### 对象图

什么是对象

对象指的是一个单独的、可确认的物体、单元或实体，它可以是具体的也可以是抽象的，在问题领域里有确切定义的角色。换句话说，对象是边界非常清楚的任何事物。

一个对象通常包含以下几部分：

标识（名字）：为了将一个对象与其他的对象区分开，通常会给对象起一个“标识”，也就是“对象名”。

状态（属性）：对象的状态包括对象的所有属性（通常是静态的）和这些属性的当前值（通常是动态的）。

行为（方法，事件）：没有一个对象是孤立存在的，对象可以被操作，也可以操作别的对象。而行为就是一个对象根据它的状态改变和消息传送所采取的行动和所做出的反应。

对象图的定义

对象图描述的是参与交互的各个对象在交互过程中某一时刻的状态。对象图可以被看作是类图在某一时刻的实例。在UML中，对象图使用的是与类图相同的符号和关系，因为对象就是类的实例。

对象图主要包括以下几个部分：

对象名：由于对象是一个类的实例，因此其名称的格式是“对象名：类名”，这两个部分是可选的，但如果是包含类名，则必须加上“：”，另外为了和类名区分，还必须加上下划线。

属性：由于对象是一个具体的事物，因此所有的属性值都已经确定，因此通常会在属性的后面列出其值。

### 构件图

构件图的定义

构件图是对面向对象系统的物理方面建模时使用的两种图之一（另一种是部署图），用于描述软件组件及组件之间的组织和依赖关系。构件图通过对组件间依赖关系的描述来估计对系统组件的修改给系统可能带来的影响。构件图用于描述系统中软件的构成，但没有描述系统中与硬件有关的构成情况。

构件图有利于：

1. 帮助客户理解最终的系统结构。

（2）使开发工作有一个明确的目标。

（3）帮助开发组的其他人员理解系统。

（4）复用软件组件。

构件图的组成元素

构件图的组成元素包括组件、接口、关系，还可以包括包和子系统。

组件

组件是系统中遵从一组接口且提供实现的一个物理部件，通常指开发和运行时类的物理实现。组件常用于对可分配的物理单元建模，这些物理单元包含模型元素，并具有身份标识和明确定义的接口，其具有很广泛的定义，以下的一些内容都可被认为是组件：程序源代码、子系统、动态链接库等。组件的图形表示法是把组件画成带有两个标签的矩形。每一个组件都必须有一个唯一的名称。

组件的类型

组件可以分为以下三种类型。

1. 实施组件。

实施组件是构成一个可执行系统必要和充分的组件，如动态链接库（Dynamic Link Library，DLL）、二进制可执行体（EXE）、ActiveX控件和JavaBean组件等。

1. 工作产品组件。

这类组件主要是开发过程的产物，包括创建实施组件的源代码文件及数据文件，这些组件并不是直接地参加可执行系统，而是开发过程中的工作产品，用于产生可执行系统。

1. 执行组件。

这类组件是作为一个正在执行的系统的结果而被创建的。

接口

组件的接口可以分为两种类型。

1. 导出接口：即为其他组件提供服务的接口，一个组件可以有多个导出接口。
2. 导入接口：即在组件中所用到的其他组件所提供的接口，一个组件可以使用多个导入接口。

组件和组件的接口可以有两种表示法。

1. 将接口用一个矩形来表示，矩形中包含与接口有关的信息。接口与实现接口的组件之间用一条带空心三角形箭头的虚线表示，箭头指向接口。

（图）

1. 用一个小圆圈来代表接口，用实线和组件连接起来。实线代表的是实现关系。

关系

在构件图中使用最多的是依赖和实现关系。

依赖关系是指组件依赖外部提供的服务（由组件到接口）。构件图中的依赖关系使用虚线箭头表示。

（图）

实现关系是指组件向外提供的服务。实现关系使用实线表示，实现关系多用于组件和接口之间。组件可以实现接口。

（图）

构件图建模及绘图的步骤

使用构件图建模可按照下列步骤进行：

（1）对系统中的组件建模。

（2）定义相关组件提供的接口。

（3）对它们间的关系建模。

（4）对建模的结果进行精华和细化。

构件图的几种使用方式

构件图用于对系统的静态实现视图建模，这种视图主要支持系统部件的配置管理。通常可以按下列4种方式之一来使用构件图。

1. 对源代码建模

采用当前大多数面向对象编程语言，将使用集成化开发环境来分割代码，并将源代码存储到文件中。可以使用构件图来为这些文件的配置建模，并设置配置管理系统。

1. 对可执行体的发布建模

软件的发布是交付给内部或外部用户的相对完整而且一致的组件系列。在组件的语境中，一个发布注重交付一个运行系统所必需的部分。当用构件图对发布建模时，其实是在对构成软件的物理部分所做的决策进行可视化、详述和文档化。

1. 对物理数据库建模

可以把物理数据库看作模式在比特世界中的具体实现。实际上，模式提供了对永久信息的应用程序编程接口，物理数据库模型表示了这些信息在关系型数据库的表中或者在面向对象数据库的页中的存储。可以用构件图表示这些以及其他种类的物理数据库。

1. 对可适应的系统建模

某些系统是相对静态的，其组件进入现场、参与执行、然后离开。另外一些系统则是较为动态的，其中，包括一些为了负载均衡和故障恢复而进行迁移的可移动的代理或组件。可以将构件图与对行为建模的UML的一些图结合起来表示这类系统。

### 包图

包

包是一种把元素组织到一起的通用机制，包可以嵌套于其他包中。包的图标是一个带标签的文件夹。包是一个命名空间，也是一个元素。可以包含在其他命名空间中。包可以拥有其他包或与其他包合并，它的元素可以导入包命名空间中。

包图

包图用于描述包与包之间的关系，包图描绘模型元素在包内的组织和依赖关系，包括包的导入和包扩展。它们还提供相应命名空间的可视化。

包之间的关系

1. 引入关系

一个包中的类可以被另一个指定包（以及嵌套于其中的那些包）中的类引用。

引入关系是依赖关系的一种，需要在依赖线上增加一个<<import>>衍型，包之间的一般依赖关系都属于引入关系。

（图）

1. 泛化关系

表示一个包继承了另一个包的全部内容，同时又补充自己增加的内容。

（图）

1. 嵌套关系

一个包中可以包含若干个子包，构成包的嵌套层次结构。

（图）

包图的建模技巧

（1）两种组包方式：

根据系统分层架构组包。

根据系统业务功能模块组包。

（2）参照类之间的关系确定包之间的关系。

1. 减少包的嵌套层次，一般不超过三层。
2. 每个包的子包控制在5~9个。
3. 如果几个包有若干相同组成部分，可优先考虑将它们合并。
4. 可通过包图来体现系统的分层架构。