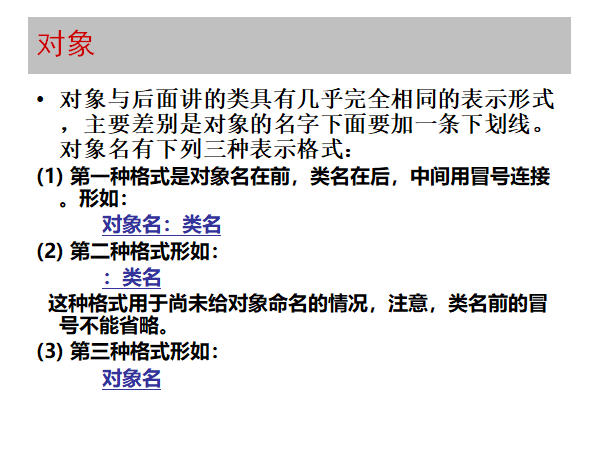
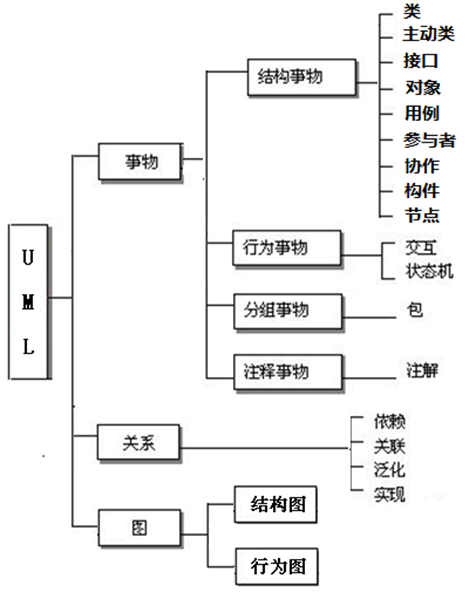
# 软件工程概论5-6

1. 五、面型对象方法和UML
   1. 1.面向对象的概念
      1. 面向对象＝对象＋类＋继承＋消息通信，如果一个系统是使用这样4个概念设计和实现的，则可认为这个系统是面向对象的。
      2. 对象
         1. 对象是包含现实世界物体特征的抽象实体，它反映了系统为之保存信息和（或）与它交互的能力。



* + - 1. 对象有两个层次的概念：
         1. (1) 现实生活中对象指的是客观世界的实体。可以是可见的有形对象，如人、学生、汽车、房屋等；也可以是抽象的逻辑对象，如银行帐号，生日。
         2. (2) 程序中对象就是一组变量和相关方法的集合，其中变量表明对象的状态，方法表明对象所具有的行为。
      2. 可以将程序中的对象分为5类：物理对象，角色，事件，交互，规格说明。
         1. (1)物理对象（Physical Objects）── 物理对象是最易识别的对象，通常可以在问题领域的描述中找到，它们的属性可以标识和测量。
         2. (2) 角色（Roles）── 一个实体的角色也可以抽象成一个单独的对象。角色对象的操作是由角色提供的技能。
         3. (3) 事件（Events）── 一个事件是某种活动的一次“出现”。
         4. (4) 交互（Interactions）── 交互表示了在两个对象之间的关系，这种类型的对象类似于在数据库设计时所涉及的“关系”实体。
    1. 类:
       1. 可以将现实生活中的对象经过抽象，映射为程序中的对象。对象在程序中是通过一种抽象数据类型来描述的，这种抽象数据类型称为类（Class）。
    2. 封装:
       1. 面向对象的封装特性与其抽象特性密切相关。封装是一种信息隐蔽技术，就是利用抽象数据类型将数据和基于数据的操作封装在一起。用户只能看到对象的封装界面信息，对象的内部细节对用户是隐蔽的。
    3. 继承
       1. 继承是一种联结类的层次模型，为类的重用提供了方便，它提供了明确表述不同类之间共性的方法。
       2. 我们将公共类称为超类(superclass)、父类（father class）、祖先（ancestor）或基类（base class），而从其继承的类称为子类(subclasses)、后代（deslendane）或导出类（derived class）。
    4. 面向对象软件开发方法的特征
       1. 方法的唯一性即方法是对软件开发过程所有阶段进行综合考虑而得到的。
       2. 从生存期的一个阶段到下一个阶段的高度连续性，即生存期后一阶段的成果只是在前一阶段成果的补充和修改。
       3. 将面向对象分析(OOA)、面向对象设计(OOD)和面向对象程序设计(OOP)集成到生存期的相应阶段。
  1. 2.UML
     1. UML的基本模型

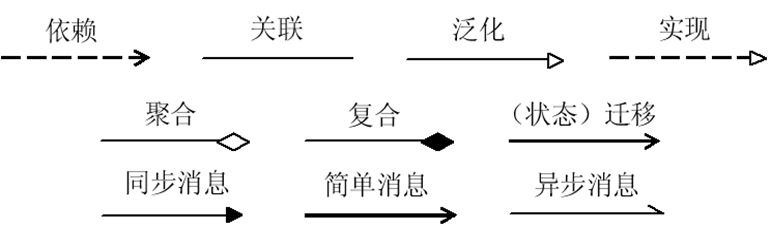


* + 1. UML的事物
       1. 事物是对模型中最具代表性成分的抽象，在UML中，可以分为结构事物、行为事物、分组事物和注释事物4类。
       2. 结构事物:
          1. 结构事物是UML模型的静态部分，主要用来描述概念的或物理的元素，包括类、主动类、接口、对象、用例、参与者、协作、构件和节点等。
       3. 行为事物:
          1. 行为事物是UML模型的动态部分，包括两类：

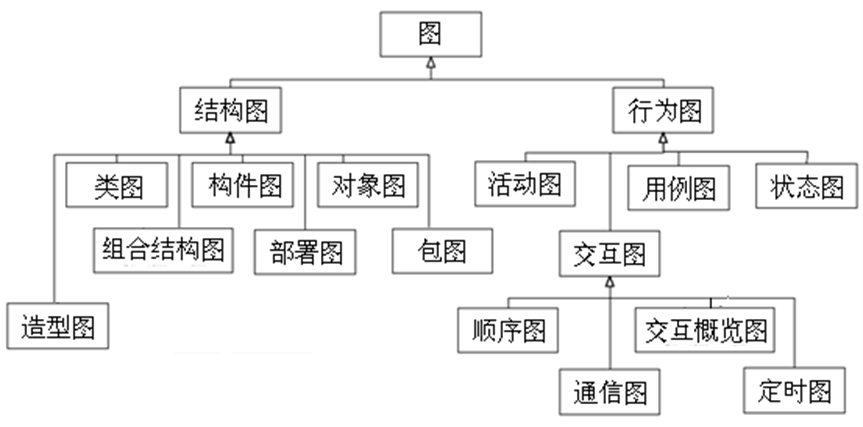
（1）交互（interaction）── 交互由在特定的上下文环境中共同完成一定任务的一组对象之间传递的消息组成。如图所示。交互涉及的元素包括消息、动作序列（由一个消息所引起的行为）和链（对象间的连接）。

（2）状态机（state machine）── 描述了一个对象或一个交互在生存周期内响应事件所经历的状态序列，单个类或者一组类之间协作的行为都可以用状态机来描述。 状态机涉及到状态、变迁和活动，其中状态用圆角矩形来表示。

* + - 1. 分组事物
         1. 分组事物是UML模型的组织部分。它的作用是为了降低模型复杂性。UML中的分组事物是包（package）。包是把模型元素组织成组的机制，结构事物、行为事物甚至其他分组事物都可以放进包内。
      2. 注释事物
         1. 注释事物是UML模型的解释部分，它们用来描述和标注模型的任何元素。通常可以用注释修饰带有约束或者解释的图。
    1. UML的关系
       1. 画图



* + - 1. 依赖关系
         1. 依赖(Dependency)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到另一个事物的语 义，它用一个虚线箭头表示。
         2. 虚线箭头的方向从源事物指向目标事物，表示源事物依赖于目标事物。
      2. 关联关系
         1. 关联(association)是一种结构关系，它描述了两个或多个类的实例之间的连接关系，是一种特殊的依赖。
         2. 关联分为普通关联（**二元关联和多元关联**）、限定关联、关联类，以及聚合与复合。
      3. 泛化关系
         1. 泛化(generalization)关系就是一般类和特殊类之间的继承关系。
         2. 在UML中，一般类亦称泛化类，特殊类亦称特化类。
         3. 泛化针对类型而不针对实例，因为一个类可以继承另一个类，但一个对象不能继承另一个对象。
         4. 泛化可进一步划分成普通泛化和受限泛化（交叠、不相交、完全和不完全 ）两类。、
      4. 实现关系
         1. 实现(implement)是泛化关系和依赖关系的结合，也是类之间的语义关系，通常在以下两种情况出现实现关系：
         2. (1)接口和实现它们的类或构件之间；
         3. (2)用例和实现它们的协作之间。
    1. UML的图
       1. 图



* + - 1. 用例图
         1. 在UML中，一个用例模型由若干个用例图来描述，用例图的主要元素是用例和执行者。用例图是包括执行者、由系统边界（一个矩形）封闭的一组用例，执行者和用例之间的关联、用例间关系以及执行者的泛化的图。
         2. 用例之间可以有泛化、扩展、使用（包含）三种关系。

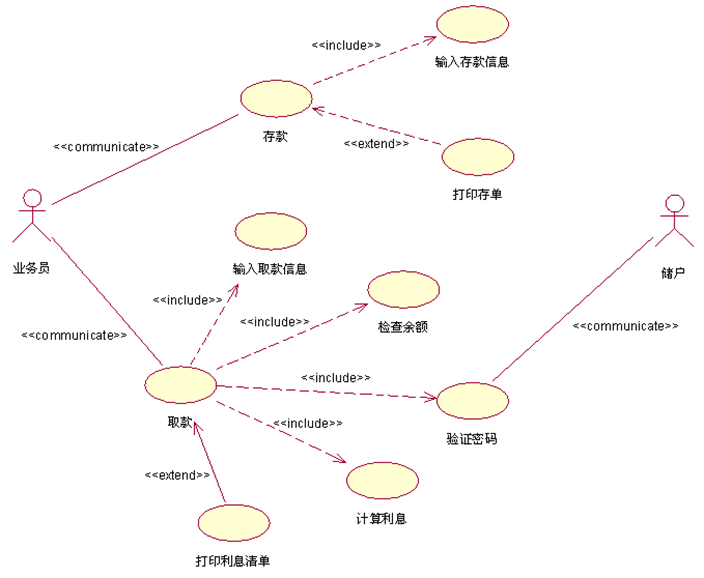
(1) 泛化关系：用例泛化是指一个用例可以被特别列举为一个或多个子用例。

(2) 扩展关系：向一个用例中加入一些新的动作后构成了另一个用例，这两个用例之间的关系就是扩展关系，后者通过继承前者的一些行为得来，通常把后者称为扩展用例。

(3) 使用（包含）关系：当一个用例使用另一个用例时，这两个用例之间就构成了使用关系。当有一大块相似的动作存在于几个用例，又不想重复描述该动作，将重复的部分分离为一个用例，两用例间关系称为使用关系。

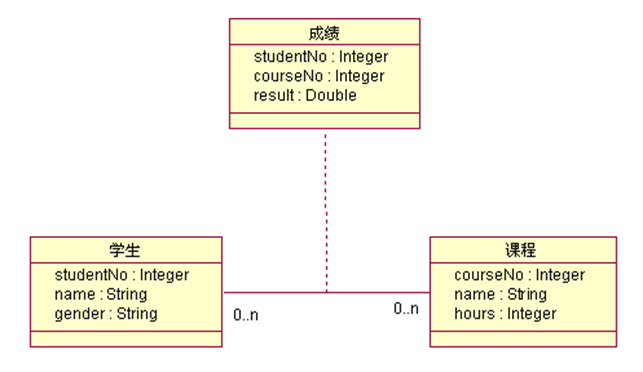
* + - * 1. 例子

银行储蓄系统的用例图



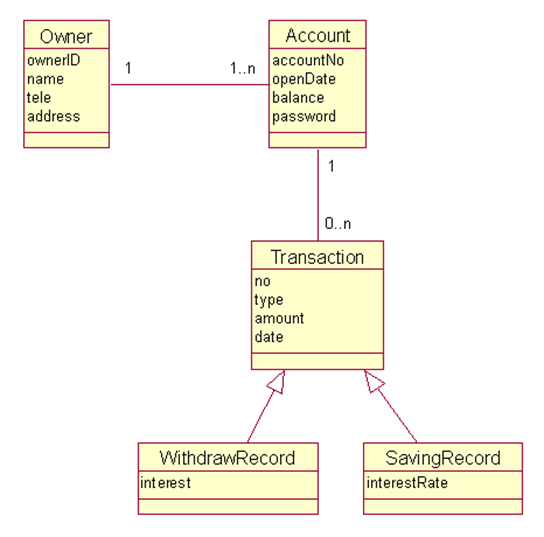
* + - 1. 类图
         1. 类图描述类和类与类之间的静态关系，它是从静态角度表示系统的，因此类图属于一种静态模型。类图是构建其他图的基础，没有类图就没有状态图、协作图等其他图，也就无法表示系统其他方面的特性。
         2. 类图显示了类（及其接口）、类的内部结构以及与其他类的联系。联系是指类元之间的联系，在类的建模中可以使用关联、聚合和泛化（继承）关系。
         3. 关联类

关联类是指表示其他类之间关联关系的类。当一个关联具有自己的属性并需要存储它们时，就需要用关联类建模。关联类用虚线连接在两个类之间的联系上

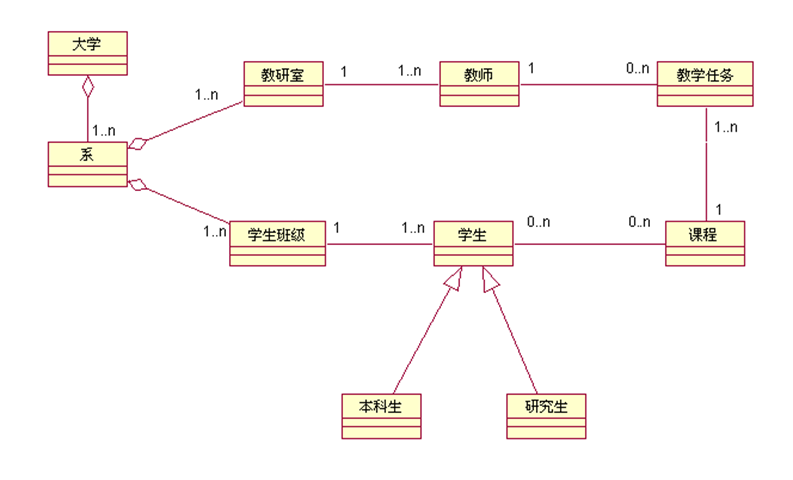


* + - * 1. 例子

银行储蓄系统的核心类图

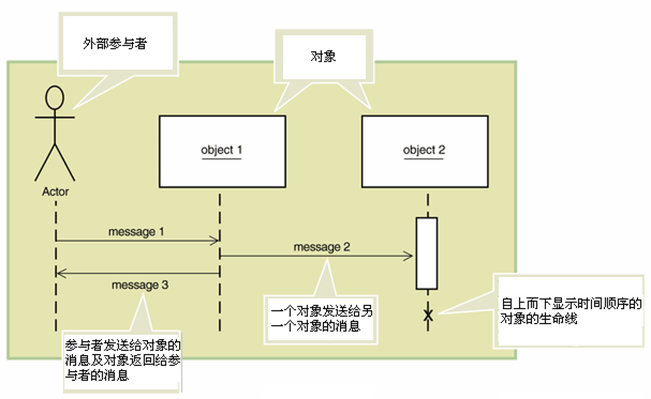


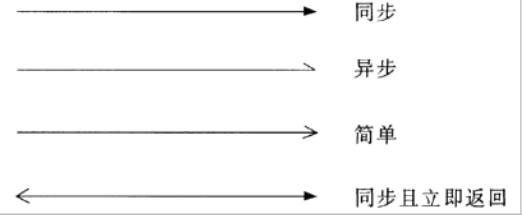
教学管理系统的类图



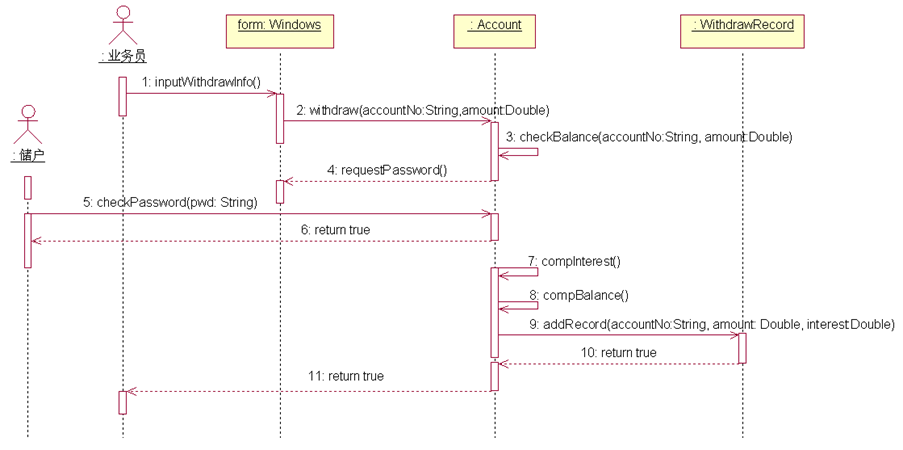
* + - 1. 交互图
         1. UML中有两种类型的交互图：顺序图和协作图。
         2. 顺序图

顺序图描述对象之间的动态交互关系，着重表现对象间消息传递的时间顺序。顺序图中的符号如下:



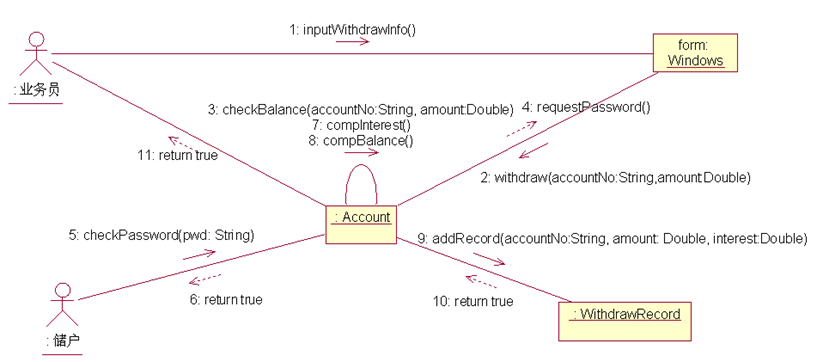


取款用例的顺序图

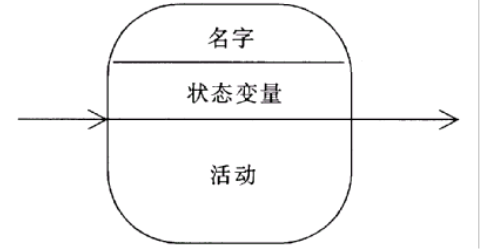


* + - * 1. 通信图

通信图是顺序图的一种变化形式，用于描述相互协作的对象间的交互关系和链接关系。

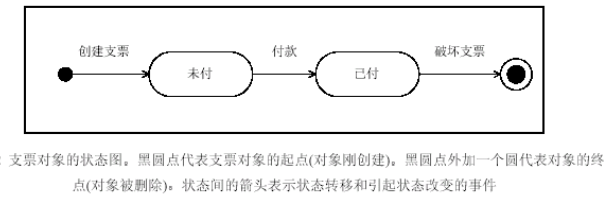


* + - 1. 状态图
         1. 状态图描述一个特定对象的所有可能的状态以及引起状态转换的事件。大多数面向对象技术都用状态图表示单个对象在其生命期中的行为。一个状态图包括一系列状态、事件以及状态之间的转移。
         2. 所有对象都具有状态，状态是对象执行了一系列活动的结果。当某个事件发生后，对象的状态将发生变化。在状态图中定义的状态可能有：初态(初始状态)、终态(最终状态)、中间状态和复合状态。
         3. 在一张状态图中只能有一个初态，而终态则可以有多个。
         4. 中间状态用圆角矩形表示，可能包含三个部分，第一部分为状态的名称；第二部分为状态变量的名字和值，这部分是可选的；第三部分是活动表，这部分也是可选的。

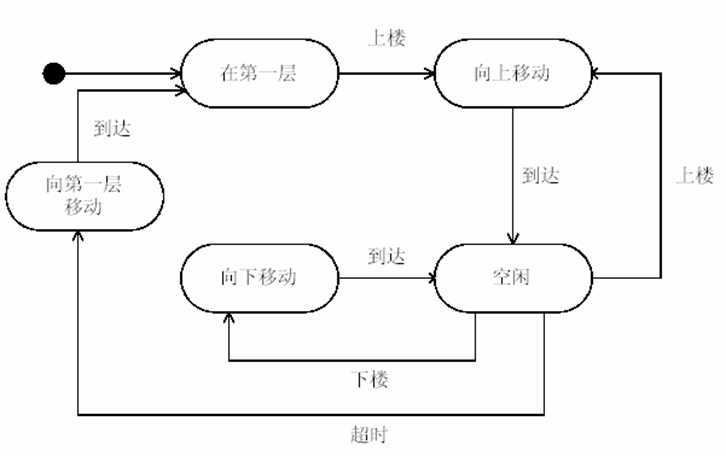


* + - * 1. 例子

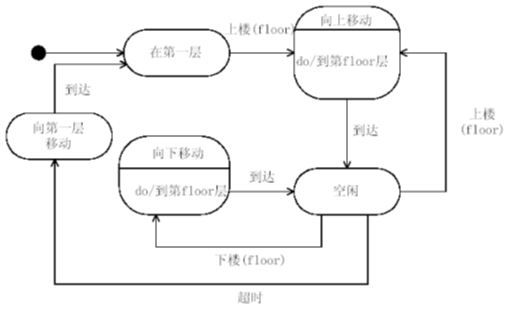
支票对象的状态图



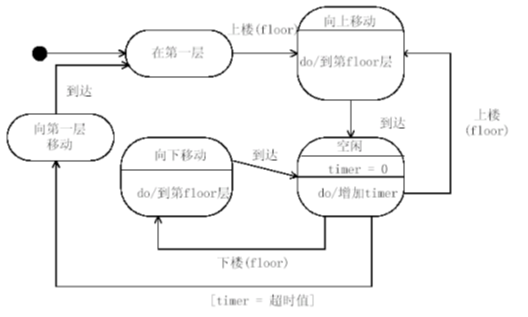
电梯的状态图(本状态图没有终点)



带有事件说明的状态转换

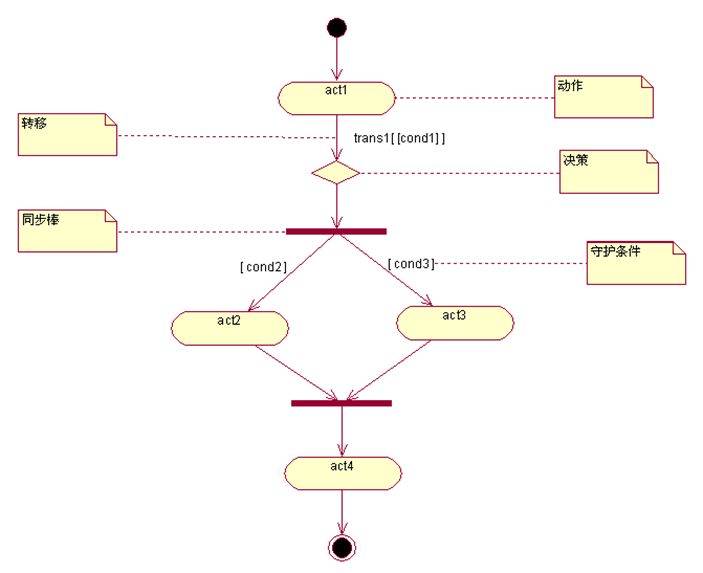


加上属性的状态转换

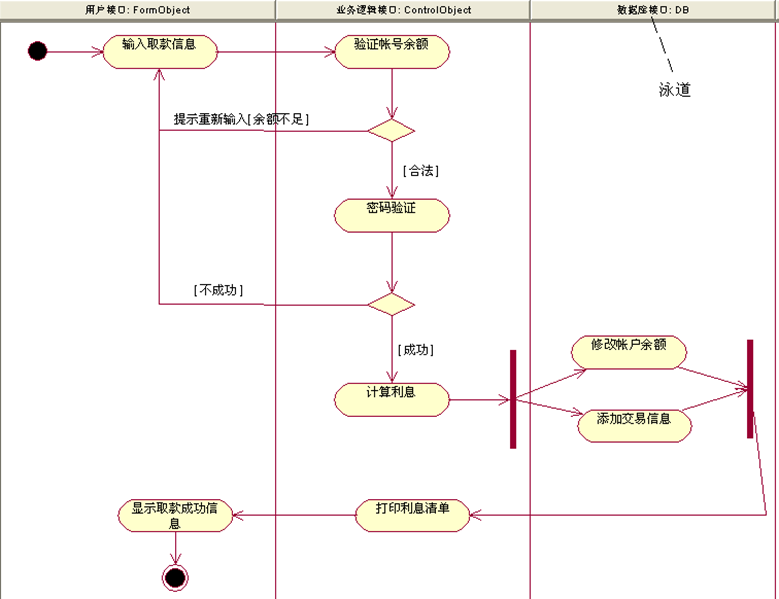


* + - 1. 活动图
         1. 活动图用来捕捉用例的活动，使用框图的方式显示动作及其结果。活动图是一个流图，描述了从活动到活动的流。它是另一种描述交互的方式，它描述采取何种动作，动作的结果是什么(动作状态改变)，何时发生(动作序列)，以及在何处发生(泳道)。
         2. 例子

活动图中的符号



取款用例的活动图



* + - 1. 构件图
         1. 构件图描述软件构件及构件之间的依赖关系，显示代码的静态结构。
         2. 构件是逻辑架构中定义的概念和功能(例如，类、对象及它们之间的关系)在物理架构中的实现。典型情况下，构件是开发环境中的实现文件。
         3. 软件构件可以是下述的任何一种构件。

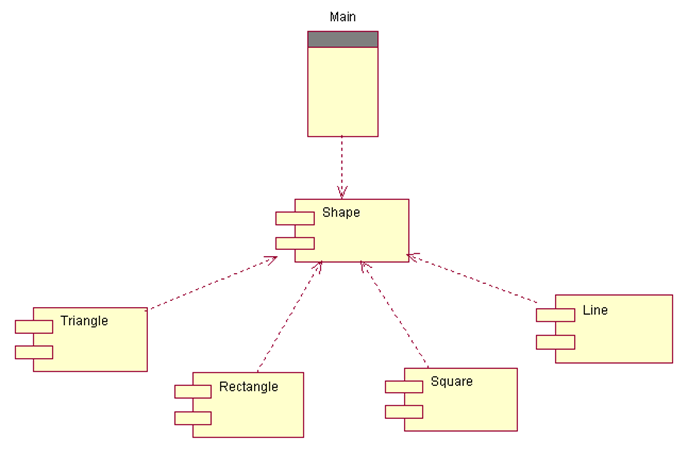
源构件：源构件仅在编译时才有意义。典型情况下，它是实现一个或多个类的源代码文件。

二进制构件：典型情况下，二进制构件是对象代码，它是源构件的编译结果。

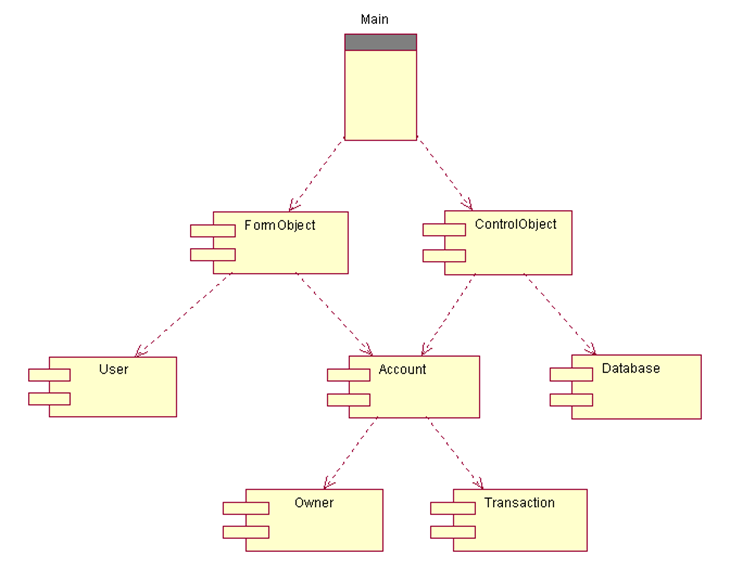
可执行构件：可执行构件是一个可执行的程序文件，它是链接所有二进制构件所得到的结果。一个可执行构件代表在处理器(计算机)上运行的可执行单元。

* + - * 1. 例子

画图系统的构件图

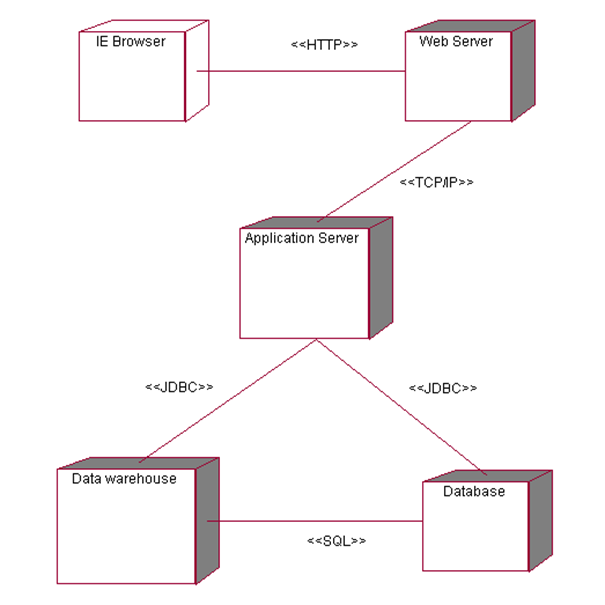


银行储蓄系统的构件图

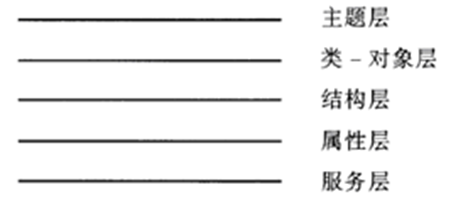


* + - 1. 部署图
         1. 部署图描述处理器、设备和连接，它显示系统硬件的物理拓扑结构及在此结构上执行的软件。
         2. 部署图可以显示计算节点的拓扑结构和通信路径、节点上运行的软件以及软件包含的逻辑单元。
         3. 例子

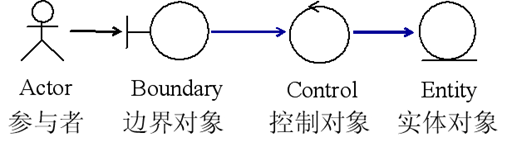
典型的部署图



1. 六、面向对象分析
   1. 1.面向对象分析概述
      1. 在结构化的分析方法中，主要采用数据流图对功能进行建模，建模时首先要建立顶层数据流图来确定系统在环境中的位置及系统的边界。在进行面向对象系统分析时，也同样需要建立环境图来确定系统边界。
      2. 面向对象分析的3种模型
         1. 用例模型：用例和场景表示的功能模型；
         2. 对象模型：用类和对象表示的静态模型；
         3. 交互模型：由状态图和顺序图表示的动态模型。
   2. 2.建立用例模型
      1. 1. 确定业务参与者
      2. 2. 确定业务需求用例
      3. 3. 创建用例图
   3. 3.建立对象模型
      1. 复杂问题(大型系统)的对象模型应该由下述5个层次组成：主题层(也称为范畴层)、类-对象层、结构层、属性层和服务层，如图所示。



* + 1. 确定类与对象
       1. 1. 找出候选的类与对象
          1. 类与对象是对问题域中有意义的事物的抽象，它们既可能是可见的物理实体，也可能是抽象的概念。我们可以将客观事物分为以下五类：
          2. 可感知的物理实体，如教学楼、教室等。
          3. 人或组织的角色，如教师、计算机系等。
          4. 应该记忆的事件，如演出、交通事故等。
          5. 两个或多个对象的相互作用，通常带有交易或接触的性质，如购买、教学等。
          6. 需要说明的概念，如保险法、政策等。
       2. 2. 筛选出正确的类与对象
       3. 3.区分实体类、边界类和控制类
          1. 实体类表示系统将跟踪的持久信息；边界类表示参与者与系统之间的交互；控制类负责用例的实现。其图形表示如图所示



* + 1. 确定结构
       1. 1. 确定泛化（继承）关系
       2. 2. 确定关联关系
    2. 确定属性
    3. 确定服务
       1. 1) 简单的服务。即每一个对象都应具备的服务，这些服务包括：建立和初始化一个新对象，建立或切断对象之间的关联，存取对象的属性值，释放或删除一个对象。这些服务在分析时是隐含的，在图中不标出，但实现类和对象时有定义。
       2. (2) 复杂的服务。它分为两种：
          1. 计算服务：利用对象的属性值计算，以实现某种功能；
          2. 监控服务：处理对外部系统的输入∕输出，外部设备的控制和数据的存取。
  1. 4.建立动态模型
     1. 在UML中动态模型的描述工具有顺序图、通信图和状态图。