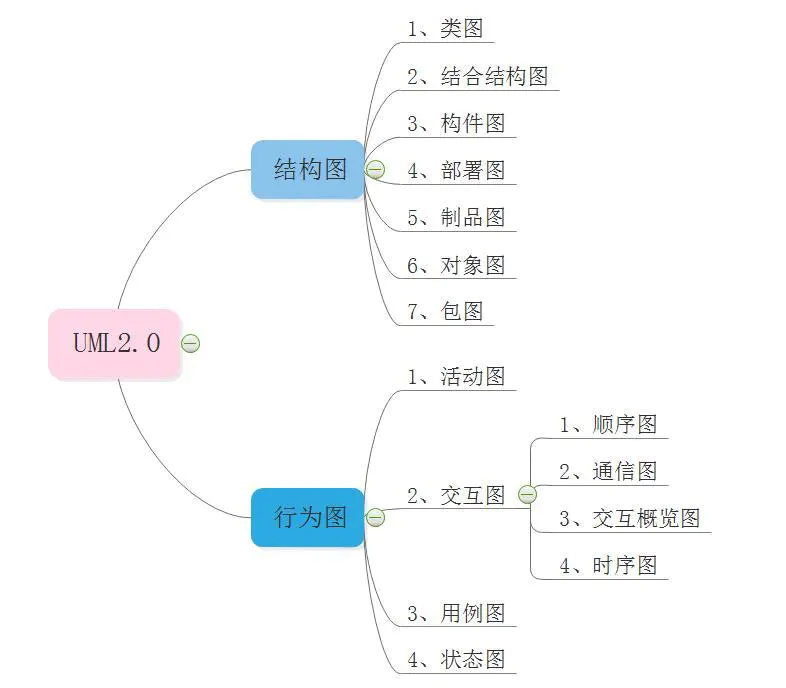
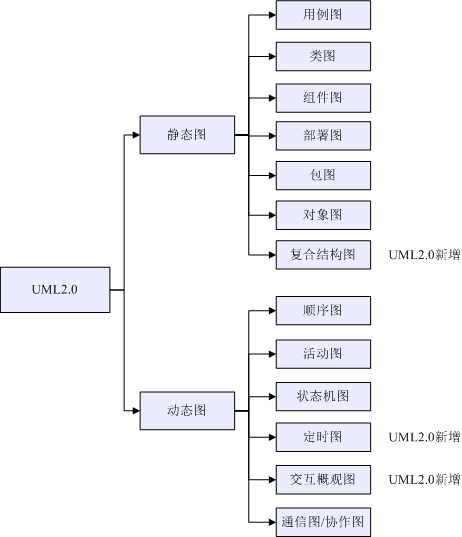
·1.7 UML2.0的特性（图种类9->13）



结构图：结构图描述了系统的静态视图或结构。它广泛用于软件体系结构文档中。强调建模要存在的元素。

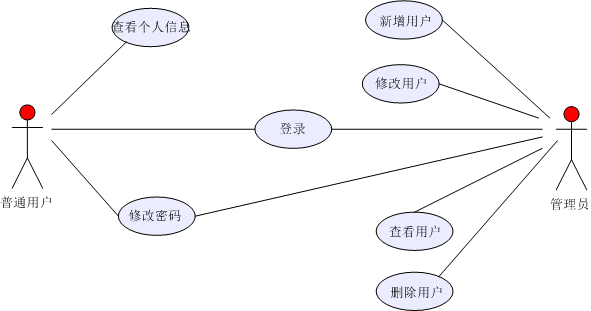
行为图：描绘了系统的动态视图或系统的行为, 描述了系统的功能，定义了系统内的交互



静态图偏向表示结构，动态图偏向表示行为

1. 用例图（行为图，静态图，有矛盾，但要辩证看待，用例图从结构上看表述功能为静态，但内部有功能描述动态系统的行为，所以被归类为静态行为图）

主要用于系统需求分析阶段，从用户角度描述系统的需求功能，方便与客户交流，保持需求唯一性。

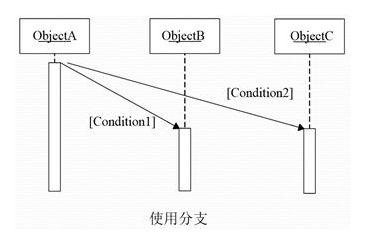


UML2中为每一个用例增加了一个称为“Subject”的特征，这项特征的取值可以作为在逻辑层面划分一组用例的一项依据。如“系统边界”就是“Subject”的一种典型案例。（未找到案例）

1. 顺序图（行为图，动态图）

UML2.0做了3方面的改进：

1. 允许顺序图中明确的表达分支判断逻辑

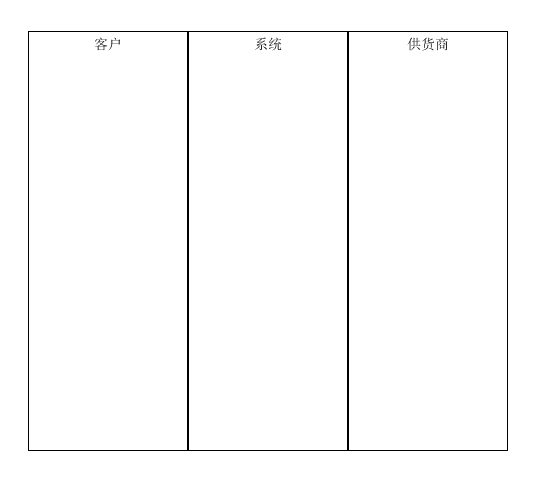


1. 允许“纵向”和“横向”对顺序图进行拆分与引用，解决了由于流程过多导致的幅面过大，浏览不便的问题。
2. 提供了一种新图，称之为“交互纵览图”（下面会介绍）
3. 活动图（行为图，动态图）

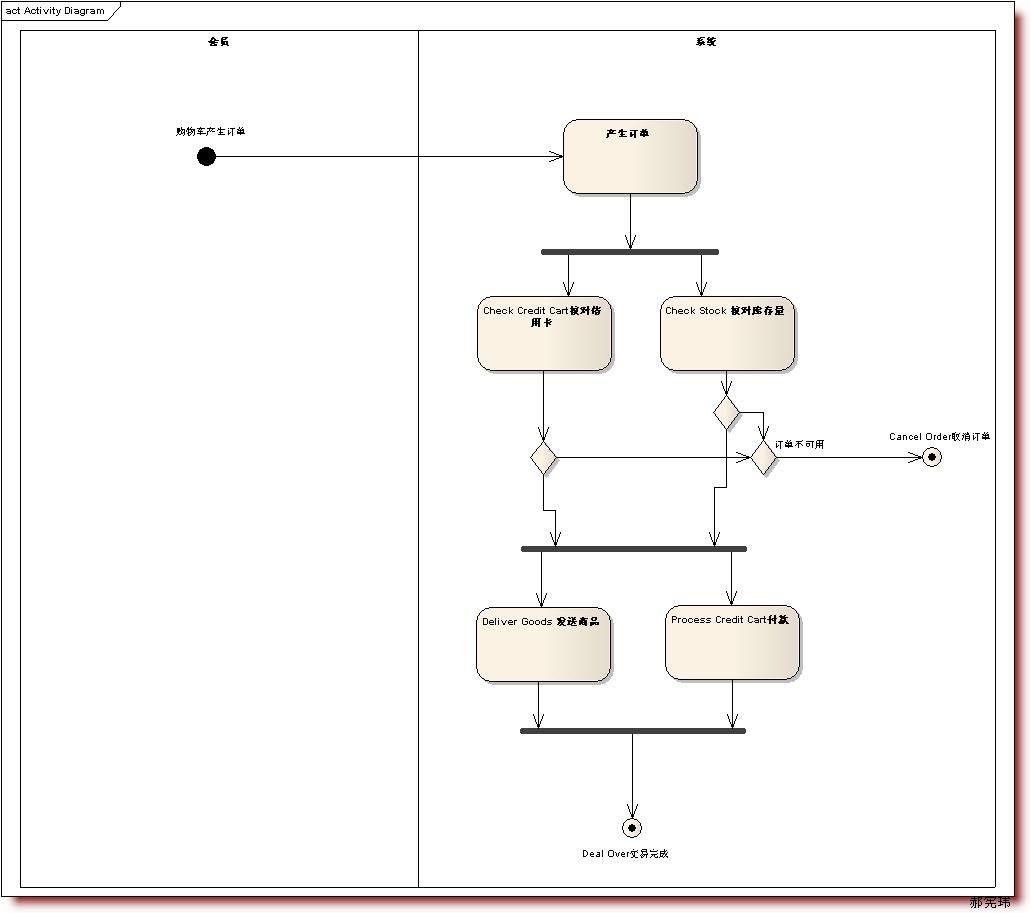
在UML2.0中，活动图增加了泳道，对象等特性

**泳道**：泳道将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。

在活动图中，泳道区分了负责活动的对象，它明确地表示了哪些活动是由哪些对象进行的。在包含泳道的活动图中，每个活动只能明确地属于一个泳道。



实例：



**对象**：用活动图描述某个对象时，可以把涉及到的对象放置在活动图中并用一个依赖将其连接到进行创建、修改和撤销的动作状态或者活动状态上，对象的这种使用方法就构成了对象流。



1. 组件图（结构图，静态图）

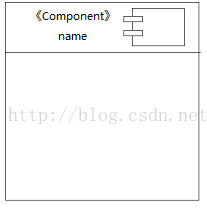
组件提供的接口，要求的接口，组件之间的依赖关系通过“组装连接器”能够更明确地表达。

组件图(Component Diagram)又称为构件图，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。

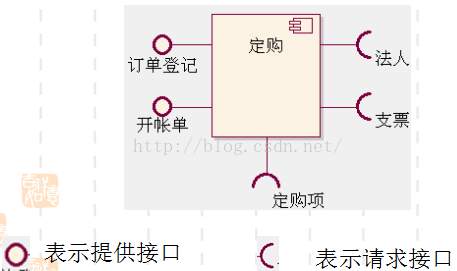
**组件**：

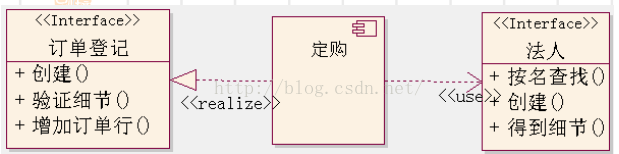


构件（UML2.0特供）：



**接口**：由一组操作组成，供接口为棒棒糖式，需接口为插座式

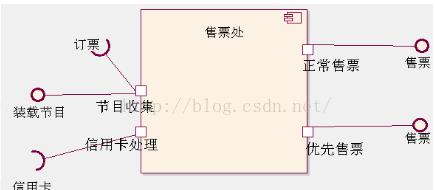




**端口**：UML2.0新引入的概念，用于描述构件与环境之间以及构件和它内部构件之间的一个显式交互点。所有进出构件的交互都需要通过端口。

提供接口说明了通过端口来提供服务，请求接口说明了通过端口需要从其它构件获得服务。

表示方式：尾部加小方框的正常接口表示，小方框就被称为端口。

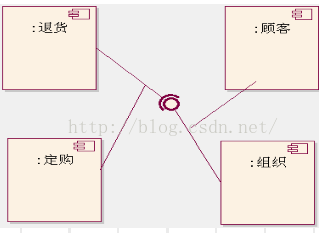


**组装连接件**：有两种表示装配连接件的方法：

1.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。

2.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－半圆”标记来表示构件实例之间的连接关系。

装配连接件是两个构件实例间的连接件，它定义一个构件实例提供服务，另一个构件实例使用这些服务。装配连接件用于把一个请求接口或端口与一个提供接口或端口的连接起来。在执行时，消息起源于一个请求端口，沿着连接件传递，被交付到一个提供端口

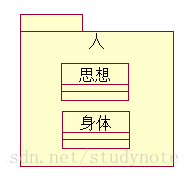


1. 新增加的图

增加了“包图”，“组合结构图”，“交互纵览图”和“计时图”

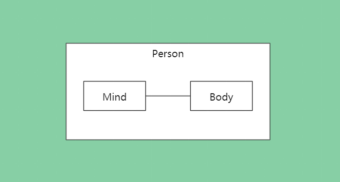
1. **包图**

包的图标就像是一个带标签的文件夹，使用包的思想就是把共同工作的元素放到这样的一个带有标签的文件夹中。例如，如果很多类或组件组成了而一个特殊的子系统，就应该将它们放到一个包中。



1. **组合结构图**

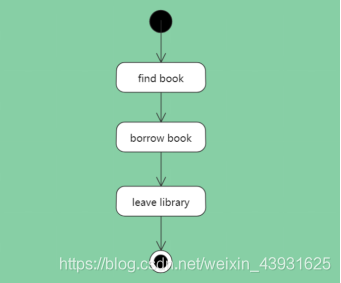
显示类的内部结构，有时一个类可由多个类组成（如：人由身体和思想组成）



1. **交互纵览图**

活动图中的活动（activity）可能包含对象之间的交互，可用对象之间的交互表示某个活动。**通俗易懂的讲就是把活动图的每一个活动都用顺序图或协作图来进一步描述**，就得到了交互纵览图

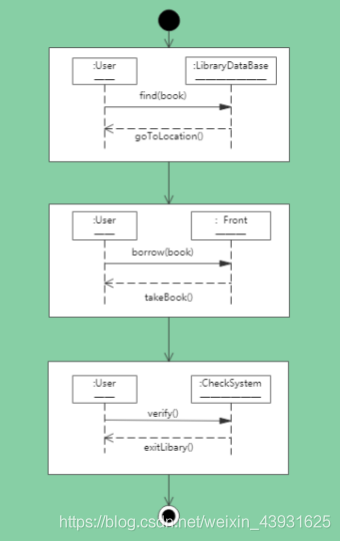
例子：用户图书馆借书流程：找书（find book）、借书（borrow book）、离开图书馆（leave library）

活动图

找书：用户（user）通过图书馆数据库（library database）查询书的位置，找到后去取书

借书：用户（user）拿书去前台（front）借阅，完成后拿走书

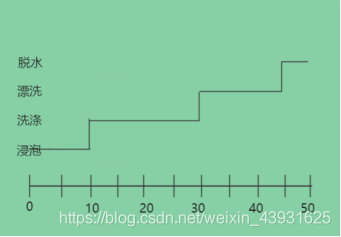
离开图书馆：用户（user）通过门禁系统（check system）自动确认后，离开图书馆

交互纵览图

1. **计时图**

顺序图描述了操作按照时间的先后顺序，但是没有指定操作持续的具体时间，可用计时图表示操作的持续时间

案例：洗衣服：浸泡（10min）、洗涤（20min）、漂洗（15min）、脱水（5min）



·1.8 系统开发阶段

系统开发有五个阶段：

1. 需求分析：了解用户需求，分析系统可行性，分析需求的一致性及正确性等。

与用户沟通确定初步需求，分析可行性建立系统开发方案，对比方案需求以及用户需求寻找统一的可行性，正确性等。

1. 设计：将需求转换为系统的重要过程，主要包括概要设计和详细设计。

概要设计：架构设计，功能模块设计，数据库设计，规范制定

详细设计：数据库详细设计，模块详细构建等

1. 实现：指的是通过程序语言，将所设计的内容转化为可以执行的软件系统

系统实现，开发工具和平台选择，数据库构建，模块编码

1. 除错：是实现活动中不可避免的工作，主要是修改程序编写过程中产生的错误。
2. 测试：是对实现的程序代码模块进行检测，检验其功能是否正确，性能是否符合要求。

模块测试，集成测试，系统测试，验证测试。