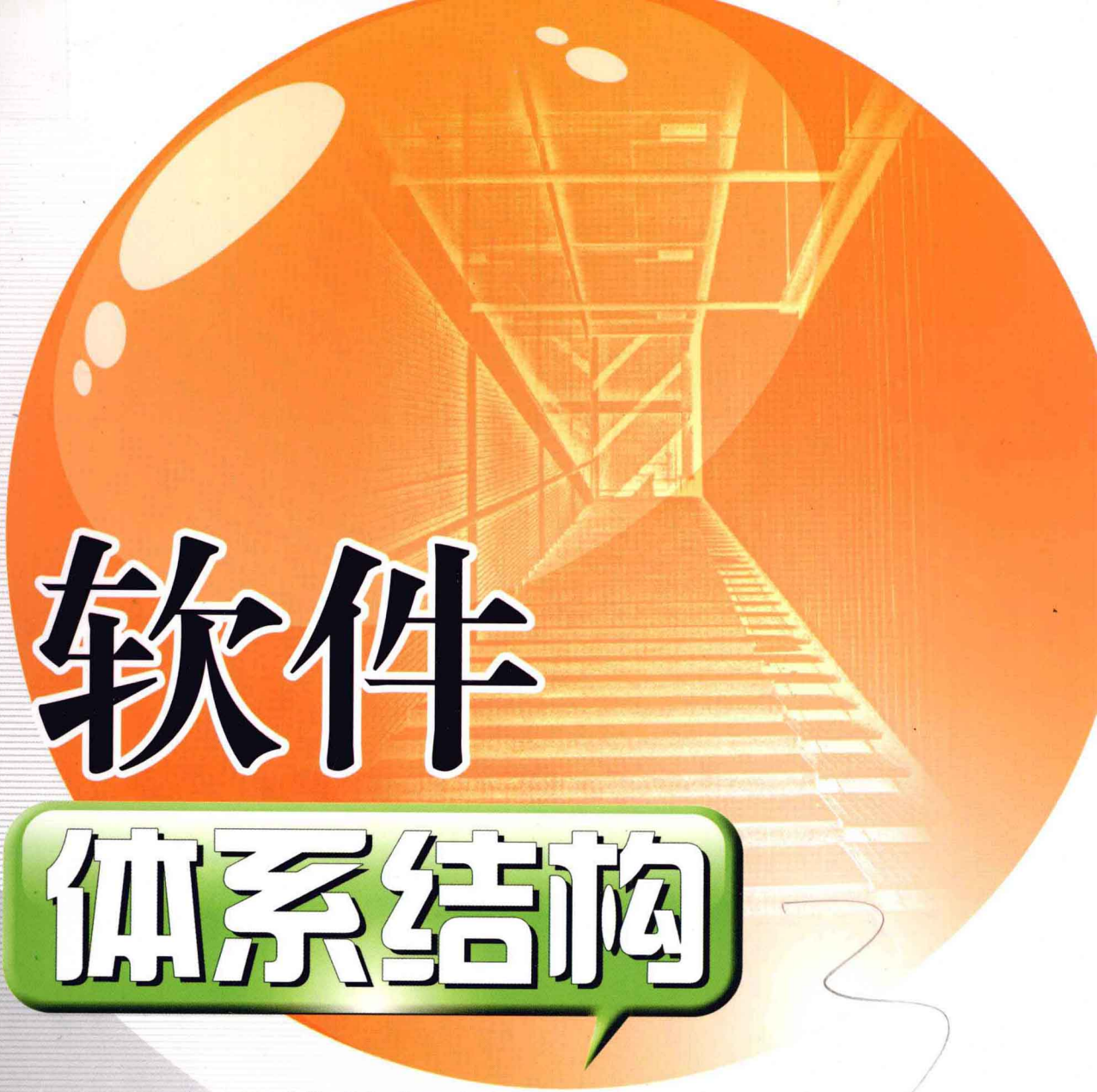


软件体系结构

——面向思维的解析方法

Software Architecture

沈　军◎编著

东南大学出版社
SOUTHEAST UNIVERSITY PRESS

软件体系结构

——面向思维的解析方法

沈　军　编著

东南大学出版社
·南　京·

内容提要

本书采用系统化思维策略，解析软件体系结构相关知识。第1章概述，给出软件体系结构的定义及其内涵、涉及的相关内容及其逻辑关系以及本书的组织结构及应有的学习策略。第2章主要解析软件体系结构赖以建立的基础——软件模型。第3章主要解析用以建立软件体系结构的基本构件——设计模式。第4章主要解析面向同族系统和异族系统的两类软件体系结构基本风格及其关系，同时解析由它们衍生的各种典型风格及其具体应用。第5章主要解析面向Web应用的新3-Tier/n-Tier体系结构的基本工作原理和面向服务的体系结构(SOA)的基本工作原理，并简单解析面向领域的体系结构的基本思想。第6章主要解析软件体系结构的若干基本描述方法，包括非形式化描述方法和形式化描述方法，并通过具体设计工具和应用案例，解析软件体系结构的基本设计方法。第7章主要解析软件体系结构的发展，基于归纳和演绎两种思维策略重点解析可恢复程序语句组件模型与SOA深入以及云计算、元模型与MDA。

本书主要面向普通高等院校计算机学院、软件学院的高年级本科生、硕士生相关课程的教学，也可以满足对计算机软件技术感兴趣的普通读者的自学需求。

图书在版编目(CIP)数据

软件体系结构：面向思维的解析方法 / 沈军编著.
—南京：东南大学出版社，2012.9
ISBN 978-7-5641-3650-5

Ⅰ. ①软… Ⅱ. ①沈… Ⅲ. ①软件—系统结构
Ⅳ. ①TP311.5

中国版本图书馆CIP数据核字(2012)第155478号

软件体系结构——面向思维的解析方法
沈　军　编著

责任编辑：张　煦
责任印制：张文礼
封面设计：毕　真

出版发行：东南大学出版社　　**出版人**：江建中
社　　址：江苏省南京市玄武区四牌楼2号　　**邮　编**：210096
经　　销：全国各地新华书店
印　　刷：南京京新印刷厂
版　　次：2012年9月第1版　　**印　次**：2012年9月第1次印刷
开　　本：787mm×1092mm　1/16　　**印　张**：29
字　　数：724千字　　**定　价**：59.00元
书　　号：ISBN 978-7-5641-3650-5

前　言

计算机的特殊结构成为驱动人类思维潜能的引擎，这种特性主要体现在软件技术的发展上。随着应用的发展，用于处理应用逻辑的软件系统也越来越庞大、越来越复杂，因此，软件体系结构应运而生并得以发展。然而，局限于本土文化带来的思维特性及其延伸的各种问题，目前国内对软件体系结构的研究与认识缺乏系统性和应有的深度，由此，导致了相应教材及著作的缺失。目前，市面上仅有的几本教材和著作，对软件体系结构涉及的知识既零碎又抽象，并且对当今软件技术发展的新思想和新方法缺乏关注或体现。更重要的是，对软件技术发展的思维本质或驱动软件技术发展的核心因素几乎没有涉及。从而，基于教材的认识及其延伸的教学设计及应用实施，严重制约了我国核心软件产业的发展，并从本质上导致了面向软件技术发展的创新思维源泉的枯竭！

计算机是由程序控制，程序固化了人类的思维。为了实现计算机应用，如何构造程序或者如何有效地描述人类思维成为一个核心问题。为此，计算模型（或称为软件模型）得到发展。不同的软件模型决定了不同的软件体系结构。尽管软件模型规定了程序的基本构造方法，然而建立在这种基本构造模型之上的稳定结构或者说这种基本构造模型的应用经验成为优秀软件体系结构的基本建筑块。降低复杂性的基本手段是分层，不同的分层策略成为软件体系结构的基本风格。基于工程学科的特性，软件体系结构的描述与设计显然是软件体系结构的最终目标之一。最后，尽管软件演绎了人类思维，人类思维也有着各种形态，然而，人类的思维一定存在着共性规律。因此，各种计算模型及其延伸的支持语言、工具和平台等，其本质存在认识的连续性。基于这些认识以及对软件体系结构对未来应用发展的重要意义的思考，作者于六年前面向硕士生开设了“软件模型”“软件体系结构”“软件开发方法和技术”“软件工程”等课程，面向本科生开设了“软件体系结构”课程。在多年的教学实践基础上，策划和构筑了本书的体系结构。特别是，结合作者对认知科学及认知计算方面的相关研究工作，将计算思维教学融入本书中，突出作者提出的二维教材设计法的内涵，更演绎软件即思想、软件即文化的本质。

本书采用系统化思维策略，解析软件体系结构相关知识。第 1 章概述，给出软件体系结构的定义，解析软件体系结构的重要性及其内涵；给出软件体系结构涉及的相关内容并解析其逻辑关系；给出本书主体部分的组织结构并解析应有的学习策略。第 2 章软件体系结构基础：软件模型，主要解析软件体系结构赖以建立的基础——软件模型，包括：软件模型的定义、软件模型对软件体系结构的作用、软件模型发展脉络的梳理、各种软件模型的基本原理解析以及对软件模型演化本质的认识和思考。第 3 章软件体系结构基本构件：设计模式，主要解析用以建立软件体系结构的基本构件——设计模式，包括：设计模式的概念、设计模式对软件体系结构的作用、常用设计模式的解析以及对设计模式抽象本质的认识和思考。第 4

章软件体系结构基本风格，主要解析面向同族系统和异族系统的两类软件体系结构基本风格及其关系，同时解析由它们衍生的各种典型风格及其具体应用，并对基本风格的思维本质及其发展和演化的脉络与其规律进行剖析。第 5 章软件体系结构案例解析，主要解析面向 Web 应用的新 3-Tier/n-Tier 体系结构的基本工作原理和面向服务的体系结构(SOA)的基本工作原理，并简单解析面向领域的体系结构的基本思想。第 6 章软件体系结构的描述与设计，主要解析软件体系结构的若干基本描述方法，包括非形式化描述方法和形式化描述方法，并通过具体设计工具和应用案例解析软件体系结构的基本设计方法。第 7 章软件体系结构的发展，主要从思维特征的角度，解析软件体系结构发展的两种路线及其带来的相应技术、方法和思想，包括：基于归纳思维策略重点解析可恢复语句组件模型、SOA 深入以及云计算，基于演绎思维策略重点解析元模型及 MDA。并且，对软件体系结构的发展本质进行深入剖析，指出并诠释了软件体系结构的发展回归到图灵机的本质——递归思想及其应用，实现哲学原理对计算机软件的直接投影。同时，也指出了文化特性对思维的影响及其给计算机软件技术发展带来的重要影响，诠释了创新的本质——文化及其作用。本书第 1 章到第 5 章为一个单元，第 6 章和第 7 章为一个单元，根据学时的安排，使用者可以灵活安排。

本书的出版，得到了东南大学出版社的支持，在此表示衷心感谢！我的学生范文、毛振洪、彭殷路和韩涛为本书的案例及图片收集整理做了大量工作。此外，参与本书写作的还有我的学生朱晓建、宋晓斐、杨先河、朱亚飞、奚子石、刘小筱、朱云斌、李金凡、张磊、高松、张旸等，在此向他们表示衷心感谢！

本书的出版，得到国家重点基础研究发展计划(“973 计划”)项目“服务可扩展的网络体系结构描述方法研究”(2009CB320501 子项)的支持。

本书中的观点都是基于作者个人的认识、理解和感悟，难免存在错误和不妥之处，希望读者来信批评与指正。作者恳切盼望各位同仁来信切磋，作者的 E-mail 地址是 junshen@seu.edu.cn。

作　者

2012 年 2 月 18 日于古都金陵

目　　录

第1章 概 述

本章主要给出软件体系结构的定义，解析软件体系结构的重要性以及涉及的主要内容。在此基础上，给出本书的组织结构以及应有的学习策略。

1.1 什么是软件体系结构

所谓**体系结构**(Architecture)，一般是指一种思想的抽象，它通过一些原则和方法等具体体现。这种解释本质上是强调了体系结构的“体系”的涵义，主要侧重于抽象级别较高的概念层面。体系结构的另一种解释，是指系统的基本组成元素及其相互关系的抽象。显然，这种解释本质上是强调了体系结构的“结构”的涵义，主要侧重于抽象级别较低的技术层面。

软件体系结构，是体系结构概念在软件上的投影或具体应用。软件体系结构通常取体系结构定义中的第二种解释，即软件体系结构是指一系列关于软件系统组织的重大决策，是软件系统结构的结构，由软件元素、元素的外部可见属性及元素间的关系组成。

对于体系结构，理解的核心在于抽象程度。首先，体系结构并不等同于系统结构的具体设计说明，而是该说明的一种抽象表示。其次，根据上下文或特定应用背景，要区分究竟侧重于“体系”还是“结构”，由此决定其具体内涵。

1.2 为什么要研究软件体系结构

得益于计算机工具的特殊结构，计算机应用都是通过构造软件系统实现。随着人类文明信息化程度的深入，企业 IT 应用越来越复杂，导致软件系统越来越庞大。众所周知，如果要建造一座摩天大厦，首先必须给出其设计蓝图，然后才能在设计蓝图的指导下，一步步按照工程要求进行建造，绝不是仅仅凭想象随意施工。与之相似，如果要构造一个庞大、复杂的软件系统，显然也必须在具体实施之前进行其蓝图的设计，即进行软件体系结构设计。这是复杂大系统构造方法与简单小系统构造方法之间的本质区别。

另外，相对于其他系统而言，软件系统有其特殊性。一方面，软件构造的基本方法和技术，随着人们对软件认识程度的深入不断发展，目前已诞生了各种各样异质的方法和技术，并且应用系统赖以执行的基础环境，包括硬件系统和系统软件平台，也存在异质性。另一方面，应用具有恒变性，业务规则多次被重新定义，新业务模型也屡屡出现。随着应用的不断发展，多年的反复改造使很多系统之间具有复杂的交叉依赖，异质度和冗余度相当高。不同时期采用不同技术和平台构建的软件系统逐渐形成“信息孤岛”，企业应用程序环境变成一个大杂烩，其维护和集成成本居高不下。特别是面对新世纪全球化、虚拟化的商业环境，许

多企业由于不能及时提供所需功能而错失良机。因此，如何匹配技术的动态性和应用的动态性，显然必须从一个战略高度进行分析。也就是说，为了有效地“重用”现有系统并与之“协同”工作，及时开发新的业务功能，减轻成本压力，必须为企业计算建立十分“灵活”和“敏捷”的“完美”结构，使企业软件环境内部保持恒久的“有序度”。企业软件体系结构正是实现这一需求的有力武器。

更进一步的认识是，软件体系结构也是软件自身发展的使然。按照事物发展的普遍规律，从软件发展的脉络来看，其目前正处于由初级阶段到高级阶段的过渡时期。软件体系结构的建立以及以软件体系结构为核心的新一代软件开发方法学的研究与发展标志着该学科的逐步成熟。

1.3 软件体系结构涉及的内容

软件体系结构研究源自于软件工程研究，目前已基本上相对独立。事实上，软件工程主要关注软件开发的整个过程，涉及软件开发整个过程的各个方面，包括人员、资源和费用等等非技术的因素。而软件体系结构主要关注软件系统本身的抽象结构定义和设计。从认识论层面可以将软件工程和软件体系结构看做针对软件开发的宏观视图和(某个)微观视图及它们的辩证关系。

按照给出的软件体系结构的定义，软件体系结构应该涉及基本软件构造模型、设计模式、基本风格、典型案例、描述与设计等内容。其中，基本软件构造模型是软件体系结构建立的基础和最小粒度的元素。不同的软件模型蕴涵了其直接支持的软件体系结构，同时，软件模型发展的轨迹也反映了软件体系结构的发展历程和进化理念，两者在思想本质上具有通约性。设计模式是面向对象软件设计中关于对象关系和结构的一种设计经验的抽象，它能有效支持软件结构对于应用的恒变特性的适应性，提高软件系统的维护能力。尽管设计模式来源于面向对象软件设计，但其思维本质对软件体系结构如何适应应用的恒变特性也有同样的指导意义。软件体系结构中往往在某个局部大量采用设计模式，以重用久经考验的设计经验。软件体系结构基本风格抽象了一些经过实验验证的有效的软件体系结构基本设计方法，这些方法广泛应用于现实软件系统的设计之中。典型案例是指软件体系结构的各种具体实现，它集成了软件模型、设计模式和软件体系结构基本风格。软件体系结构描述是指通过一定的语言或符号，从形式上定义软件体系结构，实现软件体系结构蓝图的书面具现，从而对软件体系结构进行交流、审核和分析验证。软件体系结构设计是指针对某个具体系统或应用，利用软件模型、设计模式和软件体系结构基本风格等知识以及相应描述手段，进行艺术创造，创建出能够处理该具体系统或应用的一种完美的蓝图。

软件体系结构所涉及的各部分内容从逻辑上构成一个整体，如图 1-1 所示。

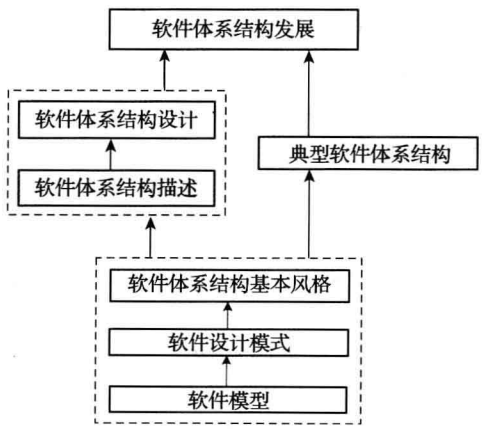


图 1-1 软件体系结构涉及的内容及其关系

1.4 本书的组织结构及学习策略

本书后面的章节，根据图1-1所示的逻辑关系展开。第2章到第4章构成一个思维单元，它们从细粒度到粗粒度，建立了软件体系结构三个层次的视图。第5章单独构成一个思维单元，解析第一个思维单元的具体应用。第6章单独构成第三个思维单元，从理论上和形式上对软件体系结构进行描述，并且在第一个思维单元的知识和描述方法与手段的基础上，解析如何针对一个具体应用问题进行其软件体系结构设计。第7章单独构成一个思维单元，解析软件体系结构的最新发展。

针对本书的知识体系，宏观上可以采用面向应用和面向设计两种基本学习策略。采用面向应用策略，主要学习第1章到第5章的内容，以理解目前软件开发领域中常用软件体系结构的相关概念和知识为核心。采用面向设计策略，应该学习所有内容，一方面理解目前软件开发领域中常用软件体系结构的相关概念和知识；另一方面，在此基础上能够设计具体的软件体系结构。通俗地讲，面向应用策略主要培养工程师，而面向设计策略主要培养设计师。微观上，应该采用基于模式的学习策略，即重点认识具体模型、方法和技术背后蕴涵的各种模式(隐性知识)及其在这些具体模型、方法和技术中的应用(模式建构)，在模式层次进行学习。具体而言，映射到每一章内容的学习，首先学习各种具体模型、方法和技术，包括它(们)的概念、术语和基本原理等。然后再深入一步，思考并发掘这些概念、术语和基本原理中共同采用的一些问题处理思想，例如分层处理思想、抽象粒度等。最后还要在此基础上，对各种概念、术语和基本原理等进行比较和分析，认识它们之间的发展和演化的脉络及其原因，例如软件模型发展和演化的脉络及其原因等。针对本书整个内容的学习，应该在普通知识(显性知识)和模式知识(隐性知识)两个层面展开学习。首先，在每个层面注意从各种具体知识点中形成知识链，例如普通知识层面中的知识关系、模式知识层面中的模式关系。然后，将两个知识面综合，形成知识球，从而达到知识的触类旁通和融会贯通。

1.5 本章小结

本章首先给出软件体系结构的定义及其重要性并解析其内涵，然后给出软件体系结构涉及的相关内容并解析其逻辑关系。在此基础上，给出本书主体部分的组织结构并解析应有的学习策略，从而为本书的展开建立思维基础。

习 题

1. 什么是软件体系结构？如何理解“系统结构”和“系统结构的结构”两者之间的区别？
2. 建立一个精美的小狗窝与建立一座庞大的摩天大厦，在策略上究竟有何本质不同？
3. 如何理解软件体系结构和软件工程之间的区别和联系？
4. 如何理解软件模型、设计模式和基本风格之间的关系？

5. 什么是应用的恒变特性？你认为对应用的恒变特性的灵活适应是不是软件体系结构设计的核心，为什么？

6. 事物初级阶段发展和高级阶段发展分别基于什么样的思维特征？

第2章　软件体系结构基础：软件模型

本章主要解析软件体系结构赖以建立的基础——软件模型。首先，给出软件模型的定义，解析软件模型对软件体系结构的作用。然后，梳理软件模型发展脉络，解析各种软件模型的基本原理。最后，对软件模型进行深入认识和思考。

2.1　什么是软件模型

所谓**模型**(Model)，一般是指客观世界中存在事物的一种抽象。事物可以是具体的，例如房子、人等；也可以是抽象的，例如思想、算法等。

模型一般都需要通过某种形式表达出来，以便交流。从形式上看，模型的表达有文字(包括自然语言和数学语言)、图形(图形语言)或图文混合。用数学语言表达的模型称为数学模型，这种模型的形式化级别最高，一般用来描述事物的抽象本质，刻画事物内在的稳定规律。例如，$s=v_0t+\frac{1}{2}at^2$ 描述了匀加速直线运动的本质，刻画了时间、加速度和路程之间的稳定变化规律。相对于数学模型，用自然语言或图形语言表达的模型的形式化级别较低，一般用来描述对事物的某种认识和理解，表示一种观点。例如，一座房子的结构、一个系统的结构或某种思想体系等。数学模型是精确的定量描述，其他模型则是一种定性描述。

所谓**软件模型**(Software Model)，是指软件的一种抽象，目前一般通过非数学模型来描述。相对于其他事物，软件具有特殊性。这主要体现在软件描述的基本元素的一致性，也就是说，无论如何描述软件，同构模型中描述的最基本单元的抽象都是统一的。本书中，将这种统一的基本单元的抽象称为软件模型，而将软件系统的抽象称为软件体系结构。因此，软件模型可以看做一种**元模型**(Meta Model)。

2.2　软件模型对软件体系结构的作用

软件模型作为软件组成的最基本单元的抽象，它既反映了软件体系结构构建的核心思想，也奠定了软件体系结构构建的基础。一方面，它定义了软件体系结构构建的基本单元元素的形态；另一方面，它定义了基本单元元素之间关系的基本形态。不同的软件模型隐式地定义了软件体系结构构建的不同方法。

2.3　软件模型的发展脉络

审视软件模型从诞生到发展的历程，尽管各种软件模型的发展存在一定的时间交叉，但

从其是否作为软件构造技术的主体支撑技术来说，软件模型的发展基本上符合图 2-1 所示的轨迹。

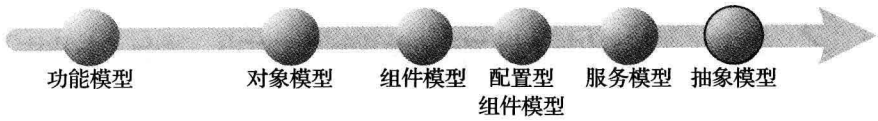


图 2-1 软件模型的发展轨迹

软件模型的发展脉络也清晰地体现了计算机应用发展的历程以及计算机技术发展的历程。计算机应用的发展和计算机技术的发展相辅相成，一方面，计算机应用的发展对计算机技术提出了新的要求，促进计算机技术的发展；另一方面，计算机技术的发展又为新型计算机应用的发展提供了基础，促进计算机应用的发展。作为计算机技术之一的软件技术在计算机应用和其他计算机技术之间建立起桥梁。因此，软件模型的发展事实上也就是不断动态地黏合应用与技术。图 2-2 给出了计算机应用、计算机技术和软件模型的关系。

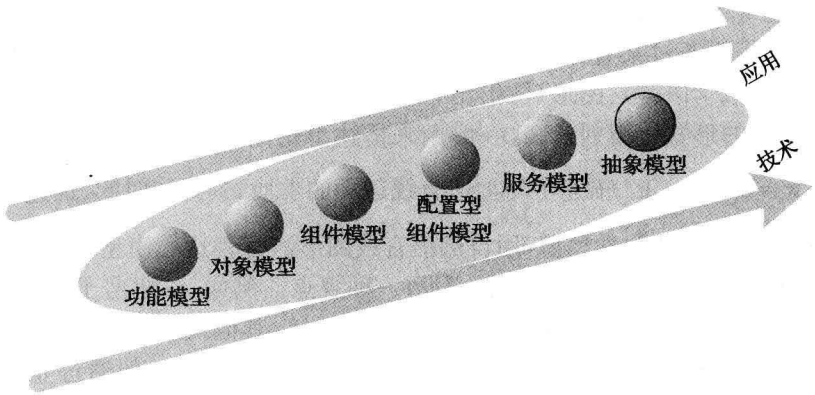


图 2-2 技术、应用和软件模型的关系

2.4 软件模型解析

本节按照软件模型的发展轨迹，主要解析各种软件模型的基本原理及其思维本质，并阐述其对软件体系结构建立的影响。

2.4.1 功能模型

功能模型(Function Model)也可以称为过程模型或函数模型，它是模型化软件构建方法的第一个基本模型。功能模型的基本原理是将一个系统分解为若干个基本功能模块，基本功能模块之间可以按需进行调用。基本功能模块集合及其调用关系集合构成一个系统的模型。

功能模型诞生于 20 世纪 60 年代，它强调了对程序中数据处理(功能)的抽象，通过功能分解和综合的方法，降低系统构造的复杂度。从而实现一体式程序体系结构向结构化程序体系结构的转变，并建立了结构化软件设计方法。

功能模型的核心之一是基本功能模块的抽象及耦合。事实上,基本功能模块是一种处理方法的抽象,这种方法独立于其处理的具体数据集,建立在抽象数据集上。通过将抽象数据集具体化,就可以实现处理方法在某个具体数据集上的作用,从而实现处理方法的重用。因此,基本功能模块的抽象一般需要定义其处理的抽象数据集。具体实现中,基本功能模块一般有**函数**(Function)和**过程**(Procedure)两种形式,前者返回处理结果,后者不返回处理结果。抽象数据集称为形式参数,具体数据集称为实际参数。图 2-3 分别给出了基本功能模块在 PASCAL 语言和 C 语言中的实现。

```
FUNCTION  函数名(形式参数列表):函数返回值数据类型;
   数据组织定义;
BEGIN
   数据处理定义;
END;
```

```
函数返回值数据类型   函数名(形式参数列表)
{
   数据组织定义;

   数据处理定义;
}
```

```
PROCEDURE  过程名(形式参数列表);
   数据组织定义;
BEGIN
   数据处理定义;
END;
```

```
void  函数名(形式参数列表)
{
   数据组织定义;

   数据处理定义;
}
```

PASCAL 语言实现　　C 语言实现

图 2-3　基本功能模块的实现

基本功能模块的耦合,是指一个模块调用另一个模块时如何进行被调模块的抽象数据集的具体化以及被调模块如何返回其处理结果给主调模块。前者一般称为参数传递,后者称为函数返回。目前,参数传递和函数返回的实现方式基本上都是通过堆栈进行,图 2-4 给出了参数传递和函数返回实现的基本思想。按照传递的方式,参数传递基本上有**值传递**和**地址传递**两种。值传递将实际参数值复制到堆栈,地址传递则是将实际参数值存放的内存地址复制到堆栈。因此,当被调模块的抽象数据集具体化后,值传递方式不会因为被调模块的处理而改变原始的实际参数值,而地址传递方式由于被调模块的处理会通过实际参数值存放的内存地址而间接地作用于原实际参数,从而改变原始的实际参数值。图 2-5 是两种参数传递方式的 C 语言实现。

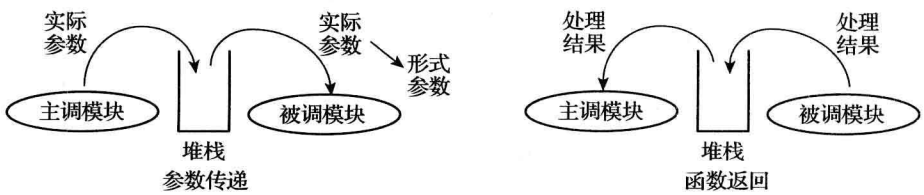


图 2-4　参数传递和函数返回实现的基本思想

```
int submodule (int p, int q)
{
  int r;
  r = p + q;
  return r;
}
mainmodule ( )
{
   int x=10, y=20, z=0;
   z += submodule (x, y);
   printf( "The Result is : %d\n" , z);
}
```

```
int submodule (int *p, int *q)
{
  int r;
  r = *p + *q;
  return r;
}
mainmodule ( )
{
   int x=10, y=20, z=0;
   z += submodule (&x, &y);
   printf( "The Result is : %d\n" , z);
}
```

图 2-5　两种参数传递方式的 C 语言实现

功能模型的核心之二是递归思想的具体实现。所谓**递归**(Recursion)，是指用同一种处理方法(该方法通过缩小待处理数据集规模)来处理不断缩小规模的数据集，并通过不断综合小规模数据集的处理结果来得到大规模数据集的处理结果的一种问题处理方法。图 2-6 给出了递归方法的基本思想。

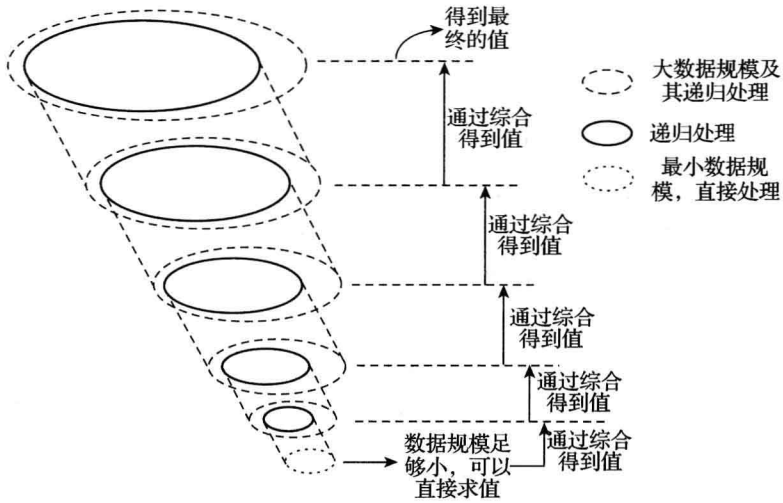
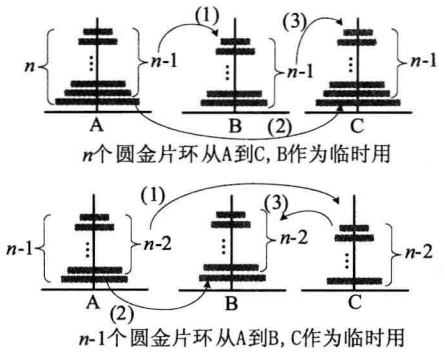


图 2-6　递归方法的基本思想

功能模型中递归思想体现在两个方面，一个是基本功能模块的递归应用，另一个是处理逻辑或数据组织方式的递归应用。基本功能模块的递归应用是将图 2-6 中的一种处理方法通过一个基本功能模块实现，将数据集规模作为基本功能模块的形式参数之一，这样在基本功能模块处理逻辑的定义中，显然需要在缩小后的数据规模上再调用其自身(使用同一种处理方法)。可见，递归是一种特殊的模块耦合关系，其主调模块和被调模块是同一个处理模块。图 2-7 给出了基本功能模块递归应用的一个具体案例。根据模块调用关系的不同，递归可以呈现多种具体应用形式，如图 2-8 所示。




```
void move (unsigned n,char a,char c,char b)
{
  if (n > 0)
  {   move(n-1, a, b,c );        -------> (1)
   printf("%d : %c -> %c\n",  n, a, c);  -------> (2)
   move(n-1, b, c,a );     -------> (3)
  }
}
```

图 2-7　基本功能模块递归应用具体案例——汉诺塔问题求解

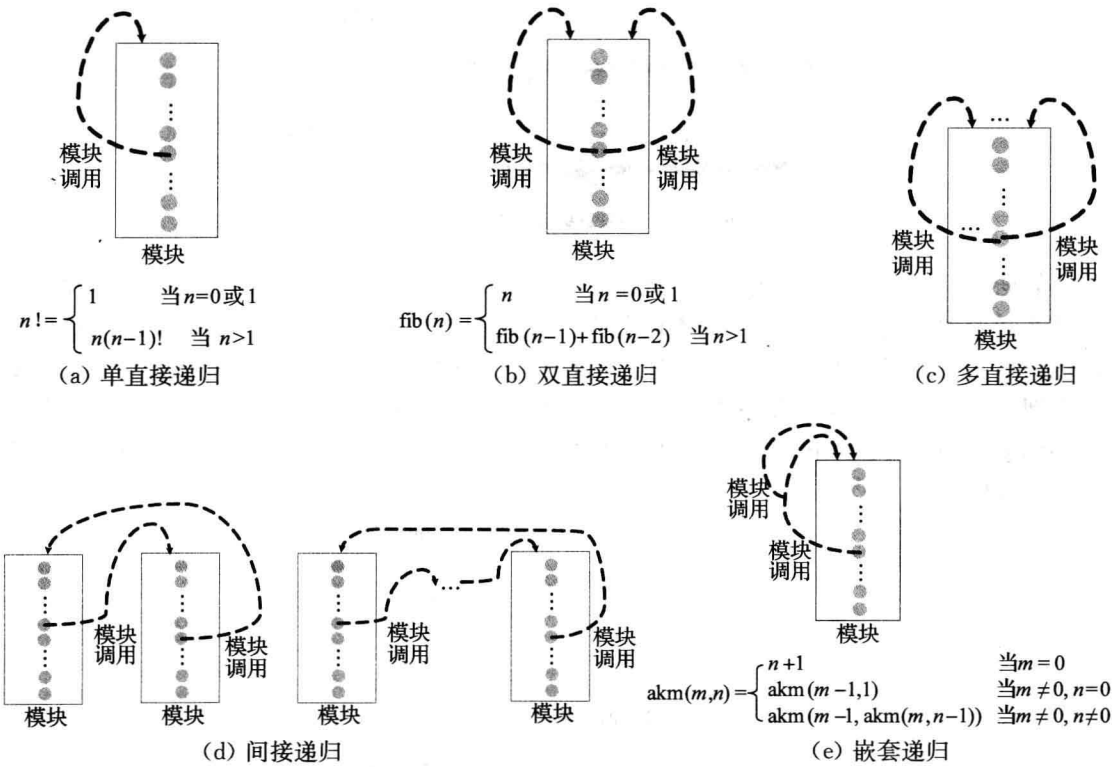


图 2-8　递归的多种具体应用形式

处理逻辑的递归应用，是指将问题的整个处理逻辑看做数据集，将基本的处理逻辑看做处理方法，从而实现用基本的处理逻辑及其组合来实现处理不同复杂度问题的整个处理逻辑。功能模型建立了三种基本的处理逻辑：**顺序**、**分支**和**循环**。图 2-9 给出了它们的语义。图 2-10 解释了处理逻辑的递归应用内涵。由图 2-10 可知，程序 A(大程序)和子程序 B(小程序)在思维上具有显式通约性。