软 件 工 程 技 术 丛 书

HZ BOOKS

软件复用与构件技术系列

基于构件的产品线工程 UML方法

Component-based Product Line Engineering with UML

（德） Colin Atkinson 等著

顾剑 钟鸣 束尧 等译

机械工业出版社 China Machine Press

中信出版社 CITIC PUBLISHING HOUSE

软 件 工 程 技 术 丛 书

软件复用与构件技术系列

基于构件的产品线工程 UML方法

Component-based Product Line Engineering with UML

（德） Colin Atkinson 等著

顾剑 钟鸣 束尧 等译

机械工业出版社 China Machine Press

中信出版社 CITIC PUBLISHING HOUSE

本书重点讲述了怎样通过将基于构件的开发与产品线方法相结合，最大程度地改善和提高构件的可重用性和软件生产效率。

全书分为五部分：第一部分大致介绍了背景知识以及KobrA方法；第二部分介绍构件建模；第三部分讲述构件的具体化；第四部分讲述产品线工程及其相关概念；最后一部分是项目监控。

本书主要适合于致力于构件重用和MDA研究的软件工程师，以及希望进一步了解基于构件进行开发或者产品线工程的关键原则及其之间相互关系的学者和学生。

本书版权登记号：图字：01-2001-4967

图书在版编目（CIP）数据

基于构件的产品线工程：UML方法 /（德）阿特金森（Atkinson, C.）等著；顾剑等译. -北京：机械工业出版社，2005.2

（软件工程技术丛书 · 软件复用与构件技术系列）

书名原文：Component-based Product Line Engineering with UML

ISBN 7-111-15655-2

Ⅰ.基… Ⅱ.①阿… ②顾… Ⅲ.①软件工程 ②面向对象语言，UML-程序设计 Ⅳ.①TP311.5 ②TP312

中国版本图书馆CIP数据核字（2004）第128332号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码 100037）

责任编辑：刘立卿

北京牛山世兴印刷厂印刷 · 新华书店北京发行所发行

2005年2月第1版第1次印刷

787mm ×1092mm 1/16 · 27.25印张

印数：0 001- 3 000册

定价：59.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书热线：（010）68326294

译者序

本书的重点在于基于构件的开发方法和产品线工程的结合。UML是一种比较成熟的标准和工具，使用UML对本书所提到的KobrA方法进行描述，更易于读者理解。

基于构件的开发是一种革命性的软件开发和维护方式。然而，现代的构件技术仅仅只在开发的最后阶段才支持构件，而早期的分析和设计工作仍然是以传统的、非面向构件的方式进行。这就使得本书所讲述的KobrA方法具有极其重要的意义：KobrA方法使软件在其整个生命周期中，都可以充分利用基于构件的开发方式的优势，并通过产品线方法支持软件开发和演进，极大地提高了构件的可重用性，改善了软件的生产效率。

本书的特点是首先提供了一个可运行的案例：图书馆系统。这个案例贯穿全书，以集成的方式举例说明了KobrA方法中的关键理念。其次，本书在内容组织上也比较有条理：先是在不考虑产品线技术的前提下，系统地讲述了基于构件开发的概念和核心原则；然后重点讲述了产品线工程及其与KobrA方法的结合，以循序渐进的方式让读者在不知不觉中掌握基于构件的开发方式与产品线方法的精髓。

由于本书原著者的母语并非英语，所以在翻译的过程中有不少地方比较晦涩难懂，我们根据实际意义进行了一定的意译。

本书主要由顾剑、钟鸣、束尧翻译。在翻译过程中，我们得到了张静、张煜、肖和平、张杰良、王景新、汪东、张英、张明军、许华、李慧霸、王凤芹等的大力支持，在此表示感谢。全书最后由顾剑统稿。Be Flying工作室负责人肖国尊负责本书翻译质量和进度的控制与管理。在本书翻译过程中，我们亦对书中的一些错误进行了更正。

敬请各位读者提供关于本书的反馈意见，我们希望通过这些意见来了解自己的不足，以求在今后的译作中更全面地考虑读者的需要。读者可以将意见发送到be-flying@sohu.com。希望各位能够从本书获益！

关于作者

本书由十位作者共同参与完成，这些作者均来自Fraunhofer实验软件工程研究院（Institute for Experimental Software Engineering，IESE）。Colin Atkinson是Fraunhofer实验软件工程研究院KobrA方法开发小组的领导者，也是本书的主要作者和编辑。他主要负责编写本书的构件模型以及KobrA中使用的UML建模方法。Joachim Bayer是产品线工程专家，也擅长关注点的分离，他主要负责编写本书的产品线工程、构件重用、模式以及方法的应用部分。Christian Bunse是实验软件工程研究院“面向对象和基于构件的开发”业务领域的经理，主要负责实施方法的编写。Erik Kamsties是实验软件工程研究院“应用工程”业务领域的经理，对本书的主要贡献在于构件规约和实现方法。Oliver Laitenberger是实验软件工程研究院“具成本效益的软件开发”业务领域的经理，主要负责KobrA方法质量保证的编写。Roland Laqua是配置管理和维护专家，主要负责KobrA方法的配置管理的写作。Dirk Muthig是产品线工程领域的专家，主要负责KobrA产品线方法的编写。他和Joachim Bayer都是本书Library案例研究的主要作者。Barbara Paech是实验软件工程研究院质量软件开发部的领导人，也是本书中KobrA方法的上下文实现部分的主要作者。Jürgen Wüst是质量建模和测量活动领域的专家，主要负责书中KobrA方法的质量建模部分。Jörg Zettel是软件过程建模专家，在本书中负责编写整个KobrA过程的描述。

Fraunhofer实验软件工程研究院的研究重点是应用研究、开发以及创新软件开发方法领域的技术转化，还研究质量和过程工程、软件产品线以及持续改进和组织学习。为了使软件产业开发人员和用户为迎接现在和未来的信息技术挑战做好准备，开发出了许多新技术、新方法、新过程和新工具，从而能基于可靠的工程原则来开展软件开发。

序　言

采用基于构件的方法进行系统的设计和构建是如此直观、诱人，因此基于构件的系统似乎是易于构建的。事实果真如此吗？这么多年以来，我们知道光凭直觉还是不够的。自20世纪60年代开始出现构件化方法，软件工程就一直在朝这个方向努力。在此期间，我们取得了长足的进步，并且在最近（几）十年里，我们已懂得如何使用小粒度的构件——对象——来开发系统。

现今，我们的工作领域发生了改变：一个大规模的、基于网络的、构建于各种粒度构件之上的系统呈现于众人眼前。本书叙述了该领域的重要进展，并重点讲述了如何架构一个大规模的、基于构件的系统这一复杂问题。本书所采用的具有很强实用性的方法来自于面向对象的方法学（比如OMT、Objectory、Fusion、Catalysis）的最佳实践，以及一些形式化的方法（比如Cleanroom方法等）。

本书还描述了用来进行架构建模的KobrA方法。KobrA是一种系统方法，它使用UML这一令人振奋的简单而又直观的工具。对于构件的作者来讲，所遇到的问题之一就是如何定义“构件”。构件这个词的应用范围非常广，所以很难给它下一个清晰的定义。为了克服这一点，KobrA引入了另外一个新术语：“Komponent”。“Komponent”是逻辑构件，表示构建一个软件系统的逻辑构件块（在本书中，Komponent都统一译成“K-构件”。——译者注）。

KobrA从概念上讲是很清晰的，因为它只采用了少数几个概念，而这些概念广泛适用于各种级别的粒度和抽象系统。因此，KobrA中所述方法从本质上讲既适用于小规模系统又适用于大规模系统，而无须在概念、产品或活动方面进行根本性的改变。这样做的一个好处是，在一个项目中，在某一点采取什么样的开发活动主要是由所开发的工件的属性所决定的，而不是由架构的规模大小和位置决定的。

实践证明KobrA是有效的，因为它的规定很自然。KobrA建议用符合逻辑的、递归的过程去开发工程工件。对每一个工件，KobrA都解释了哪些输入是必须提供的，以及哪些一致性的检查是必须完成的。这样的系统过程并没有束缚住开发者，KobrA没有要求开发者一切都从头开始，实际上KobrA更关注软件重用。

KobrA不仅关注单个的应用以及系统，它致力于产品线技术的研究。软件产品线是一个非常重要的软件开发新范型。产品线技术根据产品的成本效益对产品进行有差异的开发，使得各种组织机构都能够及时对市场做出反应和降低成本。KobrA率先提出了一种适用于框架工程的系统方法学。

KobrA与一系列的业界标准保持一致。业界已经越来越清楚地意识到，应该以一种一致的、与应用领域和中间件平台无关的方式进行软件架构的建模。在2001年春天，OMG（Object Management Group，对象管理组——一个在分布式系统和构件技术领域居领导地位的标准化组织）宣布了MDA（Model Driven Architecture，模型驱动架构），通过UML支持这种新的系统架

构方式。MDA支持跨平台的互操作性，提高了设计和实现的可重用性，降低了应用开发和维护的成本。MDA是基于UML的，所以KobrA方法可谓是支持MDA的第一个全面的方法。

本书的出版，是Colin Atkinson和他在Fraunhofer IESE的同事们为基于构件的方法学和架构设计做出的重大贡献。我希望读者也像我一样，能够从本书有所收获。如果读者希望系统地了解如何进行基于构件的系统架构设计，本书无疑是最佳的选择。

Derek Coleman
RosettaNet首席技术官
www.rosettanet.org
2001年7月

前　言

在软件开发过程中将各种级别的重用引入到产品线，是其他工程学科一直都在追求却又达不到的目标。然而，在过去的几年里，一些新的开发范型的出现彻底改变了这种状况，使得软件工程能够在工程学科这个大家族中找到一席之地。其中最引人注目的就是基于构件的软件开发和产品线工程。按照粒度从小到大排序，一端是构件——即能快速简单地组装成新系统的可重用的软件构件块，而另一端则是产品线工程——即在单一的可高度重用的软件核心内，将产品中一定结构范围内的所有公共部分合并。

遗憾的是，在企业级的软件开发中应用这些范型往往会遭遇挫折，因为缺乏规范的系统方法。目前有很多软件方法都涉及到构件或产品线，但这两种开发范型在这些软件方法之中的应用时机要么是太迟——直到软件生命周期的后期阶段（在构件的情况下），要么是太早——在软件生命周期的前期阶段（在产品线的情况下）。现有的方法几乎都不能够在软件开发的所有阶段和层次上将构件和产品线自然地协同起来。本书介绍一种新的方法——KobrA，它致力于在整个软件的生命周期，为基于构件的产品线工程提供一种简单的、规范的、系统的方法。该方法还利用UML（Unified Modeling Language，统一建模语言），以兼容于大多数主流实现技术和中间件技术（包括领先的构件技术）的方式，向这两种范型提供支持。这样，它也为将MDA技术应用于软件工程奠定了基础。

KobrA的主要目标是通过定义一个尽可能简单且正交的特征集来避免特征重载——这在其他许多方法中是存在的。尤其是，KobrA的产品和过程都是基于一个最小核心原则集的递归应用。KobrA的另一个要点和质量有关。构件和产品线的强大之处在于它们能够提高重用性，这就要求可重用的资源应该尽可能具有最好的质量。显然，系统地开发和重用低质量工件的组织机构是好景不长的。因此，具有良好定义的一致性规则和对其进行验证的系统技术都集成到了KobrA方法的方方面面中。

在软件开发方法的演变过程中，不同的时期具有不同的特征：最初是大量概念的迅速激增；接下来则是这些概念的统一和合并。UML的历史就是这种模式最典型的例子。我们希望KobrA中的理念标志着我们向基于构件的开发概念和产品线的开发概念相结合的方向又迈进了一步，也希望其有助于一些关键性原则的发现和改进。产品线工程的一大特征是分析和明确指出某个产品家族中不同成员之间的共性和差异。基于构件开发的一大特征是在分析和设计的层面上进行“局部化的”工件开发（比如UML图），着重于对单个逻辑抽象体（即构件）的描述。我们采用这两个核心理念作为本书的主干。

KobrA项目

本书中描述的方法是在开发由德国联邦教育研究部（BMBF）资助的Komponentenbasierte

Anwendungsentwicklung[⊖]（即KobrA）项目中开发出来的。在此项目中，有四个组织在协同工作：

- Fraunhofer-Institut für Experimentelles Software Engineering (IESE), 凯泽斯劳滕
- GMD-Forschungszentrum Informationstechnik (GMD-FIRST), 柏林
- PSIPENTA Software Systems GmbH, 柏林
- Softlab GmbH, 慕尼黑（项目领导人）

Fraunhofer IESE主要负责KobrA方法的研究；GMD-FIRST和Softlab负责开发一个补充性的工作台，以支持KobrA方法和商业CASE工具的配合使用；而PSIPENTA则为ERP（企业资源规划）领域的方法提供测试床。

读者对象

本书主要适用于正在寻找一种简单但规范的软件开发方法的软件工程师，尤其是关心如何利用现有软件资源去开发某个应用系列的开发人员。由于本书中的方法具有模块化的自然特性，所以其中的主要技术都能彼此独立地应用。寻求基于构件开发方式支持的开发人员可以独立地应用面向构件的理念，而不用考虑产品线理念；同样，寻求MDA支持的开发人员可以独立地应用相应的规约和实现思想，而不用考虑构件。

本书对于希望进一步了解基于构件的开发和/或产品线工程的关键原则及其之间相互关系的学者和学生同样适用。

读者最好能熟悉软件工程的基本概念、面向对象软件开发的基本原理以及UML的主要特征。

本书结构

本书按五个部分来进行组织。第一部分介绍KobrA的方法和背景，并给出KobrA关键特征的概述；第二部分重点讲述在不考虑产品线问题的情况下，如何利用UML进行逻辑构件系统的建模；第三部分描述了如何通过直接实现或者重用已有构件，比如COTS (Commercial Off-The-Shelf，商业现货）构件，得到可执行的逻辑构件，同时讨论了如何使其适应于增量开发方法，第四部分重点讲述产品线工程，并展示了如何实现对产品线的支持，最后，第五部分重点讲述了关键性的支持活动——维护、质量保证、质量建模，以及如何将KobrA引入到现实软件开发环境中。

随后是本书的两个附录。附录A介绍KobrA元模型，包括对构件结构及其之间相互关系进行管理的一致性规则的完整列表。附录B提供KobrA过程的详细规约，包括一个完整的列表，其中列出了组成该过程的活动及这些活动之间的产品流依赖关系。

⊖ 即“component-based application development”（基于构件的应用开发）的德语表示。

图书馆案例研究

一个单独的可运行的案例研究几乎贯穿于本书（第一部分除外）的全部章节，用于说明KobrA方法的应用。之所以将图书馆管理系统作为本书的研究案例，是因为几乎每一个人都熟悉它的基本概念。图书馆管理系统有助于图书馆管理员和信息专家对图书馆进行管理，包括与用户的交互、库存和账户管理。

基本的图书馆系统包括用户注册、库存书目的管理，以及跟踪所有书目的借出、归还和预订。这个基本的图书馆系统被选为图书馆管理系统家族的特殊成员，用来在本书第二部分和第三部分说明一个系统的开发。

由于对KobrA中产品线概念的解释涉及到图书馆系统的更多细节，为此，本书分析了三个图书馆系统，并用一个总的框架对三者的共性和差异进行建模。本书第四部分解释了该案例的扩展研究，其中包含了产品线技术。

完整的案例研究——包括基本的图书馆系统和图书馆系统产品线——在[BMG01]中做了描述，读者可以在本书的网站上找到它。

致谢

作者非常感激为本书中所描述的思想提供支持和做出贡献的众多友人。首先，我们要感谢KobrA项目和KobrA委员会中的同事：PSIPENTA Software Systems GmbH的Marion Dikel、Robert Hagemann、Gernot Krause、Jörg Lucas、Jens Reinhold和Torsten Sander、Softlab GmbH的Martin Dehn、Günther Merbeth、Friedrich Rössler和Uwe Zeithammer、GMD-FIRST的Matthias Anlauf、Marko Fabiunke、Joanna Filipek、Boris Groth、Stefan Jähnichen和Ronald Melster，以及Fraunhofer IESE的Günther Ruhe。其次，我们还要感谢我们在IESE的以前和现在的同事：特别是Stan Jarzabek、Dieter Rombach、Peter Rösch和Tanya Widen。第三，我们还要感谢为本书草稿提供宝贵反馈意见的审稿人员，特别是Morgan Bjorkander、Mathias Clauss、Mike Fischer、Hans-Gerhard Groβ、Bernhard Humm、Heinrich Hussman、Thomas Kühne、Cris Kobryn、Antje von Knethen和Frank Maurer。最后，还要特别感谢Brigitte Göpfert对本书案例研究中的图书馆信息领域提出的见解和专家意见，Maud Schlich为测试章节所做出的贡献，Jhair Tocancipa的EJB和CORBA实例，以及Addison-Wesley出版社的编辑Alison Birtwell对本书的鼓励和支持。最后还要感谢Fraunhofer IESE和BMBF对我们工作的支持。

网站

相关的更多信息请访问本书的网站：

www.iese.fhg.de/KobrA/book

www.kobra-project.org

目　录

第三部分 具 体 化

第一部分

概　述

每一种新方法都是在已有方法的基础上形成的。本书的第一部分为后续部分的详细描述提供了预备知识：这一部分描述了KobrA方法的背景知识，对KobrA的特点影响最深的一些方法，以及驱动KobrA开发的目标。

第1章首先解释了KobrA中所集成的范型的重要性和背景知识，并介绍了KobrA所基于的现有方法。这些都可以视为KobrA的组成部分。然后，这一章描述了组合这些部分的基本原则，这些原则被视为KobrA的“秘诀”。

第2章对KobrA进行了概述，并把它与其他一些先进方法在特点方面做了比较。最后，对KobrA的主要特性进行了总结。

第1章 背景知识

“必须改变在真实环境中所使用的程序，否则它在该环境中的用处将逐渐减少。”

——Lehman

Lehman的这条软件工程定律[LB85]最早是在1985年提出的，而在今天看来，该定律可能更加正确。如今，企业要在充满竞争的软件行业中取得成功，已经越来越依赖于这样一种能力，即对不断变化的市场条件和用户需求做出快速反应。

有多种不同的方法可以满足对变化的需要，而最有效的一种便是软件重用。显然，现有的软件资源中，能被新软件产品（或软件派生体）改造或直接包括进来的部分越多，所需的代价就越小，并且企业所处的竞争位置就越有利。然而，已经证明，系统化的、可靠的软件重用方法是难以描述的。人们花了许多时间尝试了大量的方法，但是真正系统化的软件重用与其说是一种规律，还不如说是一种例外。KobrA的核心目标之一就是帮助解决这个问题，并能通过重用提高软件开发的成本有效性。

这一章是介绍性的，其目的是为了解释KobrA方法的背景知识，为后续章节描述KobrA方法提供支持。为此，我们将简要地介绍一些先进的技术，而软件行业正寄希望于这些技术来实现更广泛的重用；另外将认识这些技术中目前所存在的一些障碍。然后，我们将介绍一些最有影响力的开发方法的长处和不足，并讨论这些方法对重用技术的支持程度。最后，我们将描述一个方法应当展示的一些理想特征，同时确定一些指导开发KobrA方法的主要原则。

1.1 重用技术

多年来，人们提出了大量的技术用以解决长期以来的软件重用问题。然而，其中只有少数几种技术成功地产生了重大的影响。如今的软件开发人员通常仍然花费了过多的时间来“重造车轮”，而他们所采用的重用也大多是出于特别原因，而不是事先计划好的。不过在近几年里，出现了几种相互关联的技术，它们可能会对重用问题产生重要的影响，尤其是下面三种技术：

- 基于构件的开发。
- 架构风格和设计模式。
- 产品线工程。

1.1.1 基于构件的开发

许多行业观察家曾经预计，面向对象将成为重用的“银弹”[Cox84]。他们认为，由继承、

多态和动态绑定等机制所支撑的类和对象，将很容易得到广泛的、制度化的重用，并能促进COTS（Commercial Off The Shelf，商业现货）构件市场的发展。然而，这些预计从来都没能够完全实现。

其中的关键问题之一是，主流的面向对象语言都采用了编译器绑定技术。这就意味着，在运行时刻，对象被埋葬在更大的、完整的工件中，这些工件不顺从于任何强大的、面向重用的、可以应用于类和对象的特性。例如在C++中，“程序”[⊖]是仅有的可执行单元，它有着非常有限的重用属性。若要重用单独的类和对象，开发人员需要经过一个冗长的周期——包括编辑、编译、链接，才能把重用的类和对象嵌入到新程序中。

构件范型通过让单个对象成为可独立运行的单元，缓解了这一问题。因此，构件范型可以看作传统的对象范型的一种扩展，它将对象从封装程序的限制中解放出来。类打着构件的幌子，这样它不仅能享有对象范型的所有优势，还能打包成可以组装且可以运行的封装体。这样，在新应用的创建过程中使用构件，已不仅仅是个编程过程，而更多的是一个组装过程。

构件范型是在探索更广泛重用的过程中所迈出的坚实一步。然而，目前它仍然缺乏足够的方法学上的支持。大多数能容纳构件的开发方法，基本上将构件仅仅视为可执行的模块，且这些模块仅仅在开发过程的最后实现和部署阶段才担当重要的角色。而这确实就是目前领先的构件技术（EJB、COM+和CORBA）所提供的构件模型。为了释放构件的所有潜力，必须在开发过程的所有阶段都支持构件范型，而且必须将它视为开发过程中必不可少的一部分，而不仅仅是最终的结果。

1.1.2 架构风格和设计模式

构件范型提倡“小规模的重用”，这是因为构件代表了软件中最小的、相互连贯的包，且它们的意义是作为独立的、可重用的实体。架构风格和设计模式可以看成是用来解决在下一级粒度层次上的重用，因为它们方便了用于组装基本构造块的重用策略。这样，它们为面向构件的方法提供了天然的补充——构件代表了用以构造解决方案的可重用构造块，而模式/风格代表了用以组合构造块的可重用策略。

通常，假定架构的规模比设计更大；与模式比起来，而风格一般视为更加面向自动化应用，但这两对概念之间存在着重叠。实际上，对于其中有些重叠部分，有些作者将它们表示在架构层[BMR96]，而有些作者将它们表示在设计层[GHJ95]，例如“模型—视图—控制器”模式；而有些重叠部分，有些作者将它们表示成风格[GS94]，而有些作者则将它们表示成模式[BMR96]，例如分层。架构风格和设计模式[⊜]基本上都是基于同一种潜在的思想，即定义通用的组装概念，而这些概念已经证明能有效地解决重复出现的软件问题。另外，还可以利用这种思想在更高或更低级别上进一步抽象，并分别以分析模式[Fowl97]和习语[BMR96]的形式表现，或者以更具体

⊖ 由语法上的“main”结构指明。

⊜ “架构风格”和“设计模式”是通用术语，在一些重要的出版物（见参考文献[GHJ94][GS94]）中介绍过它们。