本书受国家留学基金委资助
受中国驻美国休斯顿领馆教育组资助

Software Architecture

软件体系结构

覃征 何坚 谢国彤 王志敏 张丽 王向华 编著
Qin Zheng He Jian Xie Guotong Wang Zhimin ZHang Li Wang Xianghua

西安交通大学出版社
XI'AN JIAOTONG UNIVERSITY PRESS

本书受国家留学基金委资助
受中国驻美国休斯顿领馆教育组资助

Software Architecture
软件体系结构

覃征 何坚 谢国彤 王志敏 张丽 王向华 编著
Qin Zheng He Jian Xie Guotong Wang Zhimin ZHang Li Wang Xianghua

西安交通大学出版社
XI'AN JIAOTONG UNIVERSITY PRESS
·西 安·

内容提要

本书系统介绍了软件体系结构的基本概念、构建模式、组态集成、形式化描述和集成开发环境。全书共6章。第1章概要介绍了软件体系结构的研究背景、动态和面临的问题;第2章详细分析软件体系结构的构建模式及异构集成;第3章结合实例描述软件体系结构组态分析与应用;第4章从WRIGHT形式化描述语言和CSP语义学的角度对软件体系结构进行了定量的表示;第5章介绍了软件体系结构集成开发环境的设计与实现;第6章对今后软件体系结构的研究前景做了分析和展望。

本书可作为大专院校软件体系结构的教科书,也可作为从事软件工程、软件体系结构理论研究人员和从事软件研究和开发工作、软件体系结构系统设计、开发及应用工作有关人员的参考书。

图书在版编目(CIP)数据

软件体系结构/覃征等编著. —西安:西安交通大学出版社,2002.12

ISBN 7-5605-1620-3

Ⅰ.软… Ⅱ.覃… Ⅲ.软件工程—系统结构 Ⅳ.TP311.5

中国版本图书馆CIP数据核字(2002)第093555号

*

西安交通大学出版社出版发行

(西安市兴庆南路25号 邮政编码:710049 电话:(029)2668315)

陕西向阳印务有限公司印装

各地新华书店经销

*

开本:727mm×960mm 1/16 印张:17.375 字数:284千字

2002年12月第1版 2002年12月第1次印刷

印数:0 001~3 000 定价:38.00元

发行科电话:(029)2668357,2667874

前　言

20世纪60年代的软件危机使得软件工程的研究成为热点问题。为保证软件质量，提高软件可靠性、可重用性和可维护性，软件设计的核心已从“算法＋数据结构＝程序”的传统计算模式转向对系统的总体结构即软件体系结构的设计和规范。软件体系结构不仅确定了系统的组织结构和拓扑结构，并且说明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。

作为软件工程中的一个新兴研究领域，软件体系结构的重要性已为大家所共识。但是，由于没有统一的体系结构划分标准和设计原则，缺乏有效的计算机辅助工具帮助开发人员进行设计，造成软件体系结构很难被一般的开发人员所理解。本书正是在这种情况下开发撰写的。本书重点介绍了软件体系结构的构建模式、组态集成和形式化描述方法，提出软件集成开发环境模型，对当前主要的体系结构开发环境进行了分类比较。

本书共有6章，各章内容如下：

第1章　绪论；

第2章　软件体系结构的构建模式；

第3章　软件体系结构组态分析与应用；

第4章　软件体系结构的形式化描述；

第5章　软件体系结构集成开发环境的设计与实现；

第6章　软件体系结构研究的展望。

本书由覃征教授确定研究内容和书的整体结构，分别由何坚完成第1章，第5章，谢国彤完成第2章和第3章，王志敏完成第4章和第6章，张丽负责统稿和英文翻译，王向华负责统稿和日文翻译。由于我们水平有限，时间紧迫，加之软件体系结构的发展非常迅速，书中难免有疏漏和不妥之处，敬请读者批评指正。

作者

2002年11月

目 录

第 1 章 绪论

第 2 章 软件体系结构的构建模式

第 3 章 软件体系结构组态分析与应用

第 4 章 软件体系结构的形式化描述

第 5 章 软件体系结构集成开发环境的设计与实现

第 6 章 软件体系结构研究的展望

第1章 绪论

1.1 研究背景

1.1.1 计算机系统发展历程

所谓计算机系统就是指适当地组织在一起的一系列系统元素的集合，这些系统元素互相配合、相互协作，通过对信息的处理而完成预先定义的目标。计算机系统中通常包含的系统元素有：计算机软件、计算机硬件、人员、数据库、文档和过程。其中，软件是程序、数据结构和相关文档的集合，用于实现所需要的逻辑方法、过程或控制；硬件是提供计算能力的电子设备和提供外部世界功能的电子机械设备（例如传感器、马达、水泵等）；人员是硬件和软件的用户和操作者；数据库是通过软件访问的大型的、有组织的信息集合；文档是描述系统使用方法的手册、表格、图形及其他描述性信息；过程是一系列步骤，它定义了每个系统元素的特定使用方法或系统驻留的过程性语境。

迄今为止，计算机系统已经经历了4个不同的发展阶段。下面简要地介绍各个发展阶段的特点。

20世纪60年代中期以前，是计算机系统发展的早期时代。在这个时期通用硬件已经相当普遍，软件却是为每个具体应用而专门编写的，大多数人认为软件开发是无需预先计划的事情。这时的软件实际上就是规模较小的程序，程序的编写者和使用者往往是同一个（或同一组）人。由于程序规模小，编写起来相当容易，也没有什么系统化的方法，对软件开发工作更没有进行任何管理。这种个体化的软件环境，使得软件设计往往只是在人们头脑中隐含进行的一个模糊过程，除了程序清单之外，根本没有其他文档资料保存下来。

从20世纪60年代中期到20世纪70年代中期，是计算机系统发展的第二代。在这10年中计算机技术有了很大进步。多道程序、多用户系统引入了人机交互的新概念，开创了计算机应用的新境界，使硬件和软件的配合上了一个新的层次。实时系统能够从多个信息源收集、分析和转换数据，从而使得进程控制能以毫秒而不是分钟来进行。在线存储技术的进步导致了第一代数据库管理系统的出现。

计算机系统发展的第二代的一个重要特征是出现了“软件作坊”，广泛使用产品软件。但是，“软件作坊”基本上仍然沿用早期形成的个体化软件开发方法。随着计算机应用的日益普及，软件数量急剧膨胀。在程序运行时发现的错误必须设法改正；用户有了新的需求时必须相应地修改程序；硬件或操作系统更新时，通常需要修改程序以适应新的环境。上述种种软件维护工作，以令人吃惊的比例耗费资源。更严重的是，许多程序的个体化特性使得它们最终成为不可维护的。“软件危机”就这样开始出现了。1968 年北大西洋公约组织的计算机科学家在联邦德国召开国际会议，讨论软件危机问题，在这次会议上正式提出并使用了“软件工程”这个名词，一门新兴的工程学科就此诞生了。

计算机系统发展的第三代从 20 世纪 70 年代中期开始，并且跨越了整整 10 年。在这 10 年中计算机技术又有了很大进步。分布式系统极大地增加了计算机系统的复杂性，局域网、广域网、宽带数字通信以及对“即时”数据访问需求的增加，都对软件开发者提出了更高的要求。但是，在这个时期软件仍然主要在工业界和学术界应用，个人应用还很少。

这个时期的主要特点是出现了微处理器，而且微处理器获得了广泛应用。以微处理器为核心的“智能”产品随处可见，当然，最重要的智能产品是个人计算机。在不到 10 年的时间里，个人计算机已经成为大众化的商品。

在计算机系统发展的第四代已经不再看重单台计算机和程序，人们感受到的是硬件和软件的综合效果。由复杂操作系统控制的强大的桌面机及局域网和广域网，与先进的应用软件相配合，已经成为当前的主流。计算机体系结构已迅速地从集中的主机环境转变成分布的客户机/服务器(或浏览器/服务器)环境。世界范围的信息网为人们进行广泛交流和资源的充分共享提供了条件。

软件产业在世界经济中已经占有举足轻重的地位。随着时代的前进，新的技术也不断地涌现出来。面向对象技术已经在许多领域迅速地取代了传统的软件开发方法。

软件开发的“第四代技术”改变了软件界开发计算机程序的方式。专家系统和人工智能软件终于从实验室中走出来进入到实际应用中去，解决了大量实际问题。应用模糊逻辑的人工神经网络软件，展现了模式识别与拟人信息处理的美好前景。虚拟现实技术与多媒体系统，使得与用户的通信可以采用和以前完全不同的方法。遗传算法使我们有可能开发出驻留在大型并行生物计算机上的软件。

1.1.2　软件过程和开发方法

软件过程是为了获得高质量软件产品所需要完成的一系列任务的框架，它规定了完成各项任务的工作步骤。软件过程必须科学、合理，才能开发出高质量的软件产品。软件过程又称软件生存周期过程，是在软件生存周期内为达到一定目标而必须实施的一系列相关过程的集合。一个良好定义的软件过程对软件开发的质量和效率有着重要影响。

按照在软件生命周期全过程中应完成的任务的性质，在概念上可以把软件生命周期划分成问题定义、可行性研究、需求分析、概要设计、详细设计、编码和单元测试、综合测试以及维护等 8 个阶段。实际上，在从事软件开发工作时，软件的规模、种类、开发环境以及使用的技术方法等因素，都影响着阶段的划分。因此，一个科学、有效的软件过程应该定义一组适合于所承担的项目特点的任务集合。

瀑布模型历史悠久、广为人知，它的优势在于它是规范的、文档驱动的方法；这种模型的问题是，最终交付的产品可能不是用户真正需要的。瀑布模型见图 1.1。

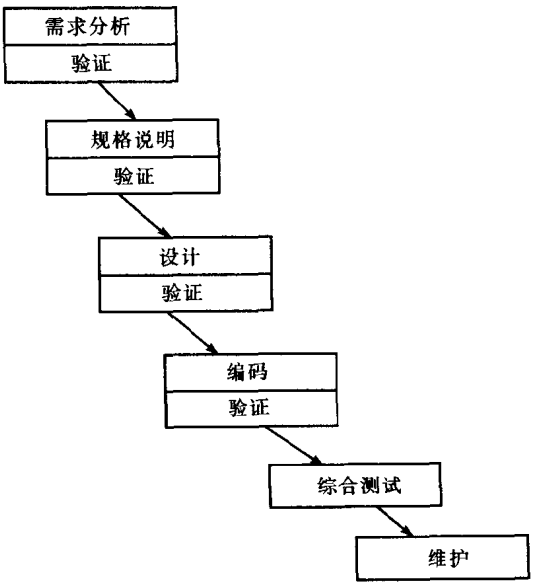

图 1.1　传统的瀑布模型

快速原型模型正是为了克服瀑布模型的缺点而提出来的。它通过快速构建起一个可运行的原型系统，让用户试用原型并收集用户反馈意见的办法，获取用户的真实需求。快速原型模型见图 1.2。

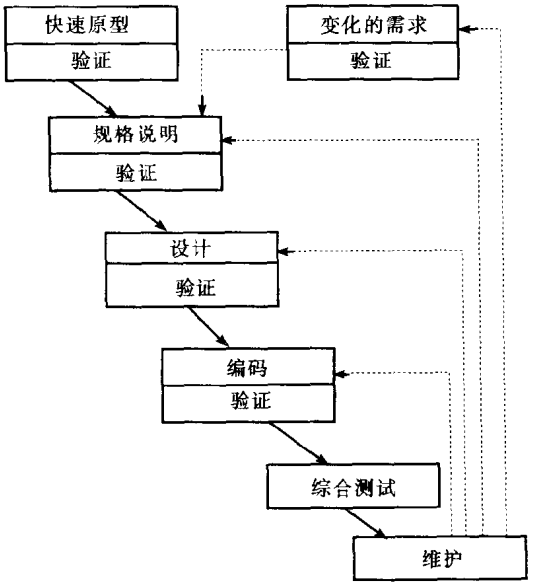

图 1.2 快速原型模型

增量模型具有能在软件开发的早期阶段，使投资获得明显回报和易于维护的优点，但是，要求软件具有开放结构是使用这种模型时固有的困难。增量模型见图 1.3。

风险驱动的螺旋模型适用于大规模的内部开发项目，但是，只有在开发人员具有分析风险和排除风险的经验及专门知识时，使用这种模型才会获得成功。螺旋模型见图 1.4。

喷泉模型是一种典型的、适合于面向对象范型的过程模型，喷泉模型软件生命周期必须是循环的，也就是说，软件过程必须支持反馈和迭代。喷泉模型见图 1.5。

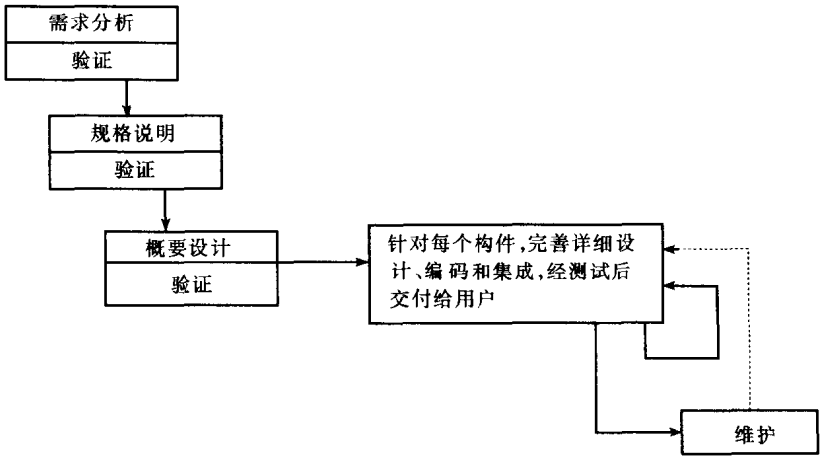

图 1.3　增量模型

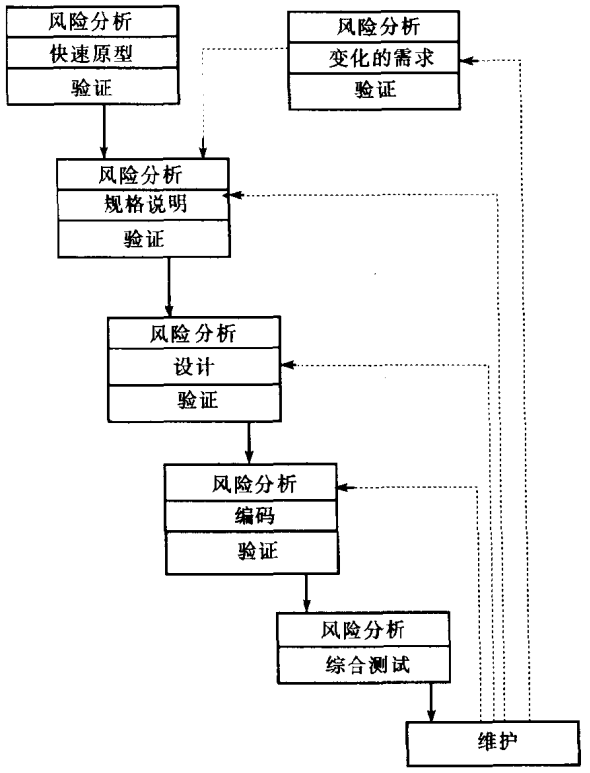

图 1.4　螺旋模型

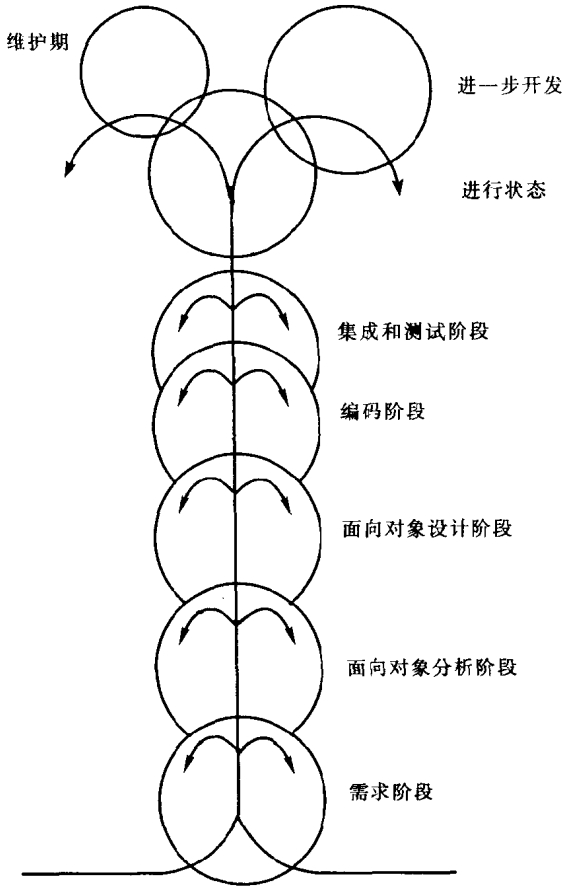

图 1.5 喷泉模型

1.2 软件复用

大多数工程项目都尽可能采用可以复用的部件。但计算机软件的工程项目有很大的特殊性，大量的软件项目，一切工作都是从头开始。这种特有的现象正是软件开发效率低下和质量问题严重的根源所在，也是软件成本高昂的因素之一。

分析传统产业的发展，其基本模式均是符合标准的零部件（构件）生产以及基于标准构件的产品生产（组装），其中，构件是核心和基础，“复用”是必需的手段。实践表明，这种模式是产业工程化、工业化的必由之路。标准零部件生产业的独立存在和发展是产业形成规模经济的前提。

机械、建筑等传统行业以及年轻的计算机硬件产业的成功发展,均是基于这种模式,这充分证明了它的可行性和正确性。这种模式是软件产业发展的良好借鉴,软件产业要发展并形成规模经济,标准构件的生产和构件的复用是关键因素。这正是软件复用受到高度重视的根本原因。

软件复用可以从多个角度进行考察。依据复用的对象,可以将软件复用分为产品复用和过程复用。产品复用指复用已有的软件构件,通过构件集成(组装)得到新系统。过程复用指复用已有的软件开发过程,使用可复用的应用生成器来自动或半自动地生成所需系统。过程复用依赖于软件自动化技术的发展,目前只适用于一些特殊的应用领域。产品复用是目前现实的、主流的途径。

依据对可复用信息进行复用的方式,可以将软件复用区分为黑盒复用和白盒复用。黑盒复用指对已有构件不需作任何修改,直接复用。这是理想的方式。白盒复用指已有构件并不能完全符合用户需求,要根据用户需求进行适应性修改后才可使用。在大多数应用的组装过程中,构件的适应性修改是必需的。

软件复用有3个基本问题:一是必须有可以复用的对象;二是所复用的对象必须是有用的;三是复用者需要知道如何去使用被复用的对象。软件复用包括两个相关的过程:可复用软件(构件)的开发(Development for Reuse)和基于可复用软件(构件)的应用系统构造(集成和组装)(Development with Reuse)。解决好这几个方面的问题,才能实现真正成功的软件复用。

实现软件复用的关键因素(技术和非技术因素)主要包括:软件构件技术(Software Component Technology)、领域工程(Domain Engineering)、软件体系结构(Softwarc Architecture)、软件再工程(Software Reengineering)、开放系统(Open System)、软件过程(Software Process)、CASE技术等以及各种非技术因素,且各种因素是互相联系、互相影响的,它们结合在一起,共同影响软件复用的实现。

1.2.1 软件构件技术

构件是指应用系统中可以明确辨识的构成成分,而可复用构件是指具有相对独立的功能和可复用价值的构件。

可复用构件应具备以下属性:①有用性是必须提供有用的功能;②可用性是必须易于理解和使用;③质量是自身及其变形必须能正确工作;④适应性是应该易于通过参数化等方式在不同语境中进行配置;⑤可移植

性是能够在不同的硬件平台和软件环境中工作。

随着对软件复用理解的深入，构件的概念已不再局限于源代码构件，而是延伸到需求、系统和软件的需求规约、系统和软件的体系结构、文档、测试计划、测试案例和数据以及其他对开发活动有用的信息。这些信息都可以称为可复用软件构件。

软件构件技术是支持软件复用的核心技术，是近几年来迅速发展并受到高度重视的一个学科分支。其主要研究内容包括：

(1) 构件获取。有目的的构件生产和从已有系统中挖掘提取构件。

(2) 构件模型。研究构件的本质特征及构件间的关系。

(3) 构件描述语言。以构件模型为基础，解决构件的精确描述、理解及组装问题。

(4) 构件分类与检索。研究构件分类策略、组织模式及检索策略，建立构件库系统，支持构件的有效管理。

(5) 构件复合组装。在构件模型的基础上研究构件组装机制，包括源代码级的组装和基于构件对象互操作性的运行级组装。

(6) 标准化。构件模型的标准化和构件库系统的标准化。

1.2.2 领域工程

领域工程是为一组相似或相近系统的应用工程建立基本能力和必备基础的过程，它覆盖了建立可复用软件构件的所有活动。领域是指一组具有相似或相近软件需求的应用系统所覆盖的功能区域。领域工程包括3个主要的阶段：

(1) 领域分析。主要目标是获得领域模型。该模型描述领域中系统之间共同的需求。本阶段主要活动包括确定领域边界，识别信息源，分析领域中系统的需求，确定哪些需求是被领域中的系统广泛共享的，哪些是可变的，从而建立领域模型。

(2) 领域设计。目标是获得领域体系结构(Domain-Specific Software Architecture，缩写为 DSSA)。DSSA 描述在领域模型中表示的需求的解决方案，它不是单个系统的表示，而是能够适应领域中多个系统需求的一个高层次的设计。建立了领域模型之后，就可以派生出满足这些领域需求的 DSSA。由于领域模型中的领域需求具有一定的变化性，DSSA 也要相应地具有变化性。

(3) 领域实现。主要行为是定义将需求翻译到由可复用构件创建的系统的机制。根据所采用的复用策略和领域的成熟与稳定程度，这种机

制可能是一组与领域模型和DSSA相联系的可复用构件,也可能是应用系统的生成器。

这些活动的产品(可复用的软件构件)包括:领域模型、领域体系结构、领域特定的语言、代码生成器和代码构件等。

1.2.3 软件再工程

软件复用中的一些问题与现有系统密切相关,如:现有软件系统如何适应当前技术的发展及需求的变化,采用更易于理解的、适应变化的、可复用的系统软件体系结构并提炼出可复用的软件构件;现存大量的遗留软件系统(Legacy Software)由于技术的发展,正逐渐退出使用,如何对这些系统进行挖掘、整理,得到有用的软件构件;已有的软件构件随着时间的流逝会逐渐变得不可使用,如何对它们进行维护,以延长生存期,充分利用这些可复用构件等等。软件再工程正是解决这些问题的主要技术手段。

软件再工程是一个工程过程,它将逆向工程、重构和正向工程组合起来,将现存系统重新构造为新的形式。再工程的基础是对系统的理解,包括对运行系统、源代码、设计、分析、文档等的全面理解。但在很多情况下,由于各类文档的丢失,只能对源代码进行理解,即程序理解。

1.2.4 开放系统技术

开放系统技术的基本原则是在系统的开发中使用接口标准,同时使用符合接口标准的技术。这些为系统开发中的设计决策,特别是对于系统的演化,提供了一个稳定的基础,同时,也为系统(子系统)间的互操作提供了保证。开放系统技术具有在保持(甚至是提高)系统效率的前提下降低开发成本、缩短开发周期的可能。对于稳定的接口标准的依赖,使得开放系统更容易适应技术的进步。当前,以解决异构环境中的互操作为目标的分布对象技术,是开放系统技术中新的主流技术。

开放系统技术为软件复用提供了良好的支持。特别是分布对象技术使得符合接口标准的构件可以方便地以"即插即用"的方式组装到系统中,实现黑盒复用。这样,在符合接口标准的前提下,构件就可以独立地进行开发,从而形成独立的构件制造业。

1.2.5 软件体系结构

软件体系结构是对系统整体结构设计的刻画,包括全局组织与控制