Prentice Hall

计算机专业教材精选

Object-Oriented Software Engineering

Conquering Complex and Changing Systems

面向对象的软件工程

——构建复杂且多变的系统

[美] B.Bruegge A.H.Dutoit 著

吴丹 唐忆 申震杰 译

张伟 审校

Pearson Education

清华大学出版社

Prentice Hall 计算机专业教材精选

面向对象的软件工程

——构建复杂且多变的系统

Object-Oriented Software Engineering

Conquering Complex and Changing Systems

[美] B. Bruegge A. H. Dutoit 著
吴丹 唐忆 申震杰 译
张 伟 审校

清 华 大 学 出 版 社

(京)新登字 158 号
北京市版权局著作权合同登记号　图字 01-2001-5318 号
EISBN 0-13-489725-0

内 容 提 要

本书融入了作者 10 多年来构造系统的经验和教授软件工程课程的教学内容。主要讨论了在复杂而多变的系统设计中如何使用软件工程的面向对象技术。全书共有 12 章，每章都是先阐述实例来介绍相关问题，再简要描述与主题相关的技术，然后通过一些简单的例子来解释基本概念并用实际系统中的例子来讲解主要技术，最后讲述管理中的问题，讨论典型的解决方案并给出相关习题。这种编排方式有助于读者更好地学习和理解。

本书内容详实，示例丰富，实用性强，适用于有一定编程基础的高年级本科生、研究生及相关工程技术人员和管理人员。

图书在版编目（CIP）数据

面向对象的软件工程/（美）布鲁志著；吴丹，唐忆等译.
—北京：清华大学出版社，2002
ISBN 7-302-05938-1

Ⅰ. 面…　Ⅱ. ①布…②吴…③唐…　Ⅲ. 软件工程　Ⅳ. TP311.5

中国版本图书馆 CIP 数据核字（2002）第 074928 号

书　名：面向对象的软件工程——构建复杂且多变的系统
出版者：清华大学出版社（北京清华大学学研大厦，邮编 100084）
　　http://www.tup.tsinghua.edu.cn
印刷者：北京市耀华印刷有限公司
发行者：新华书店总店北京发行所
开　本：异 16　　印张：30.125　　字数：658 千字
版　次：2002 年 10 月第 1 版　　2002 年 10 月第 1 次印刷
印　数：0001～5000
书　号：ISBN 7-302-05938-1/TP・3529
定　价：45.00 元

序　言

高达 8611m 的 K2 山峰耸立于西喜玛拉雅的喀拉昆仑山脉。它是世界第二高峰，并且被认为是高度在 8000m 以上最难攀登的山峰。对 K2 山峰的探险通常在夏季要持续几个月，夏天的天气是最适宜的。即使在夏季，暴风雪也是很频繁的。一次探险需要数千磅的装备，包括攀登齿链，精良的防护装备、帐篷、食物、通信工具，还有支付给数以百计的搬运工的薪水和鞋子。策划一个这样的探险需要花费攀登者很多的时间，而且还需要数十个后备者协助。一旦开始攀登，许多难以预料的突发事件，例如雪崩、工人罢工或是设备故障，将使得攀登者不得不改变方案，或者找出新的解决办法，或者放弃攀登。目前，对 K2 峰的探险成功率小于 40%。

美国国家空间系统（NAS）监管和控制全美的空中交通。NAS 拥有 18 300 多个航空站点，21 个空中交通控制中心和 460 多个控制塔。包括雷达、信息交换机、无线电收发设备，计算机系统和显示器在内，NAS 拥有总计超过 34 000 件设备。现在的基础设施老化很快。例如，支持 21 个空中交通控制中心的计算机是 20 世纪 80 年代早期的 IBM3083 大型机。1996 年，美国政府启动一个项目来更新 NAS 基础设施，使之现代化，包括改善卫星导航系统、数字控制器或飞行员通信，并希望在控制空中路线、安排飞行器着陆的秩序，以及当飞机到达或离开跑道时的地面交通控制方面取得更高的自动化。但是，让这样复杂的基础设施现代化，只能循序渐进。结果，当引进具有新功能的部分时，仍需要支持旧的部分。例如，在过渡时期，一个控制器不得不同时具备模拟和数字声音通道与飞行员通信。最后，NAS 的现代化与全球空中交通的巨大增长一致，而后者预计在未来 10~15 年后会成倍增加。被称为先进自动化系统（AAS）的 NAS 最初的现代化尝试，由于与软件有关的问题，在延误最初期望的截止期限几年并超出预算数十亿美元后于 1994 年暂停。

上面的两个例子都是讨论复杂系统，此处外部条件会引发意想不到的变动。复杂性使得问题超出任何个人可以控制的范畴。变动迫使参与者放弃众所周知的方法而创造新的方案。在上面两个例子中，数名参与者需要合作并开发新的技术来对付这些挑战。如果不这样做，将导致我们不能到达目标。

本书就是关于如何设计复杂而不断变化的软件系统的。

主题

应用域（登山计划、空中交通控制、金融系统、文字处理）通常包括许多软件开发人员不熟知的概念。而求解域（用户界面工具包、无线通信、中间件、数据库管理系统、交易处理系统、计算机等）通常是不成熟的，且会给开发人员带来很多具有挑战性的实现技

术。结果，系统和开发项目都变得复杂，并且涉及到许多不同的组件、工具、方法和人员。

当开发人员从使用者那里增进了对应用域的了解后，他们更新系统的需求。当开发人员对正在出现的技术或现行技术的缺陷了解得更多后，他们会调整系统设计和实现。当质量控制发现系统中的缺陷和用户要求增加新功能时，开发人员修改系统及与之相关的产品，从而导致不断的变化。

复杂性和不断变化代表了这样一种挑战：它使得任何个人都不可能控制系统和它的演化。如果不正确地进行控制，即使目标就在眼前，复杂性和不断变化将使得解决方案在发布前就胎死腹中。在应用域表述中，太多的错误会使得出台的解决方案对用户毫无用处，从而不得不返工。不成熟或不兼容的实现技术导致很差的稳定性和延迟。不能解决变动会给系统带来新的缺陷并使性能降低。

本书反映了作者 10 余年来构造系统和教授软件工程课程的教学内容。我们发现，学生通常孤立地学习编程技术和软件工程技术，使用小的问题作为例子。结果，他们可以很有效地解决定义明确的问题，但对他们的第一次真实开发经历，却常常束手无策。而这种真实开发是需要许多不同的技术和工具并要求不同的人共同协作的。针对这种状况，现在的本科课程通常都包括一门软件工程项目课程，并组织成一个单独的开发项目。

工具：UML，JAVA 和设计模式（Design Patterns）

我们编写本书的时候脑子里有个项目课程。但本书在其他情况下也可以使用，比如短而集中的学习培训或者短期的 R&D 项目。我们使用真实系统中的例子并检查类似于 UML（Unified Modeling Language）、基于 JAVA 的技术、设计模式、设计原理、配置管理以及质量控制等这些技术间的交互。此外，我们讨论与这些技术相关问题的项目管理以及它们对复杂问题和不断变化造成的影响。

原则

我们遵循下面 5 个原则教授软件工程：

（1）实践经验

我们相信软件工程教育必须与实践经验联系起来。对复杂问题的理解只能在身临其境地处理复杂系统的过程中才能得到，也就是说，复杂系统是一个学生无法独自完整地理解的系统。

（2）问题的解决

软件工程教育必须基于问题的解决。因此，没有对或者错的解决方案，只有相对于规定标准的比较好和比较差的解决方案。尽管我们调查实际问题的现有解决方案并且鼓励它们的复用，但也鼓励对标准解决方案的批评和改进。

（3）有限资源

如果拥有充足的时间和资源，我们或许能够建立理想的系统。这样的情况存在几个问题。首先，这是不现实的。其次，即使有足够的资源，如果在开发过程中，最初的问题很快发生变动，我们最终将会移交一个解决错误问题的系统。因此，我们假设解决问题的过程在资源方面受到限制。而且，强烈意识到缺乏资源将促进基于组件的方法、知识的复用、设计和编码。换句话说，我们支持软件开发的工程环境。

（4）跨学科性

软件工程是一个跨学科的领域。它涉及到电子工程和计算机工程、计算机科学、商务管理、图表设计、工业设计、建筑学、戏剧和写作等领域的知识。软件工程是一个应用领域。当试图理解和为应用域建模的时候，开发人员与其他人员进行有规律的相互交流，这些人包括用户和客户，其中一些人对于软件开发懂得很少。这就要求从多角度并用多术语来观察和处理系统。

（5）交流

即使开发人员只为自己建造软件，他们仍然需要相互交流。作为开发人员，我们不能只限于与同行交流。我们需要就备选方案进行讨论、阐述解决方案、协商折中的方法，以及对其他人的工作加以评审和批评。软件工程项目中的大量失败都源于信息的不准确或遗漏丢失。我们必须学会与项目的所有参与者交流，最重要的是包括客户和最终用户。

上述 5 个原则是本书的基础。它们将鼓励并使得读者能够用符合实践的、有一定技巧的解决方案来解决复杂和不断变化的问题。

关于本书

本书基于应用于软件工程的面向对象技术。它既不是一般的研究所有可能方法的软件工程书，也不是关于算法和数据结构的程序设计课本。相反，我们把重点放在有限的技术上，并且在合理的复杂环境中解释它们的应用，比如包括 20~60 个参与者的多组开发项目。因此，本书也反映了我们的偏见、实力和弱点。尽管如此，我们希望所有的读者都能够从中找到有用的东西。本书共 12 章分为 4 个部分，可以作为一个学期的课程进行教学。

第 1 部分“开始”包括三章。在这一部分，我们重点介绍软件工程环境中开发人员所必需的基本技能。

- 第 1 章“软件工程介绍”，描述了程序设计和软件工程之间的差异、本学科当前存在的挑战，以及书中所用到的基本概念的定义。
- 第 2 章“用 UML 建模”，描述了面向对象技术中用到的建模语言 UML 的基本组件。我们把建模作为处理复杂性的一种技术提出来。本章教会读者如何阅读和理解 UML 图。随后的章节将讲授如何创建系统不同部分的 UML 图。本书中，我们使用 UML 来为从软件系统到过程和工作产品的多种产品建模。

- 第 3 章“项目交流”，讨论开发人员执行的单个的最重要活动。开发人员和管理人员花了一大半时间在相互交流上，或者是面对面的，或者通过 Email、群件、电视会议或手写文件。当用建模来处理复杂性的同时，用交流来处理变动。我们描述了交流的主要方法及其应用，并讨论了是什么构成了有效的交流。

第 2 部分“处理复杂性”，我们集中于那些有助于开发人员详细说明、设计和实现复杂系统的方法和技术上。

- 第 4 章“需求提出”；第 5 章“分析”。描述了从用户角度出发的系统定义。在需求提出过程中，开发人员决定了用户需要的功能以及移交这些功能的可行方法。在分析阶段，开发人员形式化这些知识并保证其完整性和一致性。这里我们集中讨论如何使用 UML 来处理应用域的复杂性。
- 第 6 章“系统设计”，描述了从开发人员角度出发的系统定义。在这一阶段，开发人员用设计目标和子系统分解的形式定义了系统的结构。他们在这里提出全局性问题，如：系统到硬件的映射、长期数据的存储以及全局控制流。讨论集中在开发人员是如何使用设计模式、组件以及 UML 来处理求解域的复杂性。
- 第 7 章“对象设计”，描述了与求解域相关的具体的建模和构造活动。我们将需求和系统模型进行细化，精确说明了构成系统的各个类，并定义了现存类库和框架的外围结构。为了确定类的接口，我们使用 UML 的对象约束语言（Object Constraint Language）。

第 3 部分“管理变化”。我们集中讨论在整个生命周期中支持对变化的控制、评估和实现的方法和技术。

- 第 8 章“基本原理管理”，描述了设计决策的获得以及对这些决策的证明。我们在需求提出、分析和系统设计三个阶段开发出来的模型给出了一个系统该做什么和如何去做的不同视角，从而有助于对复杂问题的处理。为了能够处理变化，我们需要知道系统为什么是那样。设计决策的获得、可选方案的评估以及对它们的论证使得我们得到了系统的基本原理。
- 第 9 章“测试”，对照系统模型描述了系统行为的合法性。通过测试可以发现系统中的错误，包括那些在系统或者需求发生变化时产生的错误。测试包括单元测试、集成测试和系统测试。我们介绍了几种测试技术：白盒测试、黑盒测试、路径测试、基于状态的测试、检查。
- 第 10 章“软件配置管理”，描述了对项目的发展过程进行建模的技术和工具。配置管理有助于我们对变动的处理，这样就对基本原理进行了补充。版本管理记录了系统的开发过程，保证了同一个版本的各个组件的质量和它们之间的一致性。变动管理使得对系统的修改与项目的目标始终保持一致。

- 第 11 章“项目管理”，描述了一些在启动软件开发项目、跟踪其进度以及处理遇到的危险和意外事件中所必需的技术。焦点集中在有助于大量参与者按照计划合作并移交一个高质量的系统的组织、角色和管理活动上。

第 4 部分“尾声”。我们依次回顾了前面章节中提到的概念。第 12 章“软件生命周期”，讲述了软件的生命周期，例如 Boehm 的螺旋模型和标准的软件开发过程，并且提供了开发活动的抽象模型。在这一章中，我们还讲述了能力成熟度模型，可以用它来评价组织的成熟性。我们用两个软件生命周期的例子来结束本章，它们可以用于课堂教学实验。

上面这些主题是密切相关的。为了强调这一点，我们反复讲述了它们之间的关系。每一章都由 5 节组成。第 1 节阐述一个例子来介绍相关问题。第 2 节简要描述与主题相关的技术。第 3 节通过一些简单的例子来解释基本概念。第 4 节用真实系统中的例子来详细讲解主要技术。最后，我们讲述管理中的问题并讨论典型的折中方案。通过越来越复杂的例子来重复讲述同一个概念，我们希望读者能获得面向对象软件工程的实际知识。

课程

本书是为本科毕业班学生或研究生编写的，适用于一个学期的软件工程课程。学生应该具有一定的编程能力，例如会用 C，C++，Ada 或 Java。我们希望学生具有必备的攻克技术问题的能力，但是不要求他们有应付系统开发中的复杂多变情况的能力。不过，本书也可以用于其他类型的课程，如短期高强度的专业课程等。

项目和高级课程

实验课程应该包括书中的所有章节，最好按顺序讲述。老师可以考虑在介绍课程的时候较早地讲解第 11 章中项目管理的概念，让学生熟悉制订计划和编写状况报告。

介绍性的课程

一个介绍性的课程应该把重点集中在每一章的前 3 节。第 4 节可以用来做作业，也可以在该节中通过使用 UML 图表、文档和代码的相关技术来模拟一个小系统的构造过程。

简短的技术课程

本书也可以用作短而精的专业性指导。一个技术性的课程的重点在于 UML 和面向对象的方法，所用到的部分按照顺序为第 1，2，4，5，6，7，8，9 章，包括了从需求提出到测试的所有过程。高级课程还应该包含第 10 章“软件配置管理”。

简短的管理课程

本书还可以用作旨在面向管理者的比较简短强度密集的指导性培训课程。管理性课程的重点集中在交流、风险管理、基本原理、成熟度模型和 UML 这些与管理相关的部分，所用到的部分按照顺序为第 1，2，11，3，4，8，10，12 章。

关于作者

Bernd Bruegge 博士曾在卡耐基·梅隆大学学习和教授软件工程长达 20 年之久，在那里他获得了硕士和博士学位。他还曾经获得汉堡大学的毕业证书。目前，他是某大学的计算机科学教授并是 Technische Universitat Munchen 的应用软件工程首席专家，并且是卡耐基·梅隆大学的客座教授。本书中所讲的内容，是他以书本和网络教学的方式进行了 10 年的面向对象的软件工程课程教学经验的结晶。在 1995 年的卡耐基·梅隆大学教学奖中，他获得了 Herbert A. Simon 优秀奖。Bruegge 还是一名国际级顾问，他使用本书中介绍的技术设计并实现了很多实际系统，包括：DaimlerChrysler 的工程反馈系统、E.P.A.的环境建模系统、市警局的意外事件管理系统等。

Allen Dutoit 博士是 Technische Universitat Munchen 的科研人员。他在卡耐基·梅隆大学获得了硕士和博士学位，在 Lausanne 的瑞典联合技术学会获得荣誉证书。从 1993 年开始，他就与 Bruegge 教授一起，在卡耐基·梅隆和 Technische Universitat Munchen 大学教授软件工程课程，他们使用并完善了本书中所描述的方法。Dutoit 的研究包含了软件工程和面向对象系统的多个方面，包括：知识管理、基本原理管理、分布式决策支持和基于原型的系统。他曾经是软件工程学会和卡耐基·梅隆大学复杂工程系统学会的会员。

目　录

第 1 部分　开始