软件复用与软件构件技术丛书
Software Reuse and Software Component Technology

Reuse Oriented Requirements Modeling

面向复用的需求建模

杨芙清 梅 宏 主编
赵海燕 张 伟 麻志毅 编著

清华大学出版社

软件复用与软件构件技术丛书
Software Reuse and Software Component Technology

Reuse Oriented Requirements Modeling

面向复用的需求建模

杨芙清 梅 宏 主编
赵海燕 张 伟 麻志毅 编著

清华大学出版社
北京

内 容 简 介

本书以复用技术为主线，系统地介绍面向复用的软件开发过程中最关键的一步——需求建模所涉及的基本思想和方法，包括传统的需求工程、领域工程方法、面向特征的领域建模方法以及基于 UML 的需求建模。同时，结合金融信贷、奥运信息管理系统、文档编辑器、网上商店等领域的实际案例，应用面向复用的需求建模理论和方法进行了深入的建模实践。本书内容兼顾理论与实践两方面，可使读者在获得面向复用的需求建模理论知识的同时，学会如何将理论知识应用于实践。

本书适用于计算机软件及相关专业的本科生或研究生，也适合高级计算机软件开发人员使用。

图书在版编目(CIP)数据

面向复用的需求建模/赵海燕，张伟，麻志毅编著. —北京：清华大学出版社，2008.10
(软件复用与软件构件技术丛书/杨芙清，梅宏主编)
ISBN 978-7-302-17644-2

Ⅰ. 面…　Ⅱ. ①赵… ②张… ③麻…　Ⅲ. 软件工程　Ⅳ. TP311.5

中国版本图书馆 CIP 数据核字(2008)第 065322 号

责任编辑：丁　岭　赵晓宁
责任校对：李建庄
责任印制：何　芊
出版发行：清华大学出版社　　**地　　址**：北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**邮　　编**：100084
社　总　机：010-62770175　　**邮　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质　量　反　馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：北京市清华园胶印厂
装 订 者：北京市密云县京文制本装订厂
经　　销：全国新华书店
开　　本：185×230　**印　张**：17.5　**字　数**：359 千字
版　　次：2008 年 10 月第 1 版　　**印　　次**：2008 年 10 月第1 次印刷
印　　数：1～3000
定　　价：29.00 元

丛书序言

软件复用是在软件开发中避免重复劳动的解决方案。通过软件复用，可以提高软件开发的效率和产品的质量。近20多年来，面向对象技术、软件构件技术出现并逐步成为主流技术，为软件复用提供了基本的技术支持。软件复用研究及实践成为热点，被视为解决软件危机、提高软件生产效率和质量的现实可行的途径。

软件复用通常可分为产品复用和过程复用两条途径。基于构件的复用是产品复用的主要形式，也是当前复用研究及实践的主流。软件构件技术被视为实现成功复用的关键因素之一。

一、为什么要复用

通常情况下，应用软件系统的开发过程包含以下几个阶段：需求分析、设计、编码、测试、维护等。当每个应用系统的每个部分都是从头开发时，在系统开发过程中就可能存在大量的重复劳动，如：用户需求获取的重复、需求分析和设计的重复、编码的重复、测试的重复和文档工作的重复等。

探讨应用系统的本质，可以发现其中通常包含三类成分：

① 通用基本构件，是特定于计算机系统的构成成分，如基本的数据结构、用户界面元素等，它们可以存在于各种应用系统中。

② 领域共性构件，是应用系统所属领域的共性构成成分，它们存在于该领域的各个应用系统中。

③ 应用专用构件，是每个应用系统的特有构成成分。应用系统开发中的重复劳动主要在于前两类构成成分的重复开发。基于此分类，可以看到，通过凝结共性，通用基本构件和领域共性构件将具有良好的可复用性，由此可以将特定软件的开发聚焦在专用构件的开发及各相关构件的集成组装工作中。

软件复用是在软件开发中避免重复劳动的解决方案，其出发点是应用系统的开发不再采用一切“从零开始”的模式，而是以已有的工作为基础，充分利用过去应用系统开发中积累的知识和经验，如：需求分析结果、设计方案、源代码、测试计划及测试案例等，从而将开发的重点集中于应用的特有构成成分。

通过软件复用，在应用系统开发中可以充分地利用已有的开发成果，消除了包括分析、设计、编码、测试等在内的许多重复劳动，从而提高了软件开发的生产率，同时，通过复用高质量的已有开发成果，避免了重新开发可能引入的错误，从而提高了软件的质量。

二、复用的基本概念

软件复用是指重复使用“为了复用目的而设计的软件”的过程。相应地，可复用软件是指为了复用目的而设计的软件。与软件复用的概念相关，重复使用“并非为了复用目的而设计的软件”的过程，可以称为“软件挽救”。在一个应用系统的不同版本间重复使用代码的过程，可以称为“代码沿用”。这两类行为都不属于严格意义上的软件复用。

以下的类比有助于进一步说明软件复用的概念。在软件演化的过程中，重复使用的行为可能发生在三个维上：

(1) 时间维。使用以前的软件版本作为新版本的基础，加入新功能，适应新需求，即软件维护。

(2) 平台维。以某平台上的软件为基础，修改其和运行平台相关的部分，使其运行于新平台，即软件移植。

(3) 应用维。将某软件(或其中构件)用于其他应用系统中，新系统具有不同功能和用途，即真正的软件复用。

这三种行为中都重复使用了现有的软件，但是，真正的复用是为了支持软件在应用维的演化，使用“为复用而开发的软件(构件)”来更快、更好地开发新的应用系统。

复用概念的第一次引入是在1968年NATO软件工程会议上，McIlroy的论文“大量生产的软件构件”中。在此以前，子程序的概念也体现了复用的思想。但其目的是为了节省当时昂贵的机器内存资源，并不是为了节省开发软件所需的人力资源。然而子程序的概念可以用于节省人力资源的目的，从而出现了通用子程序库，供程序员在编程时使用。例如，数学程序库就是一个非常成功的子程序复用的例子。

在其后的发展过程中，有许多复用技术的研究成果和成功的复用实践活动。但是，复用技术在整体上对软件产业的影响却并不尽如人意。这是由于技术方面和非技术方面的种种因素造成的，其中技术上的不成熟是一个主要原因。近20多年来，面向对象技术和软件构件技术出现并逐步成为主流技术，为软件复用提供了基本的技术支持。软件复用研究成为热点，被视为解决软件危机、提高软件生产效率和质量的现实可行的途径。

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并充分证明了这种模式的可行性和正确性。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（Black-box）复用和白盒（White-box）复用。黑盒复用指对已有构件不需作任何修改，直接进行复用。这是理想的复用方式。白盒复用指已有构件并不能完全符合用户需求，需要根据用户需求进行适应性修改后才可使用。而在大多数应用的组装过程中，构件的适应性修改是必需的。

三、实现软件复用的关键因素

软件复用有三个基本问题，一是必须有可以复用的对象，二是要被复用

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软件复用。

与以上几个方面的问题相联系，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)、领域工程(Domain Engineering)、软件体系结构(或构架)(Software Architecture)、软件再工程(Software Reengineering)、开放系统技术(Open System)、软件过程(Software Process)、CASE 技术等以及各种非技术因素。这些技术因素和非技术因素是互相联系的。如图 1 所示，它们结合在一起，共同影响软件复用的实现。

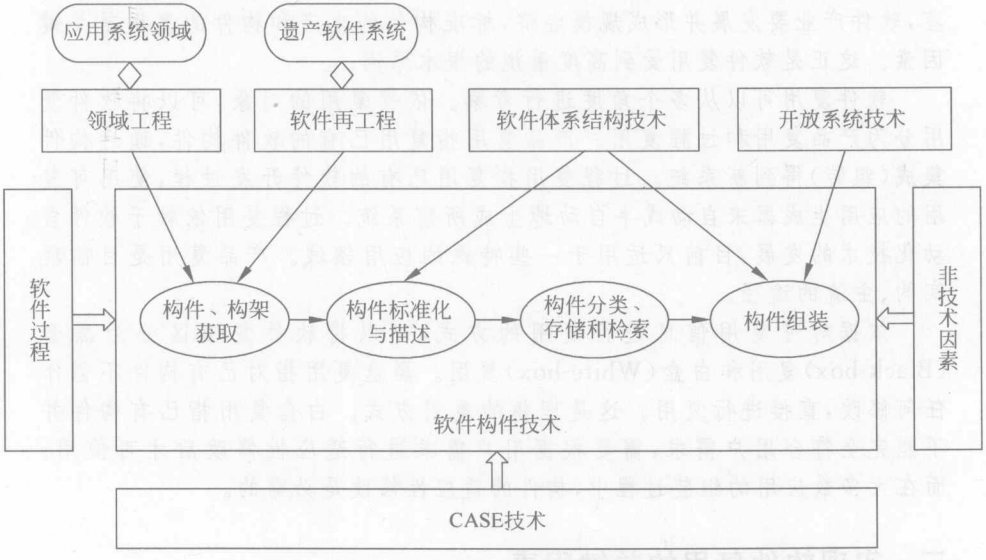

图 1　实现软件复用的关键因素

软件构件技术

构件(Component)是指应用系统中可以明确辨识的构成成分。在计算机百科全书中,软件构件被定义为软件系统中具有相对独立功能、可以明确辨识、接口由契约指定、和语境有明显依赖关系、可独立部署、且多由第三方提供的可组装软件实体。而可复用构件(Reusable Component)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(Usefulness)。构件必须提供有用的功能。

② 可用性(Usability)。构件必须易于理解和使用。

③ 质量(Quality)。构件及其变形必须能正确工作。

④ 适应性(Adaptability)。构件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(Portability)。构件应能在不同的硬件运行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。

随着 Internet 上软件技术的发展,软件服务化成为新的技术发展趋势,也出现了以 Web Service、EJB、CORBA、COM+等为代表的服务类型构件。此时,构件在使用方式上发生了变化,即从原先直接使用构件实体(如源码)转向通过网络调用构件所提供的服务进行使用,而这些服务又可以通过集成组装其他服务形成粒度更大的、新的服务提供给用户使用。我们看到,服务与传统构件既有相同性又有差异性:首先,从应用系统的开发角度来看,目前这两个概念均作为系统的基本构成单元,但是基于服务或构件进行系统的构建方式不同。基于构件的软件开发,需将构件以源码或目标代码的形式集成到最终系统中,此时构件的源码或目标代码是作为系统的组成成分的。基于服务的软件开发,服务只向服务的请求者提供计算结果,最终系

统集成的仅是服务访问地址，服务本身的源代码或目标代码不作为系统的组成成分。其次，服务是由一个或一组构件实现的，而一个或一组构件又可以被发布为不同的服务，服务是构件运行时对外提供功能的表象，用户使用服务完成特定任务，而构件是实现服务的载体。总之，服务在使用方式、构件管理方式以及集成组装方式上有别于传统构件，但是由于服务本身具有相对独立的功能和由契约指定的接口，并且可以被明确辨识、被集成组装到新的应用中，因此仍然可将服务看作为一类新型构件。服务概念的引入，扩展了构件的内涵和外延，有力地支持了基于构件的软件开发。

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。利用领域工程等技术，进行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。构件模型定义了构件的本质属性、规定了构件接口的结构以及构件与软件体系结构、构件与构件之间的交互机制，构件模型通常还提供创建和实现构件的指导原则。

(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) 构件组装。在构件模型的基础上研究如何通过构件的接口使软件构件相互连接以构造应用系统。

(6) 标准化。包括构件模型的标准化和构件库系统的标准化。

经过软件复用的研究和实践方面的努力，在软件构件技术方面已经取得一定的成果。譬如 COM、EJB、Web Services 等已经成为成熟的软件构件模型。一些政府、军方或企业发布了自己的构件库系统，如 REBOOT 系统、ComponentSource 等；IBM、Microsoft 等公司对外提供了服务注册中心——UDDI 站点。在某些领域，如科学计算，已有商用的构件存在。同时，存在大量独立于应用领域的计算机特定的软件构件，如程序设计语言的类

库、函数库等。但对大多数特定领域来说，可复用构件仍十分短缺，从而形成了一个巨大的应用软件构件市场。

软件体系结构技术

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结构，构件间通讯、同步和数据访问的协议，设计元素间的功能分配，物理分布，设计元素集成，伸缩性和性能，设计选择等。

研究软件体系结构对于软件工程具有非常重要的意义：通过对软件构件的研究，有利于发现不同系统在较高级别上的共同特征；获得正确的体系结构对于进行正确的系统设计非常关键；对各种体系结构的深入了解，使得软件工程师可以根据一些原则在不同的软件体系结构之间做出选择；从体系结构的层次上表示系统，有利于系统较高级别性质的描述和分析。特别重要的是，在基于复用的软件开发中，为复用而开发的软件体系结构可以作为一种大粒度的、抽象级别较高的软件构件进行复用，而且软件体系结构还为构件的组装提供了基础和上下文，对于成功的复用具有非常重要的意义。

软件体系结构研究如何快速、可靠地从可复用构件构造系统的方式，着重于软件系统自身的整体结构和构件间的互联。其中主要包括：软件体系结构原理和风格，软件体系结构的描述和规约，特定领域软件体系结构，基于软件体系结构的构件组装机制等。

当前，面向服务的体系结构(Service-Oriented Architecture，SOA)成为研究热点。SOA 提供了一种面向服务，大粒度、松耦合及动态绑定的分布式应用系统的构建方法。区别于传统的软件工程方法，SOA 可以看作是从面向对象技术和软件构件技术演化而来的一种新的体系结构风格。从当前的学术研究和业界实践来看，许多针对 SOA 的研究和技术来源正是原来的构件技术研究成果。如分析一个系统中哪些部分可以封装成为一个服务，是依赖于领域知识和相关的 SOA 技术的。并如前所述，服务和传统构件具有一定的共性，因此在构件技术领域所进行的构件模型、组装技术等研究都可以成为实现 SOA 的支持技术，并具有良好的统一性。例如实现服务的软件

实体大多以CORBA、EJB、COM等分布式构件技术实现，仅是在接口进行了标准协议的封装。服务和SOA的引入，扩展了构件的内涵和外延，支持了异构应用的集成与互操作，促进了软件复用在更大范围得以实施。

领域工程

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。软件复用的研究和实践表明，在特定领域的软件复用相对容易取得成功。这是由于特定领域本身的相对内聚性和稳定性所决定的。内聚性保证了领域有足够的共性；而稳定性保证了领域工程的投资可以获得足够的回报。

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。领域工程包括三个主要的阶段：

(1) 领域分析。这个阶段的主要目标是获得领域模型(Domain Model)。领域模型描述领域中系统之间的共同的需求。这个阶段的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。这个阶段的目标是获得领域体系结构(Domain-Specific Software Architecture，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的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被建模的领域需求的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。这个阶段的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和稳定程度，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。

软件再工程

软件复用中的一些问题是与现有系统密切相关的，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？现存大量的遗产软件系统(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，以延长其生命期，充分利用这些可复用构件？软件再工程(Software Reengineering)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。再工程的基础是系统理解，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。

再工程的主要行为如图 2 所示。

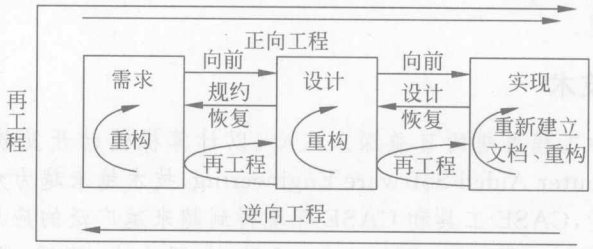

图 2　软件再工程

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标准接口，同时使用符合接口标准的实现。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,使得开发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作为目标的分布对象技术是开放系统技术的新的主流技术。它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生产业。

软件过程

软件过程又称为软件生存周期过程,是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。当前,软件过程研究以及企业的软件过程改善已成为软件工程界的热点,并已出现了实用的过程模型标准,如CMM、ISO9001/TickIT 等。

然而,基于构件复用的软件开发过程和传统的一切从头开始的软件开发过程有着实质性的不同,探讨适应于软件复用的软件过程自然就成为一个重要的问题。

CASE 技术

随着软件工程思想的日益深入人心,以计算机辅助开发软件为目标的CASE (Computer Aided Software Engineering)技术越来越为众多的软件开发人员所接受,CASE 工具和 CASE 环境得到越来越广泛的应用。CASE 技术对软件工程的很多方面,例如分析、设计、代码生成、测试、版本控制和配置管理、再工程、软件过程、项目管理等,都可以提供有力的自动或半自动支持。CASE 技术的应用,可以帮助软件开发人员控制软件开发中的复杂性,有利于提高软件开发的效率和质量。

软件复用同样需要 CASE 技术的支持。CASE 技术中与软件复用相关的主要研究内容包括:在面向复用的软件开发中,可复用构件的抽取、描述、分类和存储;在基于复用的软件开发中,可复用构件的检索、提取和组装;可

复用构件的度量等。

非技术因素

除了上述的技术因素以外，软件复用还涉及众多的非技术因素，如：机构组织如何适应复用的需求；管理方法如何适应复用的需求；开发人员知识的更新；创造性和工程化的关系；开发人员的心理障碍；知识产权问题；保守商业秘密的问题；复用前期投入的经济考虑；标准化问题等。

四、关于本丛书

软件复用经过几十年的发展，已有许多成功的研究和实践成果，正逐步成为实现软件工程化、工业化生产的首选途径。多年来，北京大学软件工程学科一直致力于软件复用及相关技术的研究和应用推广工作，并取得了一定成果。

软件复用技术将促进软件产业的变革，使软件产业真正走上工程化、工业化的发展轨道。软件复用将造成软件产业的合理分工，专业化的构件生产将成为独立的产业而存在，软件系统的开发将由软件系统集成商通过购买商用构件，集成组装而成。软件复用所带来的产业变革将会带来更多的商业契机，形成新的增长点。这对我国软件产业的发展是一个良好的机遇，要抓住机遇，需要我国产业界和学术界的共同努力。

为了促进我国学术界和产业界对软件复用技术的研究和应用，我们特编撰了《软件复用与软件构件技术丛书》，希望能对同行有一定借鉴和帮助。丛书介绍了国内外在软件复用技术研究及实践方面的重要成果，涉及软件复用与软件构件技术的主要方面，包括三个部分：面向复用的需求建模、构件化软件设计和实现、面向复用的软件资产和过程管理。

实践表明，在软件的生命周期中，复用发生的时间越早，其带来的收益也就越大。因为一般而言可复用的需求将导致可复用的设计，进一步导致可复用的代码，相应的验收测试计划和过程也能被复用。丛书的《面向复用的需求建模》以面向复用的需求建模方法和技术为主题展开，介绍了需求工

程、领域工程方法、面向特征的领域建模方法、基于 UML 的需求建模以及面向复用的需求建模实践活动。

构件化软件设计的核心思想是如何基于可复用构件设计出满足功能和非功能需求的软件体系结构；构件化软件实现则是按照软件体系结构、利用各种语言或中间件机制、将可复用构件组装起来。特别地，构件化软件不仅复用了构件，还复用了构件运行支撑平台的若干能力，因此，构件化软件的实现往往延伸到了传统软件开发阶段之后的部署和运营阶段。丛书的《构件化软件设计与实现》以软件体系结构为中心，以中间件为基础平台，介绍如何实现系统化和自动化的构件化软件设计、实现、部署、运行、维护与演化。

要使得软件组织能够长期稳定地从复用中获益，必须建立一套与复用配套的管理方法和基础设施。这里的管理主要包括两个方面：对软件资产的管理和对软件过程的管理。丛书的《面向复用的软件资产与过程管理》以软件复用中的这两类管理为题展开，第一部分介绍软件资产管理，涉及软件资产的描述、分类、存储、检索与维护的技术、方法和工具；第二部分介绍软件复用过程管理，涉及建立复用过程的相关指导原则和依照这些原则建立的一套相关过程的实例，包括：角色定义、活动描述和制品模板等。

在本丛书编撰过程中，北京大学软件工程学科众多教师、学生和相关研究人员付出了大量努力。特别感谢赖志迢(Robert. Lai)先生，赖先生对本丛书进行了认真的审稿，并对编撰工作提出了许多建设性的建议。丛书包含了作者们多年来的研究成果和实践经验，希望这套丛书能够为同行提供有用的参考，为培养一批有志于推进中国软件工业化生产的技术人才和管理人才做出贡献！由于时间关系，丛书中难免存在错误，希望得到来自广大读者的批评和意见。

本丛书受到国家“863”计划(No. 2007AA010301，No. 2006AA01Z189)、国家“973”计划(No. 2002CB312003)和国家科技支撑计划(No. 2006BAH02A02)等项目的资助。在北京大学多年的软件复用与软件构件技术研发和应用推广过程中，得到了国家科技部、发改委、信产部等领导部门的指导和支持，以及北京市、上海市、广东省、云南省、青岛市等地方科技主管部门的大力支持。在此对于所有给予我们支持和指导的领导部门、科研院所和企业一并表示感谢。

前 言

如何提高软件产品的质量和生产效率是软件产业发展过程中始终面临的一个重要问题。而软件项目要取得成功，最重要的莫过于了解要开发的软件需要解决哪些问题，此即软件需求所要解决的问题。它的基本任务是准确地定义未来系统的目标，确定为了满足用户的需要系统必须做什么。在此基础上通过对问题及其环境的理解与分析，为问题涉及的信息、功能及系统行为建立模型，将用户需求精确化、完全化，最终形成需求规约，为后续的软件设计、实现、测试直至维护提供基础。

同时，软件复用作为避免重复劳动、提高软件质量和生产效率的解决方案，其出发点是软件产品的开发不再采用“从零开始”的模式，而是充分利用过去软件开发中积累的资产，诸如源代码、设计方案、需求规约以及测试用例等，实现对软件开发过程中可复用成分最大程度的复用。实践表明在软件的生命周期中，复用发生的时间越早，其带来的收益也就越大。因为一般而言，可复用的需求将导致可复用的设计，而可复用的设计则可进一步导致可复用的代码。当需求被复用时，相应的验收测试，验收测试计划和过程也能被复用。软件复用的研究和实践表明，特定领域的软件复用活动相对容易取得成功。这是由特定领域本身的相对内聚性和稳定性所决定的。内聚性保证了领域有足够的共性；而稳定性保证了生产可复用资产的投资可以获得足够的回报。

作为软件复用的核心技术，领域工程的主要目的是实现对特定领域中可复用成分的分析、生产和管理；软件产品线的基本思想是在领域工程的基本上，利用这些可复用资产以及相关的组装和定制技术，开发出新的软件产品。本书主要关注于生产可复用软件资产的领域工程技术上，尤其集中在如下三个方面：建立领域需求模型的领域分析技术；对领域需求模型的复用技术；基于领域需求模型的高层软件体系结构设计方法。本系列丛书的《构件化软件的设计与实现》将涉及领域设计阶段和实现阶段的技术；本系列丛书的《软件复用过程与资源管理》将介绍面向复用的开发和基于复用的开发中的过程管理，以及所生产资源的管理技术。

本书将以面向复用的需求建模方法和技术为题展开，全书共分5个部分。

第1部分“领域工程概述”包括第1章和第2章，主题是面向复用的需求的基本概念。第1章主要介绍传统软件开发过程中与需求相关的一系列活动，包括需求的捕获、分析、规约、确认和管理等；第2章则对系统化地生产可复用资产的领域工程技术的起源和背景、相关概念、关键问题等方面进行了概略的介绍。

第2部分“领域工程方法”由第3～第6章组成，分别介绍4种具有代表性的领域工

程方法：面向特征领域分析方法(FODA)；面向特征的复用方法(FORM)；领域特定的软件体系结构方法(DSSA)；青鸟领域工程方法。

第3部分“面向特征的领域建模方法(FODM)”主要介绍北京大学软件所提出的一种领域工程方法。此部分由3章组成，其中，第7章介绍FODM面向特征的领域分析方法；第8章介绍FODM以特征为驱动的体系结构设计方法；第9章介绍一个支持FODM领域分析方法的软件工具。

第4部分“基于UML构件规约的需求建模”包括第10章～第12章，其中第10章讲述UML2.0支持的构件模型及规约；第11章描述如何基于UML构件规约进行需求建模；第12章介绍一种面向构件的建模工具。

第5部分“面向复用的需求建模实践”主要以若干实例为主线来演示第4部分所描述的FODM方法。其中第13章和第14章分别针对文档编辑器和网上商店领域进行需求建模实践，并在此基础上基于特征进行体系结构的设计；第15章～第17章则依次针对奥林匹克运动会系统、银行信贷系统以及中间件领域进行相应的需求建模，对应的体系结构设计则在本丛书的《构件化软件的设计与实现》中予以阐述。

本书在编写过程中参考了大量的文献。我们也尽可能地将这些文献列于书后。但对于因疏漏而未能列出的参考文献，在此表达深刻的歉意。同时也对所有文献的作者们表示诚挚的感谢。由于时间仓促，加之笔者水平所限，书中难免有不妥或错误之处，敬请读者不吝赐教。

另外，本书受到国家863计划(No. 2006AA01Z156)和国家自然科学基金(No. 60528006，60703065)等项目的资助，在此一并表示感谢。

编著

2008年8月