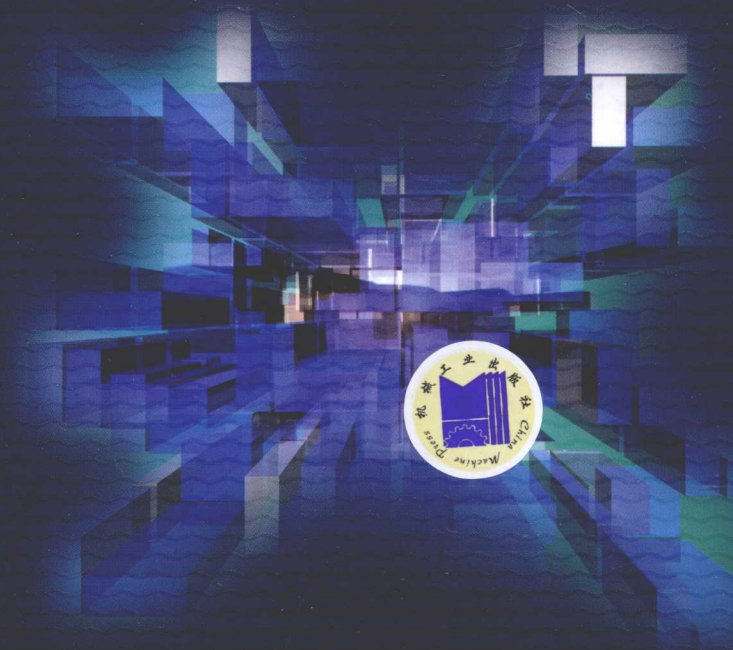
免费提供电子教案

高等院校规划教材
软件工程系列

软件构件与体系结构
——原理、方法与技术

王映辉　编著



机械工业出版社
CHINA MACHINE PRESS

高等院校规划教材·软件工程系列

软件构件与体系结构

——原理、方法与技术

王映辉　编著

机械工业出版社

本书主要包括两大部分内容：软件构件和软件体系结构，重在阐述它们的基本原理、方法和技术。

首先，本书结合软件复用，详细介绍了构件的基本概念、构件模型、面向构件、基于构件和领域工程。其次，在以构件为基础的软件体系结构基本模型的基础上，阐述了软件体系结构的基本概念、软件体系结构模型、模式系统、软件产品线、软件体系结构的设计和描述、软件体系结构编档、软件的质量属性、软件体系结构的评估，以及基于构件和软件体系结构的软件演化。最后结合软件框架给出了一个具体的应用开发实例。

本书可作为计算机专业或者软件工程专业高年级本科生的教材，也可作为计算机软件与理论专业或者计算机应用技术专业研究生的教材，还可作为软件架构师、开发人员和软件工程技术人员的参考用书。

图书在版编目(CIP)数据

软件构件与体系结构——原理、方法与技术/王映辉编著. —北京：机械工业出版社，2009.9
(高等院校规划教材·软件工程系列)
ISBN 978-7-111-27970-9

Ⅰ. 软… Ⅱ. 王… Ⅲ. 软件工程-高等学校-教材 Ⅳ. TP311.5

中国版本图书馆 CIP 数据核字(2009)第 135249 号

机械工业出版社(北京市百万庄大街 22 号 邮政编码 100037)
策划编辑：唐德凯
责任编辑：唐德凯 吴超莉
责任印制：李 妍
北京诚信伟业印刷有限公司印刷

2009 年 9 月第 1 版·第 1 次印刷
184mm×260mm·22 印张·544 千字
0001—3000 册
标准书号：ISBN 978-7-111-27970-9
定价：36.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
销售服务热线电话：(010)68326294 68993821
购书热线电话：(010)88379639 88379641 88379643
编辑热线电话：(010)88379753 88379739

出 版 说 明

计算机技术的发展极大地促进了现代科学技术的发展，明显地加快了社会发展的进程。因此，各国都非常重视计算机教育。

近年来，随着我国信息化建设的全面推进和高等教育的蓬勃发展，高等院校的计算机教育模式也在不断改革，计算机学科的课程体系和教学内容趋于更加科学和合理，计算机教材建设逐渐成熟。在“十五”期间，机械工业出版社组织出版了大量计算机教材，包括“21世纪高等院校计算机教材系列”、“21世纪重点大学规划教材”、“高等院校计算机科学与技术‘十五’规划教材”、“21世纪高等院校应用型规划教材”等，均取得了可喜成果，其中多个品种的教材被评为国家级、省部级的精品教材。

为了进一步满足计算机教育的需求，机械工业出版社策划开发了“高等院校规划教材”。这套教材是在总结我社以往计算机教材出版经验的基础上策划的，同时借鉴了其他出版社同类教材的优点，对我社已有的计算机教材资源进行整合，旨在大幅提高教材质量。我们邀请多所高校的计算机专家、教师及教务部门针对此次计算机教材建设进行了充分的研讨，达成了许多共识，并由此形成了“高等院校规划教材”的体系架构与编写原则，以保证本套教材与各高等院校的办学层次、学科设置和人才培养模式等相匹配，满足其计算机教学的需要。

本套教材包括计算机科学与技术、软件工程、网络工程、信息管理与信息系统、计算机应用技术以及计算机基础教育等系列。其中，计算机科学与技术系列、软件工程系列、网络工程系列和信息管理与信息系统系列是针对高校相应专业方向的课程设置而组织编写的，体系完整，讲解透彻；计算机应用技术系列是针对计算机应用类课程而组织编写的，着重培养学生利用计算机技术解决实际问题的能力；计算机基础教育系列是为大学公共基础课层面的计算机基础教学而设计的，采用通俗易懂的方法讲解计算机的基础理论、常用技术及应用。

本套教材的内容源自教学与科研一线的骨干教师与资深专家的实践经验和研究成果，融合了先进的教学理念，涵盖了计算机领域的核心理论和最新的应用技术，真正在教材体系、内容和方法上做到了创新。同时本套教材根据实际需要配有电子教案、实验指导或多媒体光盘等教学资源，实现了教材的“立体化”建设。本套教材将随着计算机技术的进步和计算机应用领域的扩展而及时改版，并及时吸纳新兴课程和特色课程的教材。我们将努力把这套教材打造成国家级或省部级精品教材，为高等院校的计算机教育提供更好的服务。

对于本套教材的组织出版工作，希望计算机教育界的专家和老师能提出宝贵的意见和建议。衷心感谢计算机教育工作者和广大读者的支持与帮助！

机械工业出版社

前　言

在本书出版之际，首先感谢本书所有参考文献的作者，可以说，没有他们的思想就不会有本书的成功撰写！

本书是笔者从事软件构件技术、软件体系结构和现代软件工程本科生、研究生课程教学十多年经验的积累和总结。软件构件技术和体系结构是密不可分的两项关键软件技术，它们在不同的层面上为软件的成功复用提供了支撑。软件构件是目前软件复用的基本单元，相对而言，软件体系结构为软件的大粒度复用和软件的整体骨架复用提供了机会，使软件复用从单元复用上升到了产品线的复用，进一步提升了软件复用的能力。

计算机科学与技术专业和软件工程专业已经在各个高等学校开设，而软件构件与软件体系结构是计算机专业特别是软件工程专业基础课程的主要内容之一。此外，软件构件与软件体系结构的相关原理、方法和技术也是计算机软件与理论、计算机应用技术专业的硕士研究生和博士研究生学习和研究的关键内容之一，同时也是软件技术人员所关注的主要内容之一。

本书是目前国内外出版的相关书籍和研究成果的总结，具有系统性和针对性强的特点。此外，本书将软件体系结构模型描述为构件基础上的一个拓扑结构，从而在理论、方法和技术方面，将目前解决软件危机最为有效的两种技术——软件构件技术和软件体系结构技术进行了统一，为教学、科研和工程开发等提供了全面、有效和系统的支撑。

本书中标有（*）的部分为选学内容，由读者根据情况进行选读和酌情处理。

本书的内容在撰写过程中经过了反复的修改，力求以教材的形式呈现给读者，但由于水平所限，书中难免有不足之处，敬请广大读者批评指正。

编　者

目　录

第1章 软件复用

本章重点介绍了软件复用的发展历史和定义，引出构件、软件体系结构和领域工程的概念，并阐述了它们之间的关系。

1.1 软件复用的概念

自从1968年D. McIlroy第一次提出共享构件（Shared Component）的概念以来，软件复用一直被认为是有明显回报的软件开发思想，它通过已有的高质量的软件元素来构建软件系统，提高开发效率，节约开发成本。由于涉及的因素较为复杂，诸如技术、过程和组织等，所以如何获得高质量的软件复用一直是困扰人们的一个难题，而软件体系结构为推进该问题的解提供了较为理想的途径。

简单地说，软件复用是指在两次或多次不同的软件开发过程中重复使用相同的或相近的软件元素的过程。广义的理解，软件复用就是开发粒度合适的构件，然后重复使用这些构件，进而扩展“构件组成的体系”，并将其从单纯的代码范畴扩展到需求与分析模型、设计和测试等范畴。所以软件开发过程的所有阶段都是“复用”的主角。因此，软件元素可包括程序代码、测试用例、设计文档、设计过程、需求分析文档和领域知识等。可复用的软件元素越大，我们就说可复用的粒度越大。按照不同的抽象级别，软件复用可划分为如下几类：

（1）代码的复用

代码复用是软件复用中最为常见的一种形式，包括目标代码和源代码的复用。其中目标代码的复用级别最低，历史最久，大部分编程语言的运行支持环境都提供了连接（Link）、绑定（Binding）等功能来支持这种复用。源代码的复用级别略高于目标代码的复用，程序员在编程时把一些想复用的代码段复制到程序中，但这样做往往会产生一些新旧代码不匹配的错误。要大规模地实现源程序的复用，只有依靠含有大量可复用构件的构件库，如“对象链接与嵌入”（OLE）技术，既支持在源程序级上定义构件以构造新的系统，又使这些构件在目标代码级上仍然是一些独立的可复用构件，能够在运行时被灵活地重新组合为各种应用系统。

（2）设计的复用

设计结果比源程序的抽象级别更高，因此它的复用受实现环境的影响较少，从而使可复用构件被复用的机会更多，并且所需修改更少。这种复用有3种途径，第一种途径是从现有系统的设计结果中提取一些可复用的设计构件，并把这些构件应用于新系统的设计中；第二种途径是把一个现有系统的全部设计文档在新的软硬件平台上重新实现，也就是把一个设计运用于多个具体的实现；第三种途径是独立于任何具体的应用，有计划地开发一些可复用的设计构件。

（3）分析的复用

这是比设计结果更高级别的复用。可复用的分析构件是针对问题域的某些事物或某些问

题的抽象程度更高的方法，受设计技术及实现条件的影响更小，所以可复用的机会更大。这种复用也有3种途径，第一种途径是从现有系统的分析结果中提取可复用构件并用于新系统的分析；第二种途径是用一份完整的分析文档作为输入，产生针对不同软硬件平台和其他实现条件的多项设计；第三种途径是独立于具体应用，专门开发一些可复用的分析构件。

（4）测试信息的复用

测试信息的复用主要包括测试用例的复用和测试过程的复用。前者是把一个软件的测试用例应用于新的软件测试中，或者在软件作出修改时使用在新一轮的测试中。后者是在测试过程中通过软件工具自动记录测试的过程信息，包括测试员的每一个操作、输入参数、测试用例及运行环境等信息，并将这些过程信息应用于新的软件测试或新一轮的软件测试中。测试信息的复用级别不易同分析、设计、编程的复用级别进行准确地比较，因为被复用的不是同一事物的不同抽象层次，而是另一种信息，但从这些信息的形态来看，大体处于与程序代码相当的级别。

从软件的发展历史来看，在软件发展初期，所有人都必须从头开始编写程序。现在，软件系统的种类越来越多，规模越来越大，在已有的软件中，很多功能被重复写了成千上万次，这些重复的代码在当今软件的开发中可以不断被拿来使用。AT&T、爱立信、惠普、IBM、摩托罗拉、NEC和东芝等公司的经验表明，非正式的代码复用率为15% ~ 20%，结合其他系统复用，使得软件开发的成本大大降低，开发时间得到有效缩短。

日本的一些软件公司还建立了适合使用标准部件的工程组织，一直追求更正式的复用。20世纪80年代中期，日本软件工程的复用率已经接近50%。美国的惠普公司从1984年初开始之后的10年里，在仪表和打印机固件方面的复用率达到25% ~ 50%，其中有一条仪表生产线达到了83%。

由此可见，使用软件复用技术可以减少软件开发活动中大量的重复性劳动，提高软件生产效率，降低开发成本，缩短开发周期。同时，由于软件构件大都在实际运行环境中得到了多次校验，并经过了严格的质量认证，因此，复用这些构件有助于改善软件质量。此外，大量使用软件构件，还有助于提高软件的灵活性和标准化程度。而且，由于软件生产过程主要是正向过程，即大部分软件的生产过程是使软件产品从抽象级别较高的形态向抽象级别较低的形态演化，级别较高的复用容易带动级别较低的复用，因而复用的级别越高，可得到的回报也就越大，因此分析结果和设计结果在目前很受重视。用户可购买生产商的分析构件和设计构件，自己设计或编程，掌握系统的剪裁、扩充、维护和演化等活动。

1.2 软件复用的实现(*)

1.2.1 软件复用的基本问题

尽管有许多成功运用复用的例子，但要想取得实际成就却是很困难的。主要障碍来自4个方面：工程、过程、组织和资金。

（1）工程方面

工程方面主要体现在技术和方法上。

- 缺少能够清晰标识可复用模型要素的手段。诸如，需求、体系结构、分析、设计、测试和开发实现的模型要素。模型要素是可复用构件系统的基础。

- 缺少能够复用的构件。诸如，不能优化选择供复用的构件；缺少打包、形成文档、分类和标识构件的手段；不适当的构件库系统的设计和实现导致复用人员不能很好地访问构件库等。
- 可复用的构件缺乏灵活性。构件的不灵活导致构件很少或没有复用的机会；用来设计分层体系结构的方法不灵活也不成熟，使构件满足新的需求或新的体系结构的能力受到了限制。
- 缺乏执行复用过程的工具。能够集成到面向复用支持环境的新工具是不可缺少的。

（2）过程方面

在工程和技术层次上，软件开发的传统过程本身缺乏鼓励复用的机会。在今天的大多数开发过程中，开发人员应该自问："在以前已经完成的部分中，我们可以把哪些部分分离开，并使用可复用构件来替代?"。结构设计师在复用中的潜在角色没有被定义；类似地，复用工程师或可复用构件工程师的角色也没有被定义。部分复用的实现也仅仅依赖于软件设计人员和开发人员的经验。

（3）组织方面

在组织方面，只有很少的机构为系统地实现复用提供最佳实践基础。原因是：软件开发机构将注意力一次只集中到一个项目上，而复用要求扩大注意范围，管理层的注意力必须覆盖一个应用领域的一组项目。也就是说，领域工程师必须认识到这一组项目的共同特性，并标识这个领域中可复用的要素，并从这里开始规划复用。这样，在研发队伍中，持复用观点的人员和把注意力集中到一个项目上的人员之间就会产生冲突，其中一个更为深入的原因是人们对他人创建的构件缺乏信任，或者管理者缺乏组织实现复用方面的知识。

（4）资金方面

复用需要资金投入。诸如，领域工程构建足够强的构件和构件系统，需要资金投入，创建公司内的构件库需要资金投入，教育、培训以及访问厂商提供的构件也需要资金投入。当最初的领域没有很好地定义，或者必须合并不同机构中的相似领域时，在这种不稳定的领域中竞争需要资金。共享和不共享软件在法律和社会方面也可能需要付出。另外，单个开发人员仅凭个人能力对大规模复用起不了多大作用，必须建立完善的机制，在此基础上为开发人员提供组织和资金支持。

1.2.2 软件复用的关键因素

复用依赖于软件体系结构：从小型的、一两个程序开发人员就能完成的软件应用程序，到由若干团队完成的数十万行源代码的一个系统，再到由多个系统协同的系统系列，软件的复杂性在不断增加。构建整个系统系列中可复用的构件系统，并对构件增加可变性机制时，软件的复杂性又向前迈了一步。这种一直增长的复杂性是大规模可复用系统所固有的，我们不能回避这种复杂性的增长，因此，使用软件体系结构来规划和管理这种复杂性是最好的解决方法。

首先，软件体系结构要定义一种宏观蓝图，软件构件必须遵循这种设计蓝图。其次，体系结构必须定义构件之间的标准接口，以便构件工程师开发的构件被不同的小组或在不同地点开发的应用系统使用。因此，好的体系结构的最大优点是构件系统的下层设计和依赖构件系统的应用系统可以得到更好的理解。

建立良好的复用概念：软件复用对提高软件开发生产率、缩短产品上市时间及减少软件产品中的缺陷数量都有着极大的帮助。同时，软件复用也是一个充满误解的领域，因此有必要建立良好的复用概念。下面给出一些常见的概念及回答。

1）启动软件复用仅仅是为了引入适当的技术——错误，启用软件复用还需要其他很多东西，如管理、机构、体系结构、过程、投资和坚持等。

2）复用没有什么作用——错误，有些公司过去没有使复用发挥作用，但是现在很多公司正在使复用发挥作用。

3）复用可能会有用，但是太昂贵——正确，复用最初确实需要花费资金，但是以后会得到回报，其成本在多次复用中被分摊而逐步降低。

4）复用要求我们进行的变革风险太大——错误，变革对于近期来说是有“风险”，但不进行变革对于长期来说则风险“更”大。

5）只有机构达到 SEI 能力成熟度第 3 级后，才能启动系统化复用——错误，应该从现在开始，取得复用进展需要采取与提高 SEI 能力成熟度级别同样的步骤。

6）采用面向对象的程序设计语言会导致系统化的复用——错误，面向对象的程序设计语言本身并不会导致系统化的复用，实际上，任何程序语言本身并不能导致复用。

7）将复用局限在代码构件上——错误，编写代码通常只占整个开发成本的 10% ~20%。

8）开发人员要从数以万计的小构件中选取要复用的构件——错误，开发人员是人，人的大脑一次只能考虑 5 ~9 个对象。

9）被复用的代码构件速度太低——正确，这是构件设计上的缺陷。一般情况下，构件应该在比“一般”的程序更为严格的条件下开发，其性能、规模和质量有较高的标准。

10）建立可复用构件库会促使开发人员进行构件复用——错误，这里还有开发人员的相互信任问题。

1.3 软件复用与构件技术

一般认为，构件是指语义完整、语法正确和有可复用价值的软件单元，是软件复用过程中可明确辨识的成分；在结构上，它是语义描述、通信接口和实现代码的复合体。简单地说，构件是具有一定功能、能够独立工作或同其他构件装配起来协调工作的程序体，构件的使用同它的开发、生产无关。从抽象程度来看，面向对象技术已达到了类级复用（代码复用），它以类为封装单位，这样的复用粒度还太小，不足以解决异构互操作的效率和更高的复用。构件将抽象的程度提到一个更高的层次，可以看做是对一组类的组合和封装，并代表完成一个或多个功能的特定服务，进而为用户提供了接口。整个构件隐藏了具体的实现，只用接口对外提供服务。

近年来，构件技术发展迅速，已形成 3 个主要流派，分别是 IBM 的 CORBA-CCM，Sun 的 Java 平台和 Microsoft 的 COM + 。如果把软件系统看成是构件的集合，那么从外部形态看，构件可分为 5 类：

（1）独立而成熟的构件

这种构件得到了实际运行环境的多次校验。该类构件隐藏了所有接口，用户只需按规定好的方式使用即可。例如，数据库管理系统和操作系统等。