

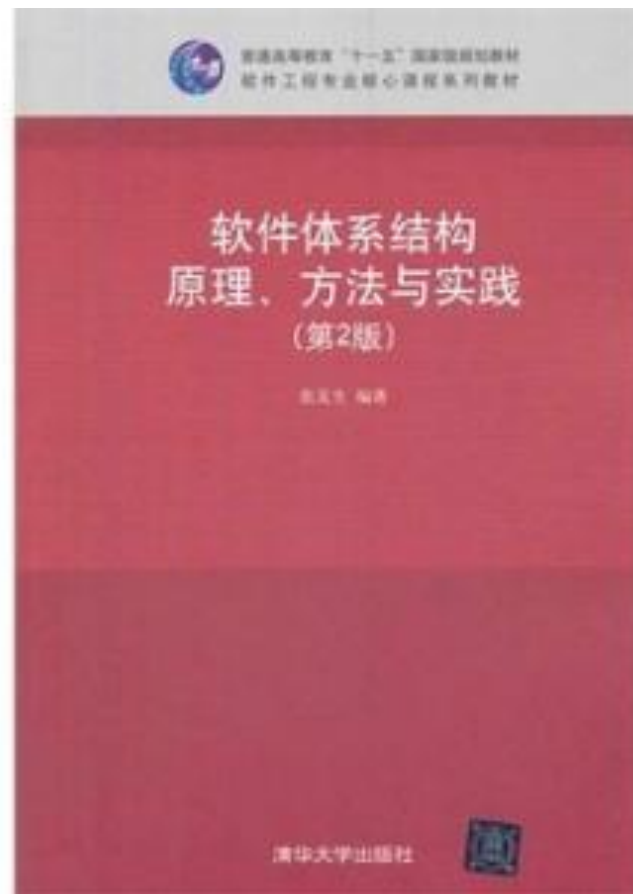


01

软件体系结构概论

关于教材

- 出版社：清华大学出版社
- 作者：张友生 李雄





课程内容

- 软件体系结构概论
- 软件体系结构建模
- 软件体系结构风格
- 软件体系结构描述
- 动态软件体系结构
- Web服务体系结构
- 基于体系结构的软件开发
- 软件体系结构的分析与测试
- 软件体系结构评估
- 软件产品线



1.1 从软件危机谈起

软件危机的表现

- 软件成本日益增长
- 开发进度难以控制
- 软件质量差
- 软件维护困难

1.1 从软件危机谈起

软件危机的表现

- 软件成本日益增长

20世纪50年代，软件成本在整个计算机系统成本中所占的比例为10%-20%。到20世纪60年代中期，软件成本在计算机系统中所占的比例已经增长到50%左右。

而且，该数字还在不断地递增，下面是一组来自美国空军计算机系统的数
据：1955年，软件费用约占总费用的18%，1970年达到60%，1975年达到72%，1980年达到80%，1985年达到85%左右。

1.1 从软件危机谈起

软件危机的表现

- 开发进度难以控制

由于软件是逻辑、智力产品，软件的开发需建立庞大的逻辑体系，这是与其他产品的生产不一样的。

在软件开发过程中，用户需求变化等各种意想不到的情况层出不穷，令软件开发过程很难保证按预定的计划实现，给项目计划和论证工作带来了很大的困难。

盲目增加软件开发人员并不能成比例地提高软件开发能力。相反，随着人员数量的增加，人员的组织、协调、通信、培训和管理等方面的问题将更为严重。

1.1 从软件危机谈起

软件危机的表现

- 软件质量差

软件项目即使能按预定日期完成，结果却不尽人意。1965年至1970年，美国范登堡基地发射火箭多次失败，绝大部分故障是由应用程序错误造成的。

在“软件作坊”里，由于缺乏工程化思想的指导，程序员几乎总是习惯性地以自己的想法去代替用户对软件的需求，软件设计带有随意性，很多功能只是程序员的“一厢情愿”而已，这是造成软件不能令人满意的重要因素。

1.1 从软件危机谈起

软件危机的表现

- 软件维护困难

由于在软件设计和开发过程中，没有严格遵循软件开发标准，各种随意性很大，没有完整的真实反映系统状况的记录文档，给软件维护造成了巨大的困难。

特别是在软件使用过程中，原来的开发人员可能因各种原因已经离开原来的开发组织，使得软件几乎不可维护。

有资料表明，工业界为维护软件支付的费用占全部硬件和软件费用的40%-75%。



1.1 从软件危机谈起

软件危机的原因

- 用户需求不明确
- 缺乏正确的理论指导
- 软件规模越来越大
- 软件复杂度越来越高

1.1 从软件危机谈起

软件危机的原因

- 用户需求不明确

在软件开发完成之前，用户不清楚软件的具体需求；

用户对软件需求的描述不精确，可能有遗漏、有二义性、甚至有错误；

在软件开发过程中，用户还提出修改软件功能、界面、支撑环境等方面的要求；

开发人员对用户需求的理解与用户本来愿望有差异。

1.1 从软件危机谈起

软件危机的原因

- 缺乏正确的理论指导

缺乏有力的方法学和工具方面的支持。由于软件不同于大多数其他工业产品，其开发过程是复杂的逻辑思维过程，其产品极大程度地依赖于开发人员高度的智力投入。由于过分地依靠程序设计人员在软件开发过程中的技巧和创造性，加剧软件产品的个性化，也是发生软件危机的一个重要原因。

1.1 从软件危机谈起

软件危机的原因

- 软件规模越来越大

随着软件应用范围的增广，软件规模愈来愈大。大型软件项目需要组织一定的人力共同完成，而多数管理人员缺乏开发大型软件系统的经验，而多数软件开发人员又缺乏管理方面的经验。各类人员的信息交流不及时、不准确、有时还会产生误解。

软件项目开发人员不能有效地、独立自主地处理大型软件的全部关系和各个分支，因此容易产生疏漏和错误。

1.1 从软件危机谈起

软件危机的原因

- 软件复杂度越来越高

软件不仅仅是在规模上快速地发展扩大，而且其复杂性也急剧地增加。软件产品的特殊性和人类智力的局限性，导致人们无力处理“复杂问题”。

所谓“复杂问题”的概念是相对的，一旦人们采用先进的组织形式、开发方法和工具提高了软件开发效率和能力，新的、更大的、更复杂的问题又摆在人们的面前。



1.1 从软件危机谈起

如何克服软件危机

人们面临的不光是技术问题，更重要的是管理问题。管理不善必然导致失败。

要提高软件开发效率，提高软件产品质量，必须采用工程化的开发方法与工业化的生产技术。

在技术上，应该采用基于重用的软件生产技术；在管理上，应该采用多维的工程管理模式。



1.2 构件与软件重用

构件模型及实现

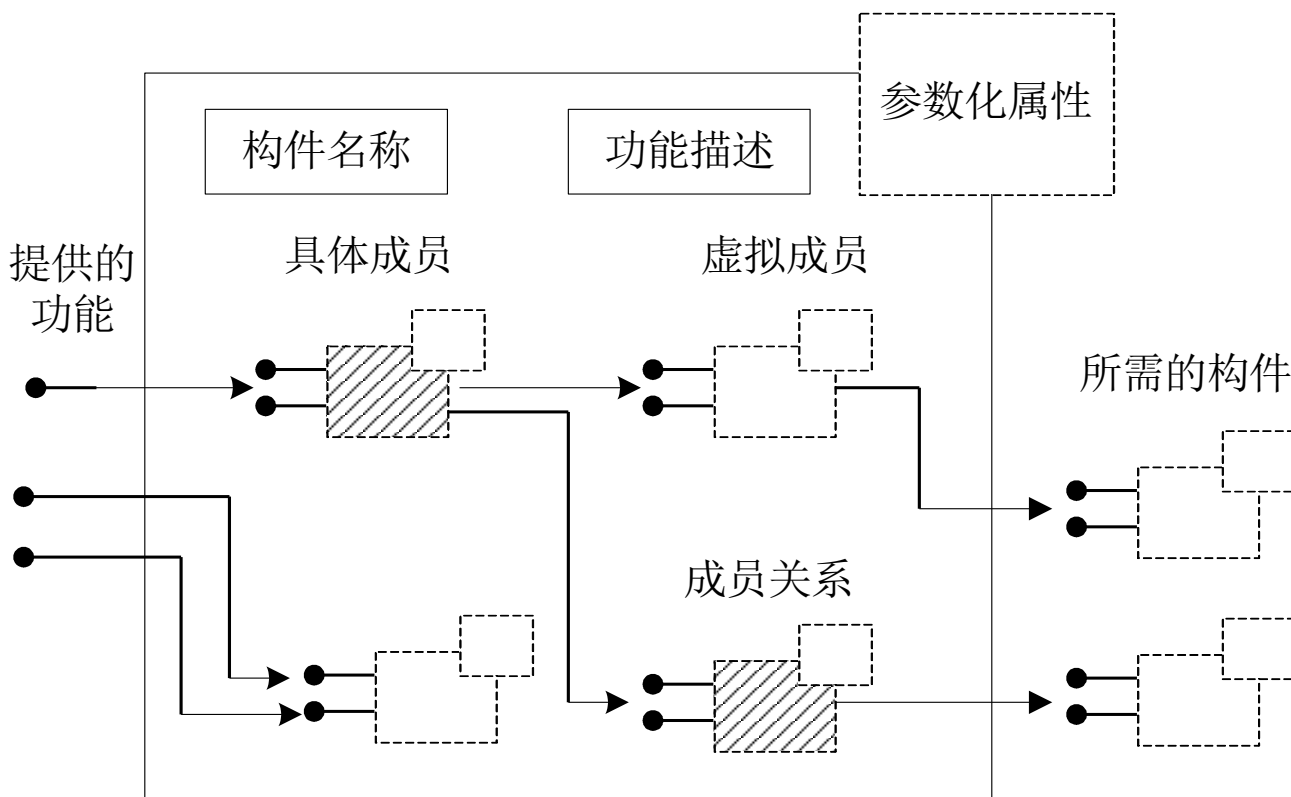
- 构件的定义

构件是指语义完整、语法正确和有可重用价值的单位软件，是软件重用过程中可以明确辨识的系统；结构上，它是语义描述、通讯接口和实现代码的复合体。

1.2 构件与软件重用

构件模型及实现

- 青鸟构件模型





1.2 构件与软件重用

构件获取

从现有构件中获得符合要求的构件，直接使用或作适应性修改，得到可重用的构件；

通过遗留工程，将具有潜在重用价值的构件提取出来，得到可重用的构件；

从市场上购买现成的商业构件，即COTS（Commercial Off-The-Shell）构件；

开发新的符合要求的构件。



1.2 构件与软件重用

构件管理

- 构件描述
- 构件分类与组织
- 人员及权限管理

1.2 构件与软件重用

构件管理

- 构件描述

构件模型是对构件本质的抽象描述，主要是为构件的制作与构件的重用提供依据；

从管理角度出发，也需要对构件进行描述，例如：实现方式、实现体、注释、生产者、生产日期、大小、价格、版本和关联构件等信息，它们与构件模型共同组成了对构件的完整描述。



1.2 构件与软件重用

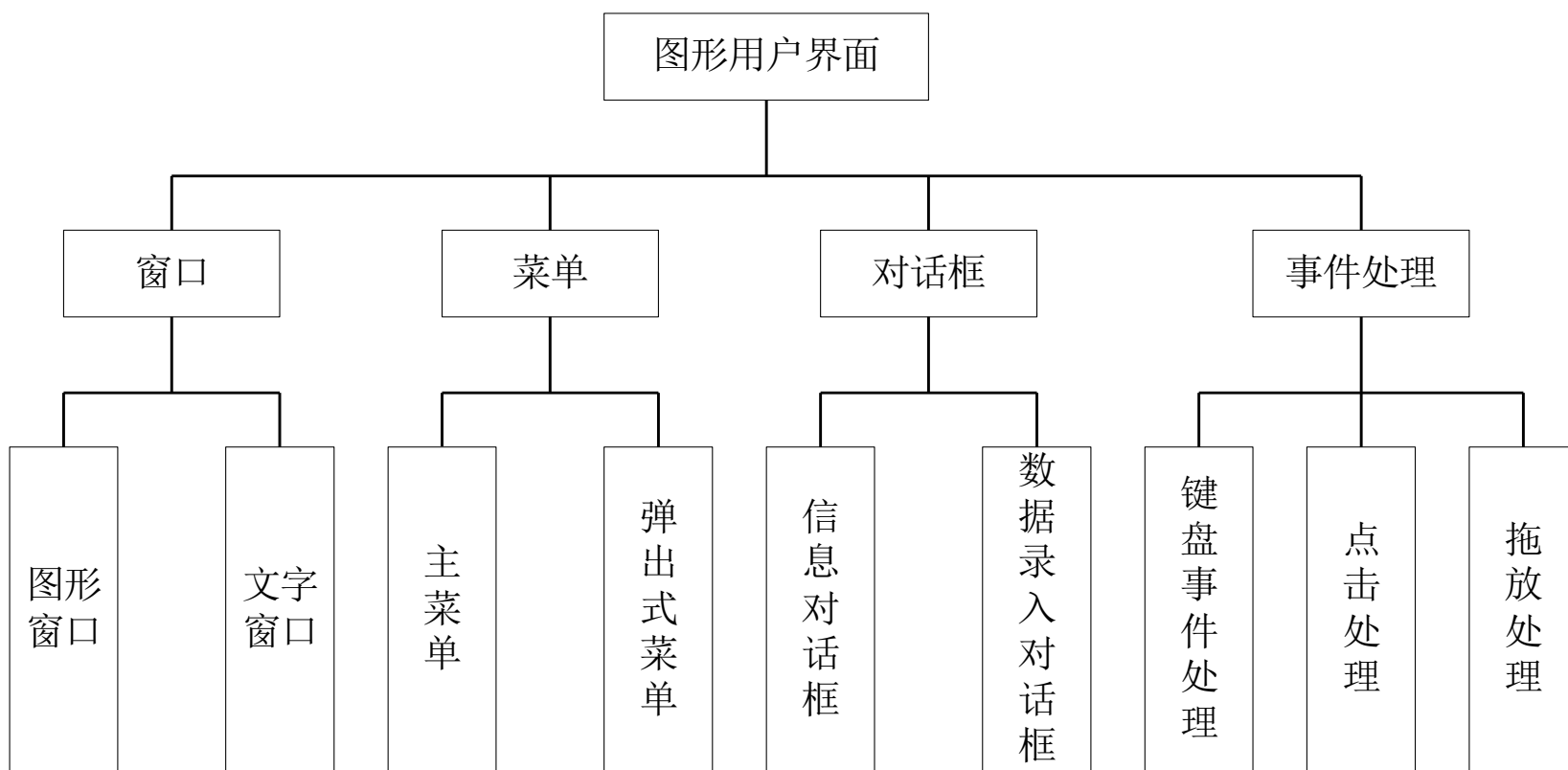
构件管理

- 构件分类与组织
 - ◇ 关键字分类法
 - ◇ 刻面分类法
 - ◇ 超文本组织方法

1.2 构件与软件重用

构件管理

- 关键字分类法





1.2 构件与软件重用

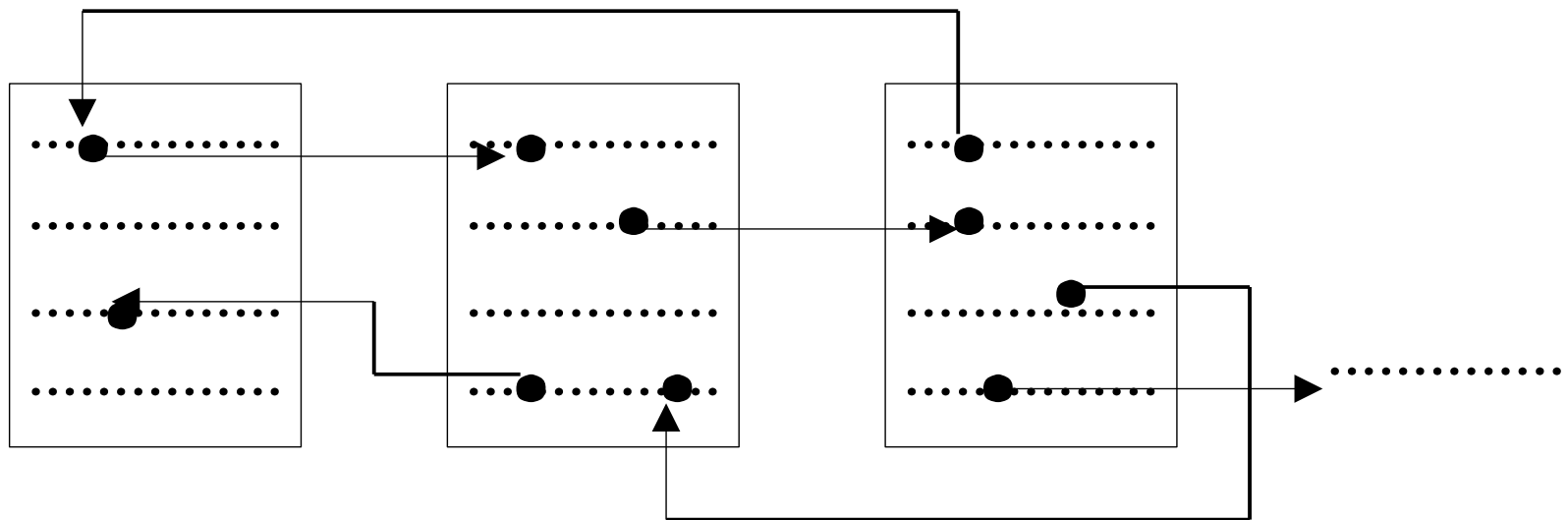
构件管理

- 刻面分类法
 - 使用环境
 - 应用领域
 - 功能
 - 层次
 - 表示方法

1.2 构件与软件重用

构件管理

- 超文本组织法





1.2 构件与软件重用

◇ 构件管理

- 人员及权限管理

一般来讲，构件库系统可包括五类用户，即注册用户、公共用户、构件提交者、一般系统管理员和超级系统管理员。



1.2 构件与软件重用

◇ 构件重用

- 检索与提取构件
- 理解与评价构件
- 修改构件
- 构件组装



1.2 构件与软件重用

◇ 构件重用

- 检索与提取构件
 - 基于关键字的检索
 - 刻面检索法
 - 超文本检索法
 - 其他检索方法



1.2 构件与软件重用

◇ 构件重用

- 理解与评价构件
 - 构件的功能与行为
 - 相关的领域知识
 - 可适应性约束条件与例外情形
 - 可以预见的修改部分及修改方法

1.2 构件与软件重用

◇ 构件重用

- 修改构件

理想的情形是对库中的构件不作修改而直接用于新的软件项目。

但是，在大多数情况下，必须对构件进行或多或少的修改，以适应新的需求。

为了减少构件修改的工作量，要求开发人员尽量使构件的功能、行为和接口设计更为抽象化、通用化和参数化。



1.2 构件与软件重用

◇ 构件重用

- 构件组装
 - 基于功能的组装技术
 - 基于数据的组装技术
 - 面向对象的组装技术

1.2 构件与软件重用

◇ 构件重用

- 构件组装
- 基于功能的组装技术

基于功能的组装技术采用子程序调用和参数传递的方式将构件组装起来。它要求库中的构件以子程序/过程/函数的形式出现，并且接口说明必须清晰。当使用这种组装技术进行软件开发时，开发人员首先应对目标软件系统进行功能分解，将系统分解为强内聚、松耦合的功能模块。然后根据各模块的功能需求提取构件，对它进行适应性修改后再挂接在上述功能分解框架中。

1.2 构件与软件重用

构件重用

- 构件组装
 - 基于数据的组装技术

首先根据当前软件问题的核心数据结构设计出一个框架，然后根据框架中各结点的需求提取构件并进行适应性修改，再将构件逐个分配至框架中的适当位置。此后，构件的组装方式仍然是传统的子程序调用与参数传递。这种组装技术也要求库中构件以子程序形式出现，但它所依赖的软件设计方法不再是功能分解，而是面向数据的设计方法，例如 Jackson 系统开发方法。

1.2 构件与软件重用

构件重用

- 构件组装

- 面向对象的组装技术

- 构造法

- 在子类中引进基类的对象作为子类的成员变量，然后在子类中通过成员变量重用基类的属性和方法。

- 子类法

- 将新子类直接说明为库中基类的子类，通过继承和修改基类的属性与行为完成新子类的定义。



1.3 体系结构的兴起和发展

背景资料

随着软件系统规模越来越大、越来越复杂，整个系统的结构和规格说明显得越来越重要。

对于大规模的复杂软件系统来说，对总体的系统结构设计和规格说明比起对计算的算法和数据结构的选择已经变得明显重要得多。

对软件体系结构的系统、深入的研究将会成为提高软件生产率和解决软件维护问题的新的最有希望的途径。



1.3 体系结构的兴起和发展

背景资料

事实上，软件总是有体系结构的，不存在没有体系结构的软件。

软件体系结构虽脱胎于软件工程，但其形成同时借鉴了计算机体系结构和网络体系结构中很多宝贵的思想和方法，最近几年软件体系结构研究已完全独立于软件工程的研究，成为计算机科学的一个最新的研究方向和独立学科分支。

1.3 体系结构的兴起和发展

◇ 软件体系结构的定义

- Dewayne Perry和Alexander Wolf

软件体系结构是具有一定形式的结构化元素，即构件的集合，包括处理构件、数据构件和连接构件。

处理构件负责对数据进行加工，数据构件是被加工的信息，连接构件把体系结构的不同部分组合连接起来。

这一定义注重区分处理构件、数据构件和连接构件，这一方法在其他的定义和方法中基本上得到保持。

1.3 体系结构的兴起和发展

◇ 软件体系结构的定义

- Mary Shaw和David Garlan

软件体系结构是软件设计过程中的一个层次，这一层次超越计算过程中的算法设计和数据结构设计。

体系结构问题包括总体组织和全局控制、通讯协议、同步、数据存取，给设计元素分配特定功能，设计元素的组织，规模和性能，在各设计方案间进行选择等。

软件体系结构处理算法与数据结构之上关于整体系统结构设计和描述方面的一些问题，如全局组织和全局控制结构、关于通讯、同步与数据存取的协议，设计构件功能定义，物理分布与合成，设计方案的选择、评估与实现等。



1.3 体系结构的兴起和发展

◇ 软件体系结构的定义

- Kruchten

软件体系结构有四个角度，它们从不同方面对系统进行描述：概念角度描述系统的主要构件及它们之间的关系；模块角度包含功能分解与层次结构；运行角度描述了一个系统的动态结构；代码角度描述了各种代码和库函数在开发环境中的组织。



1.3 体系结构的兴起和发展

◇ 软件体系结构的定义

- Hayes Roth

软件体系结构是一个抽象的系统规范，主要包括用其行为来描述的功能构件和构件之间的相互连接、接口和关系。



1.3 体系结构的兴起和发展

◇ 软件体系结构的定义

- David Garlan 和 Dewne Perry

软件体系结构是一个程序 / 系统各构件的结构、它们之间的相互关系以及进行设计的原则和随时间演化的指导方针。



1.3 体系结构的兴起和发展

◇ 软件体系结构的定义

- Barry Boehm

软件体系结构包括一个软件和系统构件，互联及约束的集合；一个系统需求说明的集合；一个基本原理用以说明这一构件，互联和约束能够满足系统需求。



1.3 体系结构的兴起和发展

◇ 软件体系结构的定义

- Bass, Clements 和 Kazman

软件体系结构包括一个或一组软件构件、软件构件的外部的可见特性及其相互关系。其中，“软件外部的可见特性”是指软件构件提供的服务、性能、特性、错误处理、共享资源使用等。

1.3 体系结构的兴起和发展

◇ 软件体系结构的定义

- 我们的定义

软件体系结构为软件系统提供了一个结构、行为和属性的高级抽象，由构成系统的元素的描述、这些元素的相互作用、指导元素集成的模式以及这些模式的约束组成。软件体系结构不仅指定了系统的组织结构和拓扑结构，并且显示了系统需求和构成系统的元素之间的对应关系，提供了一些设计决策的基本原理。



1.3 体系结构的兴起和发展

软件体系结构的意义

- 体系结构是风险承担者进行交流的手段

软件体系结构代表了系统的公共的高层次的抽象。这样，系统的大部分有关人员（即使不是全部）能把它作为建立一个互相理解的基础，形成统一认识，互相交流。

体系结构提供了一种共同语言来表达各种关注和协商，进而对大型复杂系统能进行理智的管理。这对项目最终的质量和使用有极大的影响。



1.3 体系结构的兴起和发展

·软件体系结构的意义

- 体系结构是早期设计决策的体现
 - 软件体系结构明确了对系统实现的约束条件
 - 软件体系结构决定了开发和维护组织的组织结构
 - 软件体系结构制约着系统的质量属性
 - 通过研究软件体系结构可能预测软件的质量
 - 软件体系结构使推理和控制更改更简单
 - 软件体系结构有助于循序渐进的原型设计
 - 软件体系结构可以作为培训的基础



1.3 体系结构的兴起和发展

软件体系结构的意义

- 软件体系结构是可传递和可重用的模型

软件体系结构级的重用意味着体系结构的决策能在具有相似需求的多个系统中发生影响，这比代码级的重用要有更大的好处。

1.3 体系结构的兴起和发展

软件体系结构的发展史

“无体系结构”设计阶段

以汇编语言进行小规模应用程序开发为特征

萌芽阶段

出现了程序结构设计主题，以控制流图和数据流图构成软件结构为特征

初期阶段

出现了从不同侧面描述系统的结构模型，以UML为典型代表。

高级阶段

以描述系统的高层抽象结构为中心，不关心具体的建模细节，划分了体系结构模型与传统软件结构的界限，该阶段以Kruchten提出的“4+1”模型为标志



1.3 体系结构的兴起和发展

软件体系结构的发展史

Perry和Wolf认为

未来的年代是研究软件体系结构的时代



1.4 体系结构的应用现状

软件体系结构的应用现状

- 软件体系结构描述语言
- 体系结构描述构造与表示
- 体系结构分析、设计与验证
- 体系结构发现、演化与重用
- 基于体系结构的软件开发方法
- 特定领域的体系结构框架
- 软件体系结构支持工具
- 软件产品线体系结构
- 建立评价软件体系结构的方法



1.4 体系结构的应用现状

软件体系结构的应用现状

- 软件体系结构描述语言

ADL提供了具体的语法与刻画体系结构的概念框架。ADL使得系统开发者能够很好地描述他们设计的体系结构，以便与他人交流，能够用提供的工具对许多实例进行分析。



1.4 体系结构的应用现状

◇ 软件体系结构的应用现状

- 体系结构描述构造与表示（1）

按照一定的描述方法，用体系结构描述语言对体系结构进行说明的结果则称为体系结构的表示，而将描述体系结构的过程称为体系结构构造



1.4 体系结构的应用现状

软件体系结构的应用现状

- 体系结构描述构造与表示 (2)
 - Kruchten提出的 “4+1”模型。
 - Booch从UML的角度给出了一种由设计视图、过程视图、实现视图和部署视图，再加上一个用例视图构成的体系结构描述模型。
 - IEEE于1995年成立了体系结构工作组，起草了体系结构描述框架标准IEEE P1471。
 - Rational从资产重用的角度提出了体系结构描述的规格说明框架。

1.4 体系结构的应用现状

软件体系结构的应用现状

- 体系结构分析、设计与验证（1）

体系结构分析的内容可分为结构分析、功能分析和非功能分析。

非功能分析：定量分析方法、推断分析方法。

Kazman等人提出了一种非功能分析的体系结构分析方法SAAM，并运用场景技术，提出了基于场景的体系结构分析方法，而Barbacci等人提出了多质量属性情况下的体系结构质量模型、分析与权衡方法ATAM。



1.4 体系结构的应用现状

软件体系结构的应用现状

- 体系结构分析、设计与验证（2）

生成一个满足软件需求的体系结构的过程即为体系结构设计。体系结构设计过程的本质在于：将系统分解成相应的组成成分（如构件、连接件），并将这些成分重新组装成一个系统。



1.4 体系结构的应用现状

软件体系结构的应用现状

- 体系结构发现、演化与重用 (1)

体系结构发现解决如何从已经存在的系统中提取软件的体系结构，属于逆向工程范畴。

Waters等人提出了一种迭代式体系结构发现过程，即由不同的人员对系统进行描述，然后对这些描述进行分类并融合，发现并解除冲突，将体系结构新属性加入到已有的体系结构模型中，并重复该过程直至体系结构描述充分。



1.4 体系结构的应用现状

软件体系结构的应用现状

- 体系结构发现、演化与重用（2）

由于系统需求、技术、环境、分布等因素的变化而最终导致软件体系结构的变动，称之为软件体系结构演化。

软件系统在运行时刻的体系结构变化称为体系结构的动态性，而将体系结构的静态修改称为体系结构扩展。体系结构扩展与体系结构动态性都是体系结构适应性和演化性的研究范畴。



1.4 体系结构的应用现状

软件体系结构的应用现状

- 体系结构发现、演化与重用 (3)

体系结构重用属于设计重用，比代码重用更抽象。由于软件体系结构是系统的高层抽象，反映了系统的主要组成元素及其交互关系，因而较算法更稳定，更适合于重用。

体系结构模式就是体系结构重用研究的一个成果，而体系结构参考模型则是特定域软件体系结构的重用的成熟的象征。



1.4 体系结构的应用现状

软件体系结构的应用现状

- 基于体系结构的软件开发方法（1）

在引入了体系结构的软件开发之后，应用系统的构造过程变为“问题定义—>软件需求—>软件体系结构—>软件设计—>软件实现”，可以认为软件体系结构架起了软件需求与软件设计之间的一座桥梁。

1.4 体系结构的应用现状

软件体系结构的应用现状

- 基于体系结构的软件开发方法（2）

软件开发模型是跨越整个软件生存周期的系统开发、运行、维护所实施的全部工作和任务的结构框架，给出了软件开发活动各阶段之间的关系。

目前，常见的软件开发模型大致可分为三种类型：

- 以软件需求完全确定为前提的瀑布模型。
- 在软件开发初始阶段只能提供基本需求时采用的渐进式开发模型，如螺旋模型等。
- 以形式化开发方法为基础的变换模型。

1.4 体系结构的应用现状

软件体系结构的应用现状

- 基于体系结构的软件开发方法（3）

所有开发方法都是要解决需求与实现之间的差距。但是，这三种类型的软件开发模型都存在这样或那样的缺陷，不能很好地支持基于软件体系结构的开发过程。

在基于构件和基于体系结构的软件开发逐渐成为主流情况下，已经出现了基于构件的软件工程。

但是，对体系结构的描述、表示、设计和分析以及验证等内容的研究还相对不足，随着需求复杂化及其演化，切实可行的体系结构设计规则与方法将更为重要。

1.4 体系结构的应用现状

软件体系结构的应用现状

- 特定领域的体系结构框架

特定领域的体系结构是将体系结构理论应用到具体领域的过程，常见的DSSA有：CASE体系结构、CAD软件的参考模型、信息系统的参考体系结构、网络体系结构DSSA、机场信息系统的体系结构和信息处理DSSA等。国内学者提出的DSSA有：北京邮电大学周莹新博士提出的电信软件的体系结构，北京航空航天大学金茂忠教授等人提出的测试环境的体系结构等。



1.4 体系结构的应用现状

软件体系结构的应用现状

- 软件体系结构支持工具

几乎每种体系结构都有相应的支持工具，如Unicon，Aesop等体系结构支持环境，C2的支持环境ArchStudio，支持主动连接件的Tracer工具等。

支持体系结构分析的工具，如支持静态分析的工具、支持类型检查的工具、支持体系结构层次依赖分析的工具、支持体系结构动态特性仿真工具、体系结构性能仿真工具等。

1.4 体系结构的应用现状

◇ 软件体系结构的应用现状

- 软件产品线体系结构（1）

产品线代表着一组具有公共的系统需求集的软件系统，它们都是根据基本的用户需求对标准的产品线构架进行定制，将可重用构件与系统独有的部分集成而得到的。

软件产品线是一个十分适合专业的软件开发组织的软件开发方法，能有效地提高软件生产率和质量、缩短开发时间、降低总开发成本。



1.4 体系结构的应用现状

软件体系结构的应用现状

- 软件产品线体系结构（2）

软件体系结构有利于形成完整的软件产品线。

体系结构在软件产品线的开发中具有至关重要的作用，在这种开发生产中，基于同一个软件体系结构，可以创建具有不同功能的多个系统。



1.4 体系结构的应用现状

软件体系结构的应用现状

- 建立评价软件体系结构的方法

目前，常用的三个软件体系结构评估方法是：

- 体系结构权衡分析方法（ATAM方法）
- 软件体系结构分析方法（SAAM方法）
- 中间设计的积极评审（ARID方法）




1.4 体系结构的应用现状

◇ 软件体系结构的应用现状

目前，软件体系结构尚处在迅速发展之中，越来越多的研究人员正在把注意力投向软件体系结构的研究。关于软件体系结构的研究工作主要在国外展开的，国内到目前为止对于软件体系结构的研究尚处在起步阶段。软件体系结构在国内未引起人们广泛注意的原因主要有两点：

- 软件体系结构从表面上看起来是一个老话题，似乎没有新东西。
- 与国外相比，国内对大型和超大型复杂软件系统开发的经历相对较少，对软件危机的灾难性体会没有国外深刻，因而对软件体系结构研究的重要性和必要性的认识还不很充分。



本章作业与思考题

- 1、根据自己的经验，谈谈对软件危机的看法。
- 2、就项目管理方面而言，软件重用项目与非重用项目有哪些不同之处。
- 3、实际参与/组织一个软件重用项目的开发，然后总结你是如何组织该项目的开发的。
- 4、为什么要研究软件体系结构？
- 5、根据软件体系结构的定义，你认为软件体系结构的模型应该由哪些部分组成？
- 6、在软件体系结构的研究和应用中，你认为还有哪些不足之处？