rational Rose 2003 的安装过程。【课本 263】

【答】首先下载 Rational Rose 2003, 然后, 双击 Rational Rose 2003 的安装程序, 在安装文件释放后, 进入安装向导界面。单击“下一步”按钮, 进入产品选择界面。此时用户可以选择要安装的产品, 一般选择 Rational Rose Enterprise Edition; 接下来选择 Rational Rose Enterprise Edition 类型。再单击“下一步”按钮, 然后选择 Desktop installation from CD 选项, 表示创建一个本地的应用程序而不是网络的; 然后基本上一路回车下来即可。

6. 如何使用 Rational Rose 模型的导出和导入功能? 【课本 269】

【答】导入模型及模型元素, 通过选择 File→Import Model 命令可以导入模型、包或类等, 可供选择的文件类型包括“\*.mdl”、“\*.ptl”、“\*.sub”或“\*.cat”等。Rose 会将导入的元素和当前模型中的相关元素进行比较, 提示是否要用导入的元素取代当前模型中的元素。导入元素之后, Rose 会更新当前模型中的所有模型图。导出模型及模型元素。可以通过选择 File→Export Model 命令导出模型, 导出文件的后缀名为“\*.ptl”。

7. 说出 Rational Rose 操作界面由哪几个部分组成以及各个部分的作用。【课本 266-267】

【答】Rose 的界面有菜单、浏览区、文档窗口、工具栏、状态栏、框图窗口和日志窗口等。

(1) 菜单栏。在菜单栏中包含了所有在 Rational Rose 2003 中可以进行的操作, 一级菜单共有 11 项, 分别是 File (文件)、Edit (编辑)、View (视图)、Format (格式)、Browse (浏览)、Report (报告)、Query (查询)、Tools (工具)、Add-Ins (插件)、Window (窗口) 和 Help (帮助), 打开某个一级菜单后, 就进入二级菜单。

(2) 工具栏。在 Rational Rose 2003 中工具栏的形式有两种, 分别是: 标准工具栏和编辑区工具栏。标准工具栏在任何图中都可以使用, 因此在任何图中都会显示。编辑区工具栏是根据不同的图形而设置的具有绘制不同图形元素内容的工具栏, 显示时位于图形编辑区的左侧。

(3) 浏览区和视图。Rose 浏览区是一种树型的层次结构, 可以迅速查找到各种图或者模型元素。浏览区描述了视图模型, 并且提供了在每一种视图的组件间进行访问的功能。“+”表示该图标为折叠图,“-”表示该图标已被完全展开。在浏览区中默认创建了 4 个视图, 分别是 Use Case View (用例视图)、Logical View (逻辑视图)、Component View (构件视图) 和 Deployment View (部署视图)。在这些视图所在的包或者图下, 可以创建不同的模型元素。右击每个视图, 选择“New”就可以看到这个视图所包含的一些模型元素。

(4) 框图窗口。在图的编辑区域中可以根据图形工具栏中的图形元素内容绘制相关信息。在图的编辑区域添加的相关模型元素会自动地在浏览区中添加, 从而使浏览区和编辑区的信息保持同步, 也可以将浏览区中的模型元素拖动到图形编辑区中进行添加。

(5) 文档窗口。文档窗口用于对 Rational Rose 2003 中所创建的图或模型元素进行说明, 如当对某一

个图进行详细说明时，可以将该图的作用和范围等信息置于文档窗口，那么在浏览或选中该图时就会看到该图的说明信息，模型元素的文档信息也相同。文档窗口为所选择的项和图形提供建立、浏览或修改文档的能力。当不同的选项和图形被选择时，仅允许一个文档窗口被更新。文档窗口还可以被隐藏。

（6）日志窗口。在日志窗口中记录了对模型的一些重要操作，用于提示用户。显示信息包括系统错误、用户操作记录等。

#### 8. 请说明如何实现自定义工具？【课本 267-268】

【答】选择 Tools→Options 命令，弹出 Options 对话框，打开 Toolbars（工具栏）选项卡。在 Standard toolbar 选项组中可以选择显示或隐藏标准工具栏，或者设置工具栏中的选项是否使用大图标。在 Diagram toolbar 选项组中可以选择是否显示编辑区工具栏，以及编辑区工具栏的显示样式，如是否使用大图标或小图标、是否自动显示或锁定等。在 Customize toolbars（定制工具栏）选项组中可以根据具体情况定制标准工具栏和图形编辑工具栏的详细信息。定制标准工具栏时，可以单击位于 Standard（标准）选项右侧的按钮，弹出对话框。在该对话框中可以将左侧的选项添加到右侧的列表框中，这样在标准工具栏中就会显示，当然也可以通过这种方式删除标准工具栏中不用的信息。

## 第 9 章 复习思考题

### 一、选择题

1. 软件复用的形式有（ABCD）。  
A. 代码复用      B. 分析模式复用      C. 设计模式复用      D. 测试信息复用
2. 面向对象技术为软件复用提供了（AB）。  
A. 概念上的支持      B. 方法上的支持      C. 语义上的支持      D. 数据方面支持
3. 面向对象编程重用属于（A）。  
A. 白盒复用      B. 黑盒复用      C. 语义复用      D. 信息复用
4. 软件复用的核心技术是（ABCD）。  
A. 软件构件技术      B. 领域工程      C. 软件架构      D. 开放系统技术
5. 领域工程包括三个阶段是（ABC）。  
A. 领域分析      B. 领域设计      C. 领域实现      D. 领域测试
6. 重用策略有（ABC）。  
A. 界面重用      B. 逻辑层包装原则      C. 数据层重用      D. 物理层重用

### 二、填空题

1. 软件复用的级别分为代码构建复用、设计结果复用、分析结果复用、测试信息复用。
2. 软件复用类型有横向复用、纵向复用。
3. 软件复用的技术有组装技术、生成技术和面向对象技术。
4. 面向对象程序的基本特征是：抽象、继承、封装和多态。
5. 面向对象的三要素是封装、继承和多态。

### 三、名词解释

软件复用：软件复用是指在构造新的软件系统的过程中，对已存在的软件产品(设计结构、源代码、文档等) 重复使用的技术，以缩减软件开发和维护的花费。

再工程：软件再工程是指对既存对象系统进行调查，并将其重构为新形式代码的开发过程。

架构：软件架构是软件产品、软件系统设计当中的主体结构 and 主要矛盾，软件架构是一系列相关的抽象模式，用于指导大型软件系统各个方面的设计。软件架构是一个系统的草图。

可复用构件：可以被复用的软件成分一般称作可复用构件，无论对可复用构件原封不动地使用还是作

适当的修改后再使用，只要是用来构造新软件，则都可称作复用。

#### 四、综合题

1. 软件复用有哪些优点？软件复用技术有哪些？

【答】软件复用的优点有：在软件开发中，软件复用就是充分利用已有的各项成果，避免重复劳动，增加技术积累，提高软件水平，加快开发速度，保证软件开发质量。其优点主要有以下几方面：(1)提高生产率。(2)减少维护代价。(3)提高互操作性。(4)支持快速原型。(5)减少培训开销。

软件复用技术有：(1)软件构件技术；(2)软件构架；(3)领域工程；(4)软件再工程；(5)开放系统技术；(6)CASE 技术；(7)软件过程；(8)非技术因素；

2. 软件复用的过程管理包括哪些内容？【课本 307】

【答】软件复用的过程管理(1) 构件分析与设计管理。① 领域分析和需求分析。② 建立系统模型。③ 划分构件。(2) 构件实现及局部测试管理。构件实现主要指构件的代码实现和编译封装，用软件开发工具将设计好的构件转化为可用的软件构件。(3) 基于构件的应用程序组装管理。(4) 应用系统整体测试管理。安装好构件才能运行程序，才可以进行测试。

3. 简述软件再工程的过程模型。【课本 324】

【答】软件再工程是一个工程过程，它将逆向工程、重构和正向工程组合起来，将现存软件系统重新构造为新的系统。典型的软件再工程过程模型定义了 6 类活动，如图 12 所示。

4. 实施软件再工程有什么好处？【课本 324】

【答】实施软件再工程具有如下的好处：

- (1) 再工程可帮助软件机构降低软件演化的风险。
- (2) 再工程可帮助机构补偿软件投资。
- (3) 再工程可使得软件易于进一步变革。
- (4) 软件再工程有着广阔的市场。
- (5) 再工程能扩大 CASE 工具集（如 Aidedsoft）。
- (6) 再工程是推动自动软件维护发展的动力。

5. 传统的软件再工程与面向对象的软件再工程有什么区别？【课本 321-323】

【答】对于结构化的分析和设计方法指导下的软件系统，再工程的活动主要包括数据处理环境分析、数据字典分析和程序分析，依次是对数据处理的环境（包括软件和硬件环境）、数据项的意义和标准统一、程序的静态统计、动态执行效果进行分析。在分析部分可以使用“库”的概念，作为分析结果的存储。在结构化方法软件再工程模型中，做逆向工程的时候可以采用两种方法，即代码切分和代码重复。代码切分（又为代码分离）是从程序中抽取出完成每个功能所涉及到的尽量少的代码集，通过不断地发现功能，抽取相应的代码集来达到对源代码的理解。而代码重复是通过发现及分析源代码中的代码部分来逐渐深入理解系统源代码，可以考察重复的内容并单独提出作为一个功能部分或者是目标系统的一个可重用的部分，如图 13 所示。

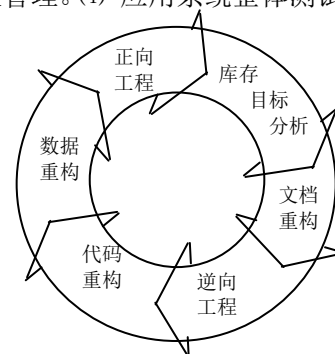


图 12 软件再工程过程

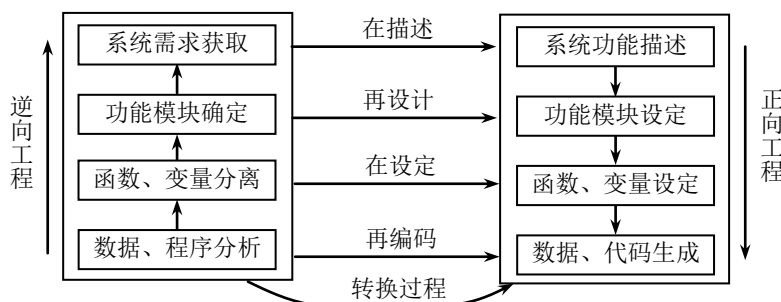


图 13 结构化软件再工程模型

对于一个面向对象方法实现的系统，实施软件再工程首先要进行功能分析、功能层次、功能需求的获取，在类结构图中进行定位。从现存系统的运行过程中发现系统的具体功能，同时将功能集中到某部分代

码集合上。然后还可以采用代码调试的方式分析系统中重要代码的功能，将固定功能的代码分离出来。此外，除了对源代码进行分析外，软件设计记录以及其他文档资源一样要分析，以得出原软件系统的整体设计视图。总体来说，对于面向对象的软件系统，软件再工程的模型可以如图 14 所示。

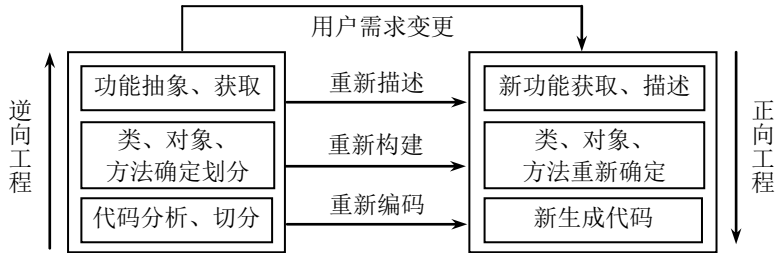


图 14 面向对象的软件再工程模型

两者具有如下的不同：(1)过程不同，见图 13 和图 14。(2)处理对象不同。传统软件再工程是面向过程的，即结构化的编程方法。面向对象的软件再工程是面向对象的，面对问题域中问题的客观存在。(3)开发效率不同。面对日益复杂的软件系统，传统的软件再工程不再具有优势。面向对象的软件再工程，它是能够适应复杂系统开发的软件再工程方法论和软件开发技术。无论是软件开发阶段的开发效率还是软件维护阶段的系统维护成本，面向对象的方法都远远优于传统的软件再工程方法。