**一、部署图：展示硬件节点和软件构件之间的映射关系**

**比如说有一个BS系统，BS系统里哪些内容和板块是要部署在应用服务器上，哪些又和前端相关。**

**解决的是系统已经开发完毕了，这个系统的元素，各个部件部署在什么位置才能实现我们要的效果。比如BS系统，把系统部署在应用服务器上，部署在数据库上来，对外提供职能在前端，只要用浏览器访问就可以了，直观的展示出哪些软件构建部署在哪些服务器或客户端上。它通常用来帮助理解分布式系统。**

**部署图的元素之间，逻辑关系非常清晰：**

**物件-被部署的东西**

**节点-部署的目标**

**物件和节点之间要表示出的是部署关系**

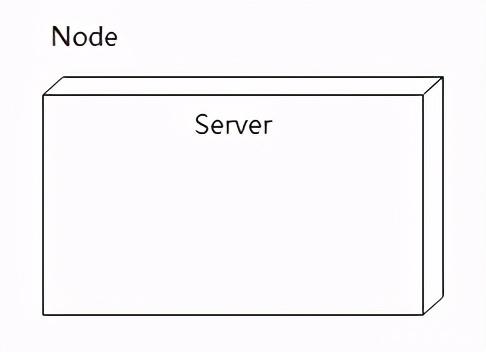
**节点与节点要表示出的是通信路径**

**二、部署图的组成**

**1）节点（Node）& 节点实例（Node Instance）**

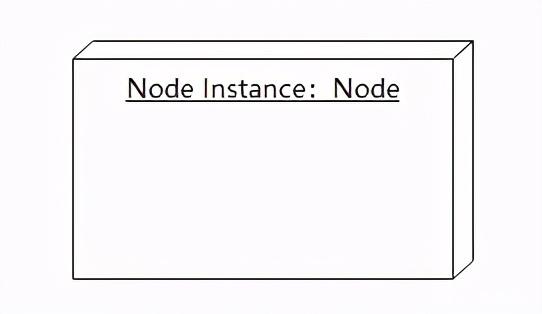
节点是存在与运行时的代表计算机资源的物理元素，可以是硬件也可以是运行其上的软件系统，比如64主机、Windows操作系统、防火墙等。

一般用立方体表示，如下图：



**注**：处理器是带阴影的立方体，设备是不带阴影的立方体

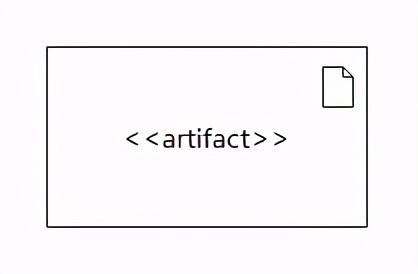
区别于节点，节点实例的名称会有下划线，并且节点类型前会有冒号（注：冒号前面可以有示例名称也可以没有示例名称），如下图：



**2）构件（Artifact）**

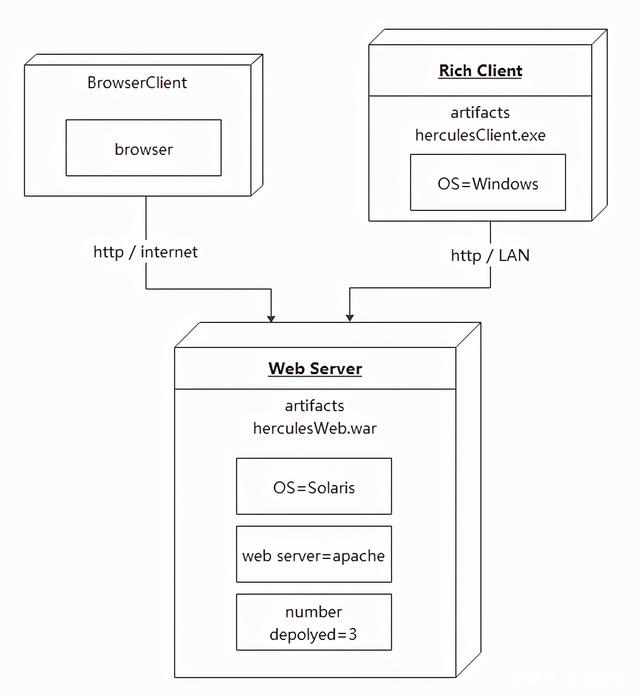
构件是软件开发过程中的产物，包括过程模型（比如用例图、设计图等等）、源代码、可执行程序、设计文档、测试报告、需求原型、用户手册等等。

构件表示如下，带有关键字artifact和文档图标：

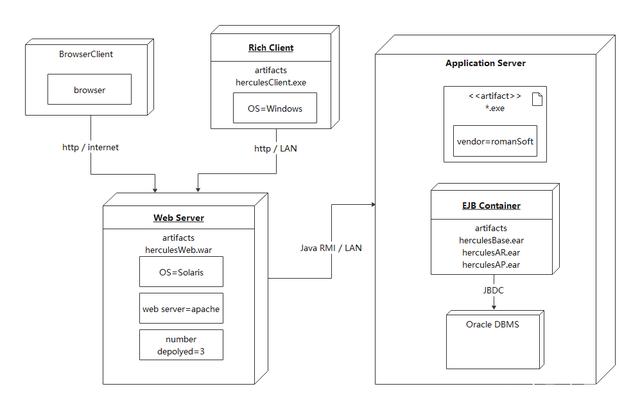


**3）连接（Association）**

节点之间的连线表示系统之间进行交互的通信路径，这个通信路径称为连接。如下图所示，连接中可以标注上网络协议：



**三、部署图实例**



**四、如何绘制部署图**

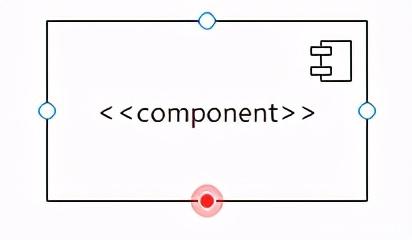
接下来进行绘制演示：

**1）新建空白画布**。在“新建”页依次选择“软件和数据库>软件>UML图”，新建一张空白画布；

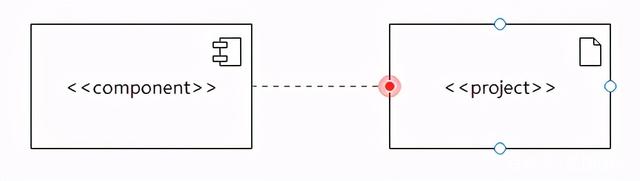
新建空白画布后，画布左侧的符号库会自动调出所需的图形符号▼

**2）符号自由拖放**。根据图形需求，从左侧符号库选择合适的符号，亿图图示支持符号自由拖放，符号支持自由缩放，以及修改图形及线条样式等操作。

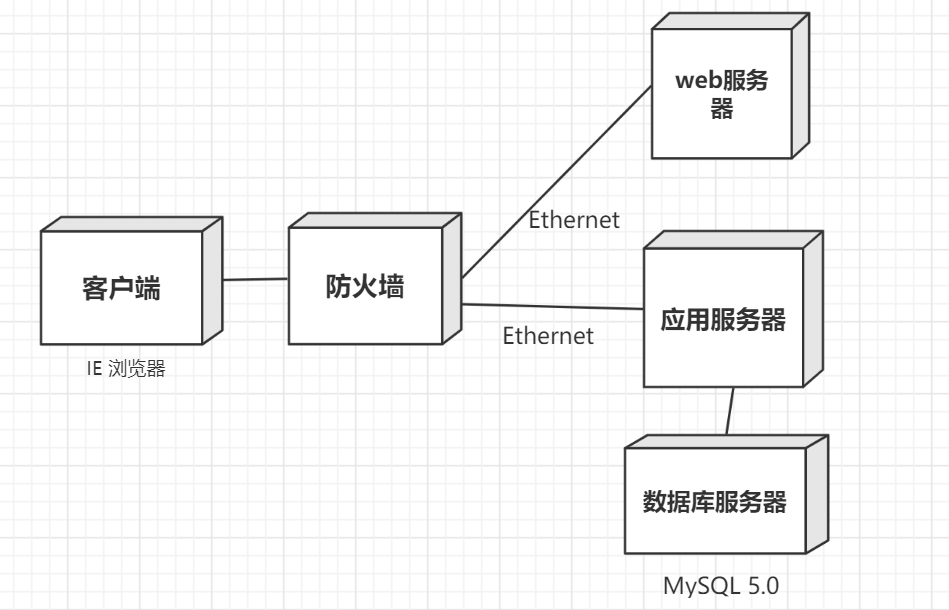
**3）连接关联符号**。除了使用用符号库的连接线符号，还可以从符号上快速生成连接线（当鼠标移至符号上方时，符号四周会自动显现蓝色的圆形连接点，选中后圆形连接点会变为红色，拖拽即可生成连接线）；

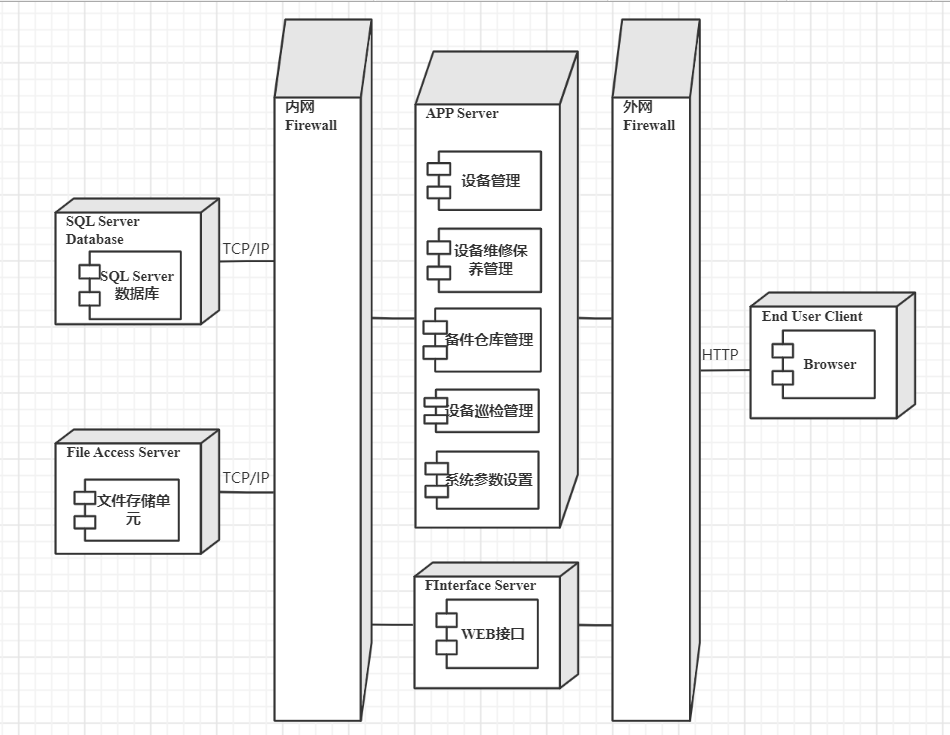


当连接线拖拽至其他符号上时同样会出现圆形连接点，支持自动吸附。



**4）给符号进行标注，以及检查图形是否完整和正确**。





当涉及音乐网站的部署图时，可以考虑以下组件和架构：

1.用户界面（User Interface）：这是用户与音乐网站交互的部分，包括网页前端、移动应用等。用户通过这些界面来搜索、播放音乐，创建播放列表，与其他用户交互等。

2.应用服务器（Application Server）：应用服务器是承载整个音乐网站的核心。它负责处理用户请求、执行业务逻辑，并在需要时从数据库检索数据。应用服务器可以采用多种技术栈，如Node.js、Python、Java等。它还可以包含缓存层来提高性能。

3.数据库（Database）：音乐网站需要存储大量的音乐数据、用户信息、播放列表等。常见的数据库选择包括关系型数据库（如MySQL、PostgreSQL）和NoSQL数据库（如MongoDB、Redis）。数据库层应该提供高可用性、可扩展性和读写性能。

4.文件存储（File Storage）：音乐文件是音乐网站的核心资源。为了确保快速且可靠的数据传输，音乐文件通常存储在专门的文件存储系统中，如Amazon S3、Google Cloud Storage等。这些文件存储系统提供了高可用性、可伸缩性和安全性。

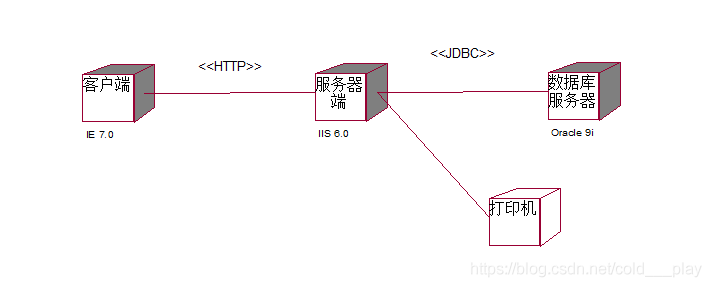
5.第三方服务集成（Third-party Service Integration）：音乐网站通常会整合其他服务，如支付网关、社交媒体平台、音乐版权服务等。这些集成点可能需要使用API或其他机制与第三方服务进行通信。

部署图通常以图形表示，可以使用工具如Microsoft Visio、Lucidchart等来创建。在图中，可以明确标识每个组件的名称和功能，并使用箭头表示它们之间的通信和依赖关系。

需要注意的是，具体的音乐网站架构会因实际需求和技术选择而有所不同。上述部署图提供了一个一般性的参考，可以根据自身情况进行调整和扩展

部署图（Deployment Diagram）描述了一个系统运行时的硬件节点，以及在这些节点上运行的软件构件将在何处物理地运行和它们将如何彼此通信的静态视图。

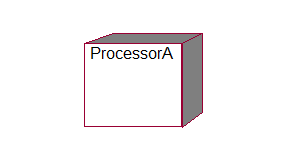
在一个部署图中，包含了两种基本的模型元素：节点（Node）和节点之间的连接（Connection）。在每一个模型中仅包含一个部署图。如下图所示，是一个系统部署图。



图中包含了客户端、服务器端、数据库服务器和打印机等节点，其中客户端和服务器端通过HTTP方式连接，服务器端与数据库服务器通过JDBC方式连接，客户端中拥有IE7.0进程，服务器端中拥有IIS 6.0进程，数据库服务器为Oracle 9i。

在Rose中可以表示的节点类型包括两种，分别是处理器（Processor）节点和设备（Device）节点。

处理器节点是指那些本身具有计算能力，能够执行各种软件节点，例如，服务器、工作站等这些都是具有处理能力的机器。在UML中，处理器的表示形式如下图：

在处理器的命名方面，每一个处理器都有一个与其他处理器相区别的名称，处理器的命名没有任何限制，因为处理器通常表示一个硬件设备而不是软件实体。

由于处理器是具有处理能力的机器，所以在描述处理器方面应当包含处理器的调度（Scheduling）和进程（Process）。调度是指在处理器处理其进程中为实现一定的目的而对共同使用的资源进行时间分配。有时候我们需要指定该处理器的调度方式，从而处理达到最优或比较优的效果。

名称 含义

HTTP 超文本传输协议

JDBC Java数据库连接，一套为数据库存取编写的Java API

ODBC 开放式数据库连接，一套微软的数据库存取应用编程接口

RMI 远程通信协议，一个Java的远程调用通信协议

RPC 远程过程调用通信协议

同步 同步连接，发送发必须等到接收方的反馈信息后才能再发送消息

异步 异步连接，发送方不需要等待接收方的反馈信息就能再发送消息

Web Service 经由诸如SOAP和UDDI的Web Service协议的通信