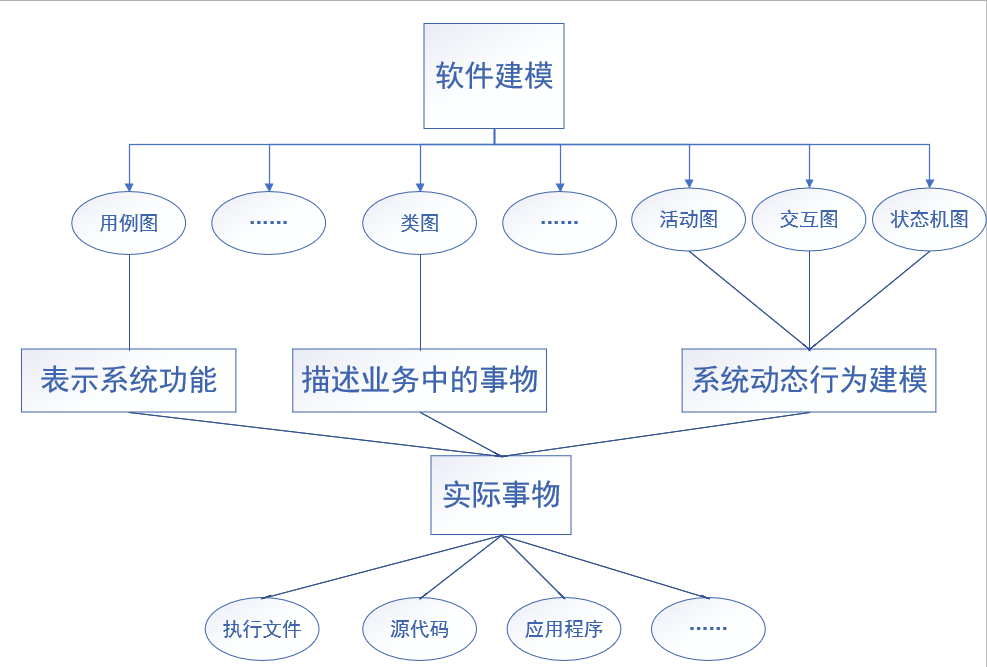
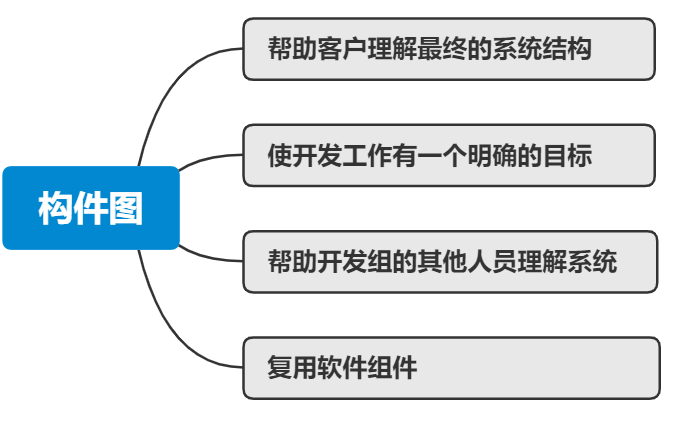
# 8.1构件图

## 8.1.1构件图概述

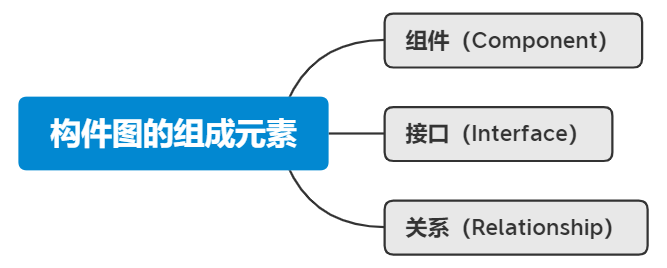
在软件建模的过程中，用例图表示系统功能，类图描述业务中的事物，活动图、交互图、状态机图对系统动态行为建模。在完成这些设计后，分析人员需要将这些逻辑设计图转化为实际的事物。在此过程中，很多组件必须重新建立，而有些组件则可以进行复用。因此，可以使用构件图来可视化物理组件及他们之间的关系，并描述其构造细节。



构件图、部署图是对面向对象系统的物理方面建模时使用的两种图。构件图用于描述软件组件及组件之间的组织和依赖关系。



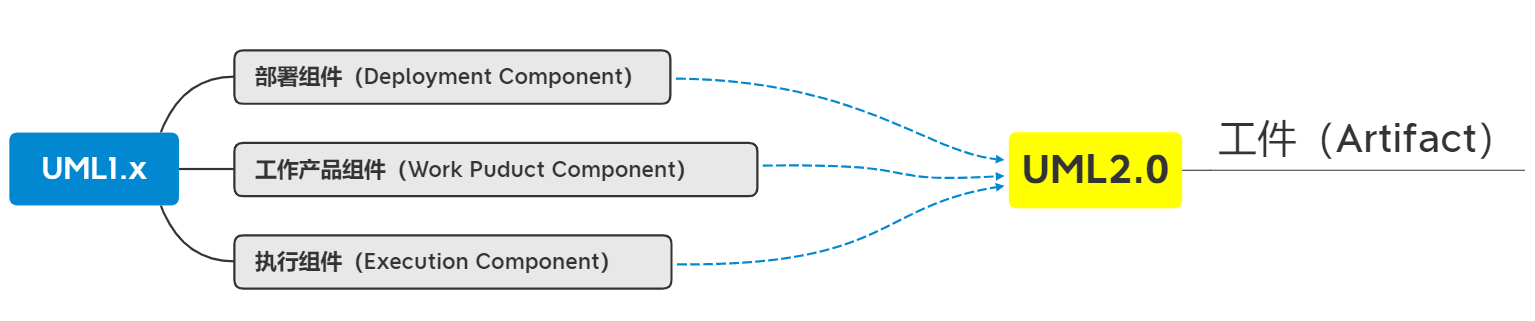
构件图的组成元素包括组件（Component）、接口（Interface）和关系（Relationship）。



## 8.1.2组件

### 8.1.2.1组件介绍

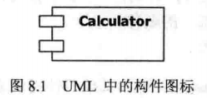
软件组件是软件系统的一个物理单元。UML1.x中数据文件、表格、可执行文件、文档等被定义为组件。在UML1.x中，建模者通常把这些组件分为部署组件、工作产品组件和执行组件。而在UML2.0中，这些组件被统称为工件。



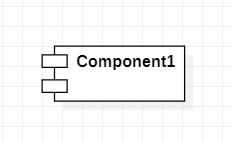
组件是系统中遵从一组接口且提供实现的一个物理部件，通常指开发和运行时类的物理实现。组件常用于对可分配的物理单元建模，这些物理单元包含模型元素，且具有身份标识和明确定义的接口。每一个组件都必须有唯一的名称。

### 8.1.2.2组件表示方法

构件图的主图标是一个左侧附有两个小矩形的大矩形框。组件的名字位于构件图标的中央，名字本身是一个文本字符串。如图8.1所示。

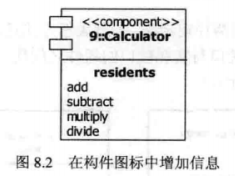


注：StarUML4.0中构件图标和Visio中的构件图标有区别

StarUML4.0：

Visio：

如果组件属于一个包，可以在组件名称的前面加上包名，还可以在另外一个隔开区域里绘出组件的操作，即该操作可以驻留在组件中，图8.2示意了这种情况。



## Q&A

问题一：组件与类的相同点？

答：

1.两者都有唯一的名称；

2.都可以实现一组接口；

3.都可以参与依赖、泛化和关联的关系；

4.都可以被嵌套；

5.都可以有实例；

6.都可以参与交互；

问题二：组件与类的不同点？

答：

1.类表示逻辑抽象，而组件表示存在于计算机中的物理抽象（组件可以存在于实际可运行的计算机上的，类不可以）

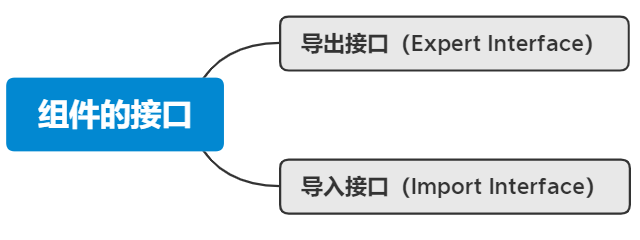
2.组件表示的是物理模块而不逻辑模块，与类处于不同的抽象级别。

3.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（虽然都可以实现接口，但是组件的服务只能通过其接口访问）

## 8.1.3接口

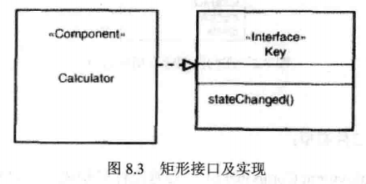
### 8.1.3.1接口介绍

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，接口不描述任何结构（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。

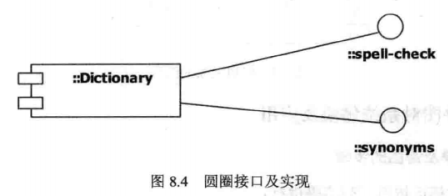


### 8.1.3.2接口表示方法

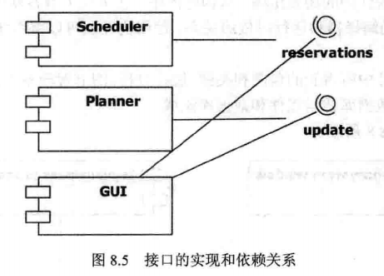
组件和组件的接口有两种表示法。一种是用矩形表示，使用带空心三角形箭头的虚线连接。如图8.3所示。



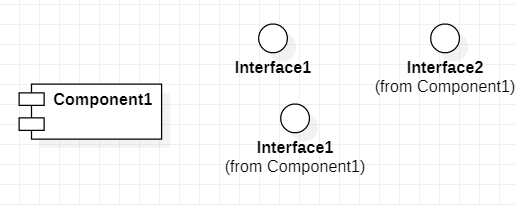
另一种表示法使用小圆圈来代表接口，用实线和组件连接起来。在这种语境中，实线代表的是实现关系。如图8.4所示。

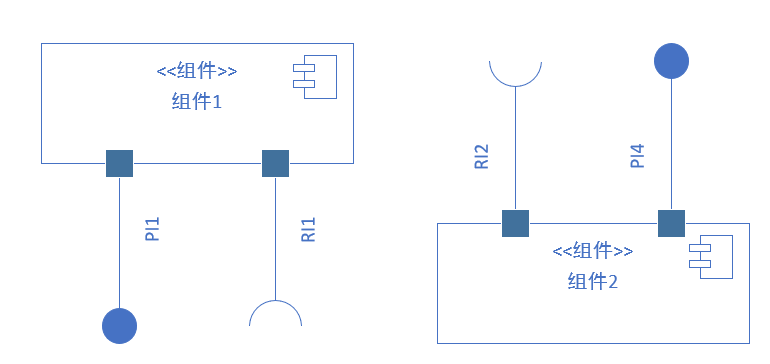


除了实现关系之外，还可以表示依赖关系即组件和它用来访问其他组件的接口之间的关系。如图8.5所示。图8.5使用了“球窝”符号。“球”代表提供的接口，“窝”代表了所需的接口。



注：在Visio和starUML4.0中，接口的表示是基本上一致的。

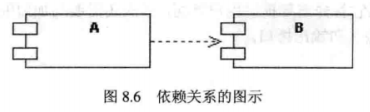
StarUML4.0：

Visio：

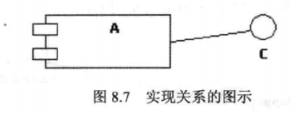
## 8.1.4关系

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关系和实现关系。

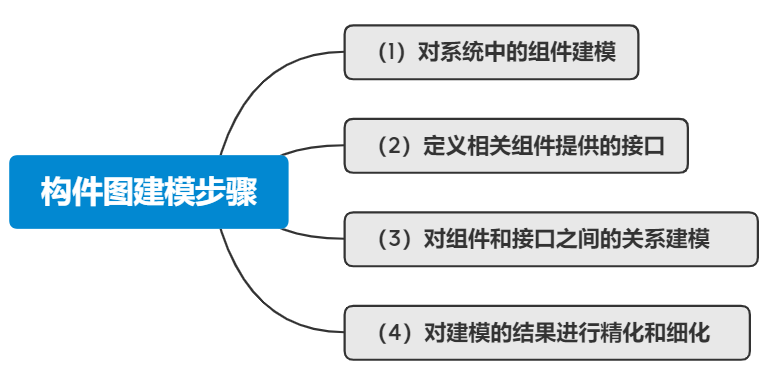
依赖关系是指组件依赖外部提供的接口（由组件到接口）。构件图中的依赖关系使用虚线箭头表示。



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



## 8.1.5构件图建模及应用



### 在线酒店预订子系统示例

将整个“在线酒店预订子系统”作为一个构件，考虑其对外接口并确定子系统对外的接口。显然它首先需要提供用户界面;其次还需要与加盟的酒店系统连接，完成预订工作，就是提供了接入和输出接口。如图8.9所示

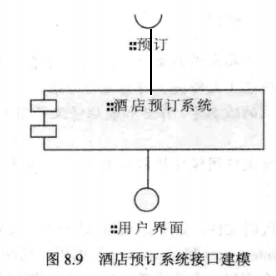
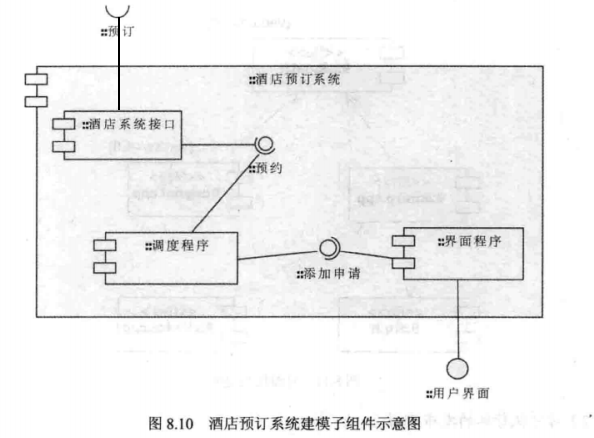


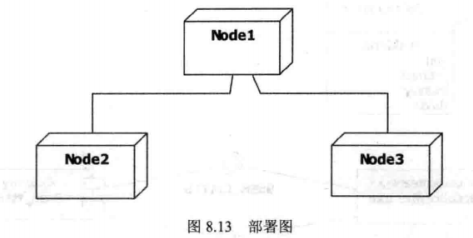
图8.10 显示如何确定子组件和接口，显然要有一个组件来实现用户界面，一个组件来完成与酒店系统的连接和预订，另外还应该有一个负责将用户的需求与酒店的供给进行匹配的“调度程序”子组件。



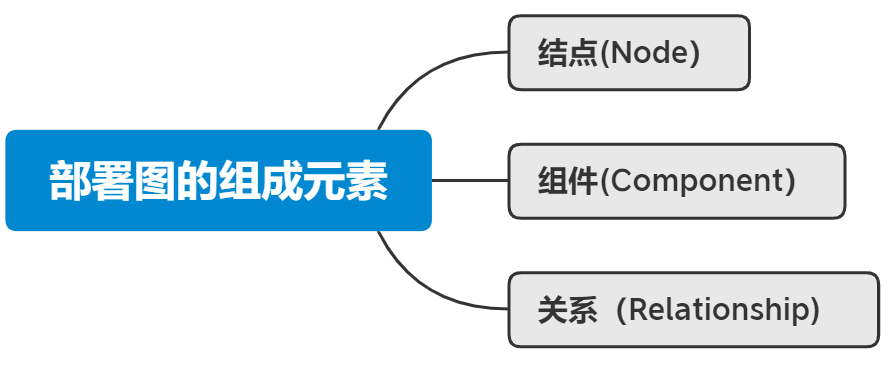
# 8.2部署图

## 8.2.1部署图概述

部署图（Deployment Diagram)用于静态建模，是表示运行时过程节点(Node）结构、组件实例及其对象结构的图。展示了第8.1节构件图中所提到的组件如何在系统硬件上部署，以及各个硬件部件如何相互连接。UML部署图显示了基于计算机系统的物理体系结构。它可以描述计算机，展示它们之间的连接，以及驻留在每台机器中的软件。每台计算机用一个立方体来表示，立方体之间的连线表示这些计算机之间的通信关系。图8.13是部署图的一个例子。



构成部署图的元素主要是节点(Node）、组件(Component）和关系（Relationship)。

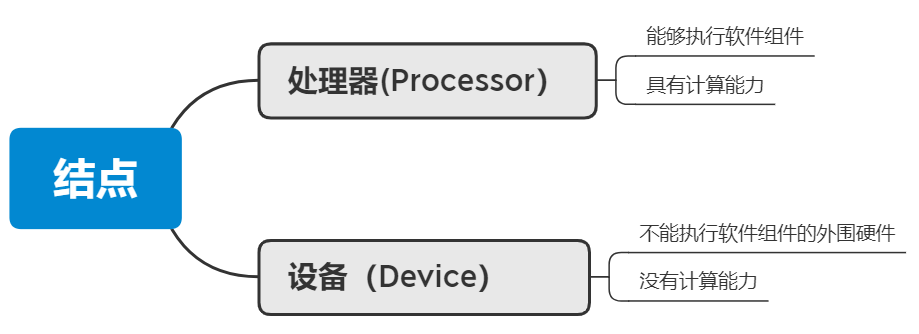


## 8.2.2结点

### 8.2.2.1结点介绍

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）。在建模过程中，可以把结点分成两种类型:处理器（Processor）和设备（Device）。

在UML1.x中，处理器(Processor）是能够执行软件组件、具有计算能力的结点。设备（Device）是不能执行软件组件的外围硬件，没有计算能力的结点，通常是通过其接口为外界提供某种服务，例如打印机、扫描仪等都是设备。尽管这种区分并没有在UML 1.x中形式化，但是它很有用。

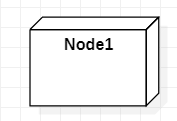


在UML2.0中用立方体来表示一个结点(与UML 1.x例图一样)。UML 2.0正式地把一个设备定义为一个执行工件(Artifact)的结点。为结点起一个名字，并添加关键字<<device>>来指明节点类型。图8.14显示了一个节点。

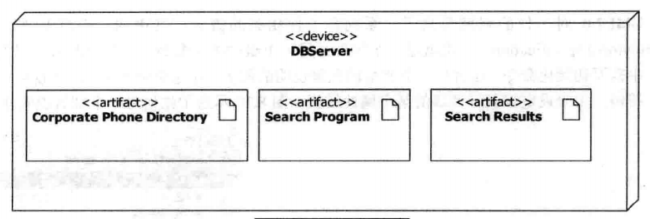


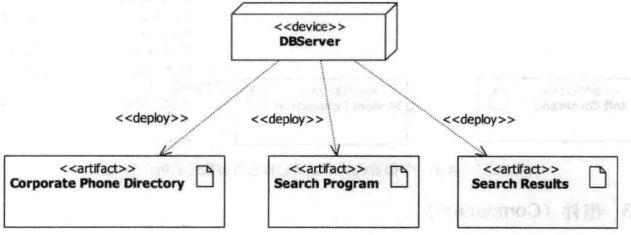
注：在Visio和starUML4.0中，部署图中结点的表示是基本上一致的，但是starUML相对Visio较为简单，Visio更贴近UML书本。

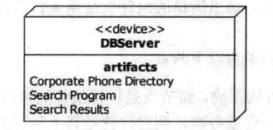
Visio

StarUML4.0

### 8.2.2.2在结点上部署工件的三种建模方式

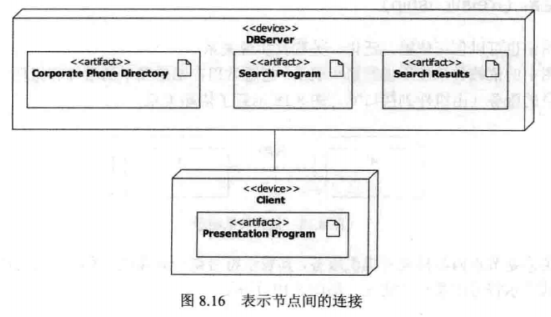






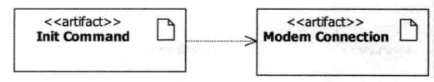
### 8.2.2.3结点间的连接

连接两个立方体的一条线，表示了两个结点相连。一个连接不一定要是一段电线或电缆。也可以表示红外线或者通过卫星的无线连接。图8.16 给出了结点间连接的例子。



### 8.2.2.4部署说明

UML2.0对工件的强调带来了一系列和工件相关的概念，其中的一个就是部署说明(Deployment Specification)，也就是一个组件为另一个组件提供参数。一些调制解调器的连接过程中需要初始化命令，这就是一个典型的部署说明的例子。在这个例子中，部署说明就是一个字符串，用来设定调制解调器的某个属性的值。下图示意了Init Command对Modem Connection进行部署说明建模。



注：Visio中支持部署说明建模。



## 8.2.3组件

部署图中还可以包含组件，这里所指的组件就是第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:

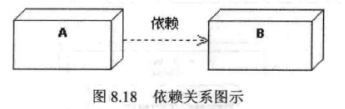
（1）组件是参与系统执行的事物，而结点是执行组件的事物。简单的说就是组件是被结点执行的事物，如假设结点是一台服务器，则组件就是其上运行的软件。

（2）组件表示逻辑元素的物理模块，而结点表示组件的物理部署。这表明一个组件是逻辑单元(如类)的物理实现，而一个结点则是组件被部署的地点。一个类可以被一个或多个组件实现，而一个组件也可以部署在一个或多个节点上。

## 8.2.4关系

部署图中也可以包括依赖、泛化、关联及实现关系。

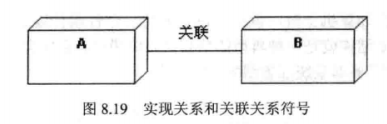
部署图中的依赖关系使用虚线箭头表示。它通常用在部署图中的组件和组件之间，组件依赖外部提供的服务（由组件到接口）。图8.18示意了依赖关系。



实现关系是结点内组件向外提供服务，其表示符号是一条实线。

关联关系是体现结点间通信关联，其表示符号也是一条实线。

如图8.19所示。

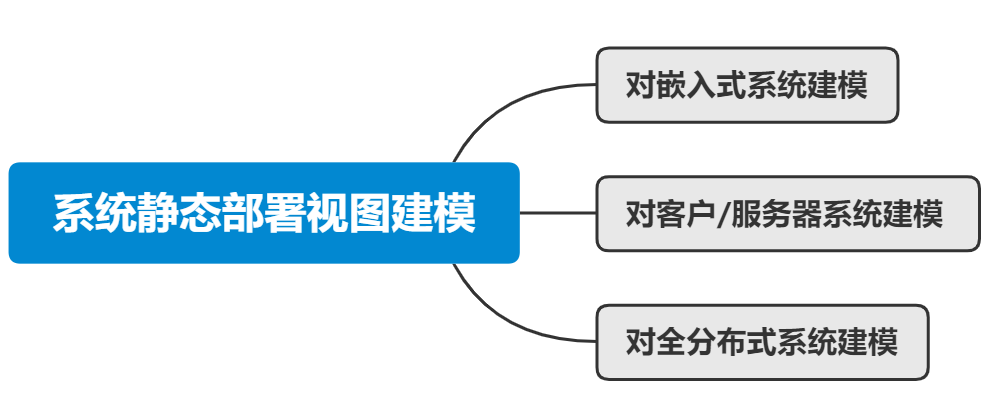


## 8.2.5部署图的系统建模及应用

部署图用于对系统的静态部署视图建模。这种视图主要用来解决构成物理系统的各组成部分的分布、提交和安装。

有些开发的系统不需要部署图，比如开发的软件是将运行在一台机器上而且只和该机器上已由宿主操作系统管理的标准设备(如键盘)相互作用，就不必要设计部署图。

如果软件交互设备是物理地分布在多个处理器上的，则使用部署图有助于思考系统中软件到硬件的映射。



### 8.2.5.1对嵌入式系统建模

嵌入式系统是一种软件密集的硬件集合。

其硬件与物理世界相互作用。

嵌入式系统包括控制设备(如马达、传动装置和显示器）的软件，又包括由外部的刺激〈如传感器输入、运动和温度变化)所控制的软件。可以用部署图对组成一个嵌入式系统的设备和处理器建模。

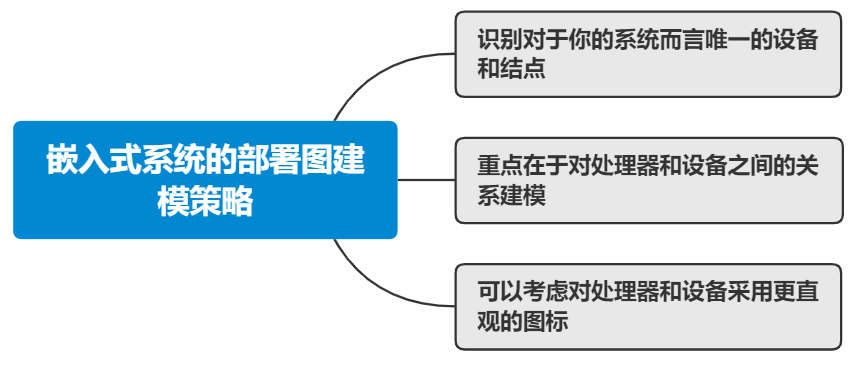
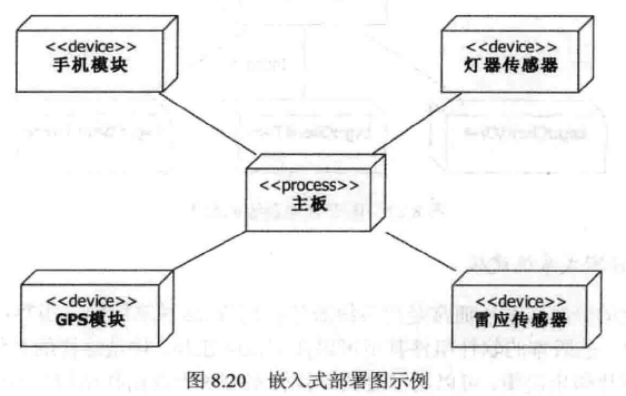
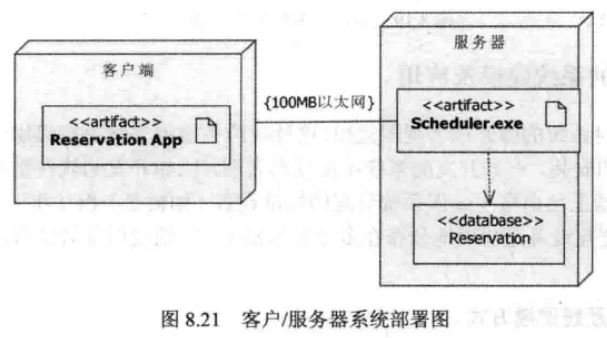


图8.20 为一个嵌入式部署图示例。



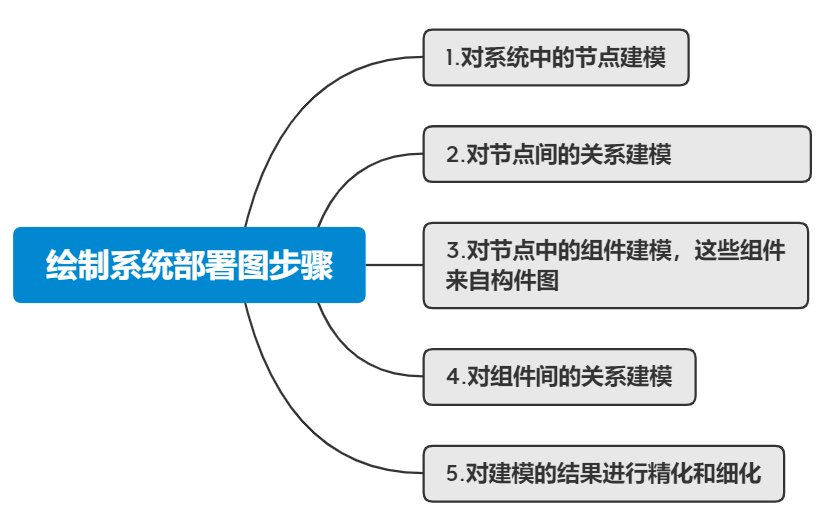
### 8.2.5.2对客户/服务器系统建模

客户/服务器系统是一种常用的体系结构，它注重于将在客户机上的系统的用户界面和在服务器的系统永久数据清晰地分开。它要求对客户/服务器间的网络连接及系统中的软件组件在结点上的物理分布做出决策。可以用部署图对这种客户/服务器系统的拓扑结构建模。当开发的软件要运行在多台计算机上时，就必须决定如何将软件组件以合理的方式部署在各个节点。其中客户机/服务器结构就是一种典型的分布式系统模型，它包含三层B/S结构、两层C/S结构。图8.21为客户/服务器系统部署图的应用。



### 8.2.5.3对全分布式系统建模

广泛意义上的分布式系统通常是由多级服务器构成。这种系统中一般存在着多种版本的软件组件，其中的一些版本的软件组件甚至可以在结点间迁移。构造这样的系统，需要对系统拓扑结构的不断变化做出决策。可以用部署图可视化系统的当前拓扑结构及组件的分布情况，并推断拓扑结构变化的影响。



## Q&A

问题一：结点间的连接是一段电线或电缆？

答：错误。一个连接不一定要是一段电线或电缆。也可以表示红外线或者通过卫星的无线连接。