软件工程基础 —— 第12章 体系结构设计

计算机学院 孟宇龙

目录

关键概念

- 敏捷和体系结构
- 原型
- 体系结构
- 体系结构决策
- 体系结构描述语言
- 体系结构描述
- 体系结构设计
- 体系结构类型
- 体系结构模式
- 体系结构风格
- 体系结构一致性
- 体系结构细化
- 检查

12.1 软件体系结构

从第一个程序被划分成模块开始，软件系统就有了体系结构

有效的体系结构及其明确的表示和设计

已经成为软件工程领域的主导主题

12.1.1 什么是体系结构

体系结构：数以千计的或大或小的决定

软件体系结构：程序或计算机系统的软件体系结构是指系统的一个或多个结构，它包括软件构件、构件的外部可见属性以及它们之间的相互关系

体系结构并非可运行的软件，确切地说，它是一种表达，使能够：

(1) 对设计在满足既定需求方面的有效性进行分析；

(2) 在设计变更相对容易的阶段，考虑体系结构可能的选择方案；

(3) 降低与软件构造相关的风险。

12.1.1 什么是体系结构

- 在体系结构层次上，不会详细说明内部属性
- **构件，可以简单和复杂**
- **构件之间的关系，可以像从一个模块对另一个模块进行过程调用那样简单，也可以像数据库访问协议那样复杂**

12.1.2 体系结构为什么重要？

- **软件体系结构的表示有助于**对计算机系统开发感兴趣的各方（利益相关者）开展交流。
- **体系结构突出了早期的设计决策**，这些决策对随后所有的软件工程工作有深远影响，同时对系统作为一个可运行实体的最后成功有重要作用。
- **体系结构“构建了一个相对小的、容易理解的模型，**该模型描述了系统如何构成以及其构件如何一起工作”[BAS03]。

12.1.3 体系结构描述

体系结构描述是一组体现系统不同视图的工作产品

- 程序员：体系结构是一个蓝图
- 体系结构为后续协商奠定基础，应该是简介易懂的
- 体系结构是对成本、可用性、可维护性、性能等属性权衡后的一种决策

- **体系结构描述必须展示出不同视角所组合的特征**

12.1.3 体系结构描述

体系结构描述是一组体现系统不同视图的工作产品

- IEEE 计算机学会提出了IEEE-Std-1471-2000，密集型软件系统体系结构描述的推荐实践做法：[IEE00]
  1. 建立软件体系结构设计过程中使用的概念性框架和词汇表，
  2. 提供表示体系结构描述的详细准则，
  3. 鼓励良好的体系结构设计实践。
- IEEE 标准将体系结构描述(AD) 定义为“记录体系结构的产品集合”。
  - 该描述本身使用多视图来表达，这里的每个视图是“从一组参与者关注点的角度观察整个系统的一种表示”。

12.1.4 体系结构决策

体系结构描述是一组体现系统不同视图的工作产品

- 视图：体系结构的一部分，解决一个特定利益相关者的关注点
- 设计师要做出各种决策
- 可将体系结构决策本身看做体系结构的一种视图
- 一个模板.....P115

12.2 体系结构类型

许多不同的体系结构风格可以用于一种特定的类型

- 类型隐含了整个软件领域中的一个特定类型。
- 在每种类别中，会有很多的子类别。
  - 例如，在建筑物类型中，大致会有以下几种一般风格：房子、单元楼、公寓、办公楼、工厂厂房、仓库等。
  - 在每一种一般风格中，也会运用更多的具体风格。每种风格有一个结构，可以用一组可预测模式进行描述。

几种体系结构类型

- 人工智能，模拟或扩大人类认知、运动或其他有机体过程的系统。
- 商业和非盈利的，工商企业营运必要的系统。
- 通信，提供用于数据传输、管理、用户连接或者展示的基础设施的系统。
- 内容创作，用于创建或管理文字或多媒体人造物品的系统
- 设备，与物理世界交互的系统，可以为个人提供某种服务
- 娱乐和运动，管理公众事件或者提供大众娱乐体验的系统
- 金融，为转账和理财及其他安全事务提供基础设施的系统
- 游戏，为个人或者群体提供娱乐体验的系统
- 行政管理，支持地方或者其他政治实体的管理和运作方式的系统

几种体系结构类型（续）

- 工业，模拟或者控制物理过程的系统。
- 法律，支持法律的系统。
- 医疗，诊断或治疗，或者有助于医学研究的系统。
- 军事，用于商议、通信等方面的系统
- 操作系统，在硬件之上提供基本软件服务的系统
- 平台，在操作系统之上提供高级服务的系统
- 科学，用于科学研究和应用的系统
- 工具，用来开发其他系统的系统
- 运输，控制水上、地面、空中等交通工具的系统
- 实用程序，与其他软件交互作用的系统

12.3 体系结构风格

体系结构风格是施加在整个系统设计上的一种变换

每种风格描述一种系统类别，包括：
（1）完成系统需要的某种功能的一组构件（例如，数据库、计算模块）；
（2）能使构件间实现“通信、合作和协调”的一组连接件；
（3）定义构件如何集成成为系统的约束；
（4）语义模型，能使设计者通过分析系统组成的已知属性来理解系统的整体性质 。

风格分类：

- 以数据为中心的体系结构
- 数据流体系结构
- 调用和返回体系结构
- 面向对象体系结构
- 层次体结构

体系结构模式和风格的区别

体系结构模式也是对体系结构设计施加的一种变换

（1）模式涉及的范围要小一些，更多集中在体系结构的某一个方面而不是整体；

（2）模式在体系结构上施加规则，描述软件如何在基础设施层次上处理某些功能性方面的问题；

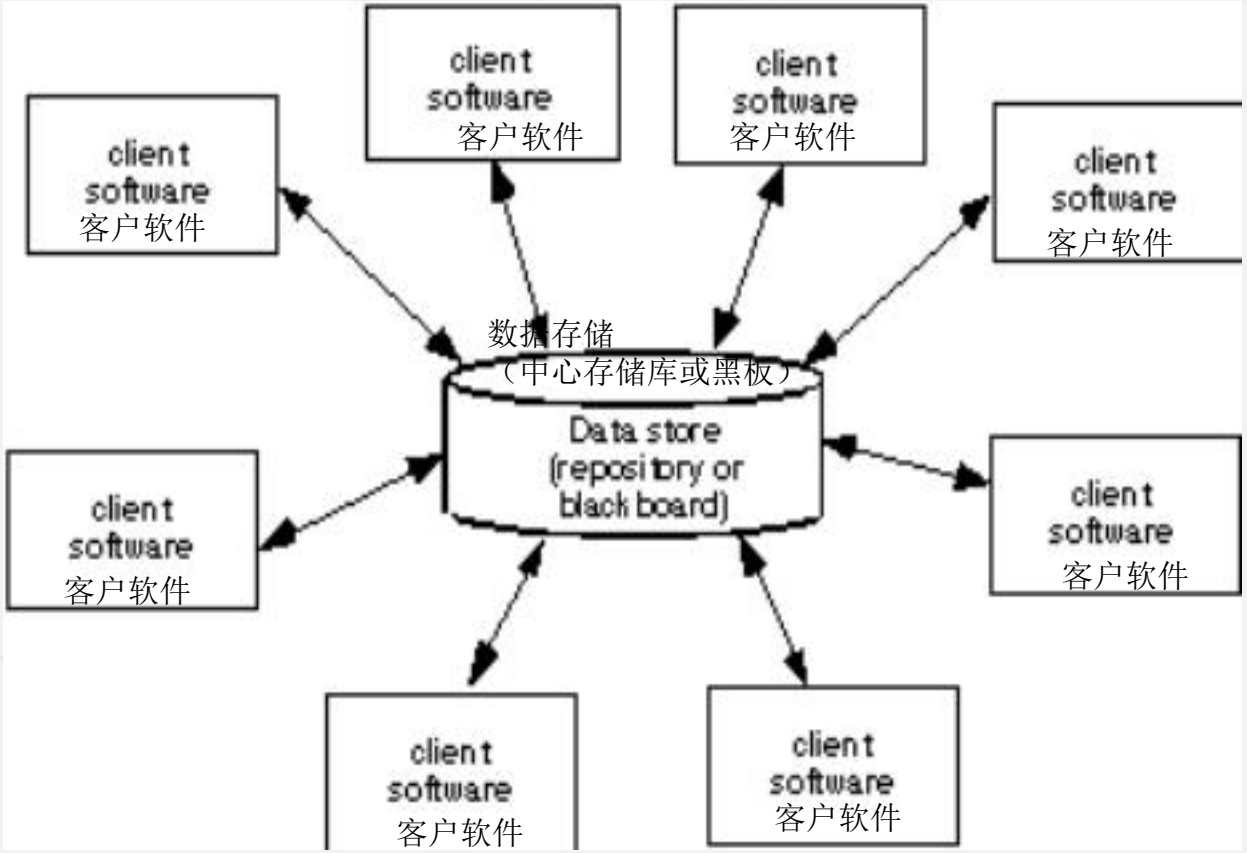
（3）体系结构模式倾向于在体系结构环境中处理特定的行为问题；

12.3.1 体系结构风格的简单分类

- 功能结构，构件表示功能或处理实体，连接件表示接口；
- 实现结构，构件可以是包、类、对象、过程、函数、方法。连接件包括传递数据和控制、共享数据等；
- 并发结构，构件表示“并发单元”，连接件包括同步于、优先级高于等；
- 物理结构，类似部署模型，构件是物理硬件，软件驻留在硬件上，连接件表示硬件之间的接口；
- 开发结构，定义软件工程过程中其他的信息源，连接件表示工作产品之间的关系。

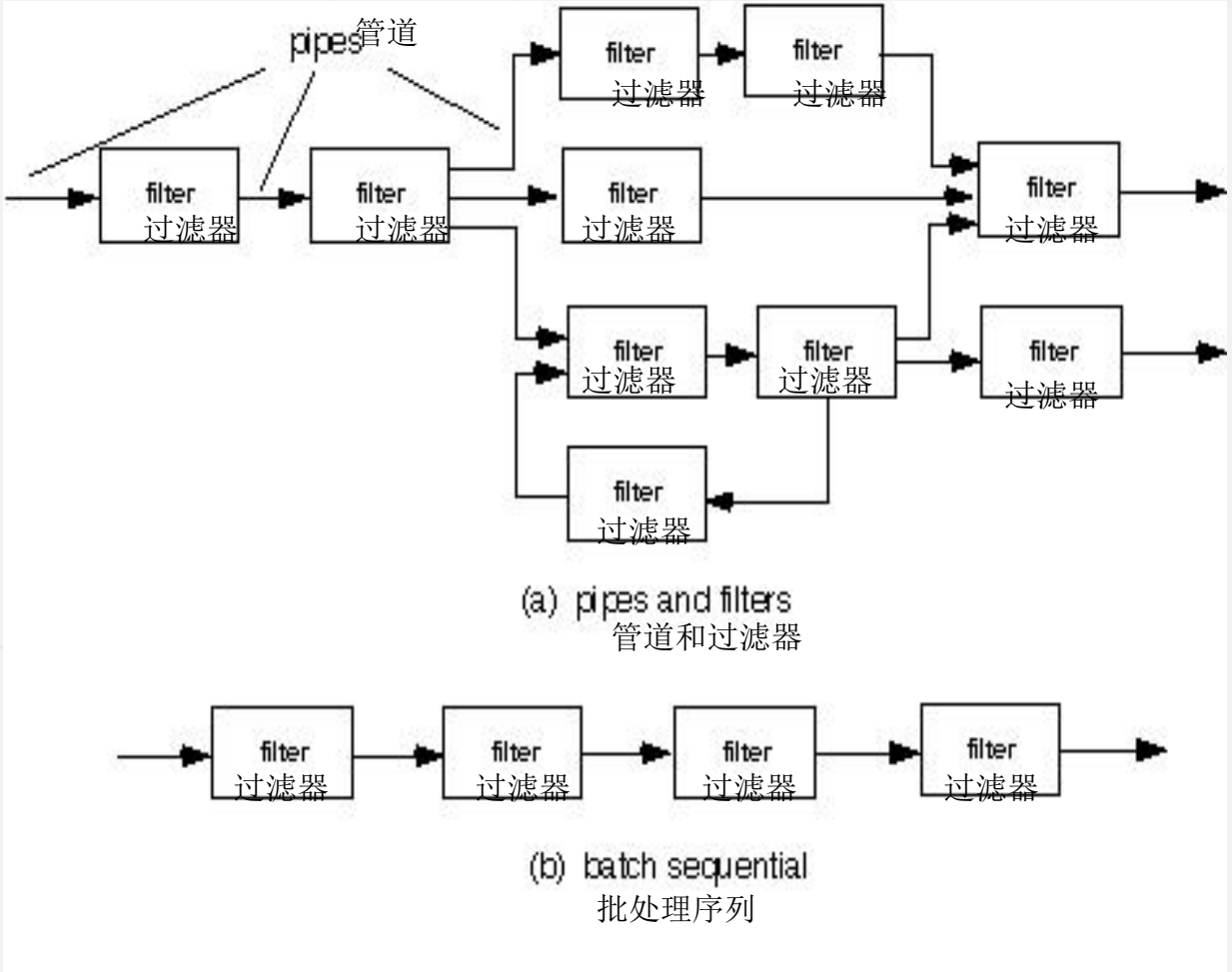
以数据为中心的体系结构

数据串联起各种功能



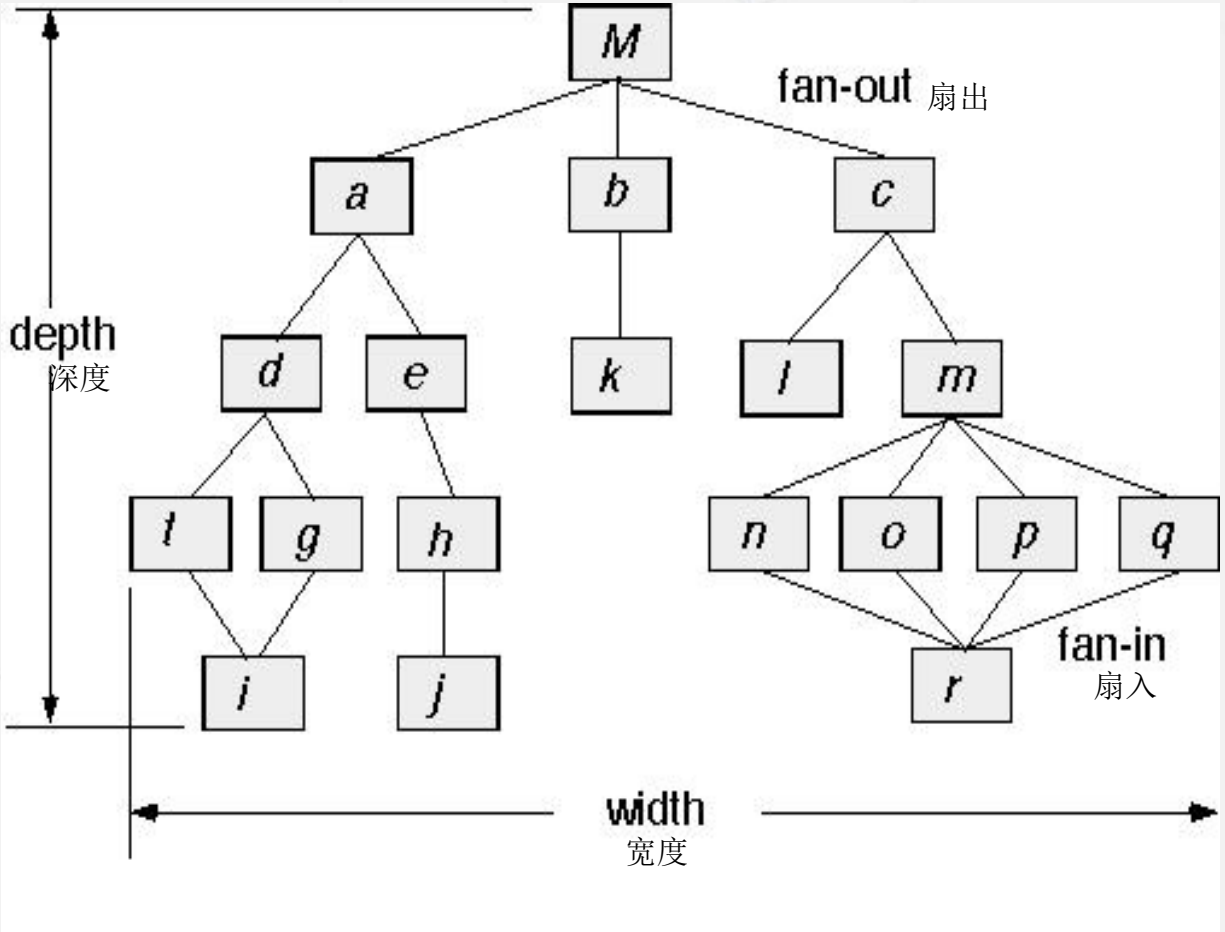
数据流体系结构

强调数据流程的



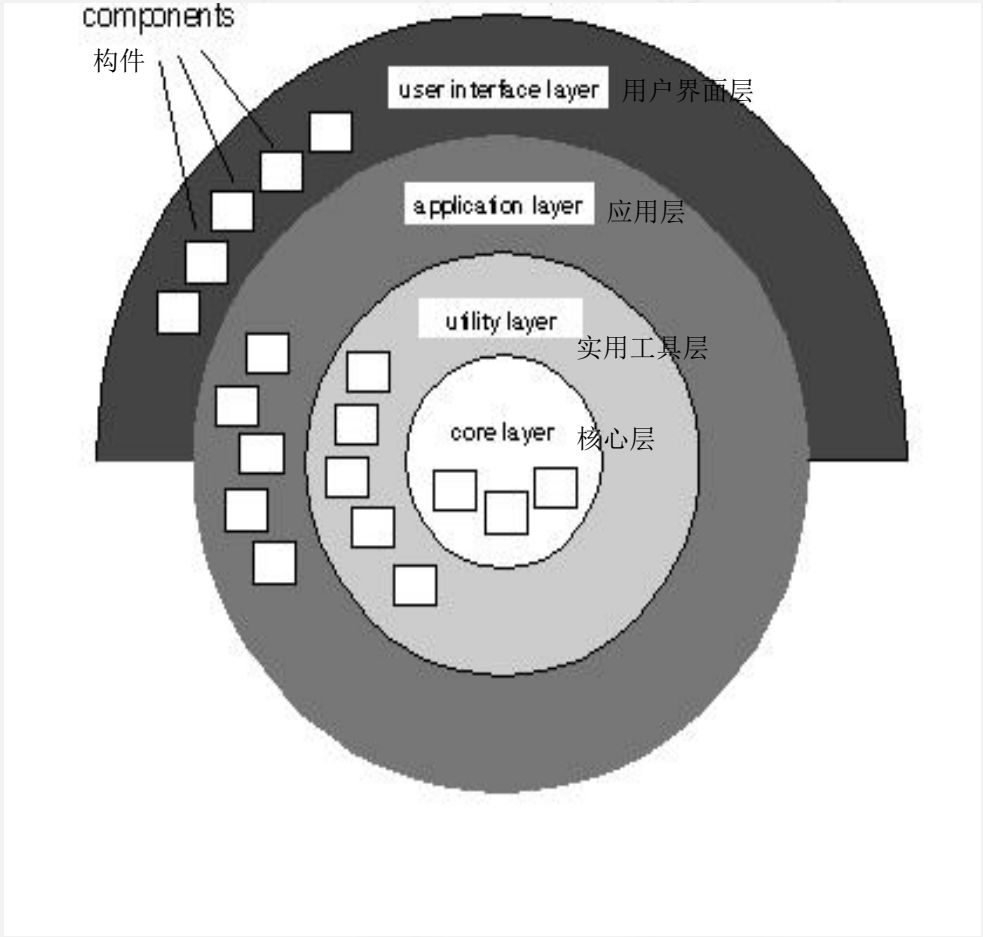
调用和返回体系结构

控制系统的组织结构，易于扩展



层次体系结构

层间独立，安全、易于移植



12.3.2 体系结构模式

在风格指导下，用模式解决具体问题

- 并发性——很多应用程序必须以模拟并行的方式来处理多任务
  - 操作系统进程管理模式
  - 任务调度模式
- 持久性——如果数据在创建它的进程运行结束之后仍然要存在，则数据时持久的。两种常见模式：
  - 数据管理系统模式将DBNS的存储功能和检索功能应用到应用系统的体系结构中
  - 应用级持久性模式在应用系统体系结构的内部构造持久性特征
- 分布性——关系到在分布式环境中系统或系统内部件相互通信的方式
  - 代理的作用是在客户端构件和服务器端构件间担当“中间人”。

12.3.3 组织和求精

- 控制
- 数据

12.4 体系结构考虑的要素

要素非独立存在，相互作用和调节

- **经济性**，简洁，依赖抽象，减少无用细节
- **易见性**，显而易见
- **隔离性**，关注点分离
- **对称性**，属性均衡
- **应急性**，紧急、自组织的行为和控制

12.5 体系结构决策

体系结构决策文档

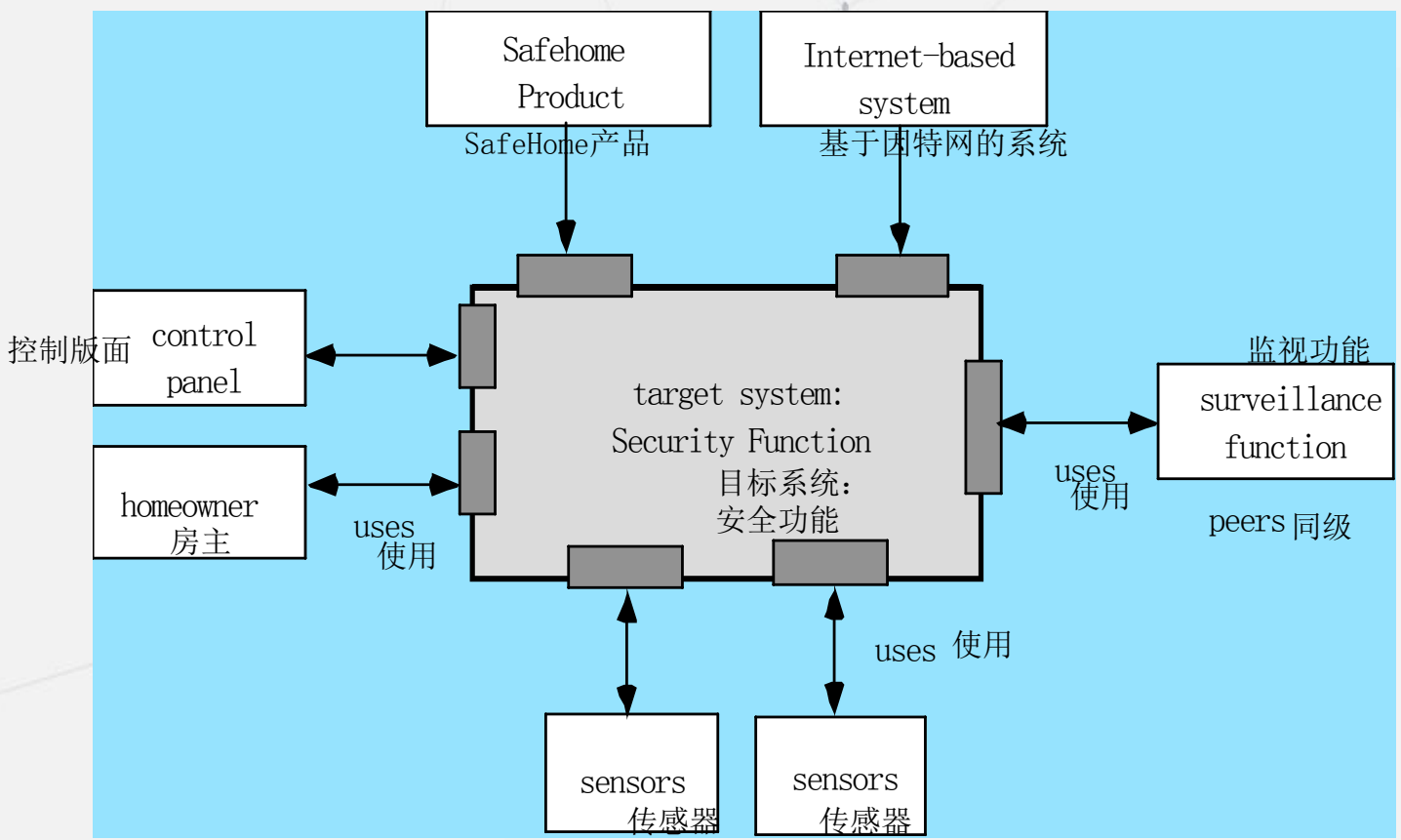
1. 确定那些信息项是一个决策所必需的
2. 定义每个决策与适当的需求之间的联系
3. 提供需要评估可替换决策时改变状态的机制
4. 定义决策之间的先觉关系，以支持可跟踪性
5. 把重大决策与从决策导出的体系结构视图相联系
6. 在作出决策时，要记录和沟通所有的决策

12.6 体系结构设计

原型本身不提供足够的细节，需要设计和细化，并持续迭代

- 待开发的软件必须放在所处的**环境**中
  - 设计应该定义与软件交互的外部实体（其他系统、设备、人）和交互的特性。
- 确定体系结构的**原型**集
  - 原型（类似于类）是表示系统行为元素的一种抽象
- 设计人员通过定义和细化实施每个原型的软件结构来指定系统的结构

12.6.1 系统环境的表示



体系结构环境图

12.6.2 原型

原型是表示核心抽象的类或模式，及时复杂系统，原型也比较小

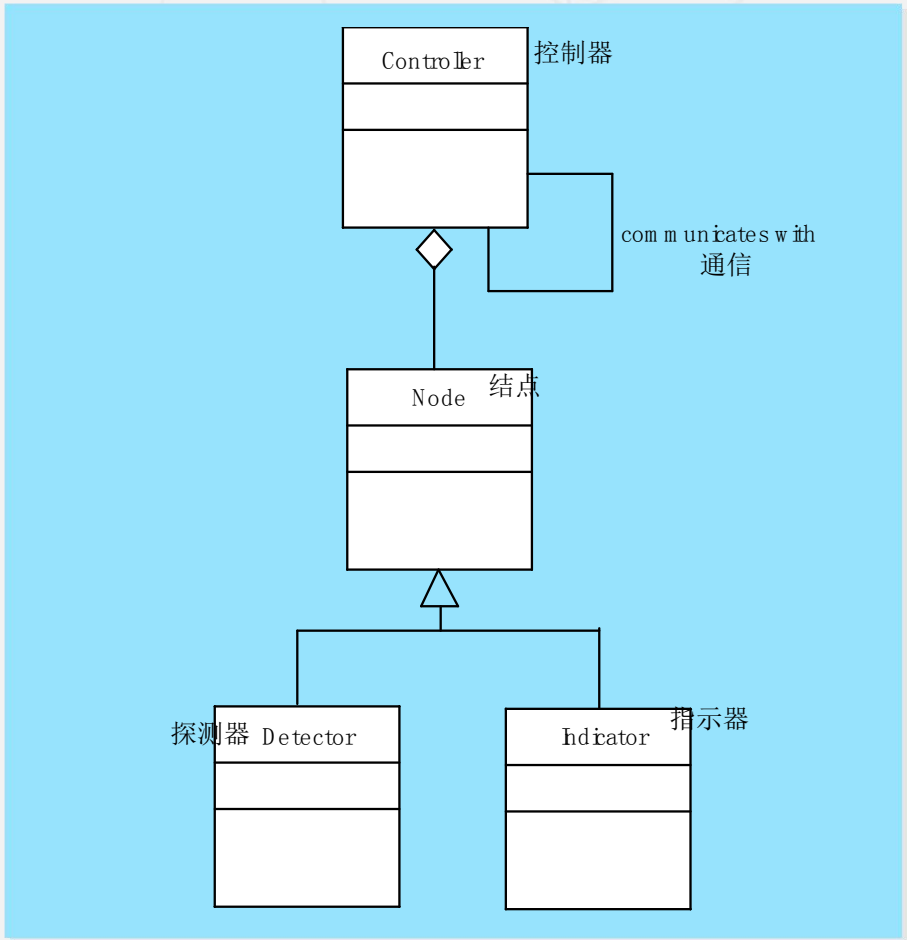
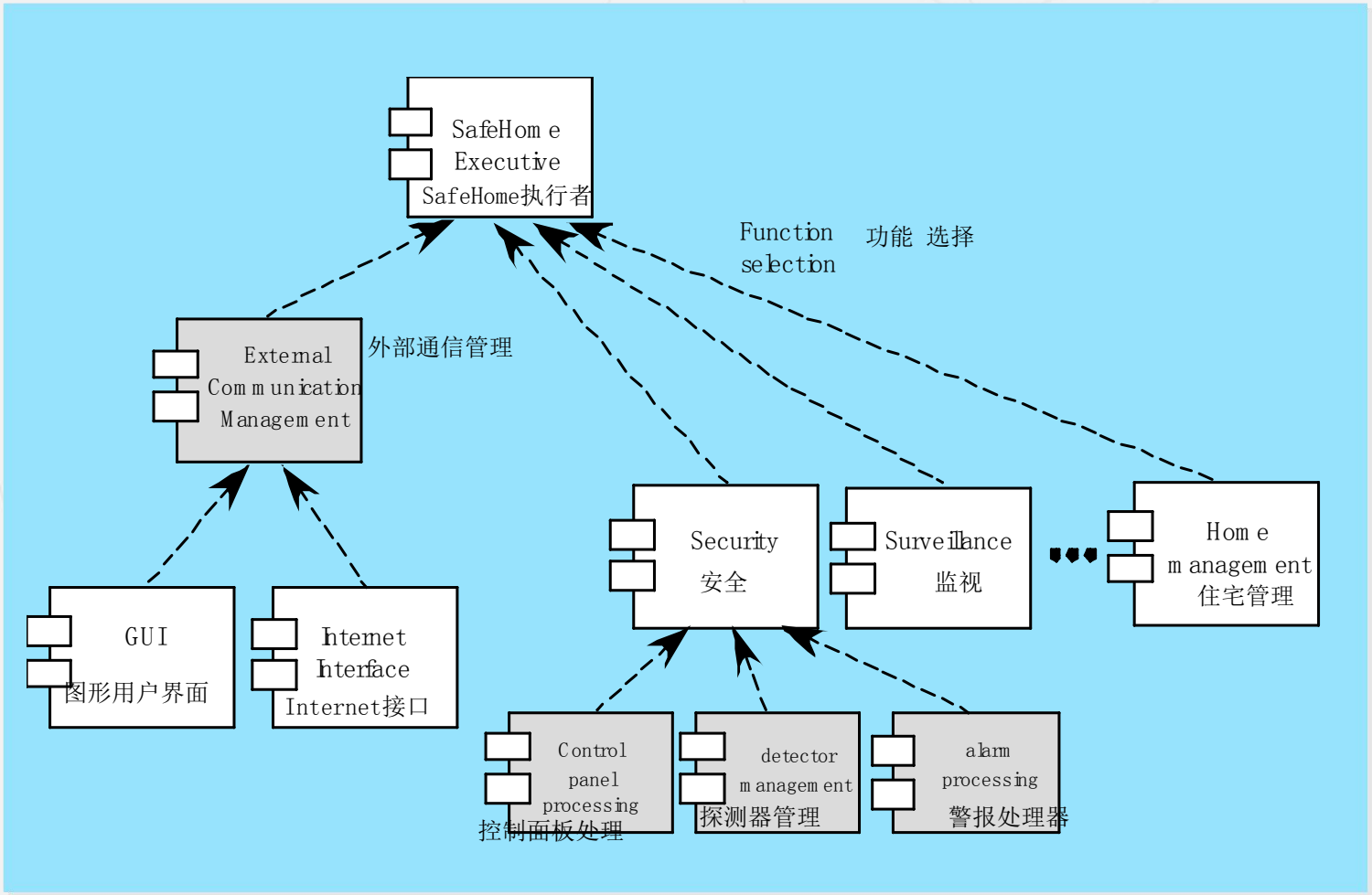


Figure 10.7 UML relationships for SafeHome security function archetypes (adapted from [BOS00])

SafHome安全功能原型的UML关系

12.6.3 体系结构细化为构件

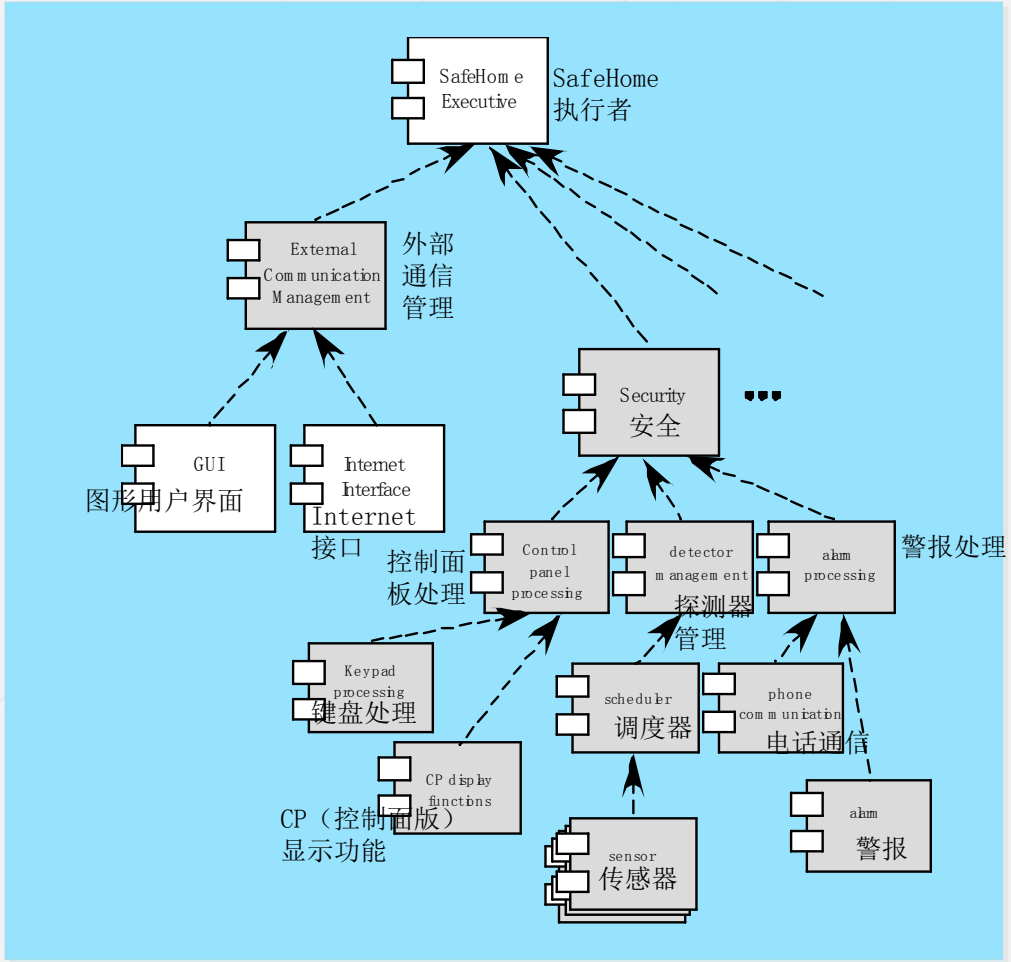
如何细化？先从类开始



构件导出和细化的源泉：

- 应用领域
- 基础设施构件

12.6.4 描述系统实例



将体系结构应用于具体问题，以证明结构和构件是合理的。

12.6.5 WebApp的体系结构设计

- B/S模式
- 多层次体系结构
- 受客户端所访问内容结构影响
- 可控的

12.6.6 移动App的体系结构设计

- 多层次体系结构
- 物理特性不同
- 软件特性不同

12.7 评估候选的体系结构设计

确定体系结构中的敏感点

1. 收集场景。开发一组用例，从用户的角度描述系统。
2. 引出需求、约束和环境描述。
3. 描述那些已经被选择用于解决场景和需求的体系结构风格/模式。
    - 模块视图：分析构件的工作任务
    - 过程视图：分析系统性能
    - 数据流视图：分析体系结构满足功能需求的程度
4. 通过单独地考虑每个属性来评估质量属性。质量属性包括：可靠性、性能、安全性、灵活性、可维护性、可测试性、可移植性、可复用性和互操作性。
5. 针对特定的体系结构风格，确定质量属性对各种体系结构属性的敏感性。可通过对体系结构进行小的变更，确定某属性对变更的敏感性。
6. 使用第5步进行的敏感性分析鉴定候选体系结构（在第3步开发的）。

体系结构复杂性

- 通过考虑体系结构中构件间的依赖关系，对提议的体系结构的整个复杂性进行评估 [Zha98]
  - 共享依赖表示使用相同资源的消费者之间或相同消费者生产的生产者之间的依赖关系。
  - 流依赖表示资源的生产者和消费者间的依赖关系。
  - 约束依赖表示在一组活动间相互控制流上的约束。

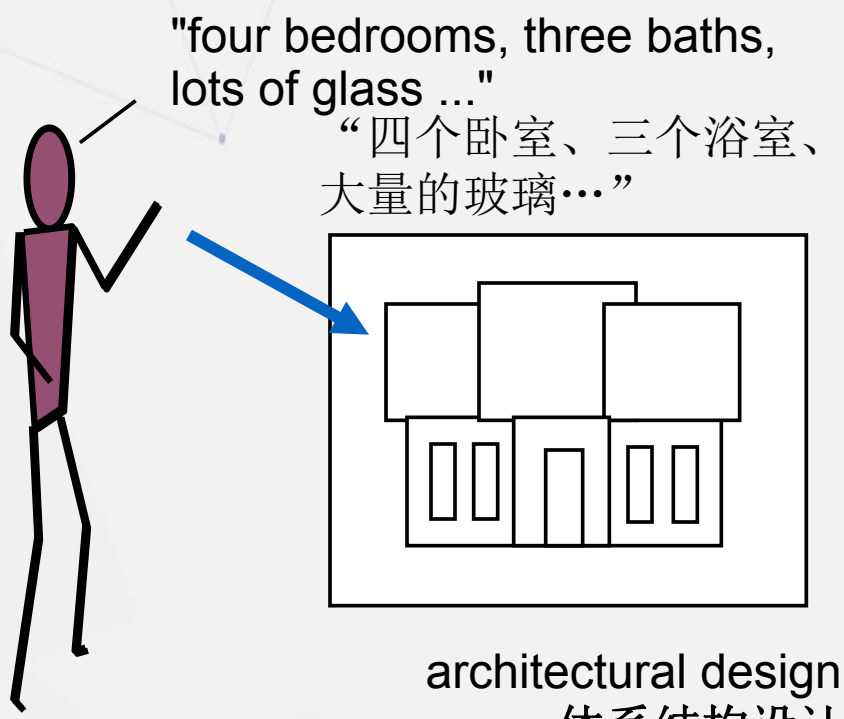
提到的共享依赖和流依赖类似于之前提到的耦合的概念。耦合可以应用在体系结构级和构建级的重要设计概念。

12.7.1 体系结构描述语言

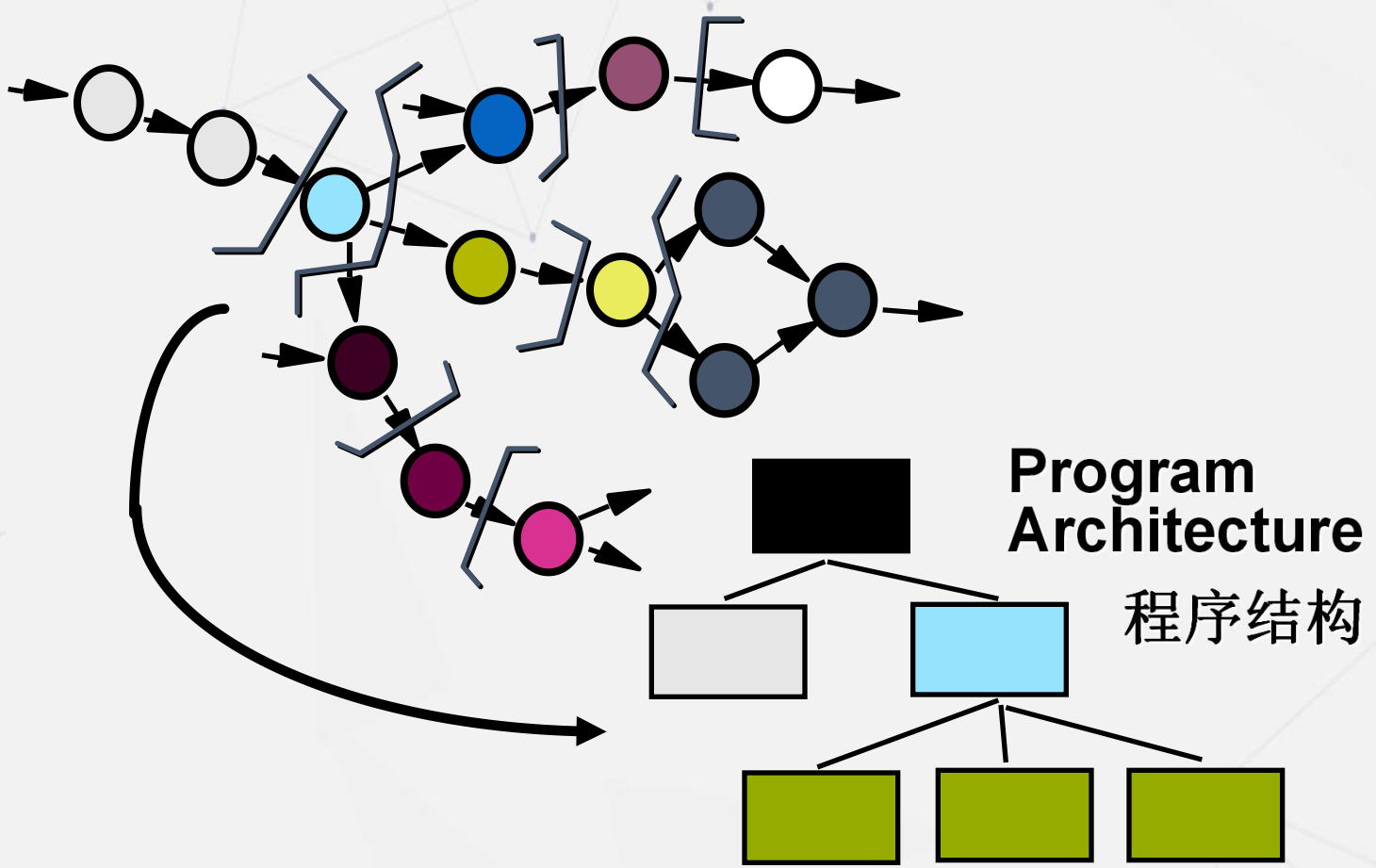
- 体系结构描述(ADL) 提供了一种描述软件体系结构的语义和语法。
- 提供设计者如下能力：
  - 分解体系结构构件
  - 将单独构件组合成大的体系结构块
  - 描述构件之间的接口（连接机制）

一种体系结构设计方法

customer requirements 客户需求

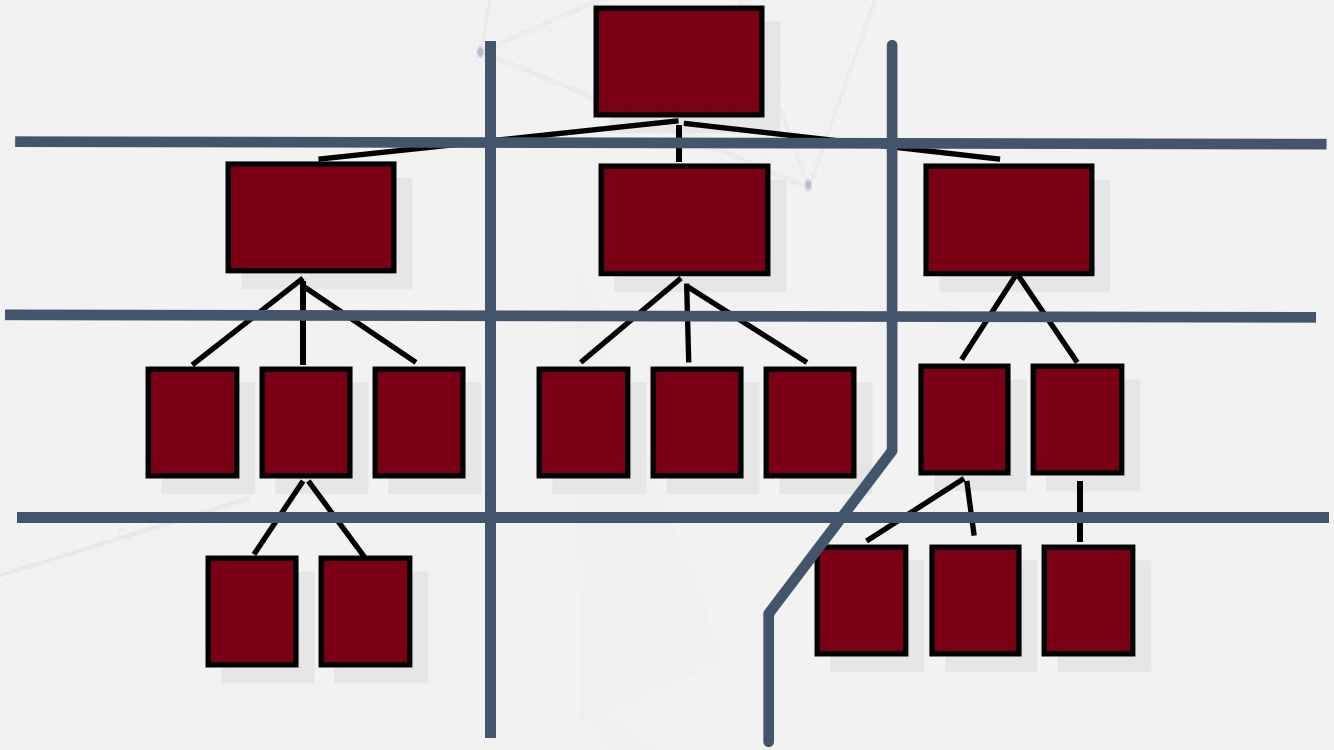


导出程序结构



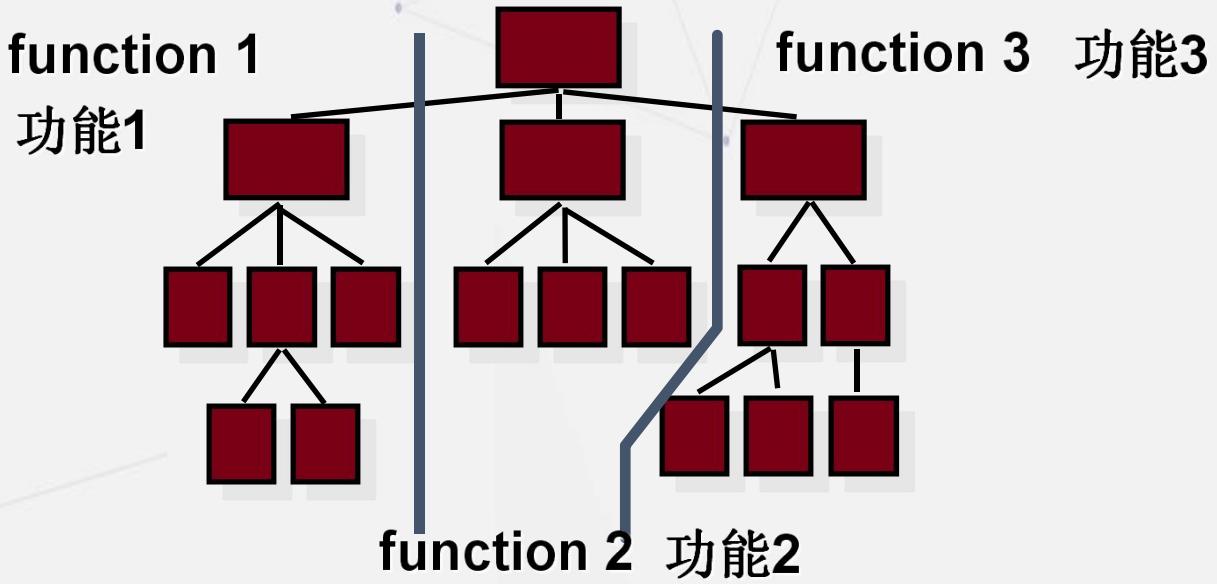
分割体系结构

- 按要求“水平”和“垂直”分割



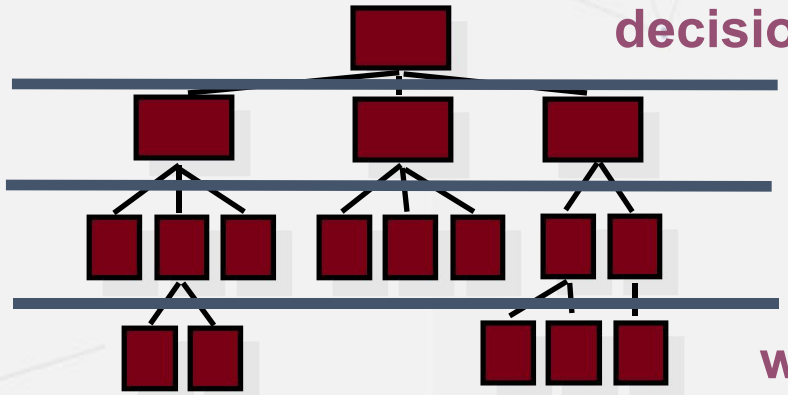
垂直分块

- 为每个主要功能定义单独的模块层次分支
- 使用控制模块协调各功能之间的通信



水平分块：分解

- 设计这样的决策，使决策者和工作者是分层的
- 决策模块应该驻留在该体系结构的顶部



为什么要分割体系结构？

- 导致软件更容易测试
- 导致软件更容易维护
- 导致较少的负作用传播
- 导致软件更容易扩展

12.9 基于模式的体系结构评审

- 评估软件体系结构满足系统质量需求和识别潜在风险的能力。
- 通过尽早检测到设计问题来降低项目成本的潜力。
- 常使用基于经验的推理、原型评估、情境评审和检查单。

12.9 基于模式的体系结构评审

1. 通过遍历用例来确定并讨论质量属性。
2. 结合需求讨论系统体系结构图
3. 识别所使用的体系结构模式，将系统结构与模式结构相匹配。
4. 使用现有的文档和用例，来确定每一个模式对质量属性的影响。
5. 识别由设计中使用的体系结构模式所引起的质量问题。
6. 针对会议上出现的问题作一个简短的汇总，并对可运行的系统骨架进行修正。

12.10 体系结构一致性检查

保证实施和演变中的系统与规划的体系结构一致

- 需求冲突
- 技术困难
- 交付期限
- 人员变动

12.11 敏捷性与体系结构

- 为了避免返工，在编码之前，使用用户故事来创建和演化体系结构模型（可运行的系统骨架）
- 混合模型有助于软件架构师把用户故事演化成故事板
- 运行良好的敏捷项目包括在每个冲刺期间交付工作产品
- 对每一次冲刺浮现出的代码进行复审是一种有用的体系结构评审方式。