# UML构件图

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。

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

* 在面向对象系统的**物理方面**进行建模要用到两种图：**组件图和配置图。**

# **二、包含的元素**

## **1.组件（Component）->构件**

* 组件是定义了良好接口的物理实现单元，是系统中可替换的物理部件。
* 组件代表系统的一个物理实现块，代表逻辑模型元素如类、接口、协同等的物理打包。
* 构件通过它的提供接口和请求接口展现行为。
* 由于在UML2.0中，构件是一种类，因此构件具有属性、操作和可见性。这些概念的含义与在类图中定义的是一样的，只是在这里把这些概念应用在构件上。

**组件的命名：**

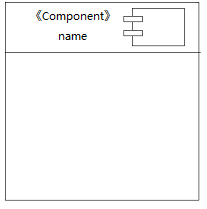
  组件的名称有两种：简单名和路径名。并依据目标操作系统可以添加相应的扩展名，例如java和dll。

**表示方式**：组件用一个左侧带有突出两个小矩形的矩形来表示。



**UML2.0中表示方式**：构件用加构造型《component》的矩形框来表示，左上角添加以前的构件符号，如果没有构件细节可在中央直接写上名字。

（构造型和左上角的图标可以二选一）



**UML2.0把构件分为基本构件和包装构件**

* **基本构件**  
  注重于把构件定义为在系统中可执行的元素。
* **包装构件**  
  扩展了基本构件的概念，它注重于把构件定义为一组相关的元素，这组元素为开发过程的一部分。也即, 包装构件定义了构件的命名空间方面。在构件的命名空间中，可以包括类、接口、构件、包、用况、依赖（如映射）和制品。按照这种扩展，构件也具有如下的含义：可以用构件来装配大粒度的构件，方法为把所复用的构件作为大粒度构件的成分，并把它们的请求和提供接口连接在一起（简单理解：组件包含组件，组拼大组件）。

**组件的种类**

1. 配置组件（Deployment Component）：运行系统需要配置的组件，是形成可执行文件的基础—操作系统、JAVA虚拟机、DBMS；
2. 工作产品组件（Work Product Component）：包括模型、源代码和用于创建配置组件的数据文件，它们是配置组件的来源—UML图、java类和数据库表；
3. 执行组件（Execution Component）：在运行时创建的组件，是最终可运行的系统产生的允许结果—.net组件

## **2.接口（Interface）**

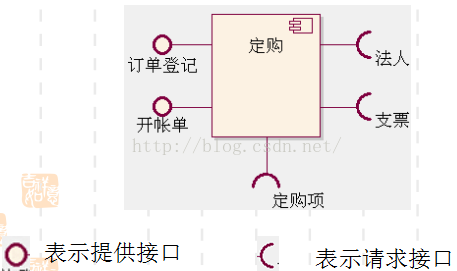
* 接口（interface）接口由一组操作组成，它指定了一个契约，这个契约必须由实现和使用这个接口的构件的所遵循。

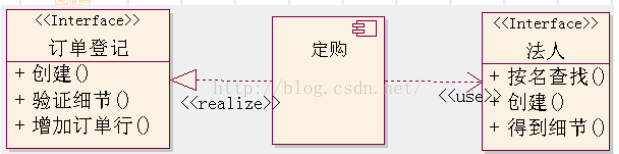
**接口分提供接口和请求接口**

* 把构件实现的接口称为提供接口（供接口），这意味着构件的提供接口是给其它构件提供服务的。实现接口的构件支持由该接口所拥有的特征，包括接口拥有的约束。
* 构件使用的接口被称为请求接口（需接口），即构件向其它构件请求服务时要遵循的接口

**表示方式：**

* 供接口用“棒棒糖”式的图形表示，即由一个封闭的圆形与一条直线组成。
* 需接口用“插座”式的图形表示，即由一个半圆与一条直线组成。





在java语言中供接口通过关键字implements来显式地表示，需接口被类所使用的任何接口类型隐式地定义。

## **3.外部接口——端口**

* 端口是UML2.0引入的概念
* 端口描述了在构件与它的环境之间以及在构件与它的内部构件之间的一个显示地交互点
* 端口是一个封装构件的显示的对外窗口，**所有进出构件的交互都要通过端口。**
* 使用端口能在更大的程度上增加构件的封装性和可替代性。
* 端口是构件的一部分，端口的实例随着它们所属的构件的实例一起被创建和撤消。

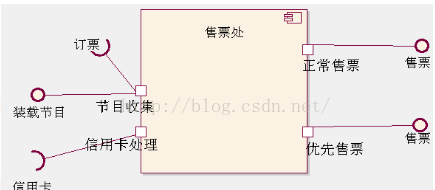
**<1>接口与端口的关系**

  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。

  一个构件可以通过一个特定端口同另一个构件通讯，而且通讯完全是通过由端口支持的接口来描述的。

**<2>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。



## **4.连接器（Connector）——连接件**

**UML2.0提供两种类型的连接器：**

1. 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。
2. 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

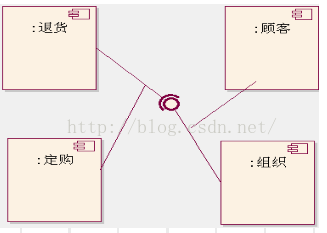
* 连接端口意味着请求端口要调用提供端口中的操作，以得到服务。
* 立端口和接口的优点在于在设计时，两个构件彼此不需要了解对方的内部，只要它们的接口是相互兼容的即可。
* 如果一个端口提供一个特定的接口而另一个端口需要这个接口，且接口是兼容的，那么这两个端口-便是可连接的。

**<1>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－穴”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



**<2>委托连接件**

  委托有这样的含义：具体的消息流将发生在所连接的端口之间，可能要跨越多个层次，最终到达要对消息进行处理的最终部件实例。这样，使用委托连接件可对构件行为的层次分解建模。

**委托连接件把外部对构件端口的请求分发到构件内部的部件实例进行处理，或者通过构件端口把构件内部部件实例向构件外部的请求分发出去。**

  构件内部的一个部件可以是另一个构件或是一个类。注意，必须在两个提供端口间或两个请求端口间定义委托连接件。

**注意事项**：因为构件是可以嵌套的，所以内部构件之间的连接（球-穴）是组装连接件，内部构件与端口之间的连接（实线箭头）是委托连接件。

## **4..依赖关系（Dependency）**

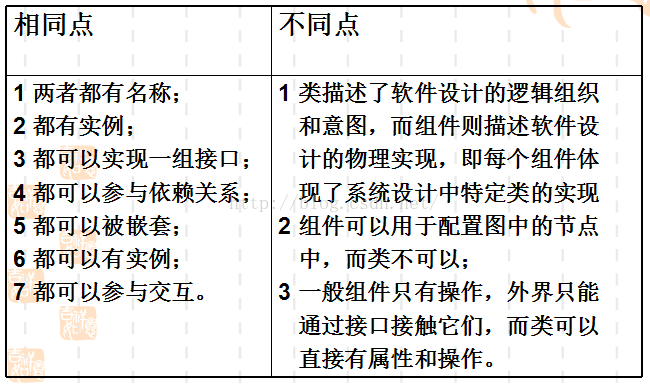
组件图用依赖关系表示各组件之间存在的关系类型。

  在UML中，组件图中依赖关系的表示方法与类图中依赖关系相同，都是一个由客户指向提供者的虚线箭头。



# **三、相关概念比较**

## **1.组件与类**



## **2.组件与包**

二者均为分组组织机制，但也有许多不同之处：

* 一个组件代表一个物理的代码模块，而包可以包含成组的逻辑模型元素，也可以包含物理的组件；可以用包来组织用例(use case)，不可以用组件来组织用例。
* 一个类可以出现在多个组件中，却只能在一个包中定义。
* 配置图(deployment diagram)中，节点(node)中可以放组件，但不可以放包。
* 包只是类型(type)，组件可以是实例也可以是类型。
* 包可以作为开发视图(development view)，用于管理。组件可作为物理视图(physical view)，用于部署。但反之不然。

# **四、组件图建模技术**

1. 对系统中的组件建模—分解系统，考虑有关系统的组成管理、软件的重用和物理节点的配置等因素，把关系密切的可执行程序和对象分别归入组件，找出相应的类、接口等模型元素。
2. 对相应组件提供的接口建模。
3. 对组件之间的依赖关系建模。
4. 将逻辑设计映射成物理实现。
5. 对建模的结果进行精化和细化。

