**包图Package Diagram**

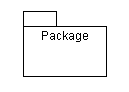
# 一、概述

当对一个比较复杂的软件系统进行建模时，会有大量的类、接口、组件、节点和图需要处理；如果放在同一个地方的话，**信息量非常的大，显得很乱，不方便查询**，所以就对这些信息进行分组，将语义或者功能相同的放在同一个包中，这样就便于理解和处理整个模型。

而包图就是描述**包与包之间的关系**。

包的内容：拥有或引用的模型元素。  
**包的实例没有任何语义。**

仅在建模时有意义，而不必转换到可执行的系统中。



# 二、详细介绍

## 2.1 名称

每个包必须有一个与其他包相区别的名称，包的名字是一个字符串：

简单名：仅含一个简单的名称。

路径名：:以包所位于的外围包的名字作为前缀的包名。



## 包的表示：

当不需要显示包的内容时，将包的名字放入主方框内；

需要显示内容时包的名字放入左上角的小方框中，将内容放入主方框内。

标以 {global} 的包叫通用包，表示系统的所有其他包都依赖于该包。

## **2.2 拥有的元素**

**包拥有的元素：类、接口、组件、节点、协作、用例、图以及其他包。**

**一个模型元素不能被一个以上的包所拥有。**

**如果包被撤销，其中的元素也要被撤销。**

**一个包形成了一个命名空间。**

**一个包的各个同类建模元素不能具有相同的名字；**

**不同包的各个建模元素能具有相同的名字，因为它们代表不同的建模元素；**

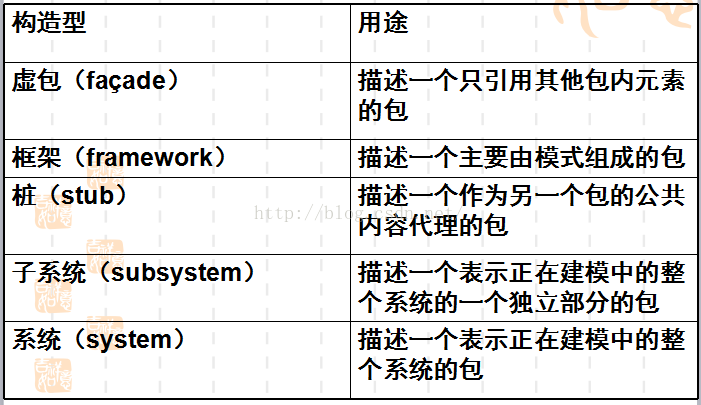
**同一个包内，不同种类的模型元素能够具有相同的名字。**

## 2.3 **可见性**

包的可见性用来控制包外界的元素对包内元素的可访问权限。

| 包的可见性 | 访问权限 |
| --- | --- |
| “+” --public | 包内的模型元素可以被任何引入了此包的其他包的内含元素访问 |
| “#”--protected | 表示此元素能被该包的子包内所含元素访问 |
| “-”--private | 表示此元素只能被属于同一包的内含元素访问 |

## 2.5 标准元素



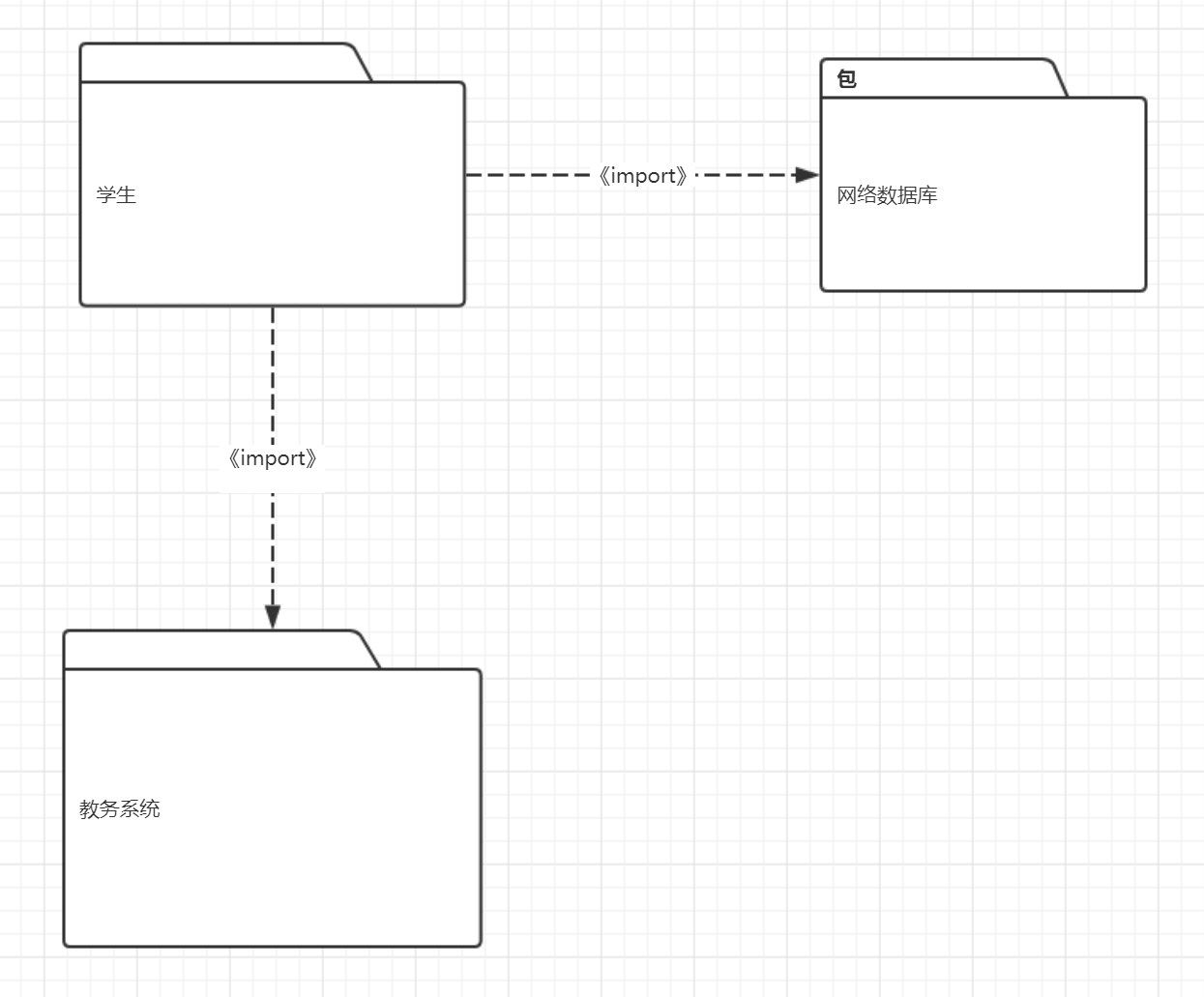
# 包之间的关系

**引入关系、泛化关系、嵌套关系**

# 一：引入关系

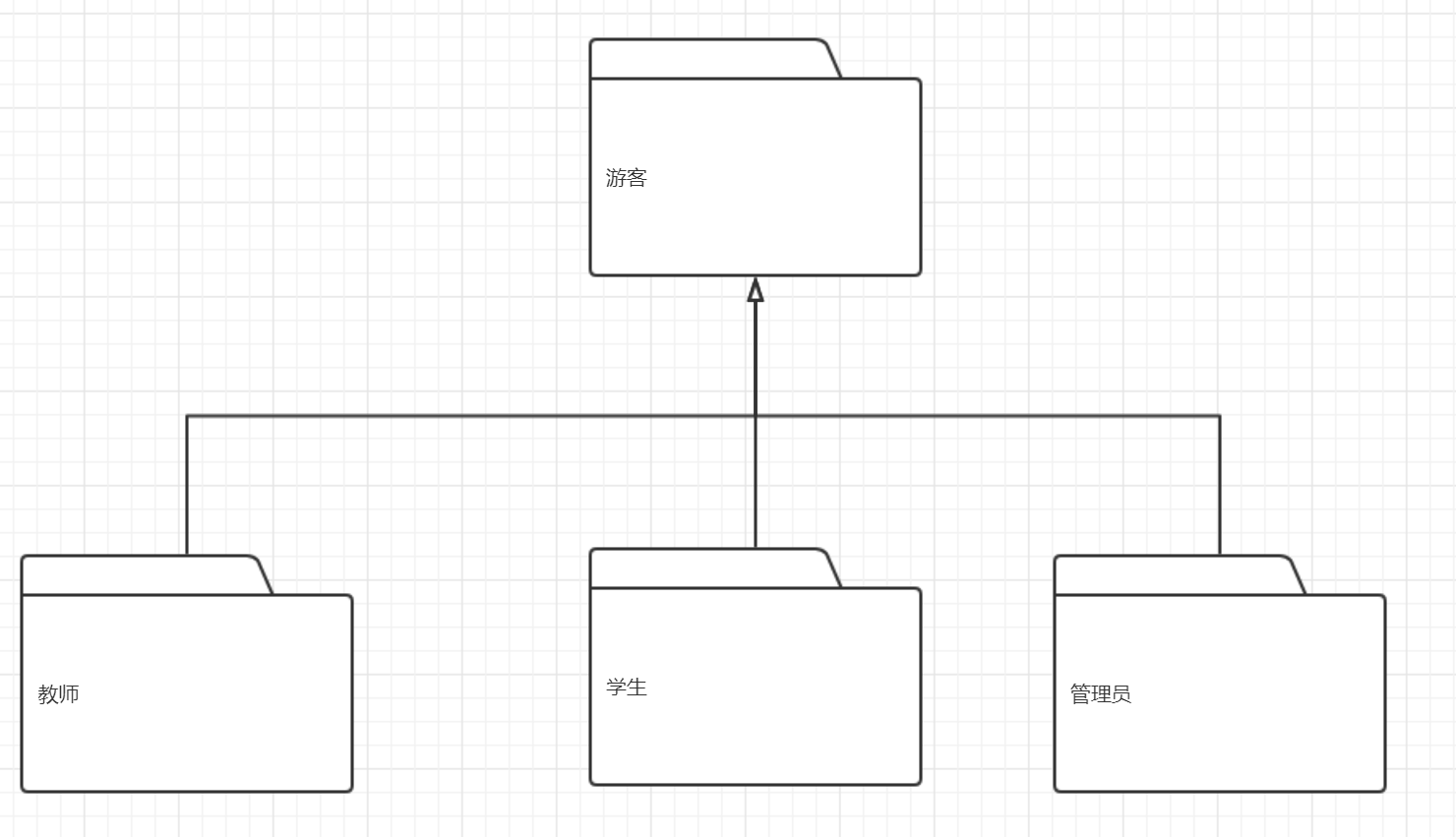
一个包中的类可以被另外一个指定包（以及嵌套于其中的那些包）中的类引用。

引入关系是依赖关系的一种需要在依赖线上增加一个《import》衍型，包之间一般依赖关系都属于引入关系。



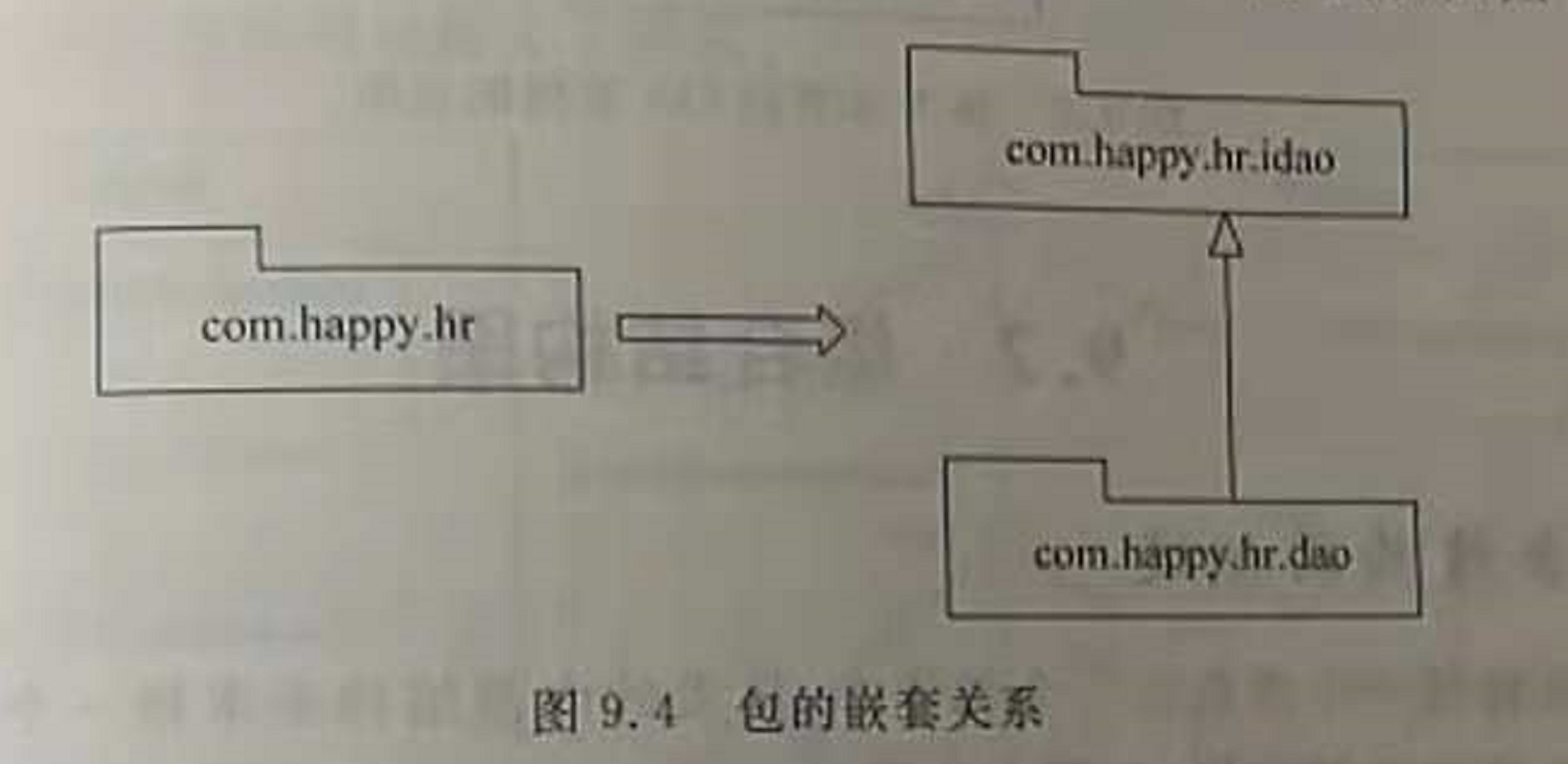
# 二：泛化关系

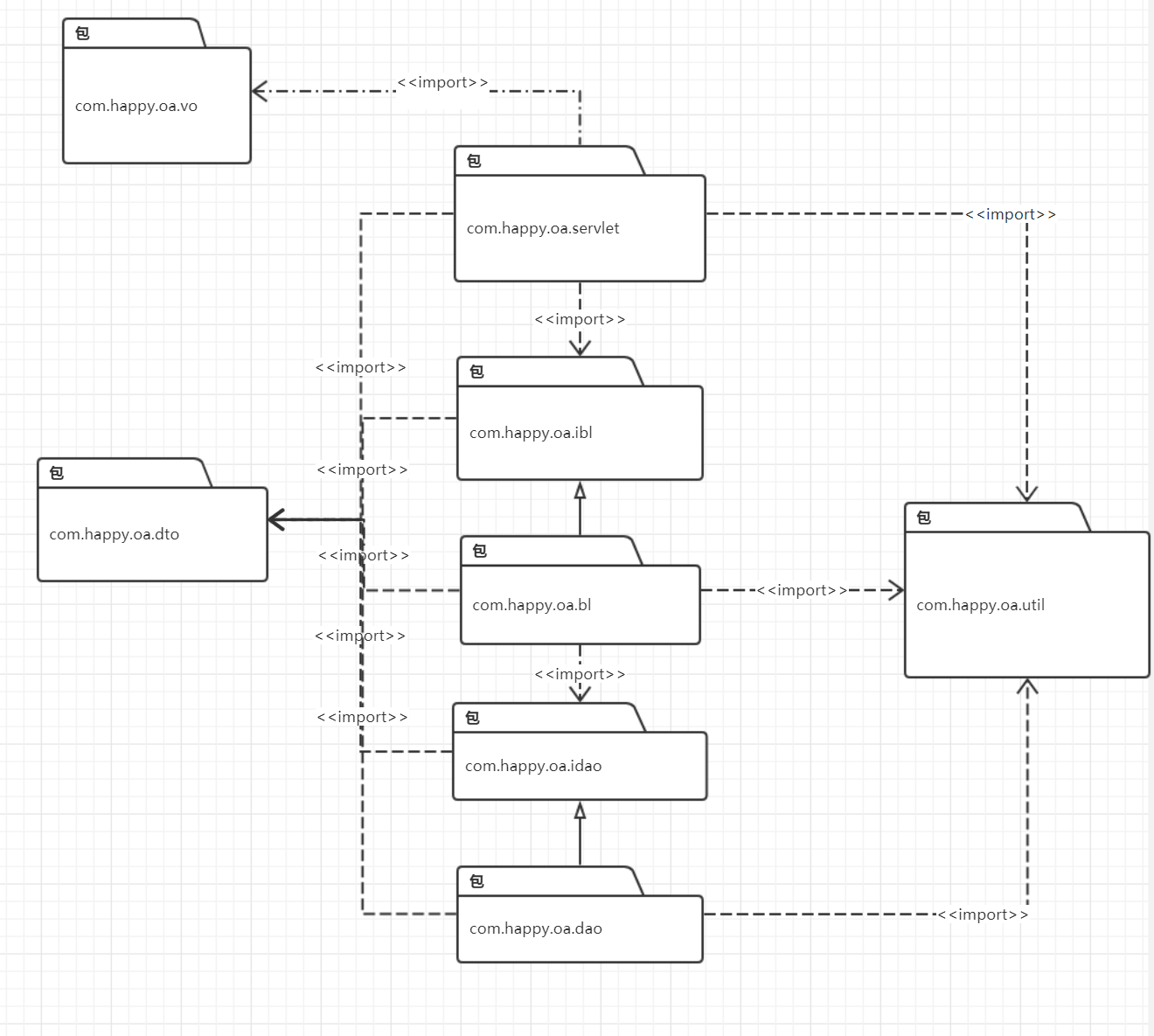
一个包继承了另一个包的全部内容，同时又补充自己增加的内容。



# 三：嵌套关系

嵌套关系:一个包中可以包含若干个子包，构成包的嵌套层次结构。

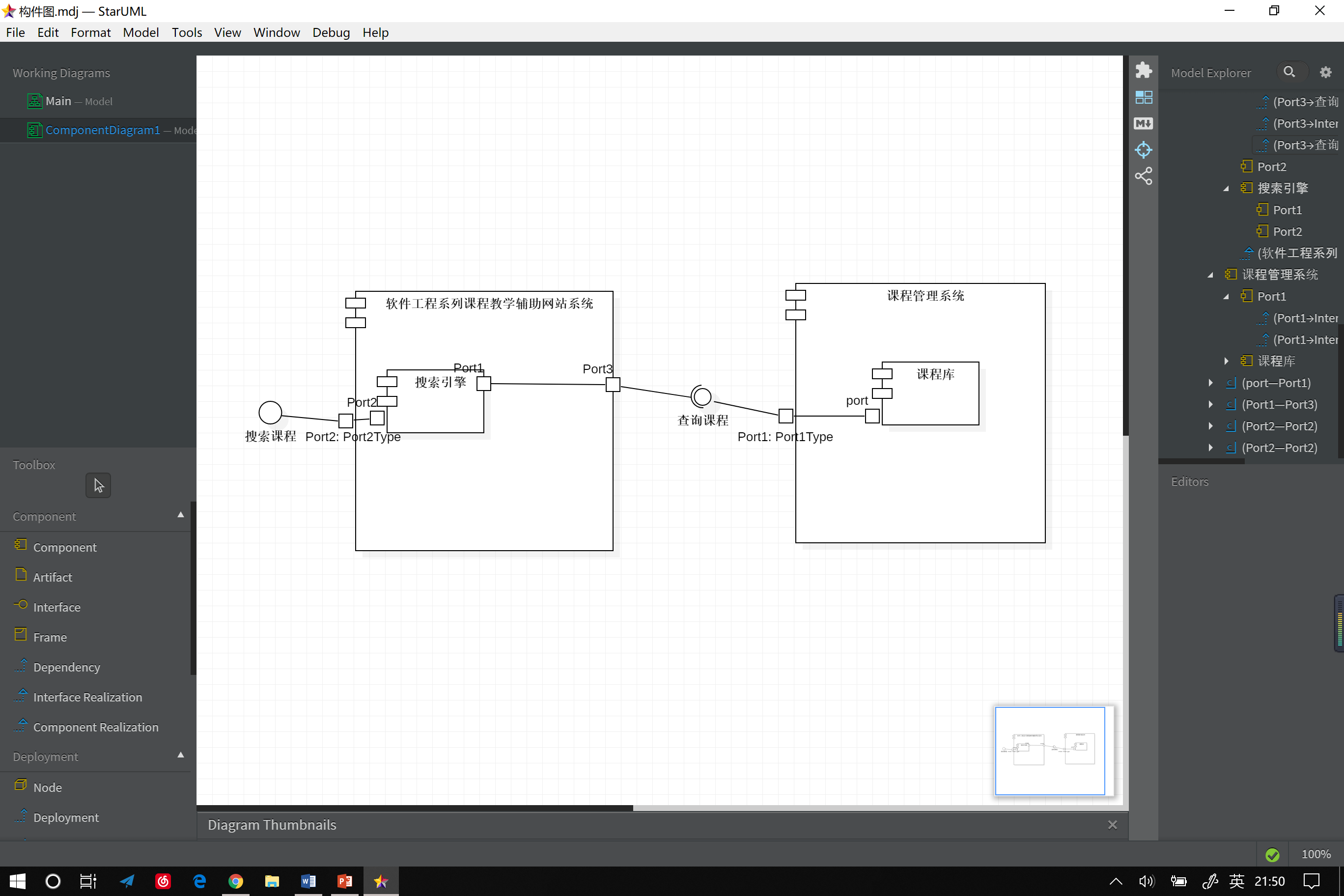




# 四、包图建模技术

**建模技术**：

* 1两种组包方式：
* ①根据系统分层架构组包（推荐使用）
* ②根据系统业务功能模块组包
* 2参照类之间的关系确定包之间的关系。
* 3减少包的嵌套层次，一般不超过三层。
* 4每个包的子包控制在7±2个
* 5如果几个包有若干相同组成部分，可优先考虑将它们 合并。
* 6可通过包图来体现系统的分层架构。



参考资料：

<https://en.wikipedia.org/wiki/Component_diagram>

<https://blog.csdn.net/fanxiaobin577328725/article/details/51647248>

https://www.jianshu.com/p/a0704aa2b092