灰色字部分可以不做进PPT

1.3UML的特点

标准建模语言UML的主要特点可以归结为以下三点。

1. UML统一了 Booch、OMT和OOSE等方法中的基本概念和符号。

注：

Booch方法是早期面向对象的软件开发方法的一种，Booch认为软件开发是一个螺旋上升的过程，每个周期包括4个步骤，分别是标识类和对象、确定类和对象的含义、标识关系、说明每个类的[接口](https://baike.baidu.com/item/%E6%8E%A5%E5%8F%A3/2886384)和实现。

OMT（Object Modeling Technique）对象建模技术。它从三个方面对系统进行建模，每个模型从一个侧面反映系统的特性，三个模型分别是：对象模型、[动态模型](https://baike.baidu.com/item/%E5%8A%A8%E6%80%81%E6%A8%A1%E5%9E%8B/888302)和[功能模型](https://baike.baidu.com/item/%E5%8A%9F%E8%83%BD%E6%A8%A1%E5%9E%8B/563406)。

OOSE（Object-oriented software engineering）即面向对象的软件工程，这是一种在OMT Object Modeling Technology的基础上用于对功能模型进行补充指导系统开发活动的系统方法。

1. UML吸取了面向对象领域中各种优秀的思想，其中也包括非OO方法的影响。

注：UML符号表示考虑了各种方法的图形表示，删掉了很多容易引起混乱的、多余的和极少使用的符号，同时添加了一些新符号。因此，在UML中凝聚了面向对象领域中很多人的思想。这些思想并不是UML的开发者们发明的，而是开发者们依据最优秀的OO方法和丰富的计算机科学实践经验综合提炼而成的。

1. UML在演变过程中还提出了一些新的概念。

在UML标准中新加了模板、职责、扩展机制、线程、过程、分布式、并发、模式、 合作、活动图等新概念，并清晰地区分类型*、类*和实例、细化、接口和组件概念。

因此可以认为，UML是一种先进实用的标准建模语言，但其中某些概念尚待实践来验证,UML也必然存在一个进化过程。

1.7UML2.0新特性

统一建模语言UML是以可视化方式描述软件系统的结构和行为的标准语言。 UML2. 0在可视化建模方面进行了许多改革和创新。它可以描述现今软件系统中存在的许多技术，例如模型驱动架构（MDA）和面向服务的架构（SOA）。UML2.0完全建立在UML1.x基础之上，大多数的UML1.x模型在UML2.0中都可以用。但UML2.0在用例图、顺序图、活动图和构件图都有所改进，特别是改善了结构建模的性能。

下面主要针对 UML2. 0上层的变化进行简要说明。

1. 用例图

用例图中的主体内容用例、参与者、通信关联并没有变化。如果用UML1. x,只能用用例图所归属的包来表达一组用例的逻辑组织关系.即用用例在模型中所处的物理位置表达 逻辑组织关系。在UML2. 0中，为每个用例增加了一个称为Subject的特征，这项特征的取 值可以作为在逻辑层面划分一组用例的一项依据。用例所属的“系统边界”就是Subject的 一种典型例子。

用例图的基本元素主要有主题( subject）、一组参与者、一组用例以及它们之间的关系。主题可以是系统、子系统、类或接口。如下图所示，主题表示为一个矩形，其中包含一组表示用例的椭圆，主题的名字标在矩形内。参与者放在矩形外面。

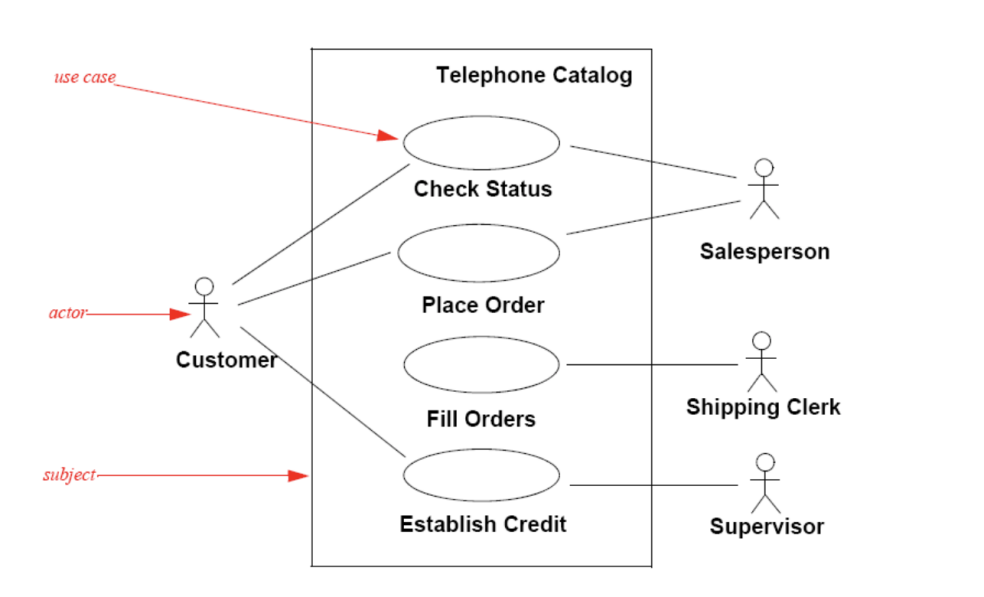


图 1用例图样例

1. 顺序图

顺序图是最常用的一种图。主要用它来描述对象间的交互关系，着重体现交互的时间顺序。对于顺序图，UML2. 0主要做了以下三方面的改进。

（1）允许顺序图中明确地表达分支判断逻辑。这样能够将以前要通过两张图才能表达的意思通过一个图就表达出来，但这并不意味着顺序图擅长表达这种逻辑，所以并不需要在顺序图中展现所有的分支判断逻辑。

（2）允许“纵向”与“横向”地对顺序图进行拆分与引用。这样就解决了以前一张图由于流程过多造成幅面过大，浏览不方便的困难。

1. 提供了一种新图，称为“交互概况图"（Interaction Overview Diagram）,可以直观地表达一组相关顺序图之间的转向逻辑，UML1. x中通常是通过活动图进行间接表达的。如下图所示：

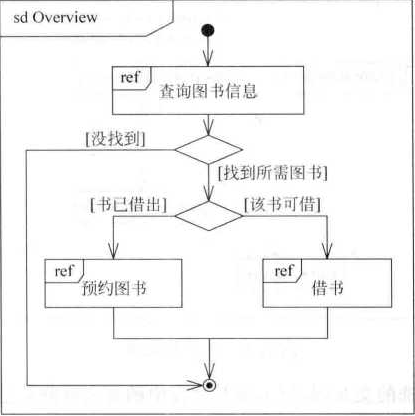


图 2交互概况图

1. 活动图

活动图是比较常用的一种图，接近于流程图。UML 2.0而言，去除了“活动图是状态图的一种特例”这一规定。在UML2.0中，活动图增加了许多新特性。例如，泳道可以划分成层次，增加丰富的同步表达能力，在活动图中引入对象等特性。

如下图所示,泳道用矩形框来表示,属于某个泳道的活动放在该矩形框内,将对象名放在矩形框的顶部,表示泳道中的活动由该对象负责

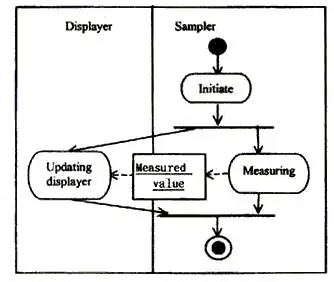
****

图 3带有泳道和对象并发的活动图

1. 构件图

构件图是在物理层面对系统结构及内容的直观描述，最接近于通常意义上的模块结构图。他描述的是在软件系统中遵从并实现一组接口的物理的、可替换的软件模块。

构件图 = 构件(Component) + 接口(Interface) + 关系(Relationship) + 端口(Port) + 连接器(Connector)

在UML2. 0中，构件图有比较明显的改进。组件本身内容的表述更清晰，包括组件所提供的接口、所要求的接口、组件之间的依赖关系通过“组装连接器”(Assembling Connector)更加明确地表达等。

UML2.0提供两种类型的连接器：

代理连接器（Delegation Connector）——委托连接件：连接外部接口的端口和内部接口。

组装连接器（Assembly Connector）——组装连接件：组件连接器表示构件之间的关系，它连接构件内部的类，将一个构件的供接口和一个构件的需接口捆绑在一起

如果要显式地把两个构件实例衔接在一起，在它们的端口之间画一条线即可。

如果两个构件实例相连是由于它们有兼容的接口，则可以使用一个“球－穴”标记来表示构件实例之间的连接关系。如下图

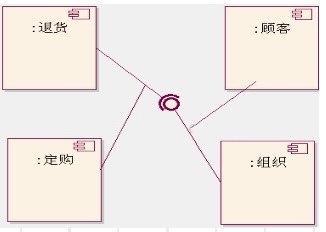


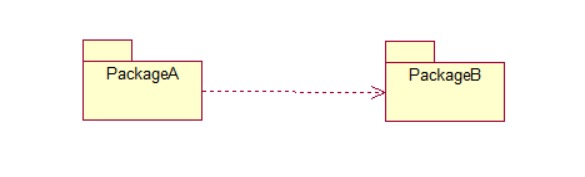
图 4构建图

1. 新增加的图

增加了“包图”、''组合结构图"、“交互概览图”和“时间图”

1. 包图

“包图”展现模型要素的基本组织单元，以及这些组织单元之间的依赖关系，包括引用关系(Packageimport)和扩展关系(PackageMerge) 。在通用的建模工具中，一般可以用类图描述包图中的逻辑内容。如下图所示：

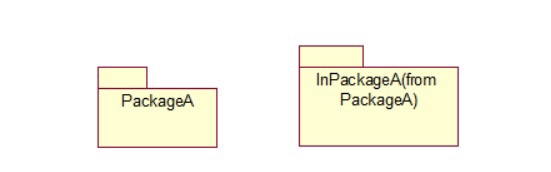


包是包图中最重要的概念，它包含了一组模型元素和图。对于系统中的每个模型元素，如果它不是其他模型元素的一部分，那么它必须在系统中唯一的命名空间内声明。包含一个元素声明的命名空间被称为拥有这个元素。包是一个可以拥有任何种类模型元素的通用命名空间。可以这样说，如果将整个系统描述为一个高层的包，那么它就直接或间接地包含了所有的模型元素。

在系统模型中，每个图必须被一个唯一确定的包所有，同样这个包可能被另一个包所包含。包是构成进行配置控制、存储和访问控制的基础。所有的UML模型元素都能用包来进行组织。每一个模型元素或者为一个包所有，或者自己作为一个独立的包，模型元素的所有关系组成了一个具有等级关系的树状图。然而，模型元素（包括包）可以引用其他包中的元素，所以包的使用关系组成了一个网状结构。

在UML中，包图的标准形式是使用两个矩形进行表示的，一个小矩形（标签）和一个大矩形，小矩形紧连接在矩形的左上角，包的名称位于大矩形的中间。

同其他模型元素的名称一样，每个包都必须有一个与其他包相区别的名称。包的名称是一个字符串，它有两种形式：简单名（Simple Name）和路径名（Path Name）。其中，简单名仅包含一个名称字符串，而路径名是以包处于的外围包的名字作为前缀并加上名称字符串。但是在Rose中，使用简单名称后加上“（from 外围包）”的形式，如下图：



1. 组合结构图

组合结构图”描述系统中的某一部分(即“组合结构”)的内部内容，包括该部分与系统其他部分的交互点，这种图能够展示该部分内容“内部”参与者的配置情况。

“组合结构图”中引入了一些重要的概念。例如，“端口” (Port), “端口 ”将组合结构与外 部环境隔离，实现了双向的封装，既涵盖了该组合结构所提供的行为(Providedlnterface), 同时也指出了该组合结构所需要的服务(Requiredlnterface)；又如"协议”(Protocol)，基于 UML中的“协作"(Collaboration)的概念，展示那些可复用的交互序列，其实质目的是描述那些可以在不同上下文环境中复用的协作模式。“协议”中所反映的任务由具体的“端口”承担。组合结构图如下图所示：

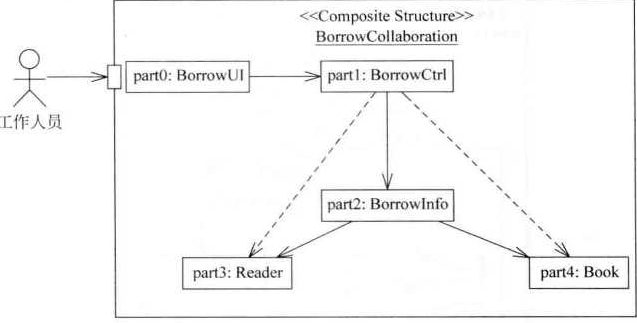
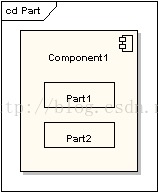
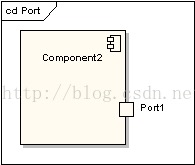


图 5组合结构图

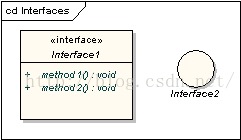
**部件**  
部件是代表一组（一个或多个）实例的元素，这组实例的拥有者是一类元实例，例如：如果一个图的实例有一组图形元素，则这些图形元素可以被表示为部件，并可以对他们之间的某种关系建模。注意：一个部件可以在它的父类被删除之前从父类中被去掉，这样部件就不会被同时删除了。  
部件在类或组件内部显示为不加修饰的方框。



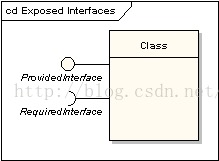
**端口**  
端口是类型化的元素，代表一个包含类元实例的外部可视的部分。端口定义了类元和它的环境之间的交互。端口显示在包含它的部件，类或组合结构的边缘上。端口指定了类元提供的服务，以及类元要求环境提供的服务。  
端口显示为所属类元边界指定的方框。



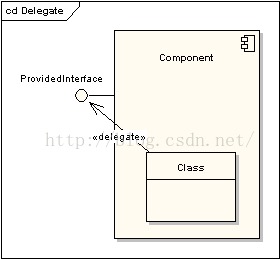
接口  
接口与类相似，但是有一些限制，所有的接口操作都是公共和抽象的，不提供任何默认的实现。所有的接口属性都必须是常量。然而，当一个类从一个单独的超级类继承而来，它可以实现多个接口。  
当一个接口在图中单列出来，它既可以显示为类元素的方框，带 «interface» 关键字和表明它是抽象的斜体名称，也可以显示为圆环。



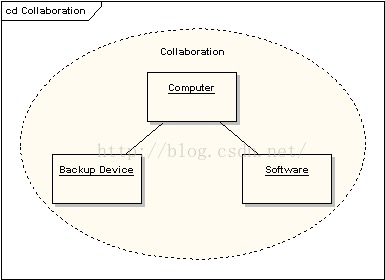
注意：圆环标注不显示接口操作。当接口显示为类所有的接口，它们会被当作暴露接口引用。暴露接口可以定义为是提供的，还是需求的。提供接口确认包含它的类元提供指定接口元素定义的操作，可通过类和接口间实现的连接来定义。需求接口说明该类元能与其他类元进行通信，这些类元提供了指定接口元素所定义的操作。需求接口可通过在类和接口间建立依赖连接来定义。  
提供接口显示为“带棒球体”，依附在类元边缘。需求接口显示为“带棒杯体”，也是依附在类元边缘。



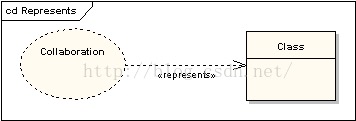
**委托**  
委托连接器用来定义组件外部端口和接口的内部工作方式。委托连接器表示为带有 «delegate» 关键字的箭头。它连接组件的外部约定，表现为它的端口，到组件部件行为的内部实现。



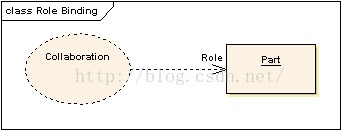
**协作**  
协作定义了一系列共同协作的角色，它们集体展示一个指定的设计功能。协作图应仅仅显示完成指定任务或功能的角色与属性。隔离主要角色是用来简化结构和澄清行为，也用于重用。一个协作通常实现一个模式。  
协作元素显示为椭圆。



**表现**  
表现连接器用于连接协作到类元来表示此类元中使用了该协作。显示为带关键字 «represents»的虚线箭头。



**发生**  
发生连接器用于连接协作到类元来表示此协作表现了（同原文）该类元；显示为带关键字«occurrence»的虚线箭头。

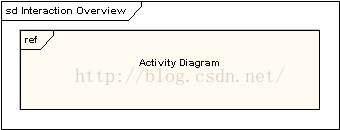


1. 交互概览图

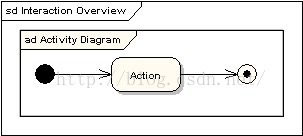
1，交互概览图是交互图与活动图的混合物，可以把交互概览图理解为细化的活动图，在其中的活动都通过一些小型的顺序图来表示；也可以将其理解为利用标明控制流的活动图分解过的顺序图。

2，交互概览图用于将一些零散的顺序图组织在一起，它采用了活动图的构造方式，利用了活动图的各种控制节点，并把活动图的每个活动结点替换为一个交互或者交互使用。每个交互或者交互使用都使用一个顺序图表示。大多数交互概览图标注与活动图一样。例如：起始，结束，判断，合并，分叉和结合节点是完全相同。并且，交互概览图介绍了两种新的元素：交互发生和交互元素。

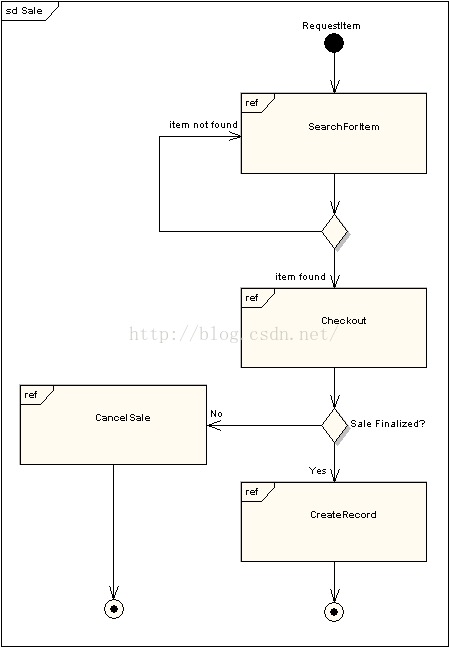
**交互发生**  
交互发生引用现有的交互图。显示为一个引用框，左上角显示 "ref" 。被引用的图名显示在框的中央。



交互元素  
交互元素与交互发生相似之处在于都是在一个矩形框中显示一个现有的交互图。不同之处在内部显示参考图的内容不同。



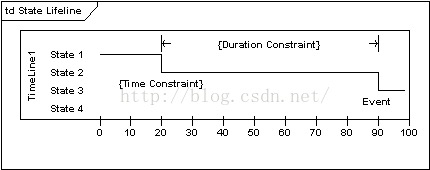
将它们放在一起  
所有的活动图控件，都可以相同地被使用于交互概览图，如：分叉，结合，合并等等。它把控制逻辑放入较低一级的图中。下面的例子就说明了一个典型的销售过程。子过程是从交互发生抽象而来。



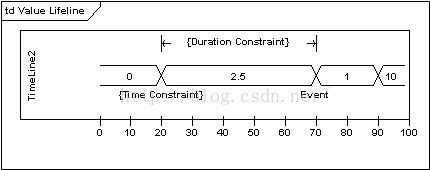
1. 时间图

时间图”是一种可选的交互图，展示交互过程中的真实时间信息，具体描述对象状态变化的时间点以及维持特定状态的时间段。UML 时间图被用来显示随时间变化，一个或多个元素的值或状态的更改。也显示时控事件之间的交互和管理它们的时间和期限约束。

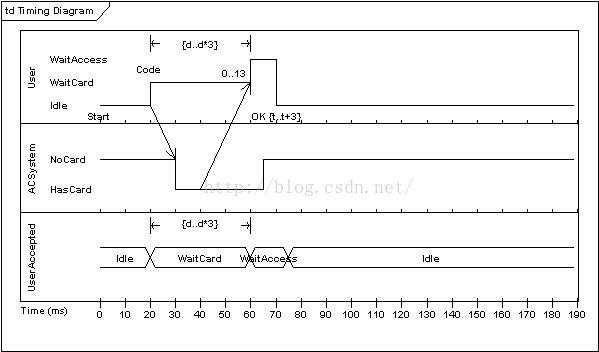
**状态生命线**  
状态生命线显示随时间变化，一个单项状态的改变。不论时间单位如何选择，X轴显示经过的时间，Y轴被标为给出状态的列表。状态生命线如下所示：



**值生命线**  
值生命线显示随时间变化，一个单项的值的变化。X轴显示经过的时间，时间单位为任意，和状态生命线一样。平行线之间显示值，每次值变化，平行线交叉。如下图所示。



将它们放在一起  
状态和值的生命线能叠加组合。它们必须有相同的X轴。 消息可以从一个生命线传递到另一个。每一个状态和值的变换能有一个定义的事件，一个时间限制是指一个事件何时必须发生，和一个期限限制说明状态或值多长时间必须有效。一旦这些已经被应用，其时间图可能显示如下。



问题：

UML2.0支持13种图，他们可以分为两大类：结构图和行为图。以下什么说法不正确：

1. 部署图是行为图
2. 顺序图是行为图
3. 用例图是行为图
4. 构件图是结构图

答案：A（分析：静态模型（类图、构件图和部署图）和动态模型（顺序、对象、用例、协作、状态、活动）

问题：某软件公司欲开发一个在线交易系统，为了能精确表达用户与系统的复杂交互过程，应该采用UML的\_\_\_\_\_来进行交互过程的建模

1. 类图
2. 序列图
3. 部署图
4. 对象图

答案：B类图描述了类、接口、协作以及它们之间的关系

序列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。可为一个协作建模。

对象图描述了类图某一时刻的实际例子，描述了这个时刻对象与对象间的关系。

由上述的内容可知题正确的选项为B