D模块（顺序图、协作图）

6.1顺序图

顺序图：

顺序图（sequence diagram）是强调消息时间顺序的交互图，它描述了对象之间传送消息的时间顺序，用于表示用例中的行为顺序。

表示：

顺序图将交互关系表示为一个二维图。横向轴代表了在协作中各独立对象的类元角色。纵向轴是时间轴，时间沿竖线向下延伸。

基本内容：

角色, 对象, 生命线, 激活期, 消息

顺序图的基本内容

1. 角色

系统角色(Actor)可以是人或其他的系统或者其子系统。

1. 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。顺序图中的对象(Object)在概念上和它在类图中的定义是一致的，它们之间可以进行交互，交互的顺序按时间的顺序。

1. 生命线

生命线(LifeLine)代表顺序图中对象在一段时间内的存在。

生命线是一个时间线，其所用的时间取决于交互持续的时间。每个对象的底部都带有生命线，对象与生命线结合在一起被称为对象的生命线。

对象在生命线上的两种状态：休眠状态和激活状态

1. 激活期

激活期(Activation)也被称为控制焦点，代表顺序图中的对象执行一项操作的时期，是顺序图中表示时间段的符号，在这个时间段内对象将执行相应的操作。

1. 消息

消息(Message)是对象之间某种形式的通信，在垂直生命线之间，用带有箭头的线并附以消息表达式方式表示。它可以激发某个操作、唤起信号或导致目标对象的创建或撤销。一个对象到另一个对象的消息用跨越对象生命线的消息线表示。对象还可以发送消息给它自己，即消息线从自己的生命线出发又回到自己的生命线。

1）同步消息：

仅当发送者要发送一个消息而且接收者已经做好接收这个消息的准备时才能传送的消息称为同步消息。



1. 异步消息：

异步消息:发送者不管接收者是否做好了接收准备都可以发送的消息称为异步消息。



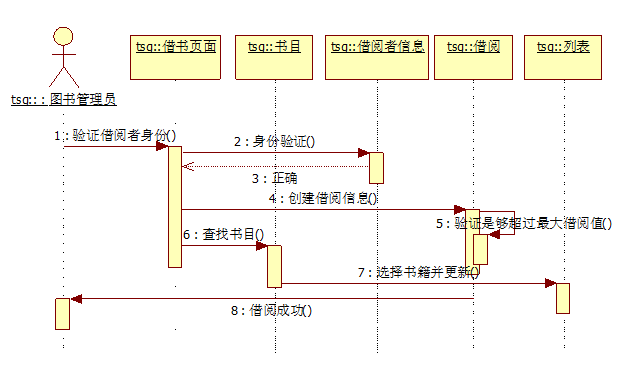
1. 返回消息：

返回消息表示从过程调用返回。



**顺序图示例：**

图书管理员处理借书顺序图



6.2通信图

通信图：

通信图(Collaboration Diagram /Communication Diagram，也叫合作图。注：UML2. 0 以后不再用协作图说法，而是明确定义为“通信图”，即Communication Diagram) 是一种交互图(Interaction Diagram)，强调的是发送和接收消息的对象之间的组织结构。

通信图作为一种给定语境中描述协作中各个对象之间的组织交互关系的空间组织结构图形化方式，在使用其进行建模时，可以将其作用分为以下三个方面。

第一，通过描绘对象之间消息的传递情况来反映具体的使用语境的逻辑表达。一个使用情境的逻辑可能是一个用例的一部分，或是一条控制流，这和顺序图的作用类似。

第二，显示对象及其交互关系的空间组织结构。通信图显示了在交互过程中各个对象之间的组织交互关系及对象彼此之间的链接。与顺序图不同，通信图显示的是对象间的关系，并不侧重交互的顺序，它没有将时间作为一个单独的维度，而是使用序列号来确定消息 及并发线程的顺序。

第三，通信图的另外一个作用是表现一个类操作的实现。通信图可以说明类操作中使用到的参数、局部变量及返回值等。当使用通信图表现一个系统行为时，消息编号对应了程序中嵌套调用结构和信号传递过程。

基本内容：

活动者, 对象, 链接, 消息

1. 活动者：

活动者发出主动操作的对象，负责发送初始消息，启动一个操作。

1. 对象：

对象是类的实例, 负责发送和接受消息。

1. 链接：

链接表示两个对象共享一个消息 , 位于对象之间或参与者与对象之间。

1. 消息：

消息用来描述系统动态的行为 , 它从一个对象向另一个或几个对象发送消息 , 或由一个对象调用另一个对象的操作。

利用消息可以完成很多任务，可以顺序执行、添加条件限制发送、创建带有消息的对象实例和执行迭代。

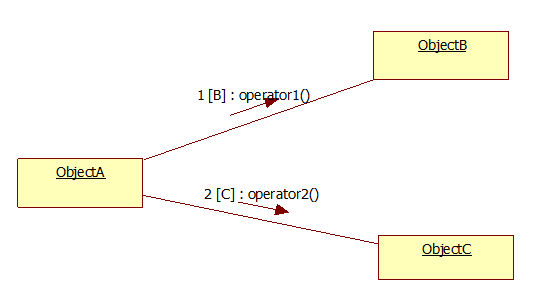
1)序列化

序列化消息只需要在消息前添加序列号，默认情况下即可。

2）控制点条件

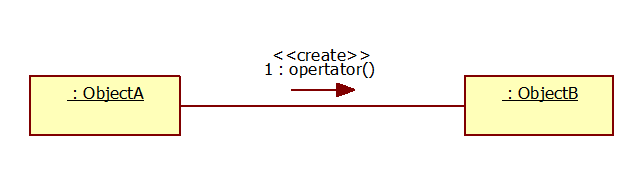
控制点条件用来根据消息表达式的计算结果来限制消息的发送。控制点包含在消息中，在序列1D号和消息文本之间。

例如：如果B计算结果为真，那么ObjectA将会把消息operator1发送给ObjectB;C同理;其他条件下不会发送任何消息



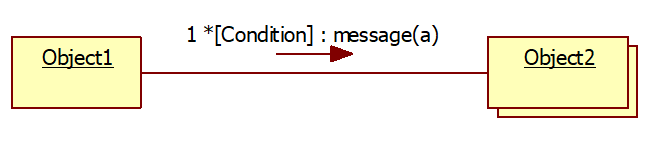
1. 创建实例

消息可以用来在协作图中创建对象实例。



1. 发送给多对象消息

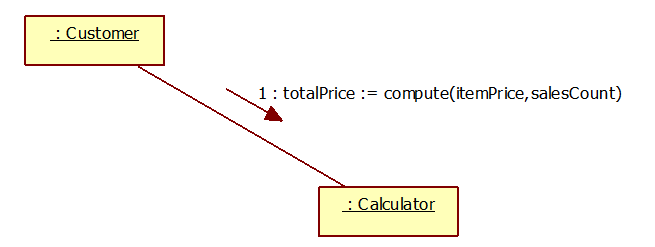
一个对象可能会向同一个类的多个对象同时发送一个消息。



1. 返回结果

消息可能是要求某个对象进行计算并返回结果的值。

例如，一个顾客对象可能请求一个计算器(Calculator)对象计算某项商品的总价，包括该项商品的价格和税款。



1. 构造型

构造型（stereotype）可以在现有的UML元素的基础上创建新的元素



通信图建模技术及应用

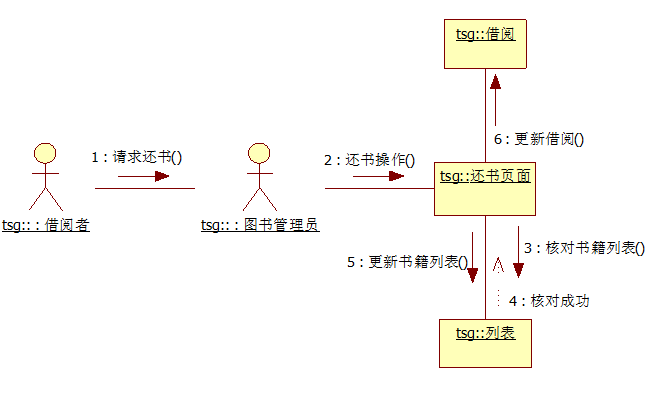
对系统动态行为建模，当按组织对控制流建模时，一般使用通信图，与顺序图一样，一个单独的通信图只能显示一个控制流。

使用通信图建模时可以遵循如下策略。

1. 确定交互过程的上下文。
2. 确定参与交互过程的活动者与对象。
3. 如果需要，为每个对象设置初始特性。
4. 确定活动者、对象之间的链接。一般先确定关联的链接，因为这是最主要的，它代表了结构的链接。然后需要确定其他链接，用合适的路径构造型修饰，这表达了对象间是如何互相联系的。
5. 从引发该交互过程的初始消息开始，将每个消息附到相应的链接上，可以用带小数点的编号来表达嵌套。
6. 细化消息内容。比如需要说明时间或空间的约束，可以用适当的时间或空间约束来修饰每个消息。

通信图示例：

图书管理员处理还书通信图



E模块（构件图和部署图）

8.1构件图

构件图：

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（另一种图是部署图），构件图用于静态建模，是表示组件类型的组织及各种组件之间依赖关系的图。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。

组成元素：

构件图的组成元素包括组件(Component)、接口 (Interface)和关系(Relationship),还可以包括包(Package)和子系统(Subsystem)

1. 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。

组件可以分为以下三种类型：

1. 实施组件
2. 工作产品组件
3. 执行组件

组件与类的异同：

一般来说，组件在许多方面都与类相同：两者都有名称；都可以实现一组接口；都可以 参与依赖、泛化和关联关系；都可以被嵌套；都可以有实例；都可以参与交互。

但是组件和类之间也有一些显著的差别：

* 类表示逻辑抽象，而组件表示存在于计算机中的物理抽象。
* 组件表示的是物理模块而不是逻辑模块，与类处于不同的抽象级别。
* 类可以直接拥有属性和操作；而一般情况下，组件仅拥有只能通过其接口访问的操作。

1. 接口

接口是一组用于描述类或组件的一个服务的操作，它是一个被命名的操作的集合，与类不同，它不描述任何结构(因此不包含任何属性)，也不描述任何实现(因此不包括任何实现操作的方法)。每个接口都有一个唯一的名称。

组件的接口可以分为以下两种类型。

* 导岀接口 (Expert Interface)
* 导入接口 (Import Interface)

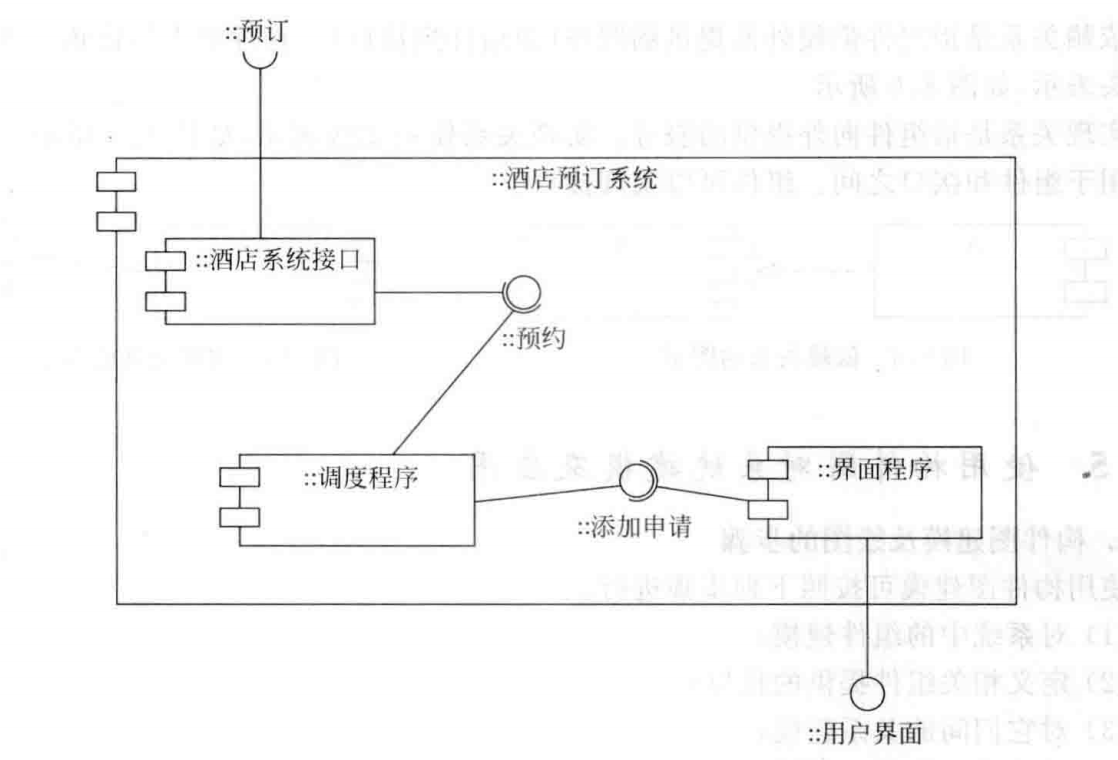
1. 关系

关系是事物之间的联系，在面向对象的建模中，最重要的关系是依赖、泛化、关联和实现，但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）。
* 实现关系是指组件向外提供的服务。

构件图示例

酒店预订系统建模子组件示意图：



8.2部署图

部署图

用于静态建模，表示运行时过程结点、组件实例以及对象结构的图。

可显示计算结点的拓扑结构，通信路径，结点上运行的软件，软件包含的逻辑单元等。

主要元素

结点 , 组件 , 关系

1. 结点

结点是存在于运行时并代表一项计算资源的物理元素，一般至少拥有一些内存，而且通常具有处理能力。它一般用于对执行处理或计算的资源建模，通常具有如下两方面内容： 能力(如基本内存、计算能力和二级存储器)和位置(在所有必需的地方均可得到)。

1. 组件

部署图中还可以包含组件，这里所指的组件就是8. 1节中介绍的构件图中的基本元素。它是系统可替换的物理部件。

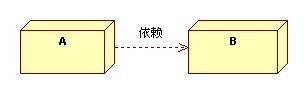
结点和组件的关系可以归纳为以下两点。

* 组件是参与系统执行的事物，而节点是执行组件的事物。
* 组件表示逻辑元素的物理模块，而节点表示组件的物理部署。

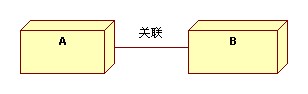
1. 关系

部署图中也可以包括依赖、泛化、关联及实现关系

依赖关系

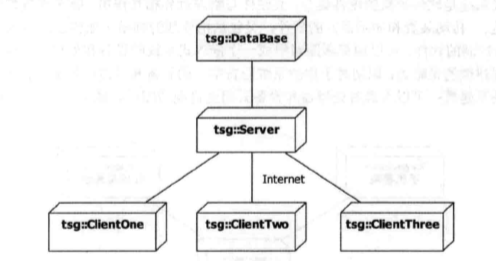


关联关系



部署图示例

图书管理系统部署图：



问题：如果需要强调时间和序列，最好选择（ ）；如果需要强调上下文相关，最好选择（ ）。

答案：序列图；通信图。