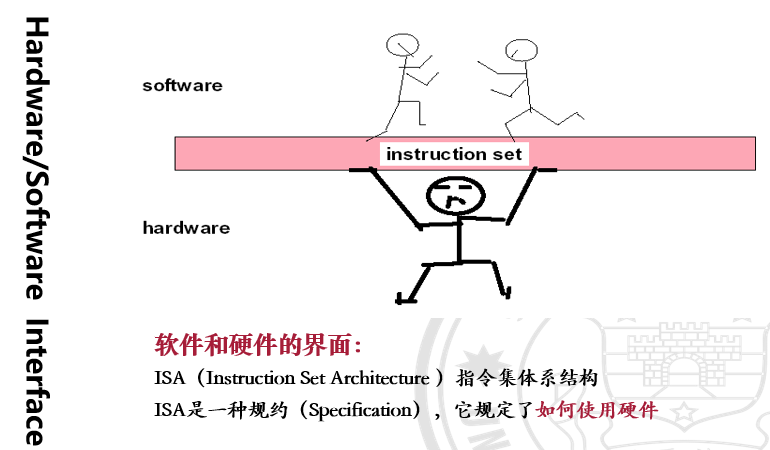
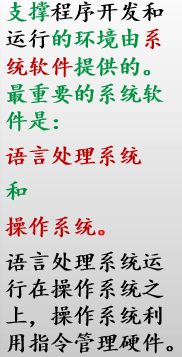


DSL（Domain Specific Language）最典型的 DSL，就是 SQL

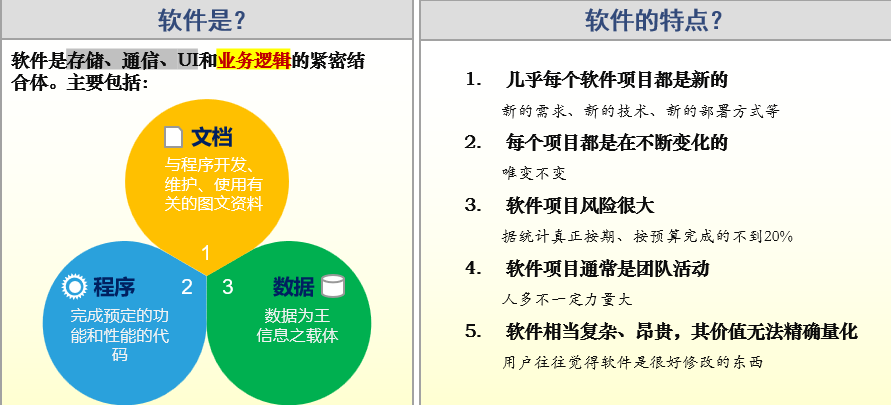


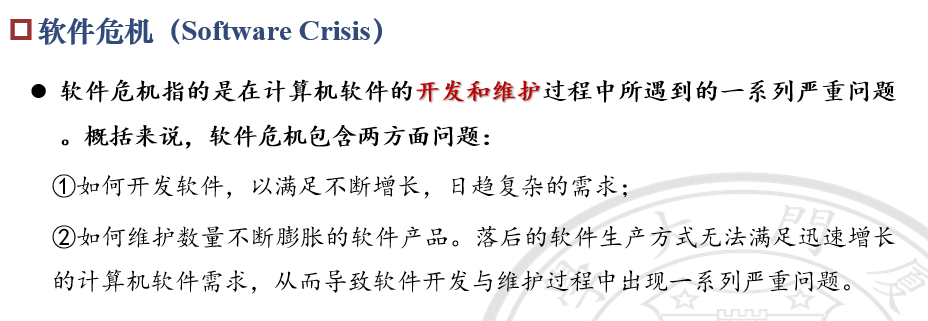




* IT（数字）素养是我们**基本**的部分，主要包括：
  + - 数据思维
    - 抽象思维
    - 系统思维

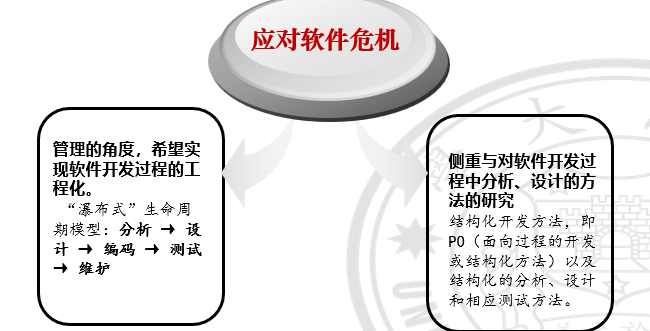


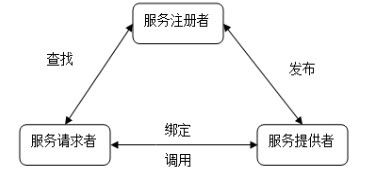


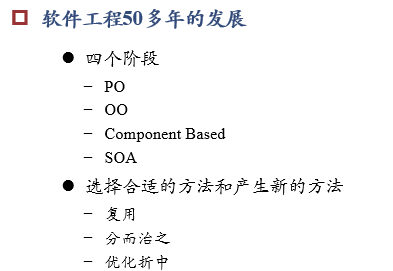


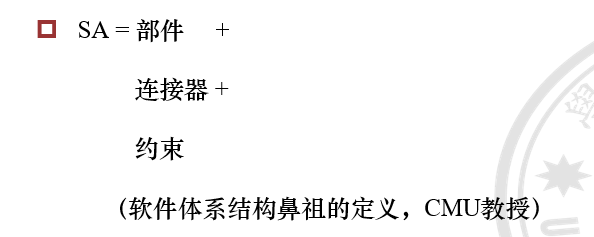
软件工程的目标是提高软件的**质量与生产率**

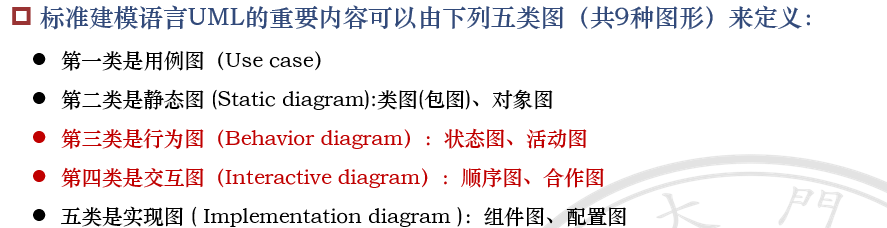






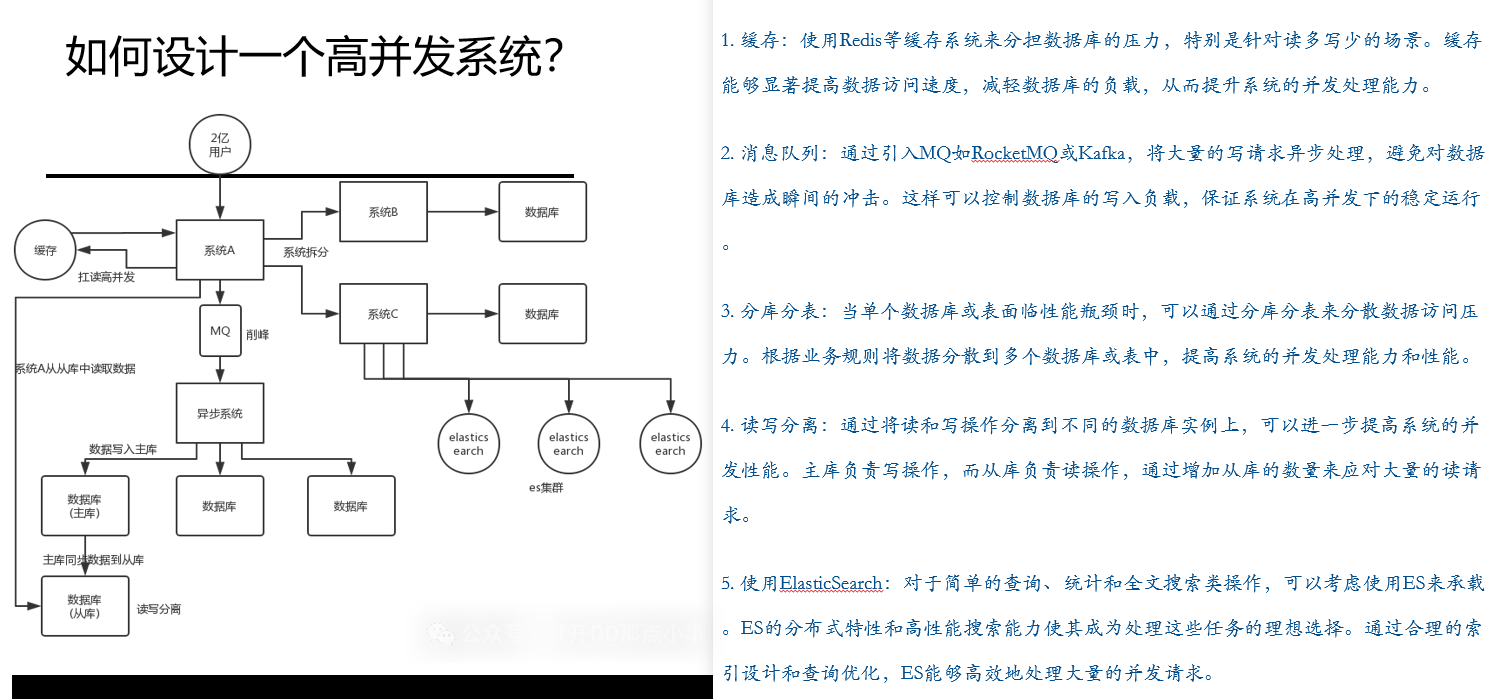


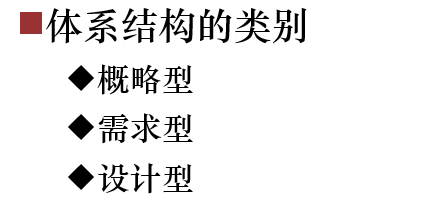


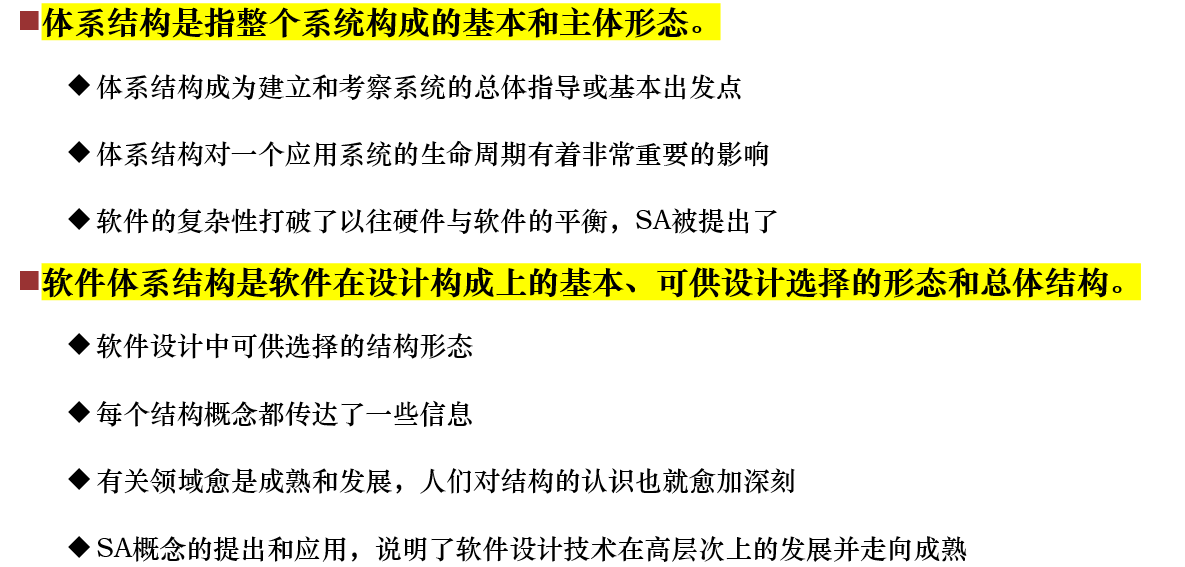


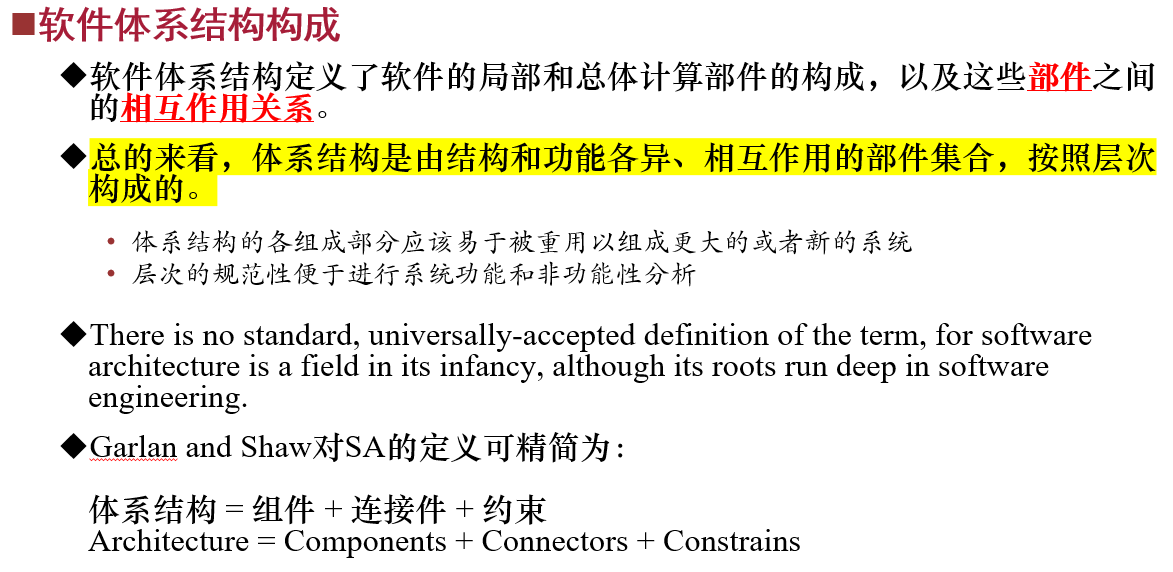


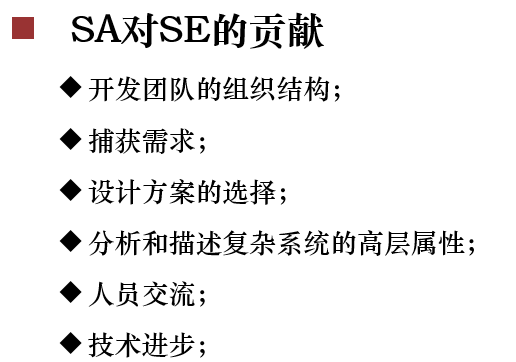
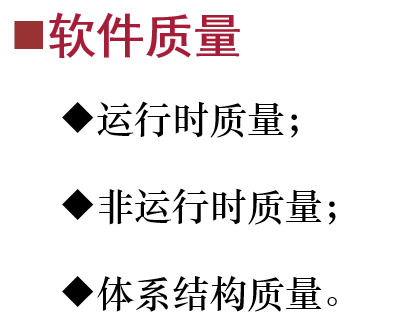




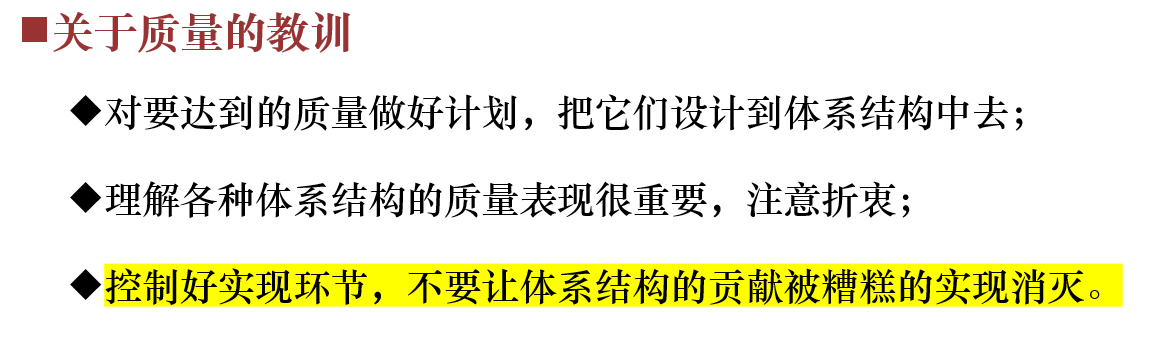


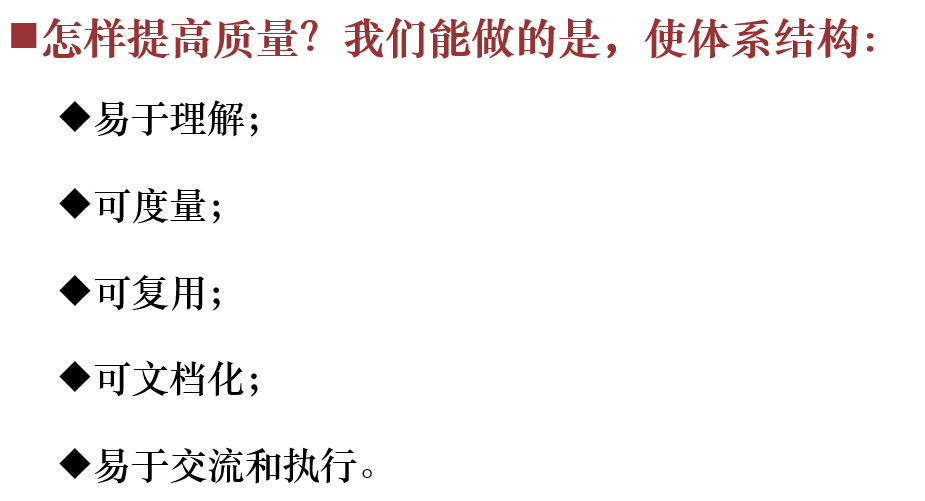


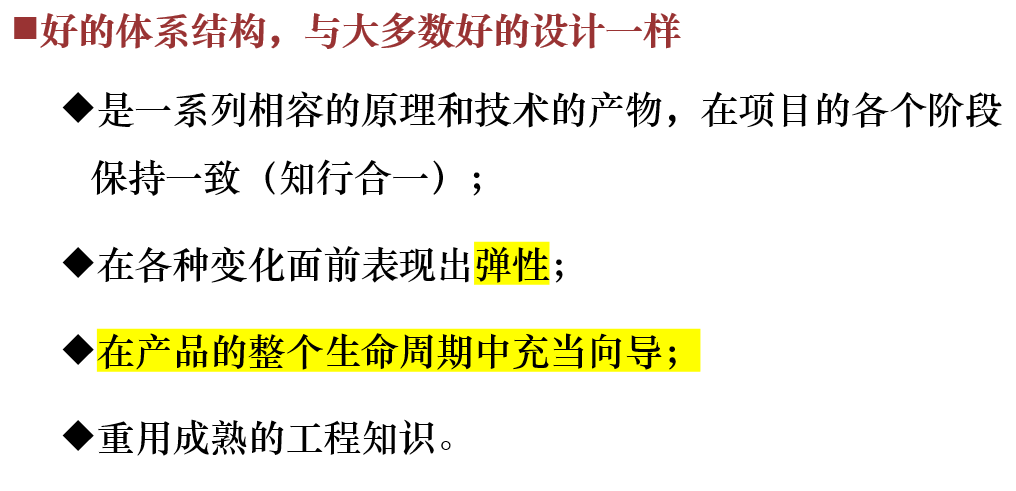












* **软件设计的层次**
  + 结构级；
  + 代码级；
  + 执行级。

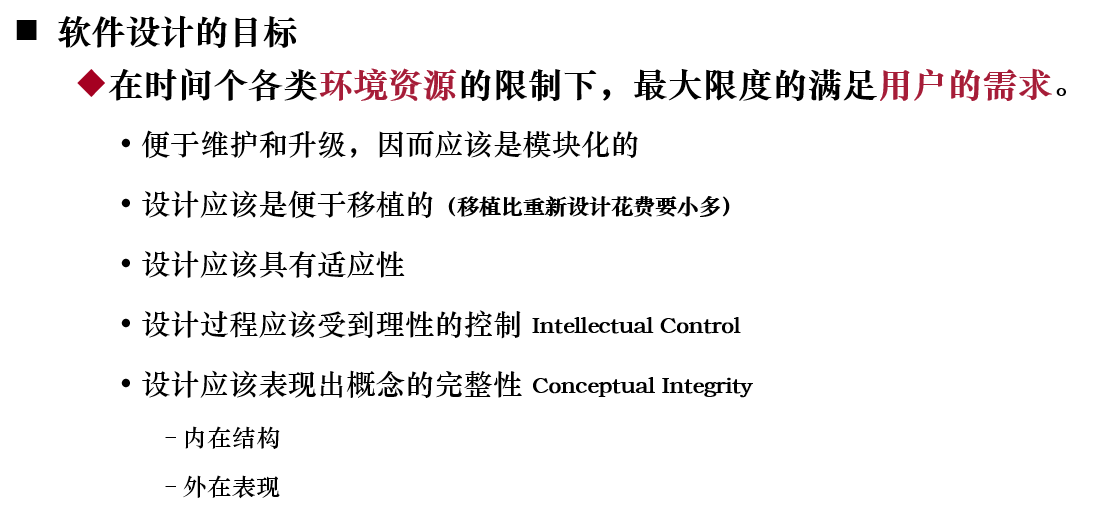
**软件复用就是具有特定性能的模式或部件在不同应用中的多次频繁使用。**



当前的软件设计问题：生产效率、可维护性、进度、质量、文档。

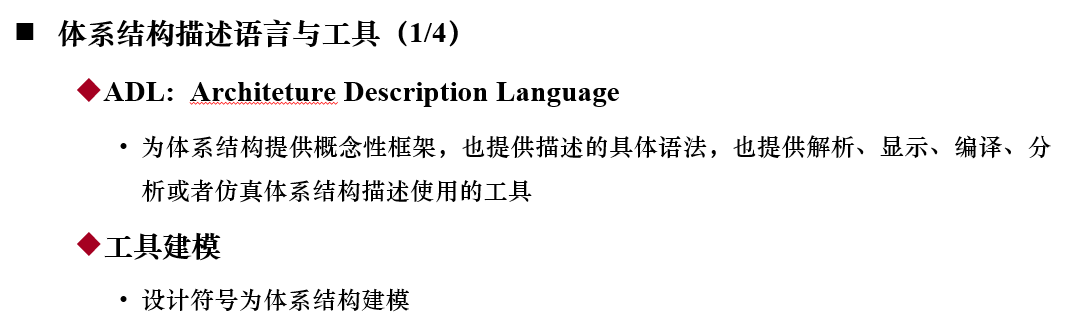
* **软件是存储、通信、UI和业务逻辑的紧密结合体。**

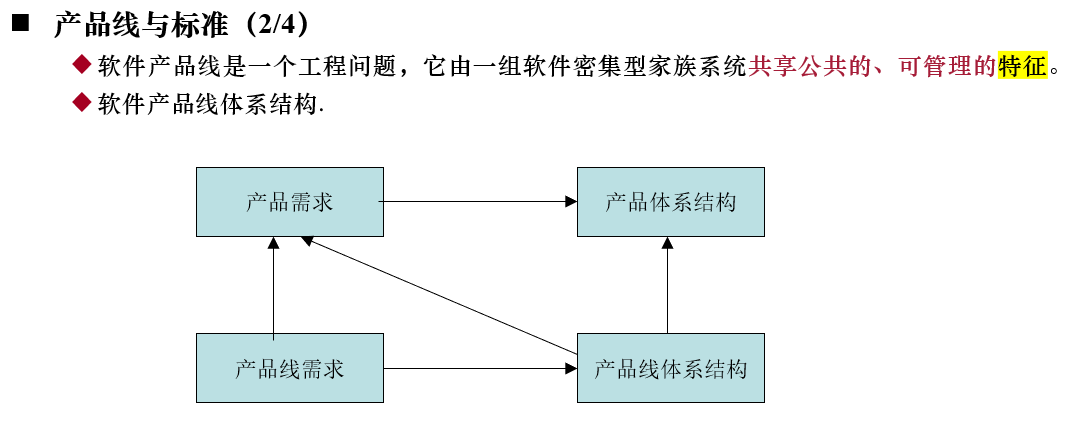
**软件即抽象，抽象是软件的本质。**

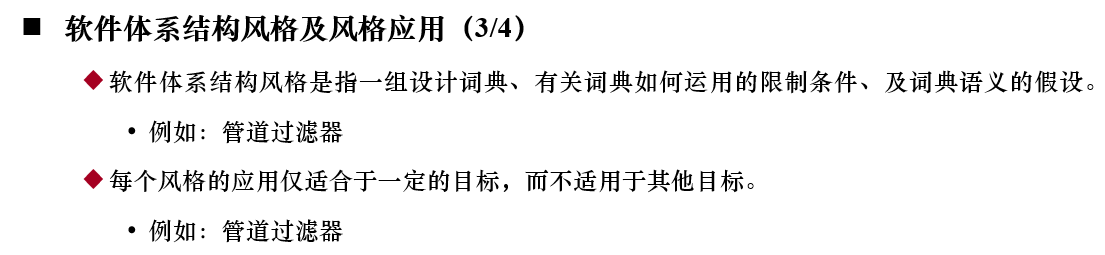


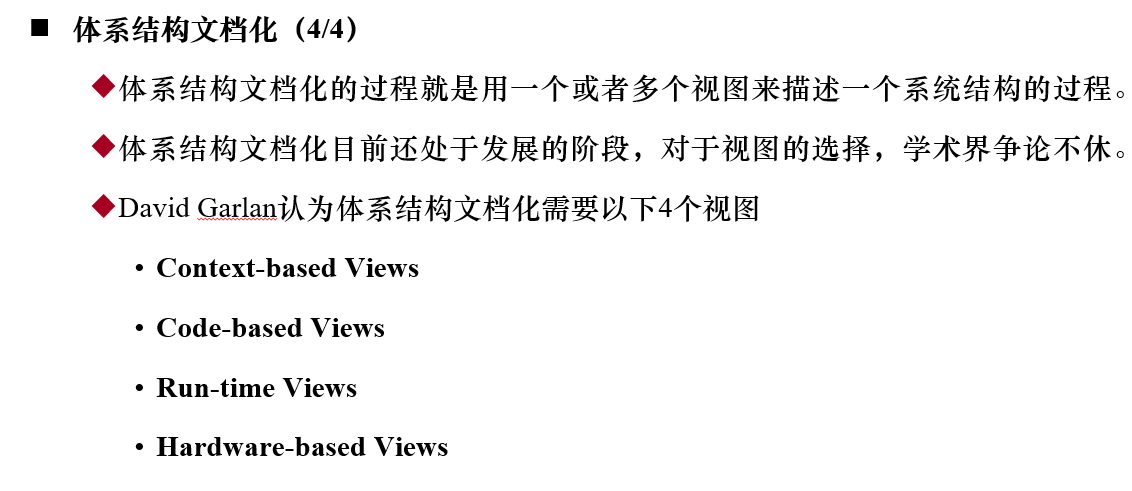
**需求来自于企业、架构来自于需求，系统来自于架构**

* + 从设计模式角度来看，框架为大粒度的可复用的部件
  + 从体系结构角度来看，框架是一个特定领域体系结构

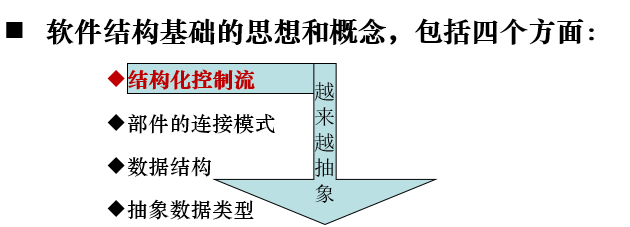


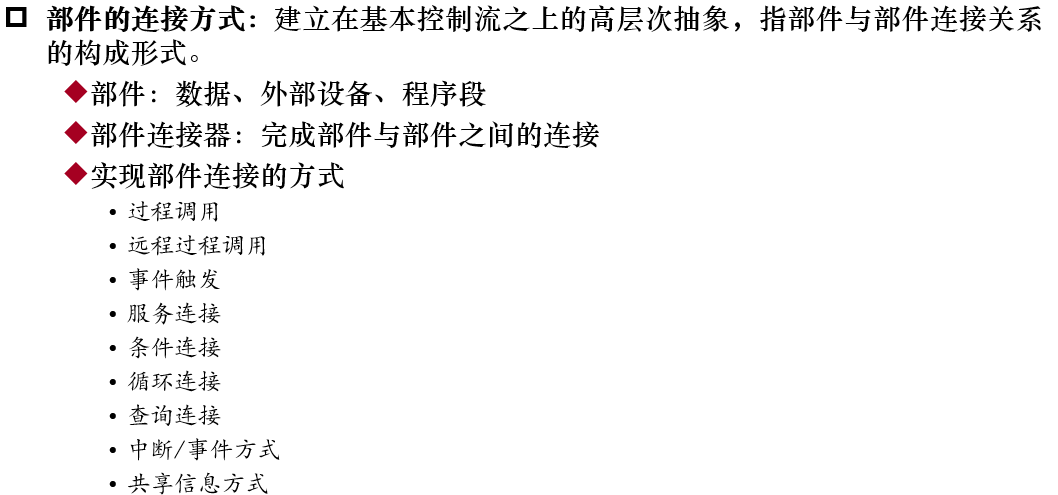


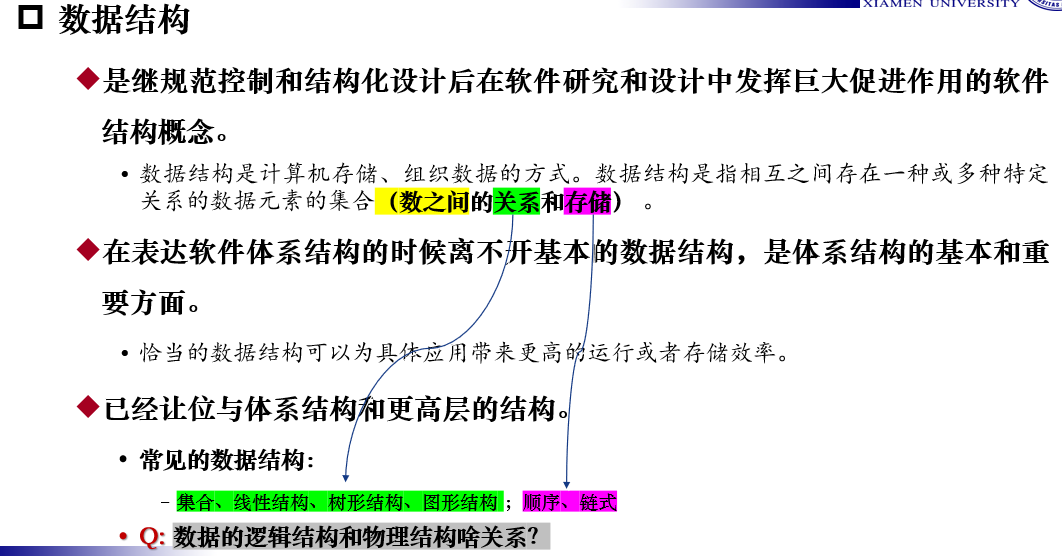


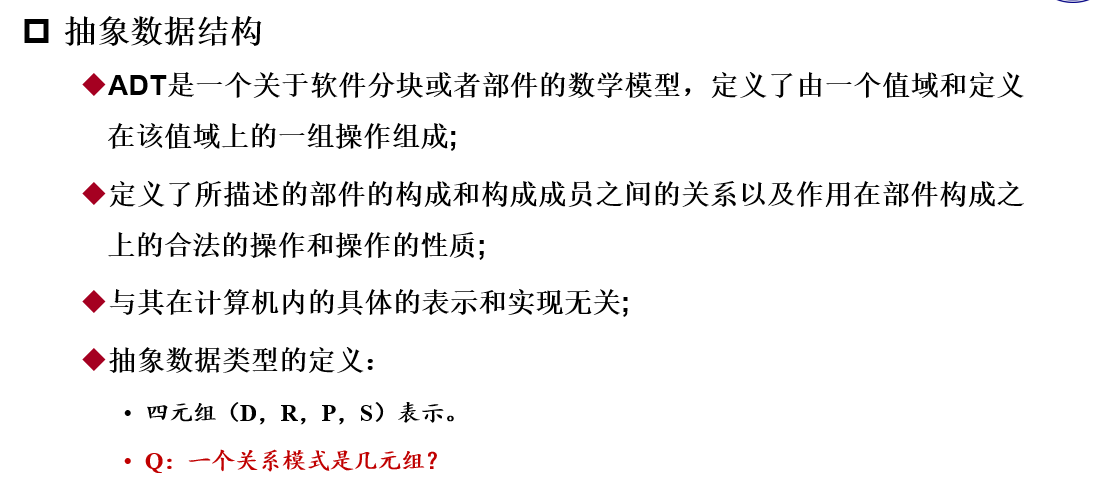


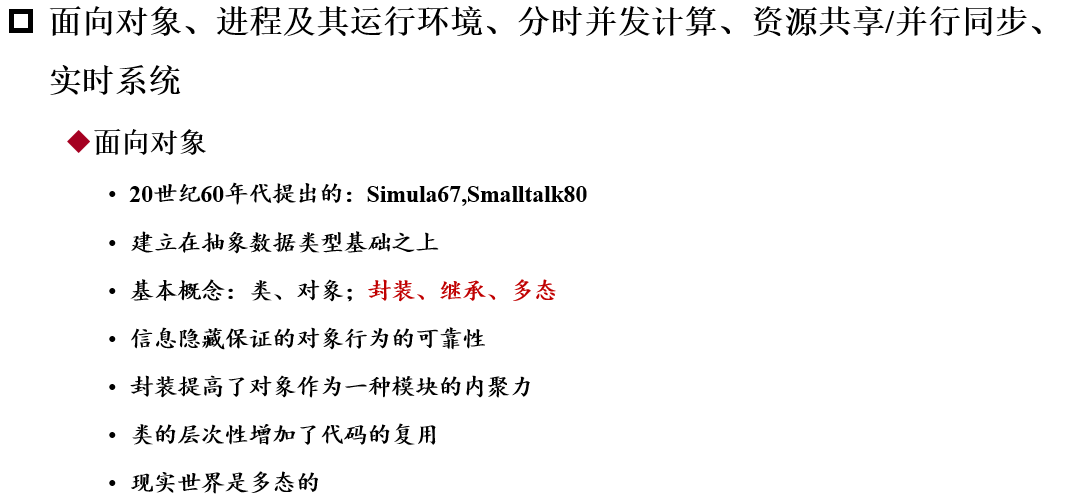
* **软件设计的物质基础是当前的计算机硬件，它决定了软件设计和实现的出发点。**





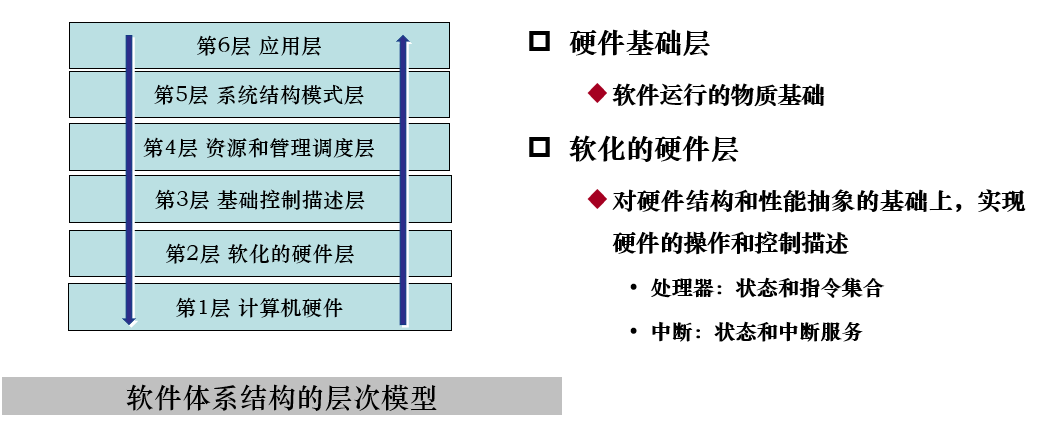


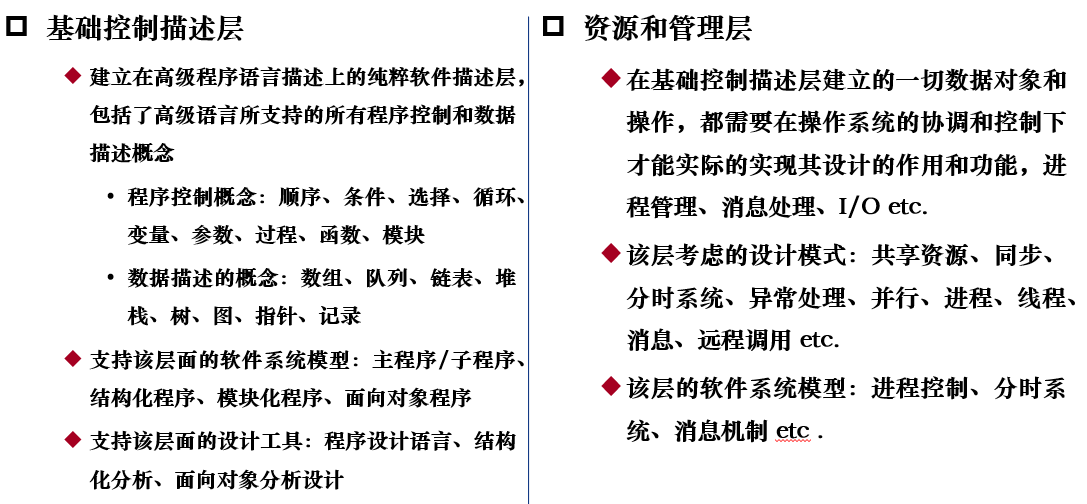


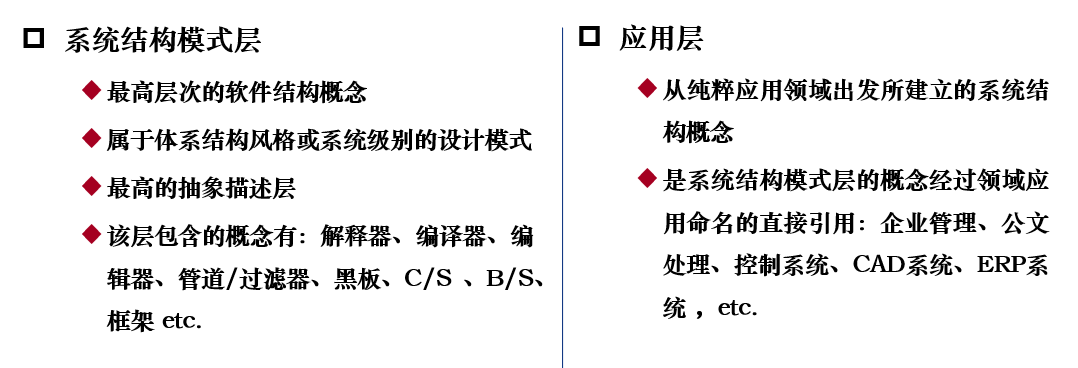


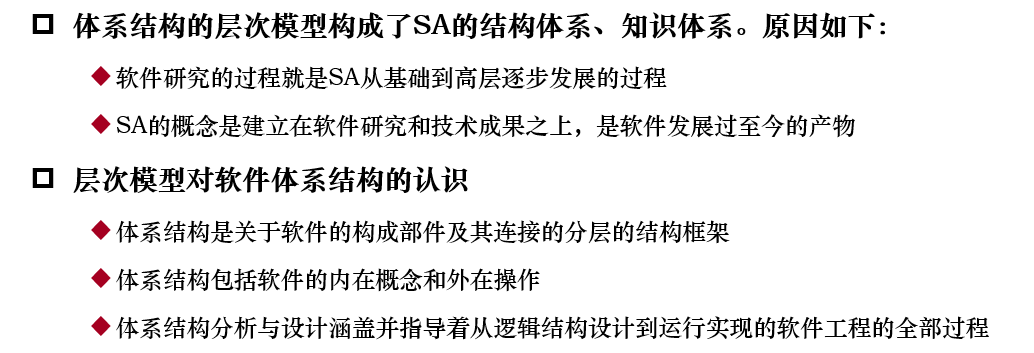
**任何具有固定组成形式的数据、代码、数据集合、代码序列、数据和代码的结合体都可以称作结构**

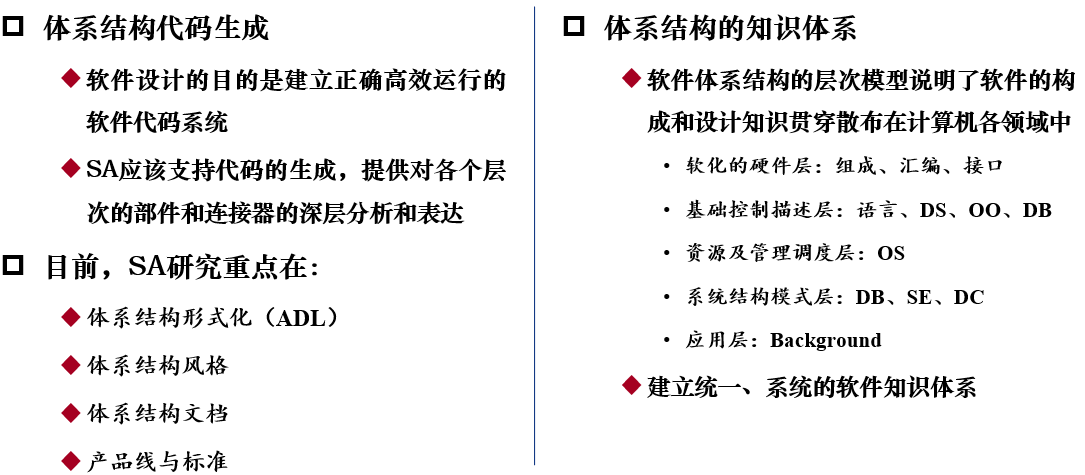
* **层次系统（Layered Systems）是一种体系结构风格**

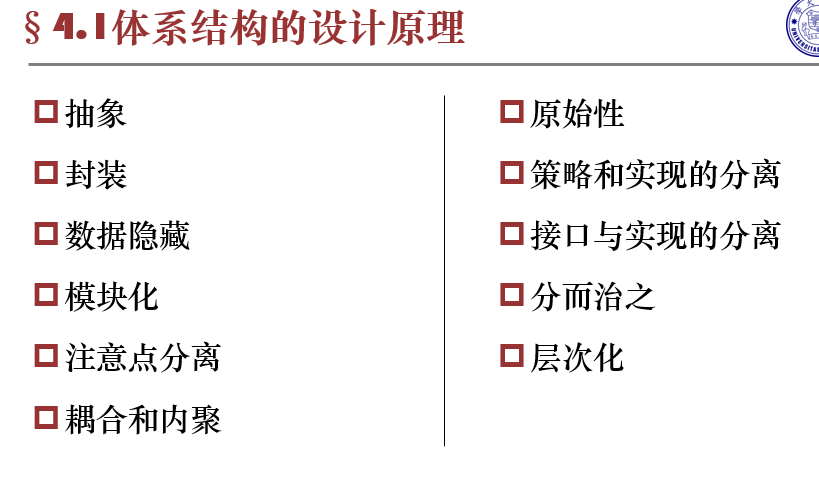


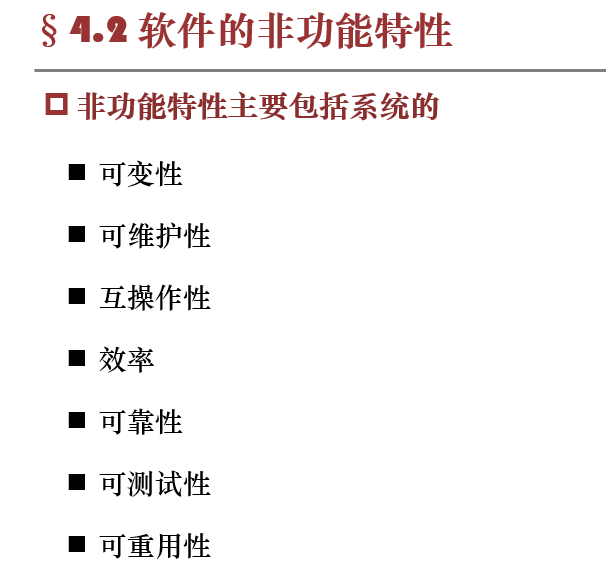




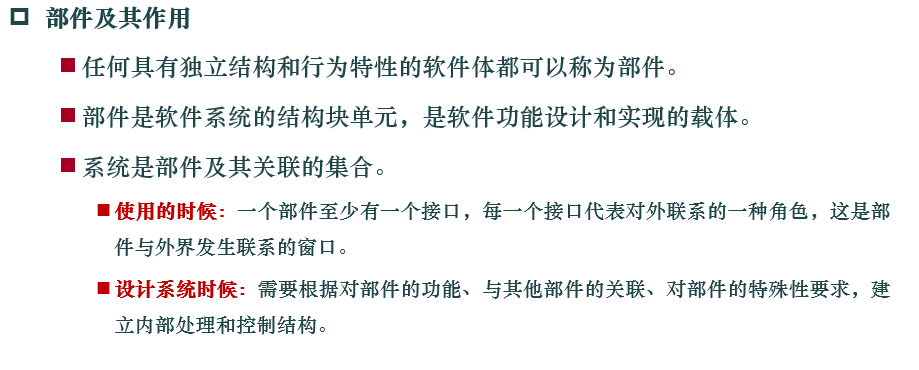


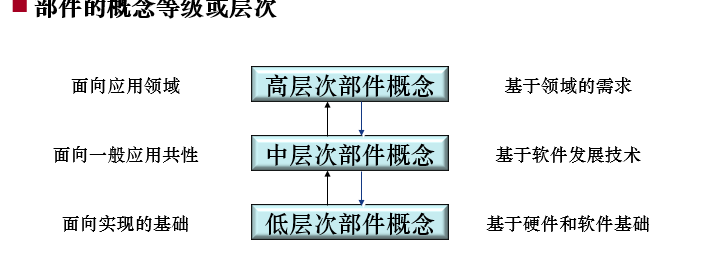


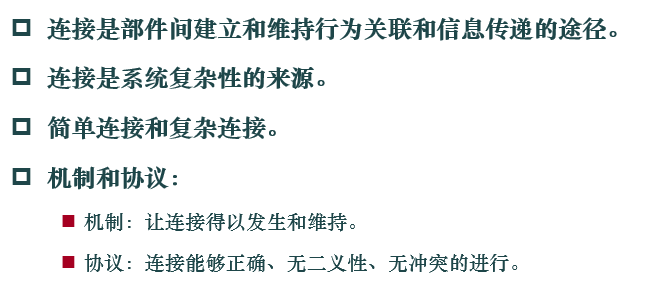
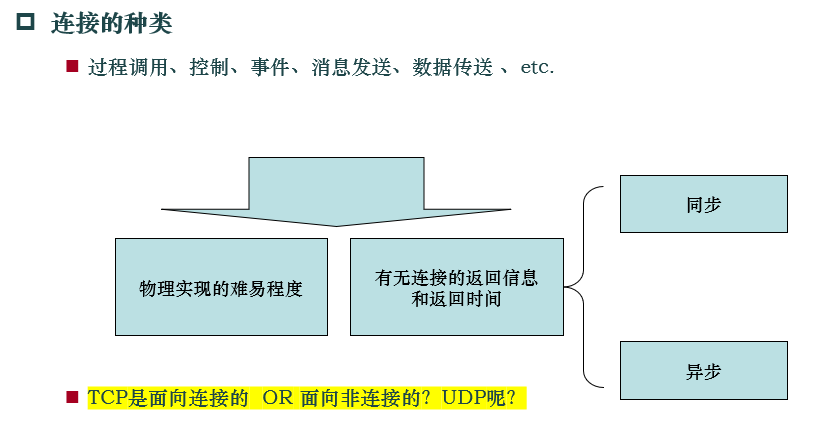




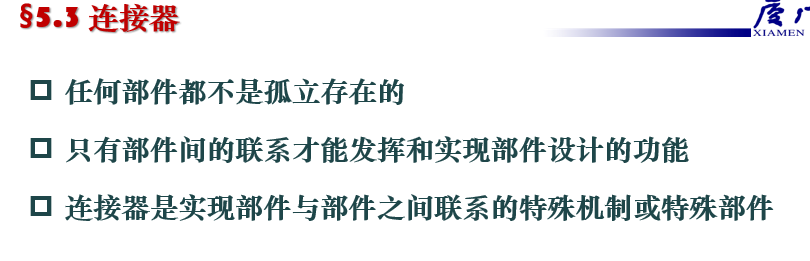
* 软件工程师的主要任务：
  + 将他们的科学和工程知识应用于技术问题的解决方案，然后在需求和约束条件的范围内，根据素材、技术、经济、法律、环境以及与人有关的条件优化这些方案。
* **体系结构设计将主要精力放在将一个系统分解为组件以及这些组件之间的相互作用上。**
* **部件和连接器被公认为体系结构的两大类构成部分：**
  + **部件是软件功能设计和实现的载体。**
  + **连接器是专门承担连接作用的特殊部件。**

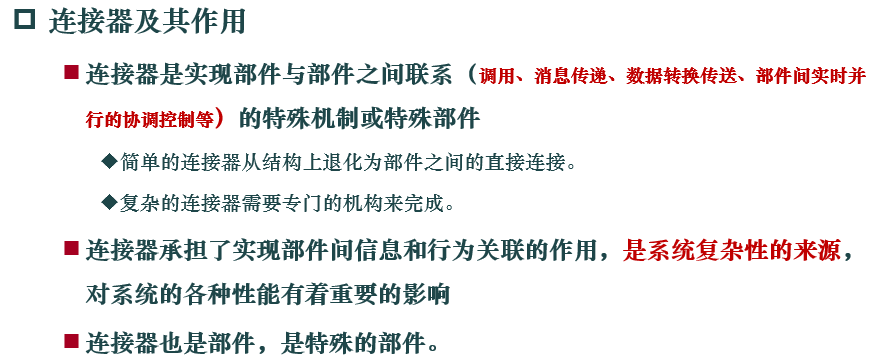




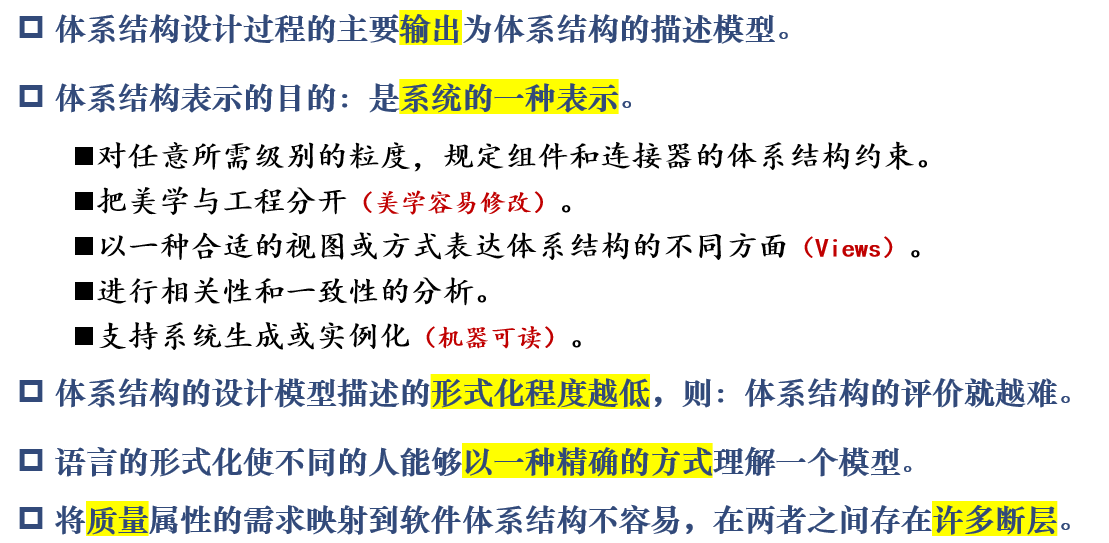
 

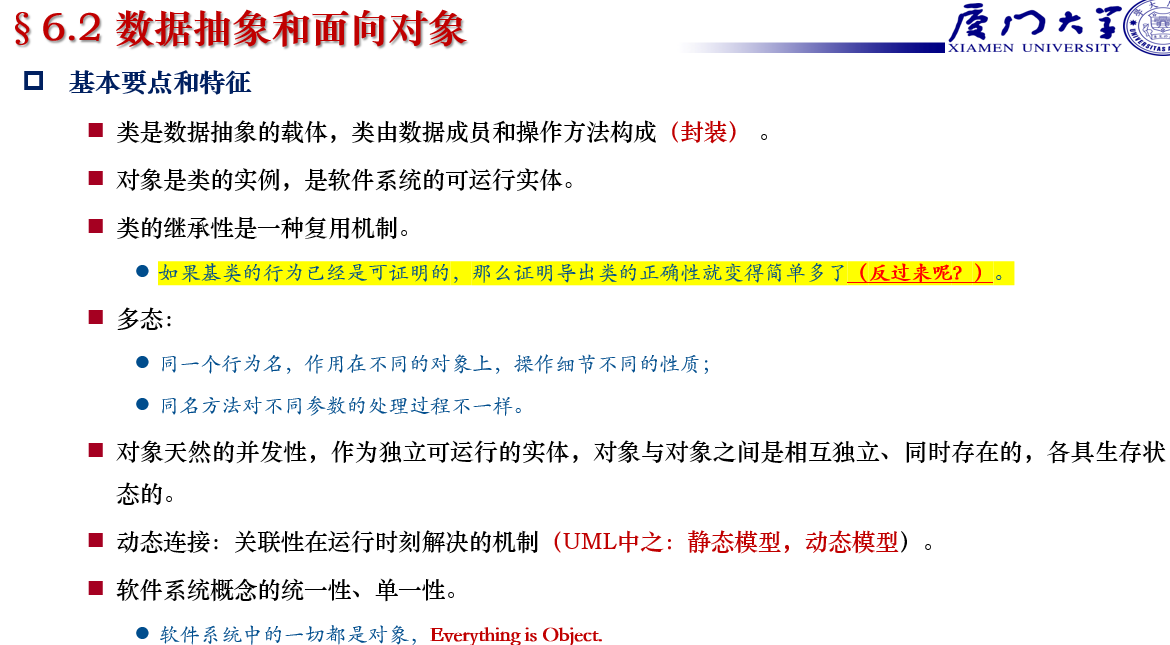
* + 无论多么复杂的连接关系，其实现都是基于以下基本连接机制：
    - 过程调用、中断、I/O、DMA、进程、线程、共享、同步、并/串行、事件、**并发、并行（区别是？）** etc.



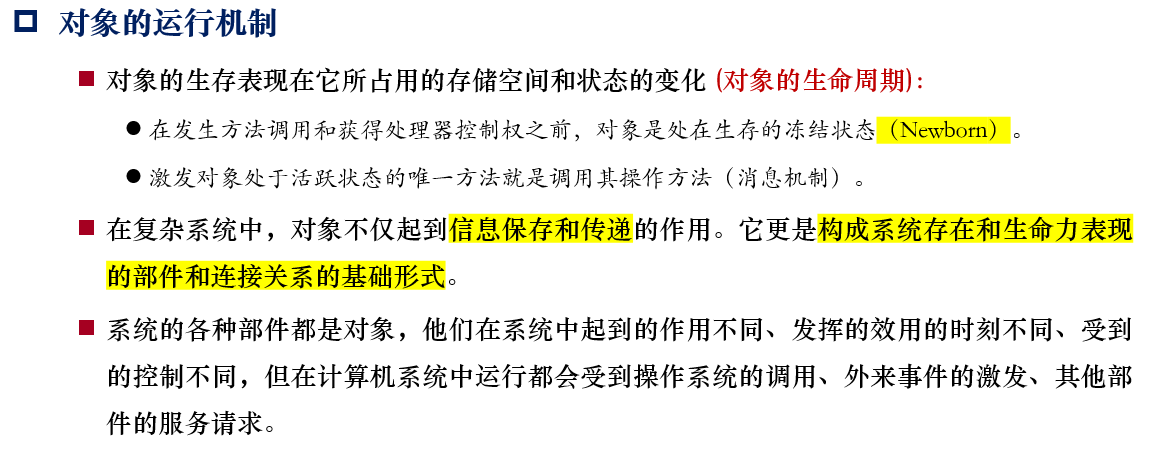


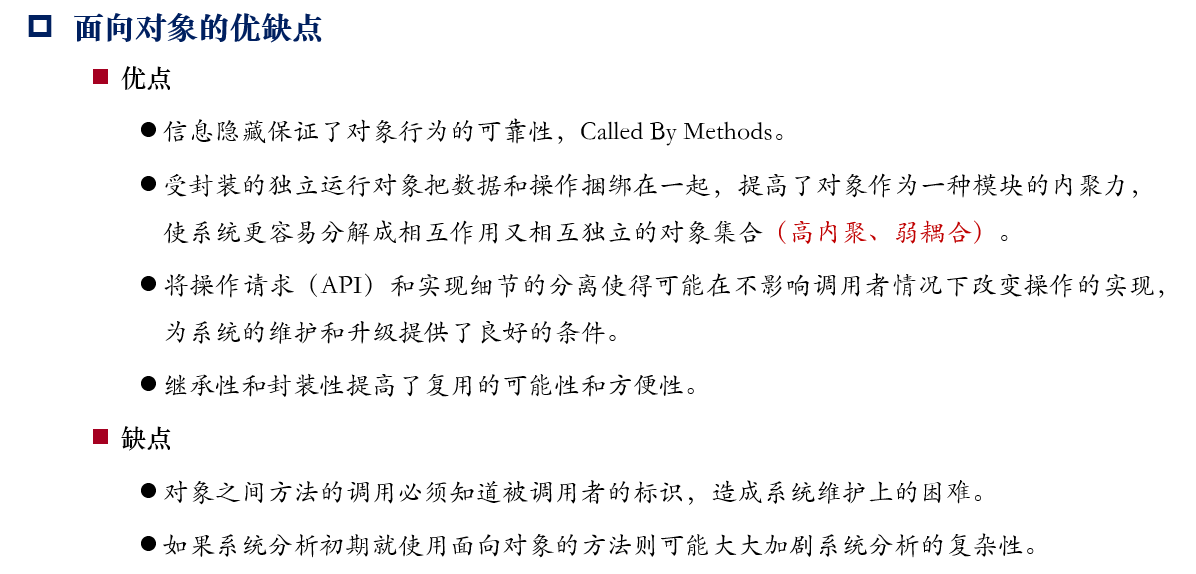
* 体系结构的设计过程由下列步骤组成：
* 理解问题；
* 确定设计元素及其关系；
* 评价体系结构；
* 转换体系结构。

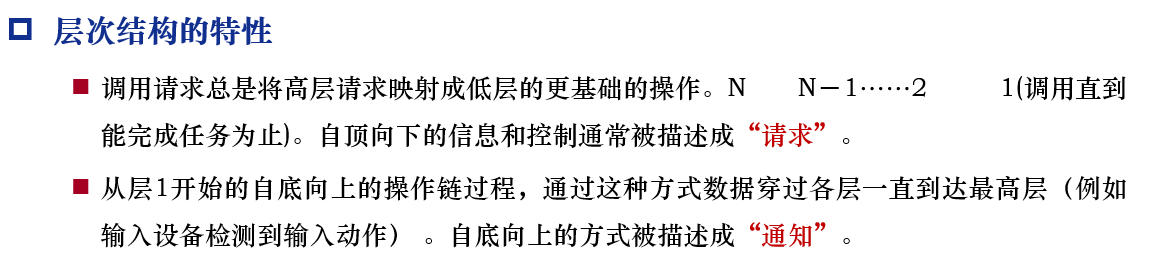
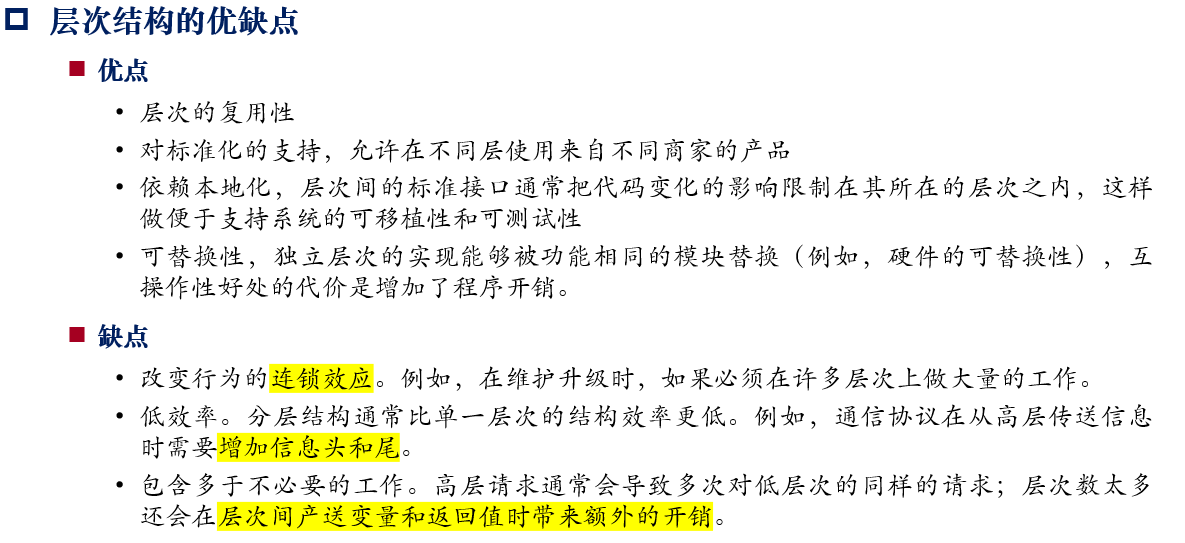
  

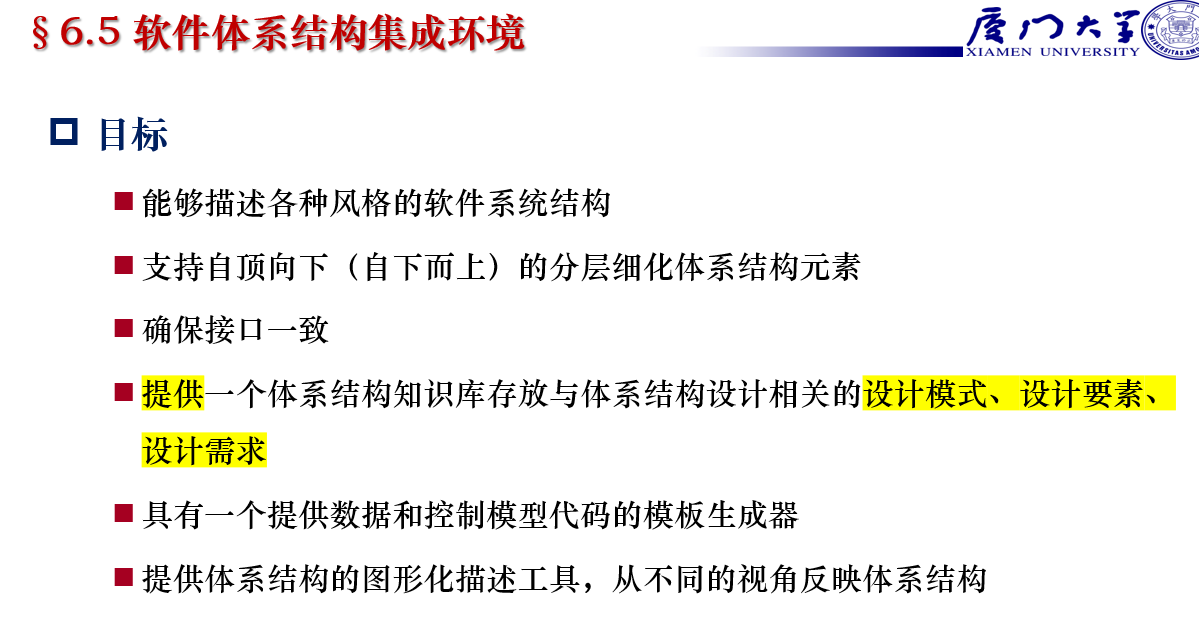



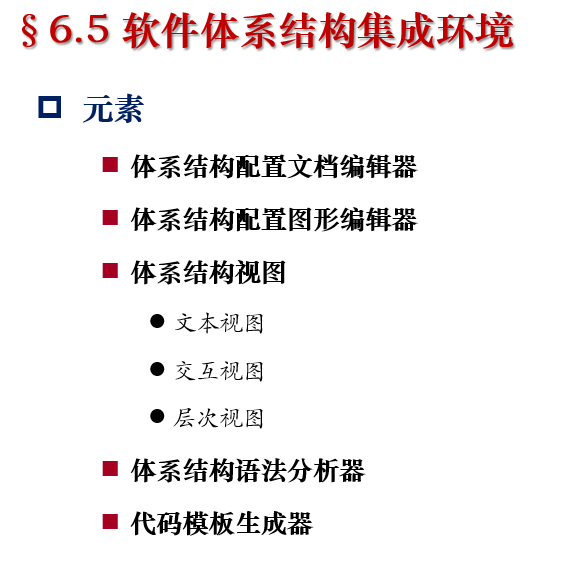
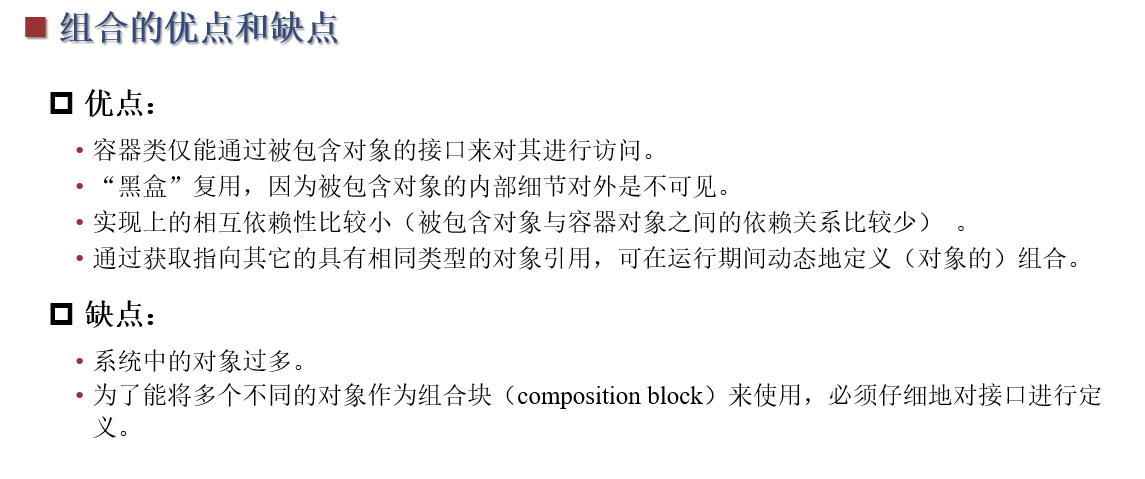
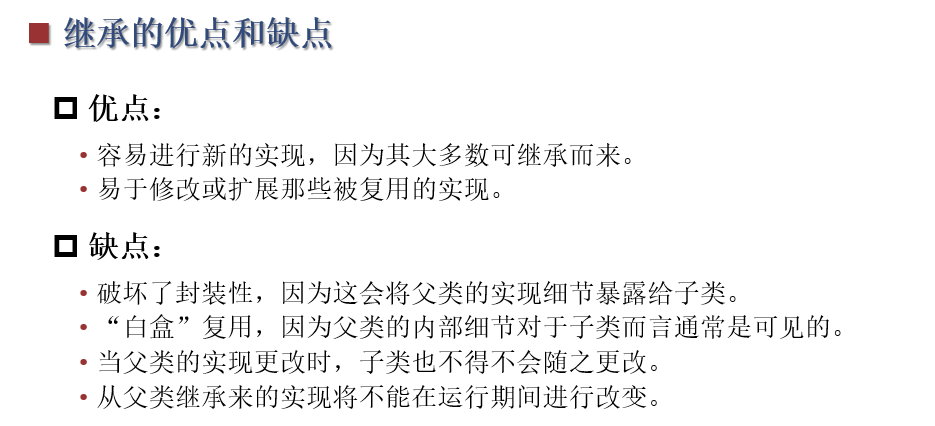




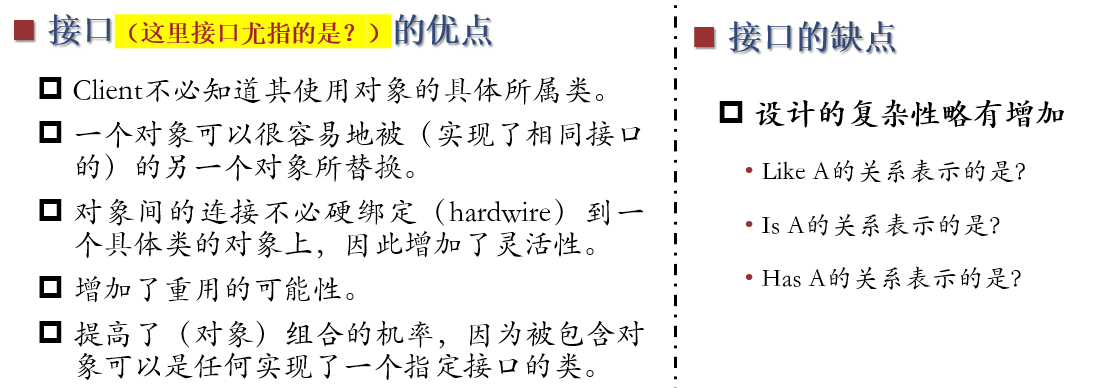


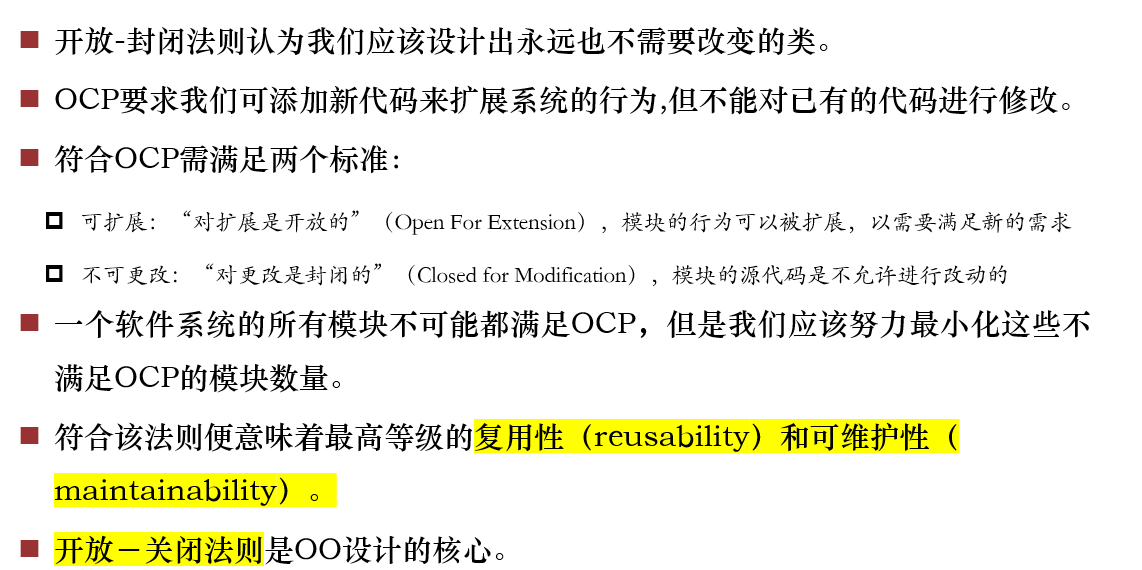
 



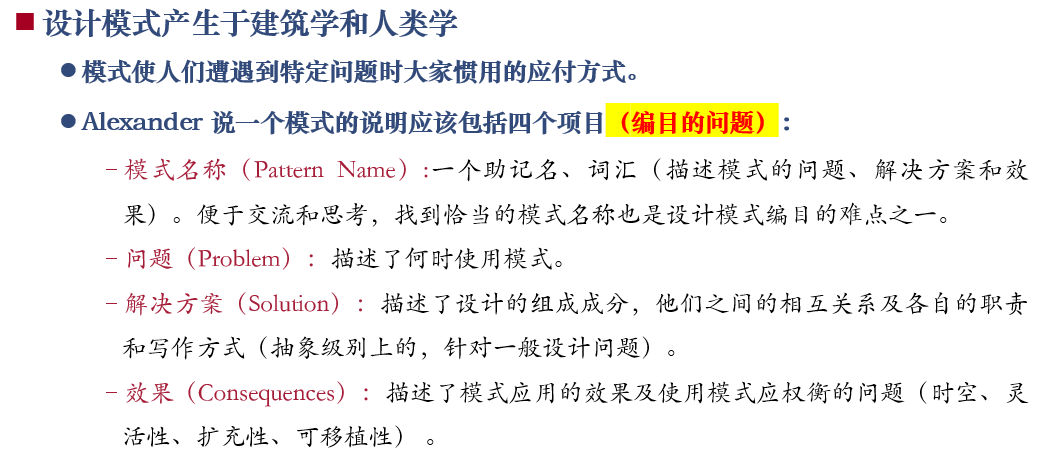
  

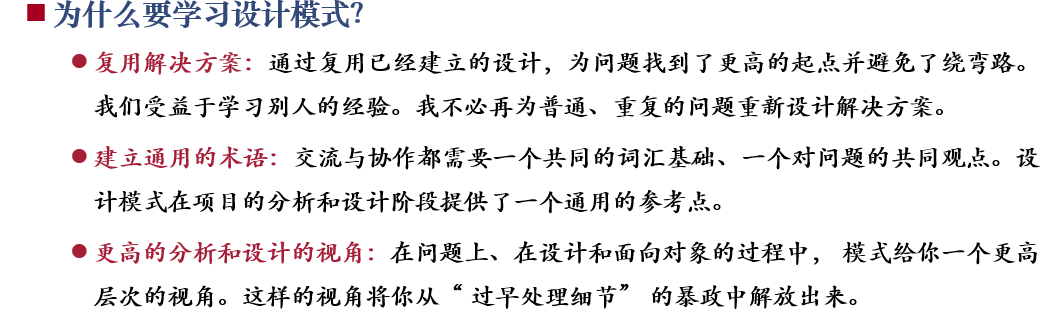
* + **接口是一个对象在对其它的对象进行调用时所知道的方法集合**

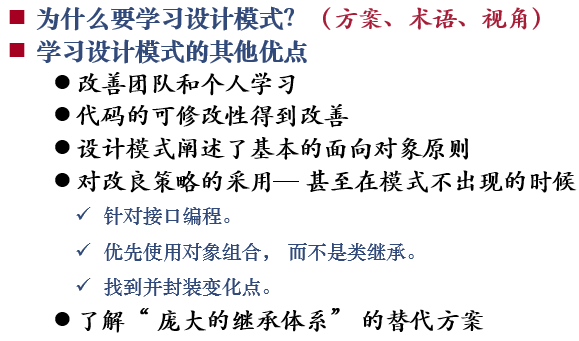




* + **LSP保证一个子类总是能够被用在其基类可以出现的地方！**





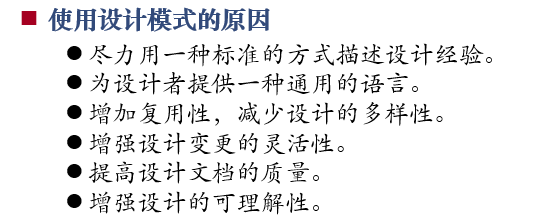


* + “设计模式”就是：**在特定环境(Context) 下﹐对特定问题(Problem)的惯用解决之道(Solution)。**
  + MVC还使用了其它的设计模式。

如：用来指定视图缺省控制器的Factory Method和用来增加视图滚动的Decorator [(装饰器)](http://www.jdon.com/designpatterns/decorator.htm) 。MVC

的主要关系是由Observer、Composite(组合)和Strategy三个设计模式给出的。

* + - * + MVC模式是一种架构模式(组合模式), 需要其他模式协作完成 。



软件模式并非仅限于设计模式，还包括架构模式、分析模式和过程模式（界面模式）

