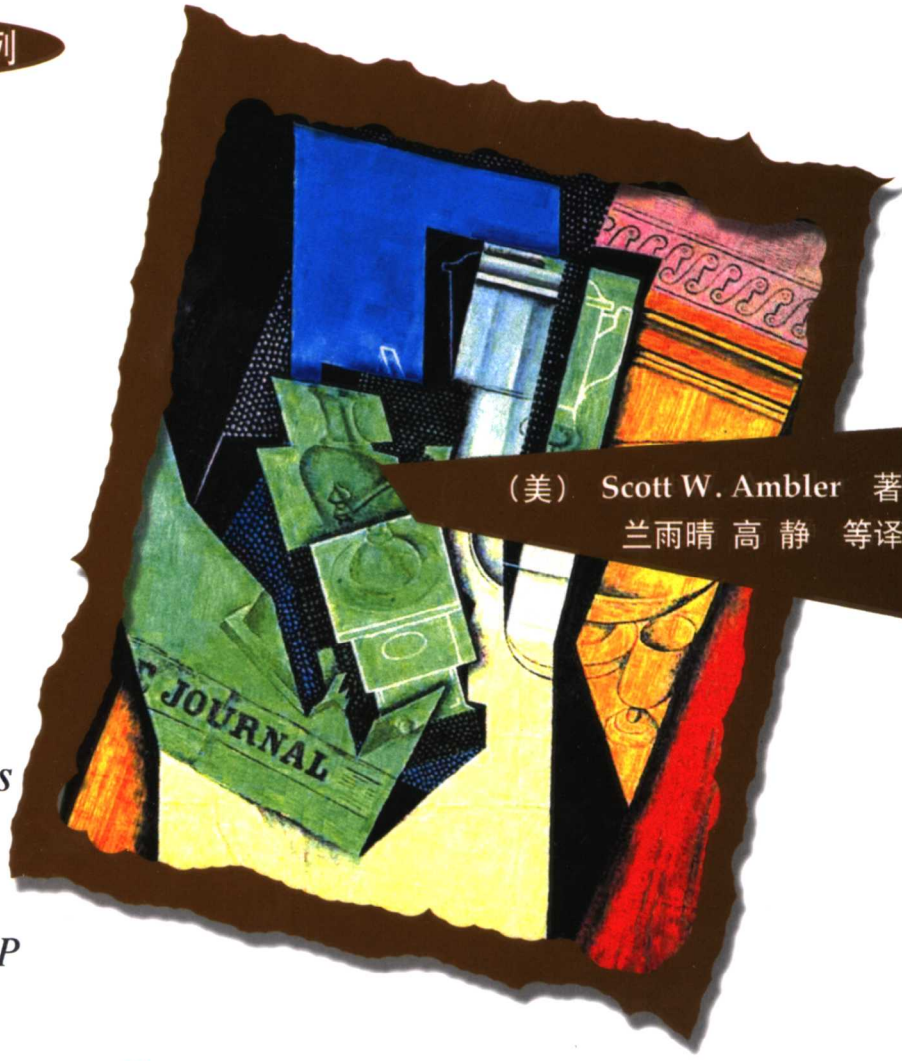
软 件 工 程 技 术 丛 书

HZ BOOKS

统一过程最佳实践
细化阶段

软件开发过程系列



（美） Scott W. Ambler 著
兰雨晴 高 静 等译

The Unified Process
Elaboration Phase
Best Practices
in Implementing the UP

机 械 工 业 出 版 社
China Machine Press

软 件 工 程 技 术 丛 书

统一过程最佳实践
细化阶段

软件开发过程系列

（美） Scott W. Ambler 著
兰雨晴 高 静 等译

The Unified Process
Elaboration Phase
Best Practices
in Implementing the UP

机械工业出版社
China Machine Press

本书重点介绍了与统一过程的细化阶段相关的最佳实践。通过对诸如代码走查、配置管理、变更控制和软件组织架构建模等最佳实践的介绍说明了实现软件过程的细节。

本书可以作为软件项目管理人员、软件开发工程师、过程工程师、系统工程师等专业人员的指导用书，也可作为高等院校计算机及相关专业学生的参考书。

Scott W. Ambler: The Unified Process Elaboration Phase: Best Practices in Implementing the UP (ISBN 1-929629-05-2).

本书版权登记号：图字：01-2003-3911

图书在版编目(CIP)数据

统一过程最佳实践　细化阶段/(美)安伯乐(Ambler, S. W.)著；兰雨晴等译. -北京：机械工业出版社，2005.4

(软件工程技术丛书-软件开发过程系列)

书名原文：The Unified Process Elaboration Phase: Best Practices in Implementing the UP

ISBN 7-111-15707-9

Ⅰ.统…　Ⅱ.①安…　②兰…　Ⅲ.软件开发　Ⅳ.TP311.52

中国版本图书馆 CIP 数据核字(2004)第 127823 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：王镇元

北京瑞德印刷有限公司印刷 · 新华书店北京发行所发行

2005 年 4 月第 1 版第 1 次印刷

787mm×1092mm　1/16·14.75 印张

印数：0001-3000 册

定价：29.00 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

序

信息技术行业最不为人知的事就是那些大型的、关键性的项目有80%～90%最终都遭遇失败。对于从事软件开发工作超过15年，并在3年前刚刚成为编辑的我来说，对此深有体会。当我刚刚开始成为 *Software Development* 杂志的技术编辑时，Scott Ambler 开始在"Thinking Objectively"专栏中发表文章。他使用日常用语来写作，这使得普通的开发者都能够阅读。在诸如过程模式、有效使用 UML 以及将对象映射到关系型数据库中等前沿技术领域，Scott 堪称专家。现在 Scott 开始将他的热情和兴趣投入到如何将大多数软件项目从高失败率的深渊中解救出来。

软件过程涉及如何标准化开发团队的工作并促进在业界长期共识的最佳实践，例如代码走查、配置管理、变更控制以及架构建模。拥有过程模式开发背景的 Scott，正处于一个对当前的面向对象的软件过程（例如 Rational 统一过程，即 RUP、OPEN 联盟所提供的 OPEN 过程以及面向对象软件过程，即 OOSP）可以提供一致而有重点的评价的理想位置。尽管 OOSP（详见 Scott 的《*Process Patterns*》和《*More Process Patterns*》）为过程模式提供了更好的内在支持，但 Scott 认识到：尽管面临挑战，统一过程是市场领导者，而且还将继续下去。正如 Scott 所说的（"增强统一过程"，*Software Development*，1999 年 10 月），"统一过程不是完美的——没有任何软件过程是完美的——但是，如果我们想要在软件领域成功，它是我们需要投入工作的一个领域。"这个系列书籍从 OPEN 过程和 OOSP 里抽取了资料来充实目前还单薄的统一过程。在我看来，一个充实的统一过程是处理这一尴尬问题的最好的方式之一，同时 Scott 就是分析解决这种问题的优秀教师。

Roger Smith

Software Development 杂志技术编辑

前　言

Software Development 杂志(www.sdmagazine.com)以及它的前身 *Computer Language* 曾经发表过大量关于如何成功开发软件的文章。曾经以及还在为该杂志写作的很多作者都是令人印象深刻的:Steve McConnell、Ed Yourdon、Larry Constantine、Steve McCarthy、Clemens Szyperski、Peter Coad 以及 Karl Wiegers 等。这些信息产业的著名专家已经通过这本令人尊敬的杂志同我们分享了他们的智慧成果。

后来,更多的组织开始逐步将更多的精力投入到提高软件过程上。这部分是因为千年虫(Y2K)的灾难,大约 80%～90%的大规模软件项目的失败,和与日俱增的对于遵循成熟软件过程是决定一个软件项目成功与否的关键的认识。从 20 世纪 90 年代中期以来,Rational 公司就开始收购和合并别的软件公司,这样做的结果是,将这些工具所支持的过程合并到单个开发过程中,即所谓的统一过程。是不是有可能将整个软件过程自动化呢?即使可以将其完全自动化,Rational 有一个完整的工具集吗?我不太确定。幸运的是,还有其他的人也在致力于定义软件过程——OPEN 联盟的 OPEN 过程就是其中之一,我自己的面向对象软件过程(OOSP)的过程模式也是一个——所以我们对于怎么去做这样的工作还是有些其他的观点。这些其他的观点能够用于驱动一个更加健壮的统一过程,并且为准确反映你的组织的真实需求的统一过程产生一个增强的生命周期。我相信在 *Software Development* 多年积累起来的智慧成果能够用来充实统一过程,并且能够统一业界的最佳实践。正因为如此,我们出版了这套丛书。

遵循一个经过证实的、成熟的过程是软件专家成功的关键。

为什么软件过程这么重要?我们后面就会讨论这个问题。假如你想建造一座房子,并且让两个建筑商来竞标。第一个告诉你通过使用新的建造技术,他能在两周内为你建造一座房子,如果他第二天就开始建造的话,这座房子将仅仅花费 100 000 美元。这个建筑商有一些顶级的木匠和管道工,并且在过去已经使用这种技术建造过一个花园洋房,并且他们也愿意日夜加班工作保证按时完成。第二个建筑商告诉你他需要跟你讨论你希望建成什么样的房子,当他一旦清楚你的需求后,将在一周内将你的需求做出一个草图,然后你可以审查并且反馈给她你的意见。这个初始阶段将耗费你 10 000 美元,而一旦你决定了想要建造的样式后,他会把详细计划和剩余工作的费用安排综合起来。你对哪个建筑商更加放心呢?想立刻开始建造的那个还是想先了解你对建造样式需求,做出模型,做出计划然后再建造的那个呢?显然,第二个建筑商有更大的机会做出满足你实际需求的房子。现在设想你正在构建软件——这是一项比建房子复杂程度高出几个数量级,并且常常是更加昂贵的工作——并且再一次假设你有两个承包商,而且他们使用上面两种方式。你对哪个更加放心呢?我希望答案仍然是第二个:一个明智的过程。不幸的是,实

践证明绝大多数的时间内，组织往往会选择第一个承包商所使用的方法。当然，实践也说明我们有大约85%的失败率，你认为两种现象有关联吗？我想是。

细化阶段

细化阶段是4个阶段的第2个。这4个阶段包括：初始阶段、细化阶段、构造阶段、交付和产品化阶段。在细化阶段，软件的发布贯穿了它的整个生命周期。这个阶段有几个目标：

- 为系统产生一个经过证实的、架构的基准。
- 让需求模型发展到“80%的完成点”。
- 为构造阶段开发一个粗粒度的项目计划。
- 确保关键的工具、过程、标准和指南已经为构造阶段准备好。
- 了解并消除项目中风险级别高的问题。

本书介绍了一组由业界专家所撰写的文章，这些文章讲述了最佳实践的前沿知识。本书的一个目标，并且也是这套书的总的目标，是为统一过程技术提供可选的、经过证实的方法。本书的另一个目标则是填补统一过程的漏洞——坦白地说，没有任何过程可以真正称为“完整”的。统一过程是开发过程而不是软件过程；因此，正因为它所选择的范围，它必然会略过一些软件专业方面的重要概念。但幸运的是，*Software Development* 中的作者们都使用了更为广阔的视角来看待过程。

关于这套书

本套书由四本书组成：分别为初始阶段、细化阶段、构造阶段以及移交阶段和产品化阶段。每本书都自成体系，但通过整个系列才能够完整了解软件过程。由于每本书中的文章都不重复，你会发现每一本书都很有价值。有时候，可能其中一本书中讲述的比较薄弱的过程工作流在其他几本书中讲述得更为详尽。例如，关于初始阶段的书中包括了大量项目计划和评估的内容——这也是开始一个项目时最关键的内容——而在其他几本书中这方面的讨论就比较少。

另外，在着手编著这套系列丛书时，我发现要比最初想像得难许多。我所碰到的问题在于，好的资料很多，但本系列丛书却只有有限的容量。所以大家可以发现本系列书中的文章都是精华中的精华。

关于我本人

作为 *Computer Language* 以及后来的 *Software Development* 杂志多年来忠实的读者，我从1995年开始为它撰写文章，并在1997年成为对象技术专栏的作家。我在20世纪80年代初期开始软件开发的工作，曾经使用Fortran以及Basic语言来编写代码，在80年代中期使用过Turing、C、Prolog以及Lisp。在80年代后期，我意识到有比编程生命周期更为长久的工作，所以我开始重新学习用户界面设计、数据建模、过程建模以及测试的技术，那时我用COBOL和第四代语言在IBM主机上编程。当我从结构/过程技术中醒悟过来后，在1990年我发现了对象，并很快地一头扎入Smalltalk开发中，接着是C++开发，最后又重新回到Smalltalk。在几个机构中担任指

导者以及架构师后，我决定将这些经历以及我所获取的技术结合，并在多伦多大学获得了一份助教的工作，在90年代中期开始从事职业培训工作。不久之后，我就意识到下面两件事情，首先，尽管我很喜欢教授培训课程(我到今天仍不曾间断)，但我并不希望做一个全职授课者。其次，更重要的是，我学习到如何使用简单易懂的方式来交流复杂的概念，例如怎样开发面向对象软件。基于这样的经验，我最初的两本书诞生了——《*The Object Primer*》(Cambridge University Press, 1995)以及《*Building Object Applications That Work*》(Cambridge University Press, 1997/1998)——这两本书从开发者的角度讲述了对象技术的基础知识。紧接着，我在《*Process Patterns*》(Cambridge University Press, 1998)一书中继续讲述了面向对象软件过程，并在另一本书《*More Process Patterns*》(Cambridge University Press, 1999)中提供给读者我在加拿大某个一流的对象技术咨询公司所获得的宝贵经验。从那时开始，我帮助过多家机构(无论大小，成立时间长短，所属行业)来提高他们的内部软件过程。我近期的著作包括本系列丛书以及另一本书：《*The Elements of Java Style*》(Cambridge University Press, 2000)。我想，我已经发现了属于我自己的体系结构。

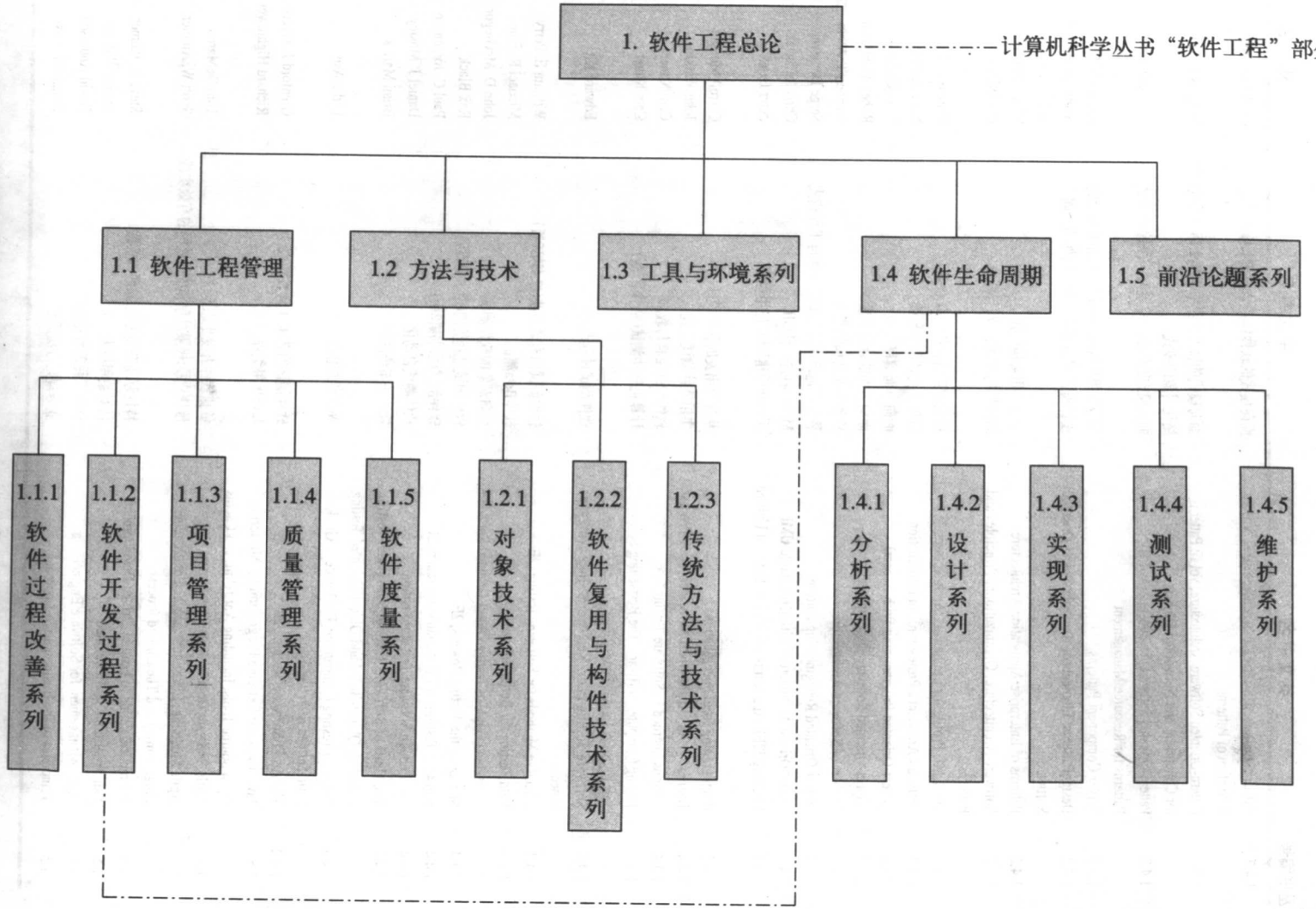
软件工程技术丛书书目

丛书编号	英文书名	中文书名	作　者
1	Object-Oriented and Classical Software Engineering, 5E	面向对象与传统软件工程(原书第5版)	Stephen R. Schach
1	Object-Oriented Software Engineering	面向对象软件工程	Timothy C. Lethbridge
1	Software Engineering: A Practitioner's Approach, 5E	软件工程:实践者的研究方法(原书第5版)	Roger S. Pressman
1	Software Engineering, 6E	软件工程(原书第6版)	Ian Sommerville
1	Software Engineering with Java	软件工程:Java语言实现	Stephen R. Schach
1	Project-Based Software Engineering:An Object-Oriented Approach	基于项目的软件工程:面向对象研究方法	Evelyn Stiller
1	Software Engineering Economics	软件工程经济学	Barry W. Boehm
1	Software Cost Estimation With Cocoma II	软件成本估算: Cocomo II模型方法	Barry W. Boehm
1	Object-Oriented Software Construction, 2E	面向对象的软件结构(原书第2版)	Bertrand Meyer
1	Software for Use: A Practical Guide to The Models and Methods of Usage-Centered Design	面向使用的软件设计	Larry L. Constantine
1.1.1	Software Process Improvement: Practical Guidelines for Business Success	软件过程改进	Sami Zahran
1.1.1	Making Process Improvement Work	软件过程改进简明实践	Neil S. Potter
1.1.2	The Road to the Unified Software Development Process	统一软件开发过程之路	Ivar Jacobson
1.1.2	The Unified Software Development Process	统一软件开发过程	Jacobson/Booch/Rumbaugh
1.1.2	The Rational Unified Process: An Introduction , 3E	RUP导论(原书第3版)	Philippe Kruchten
1.1.2	UML and The Unified Process: Practical Object-Oriented Analysis & Design	UML和统一过程:实用面向对象的分析和设计	Jim Arlow
1.1.2	The Unified Process Inception Phase	统一过程最佳实践：初始阶段	Scott Ambler
1.1.2	The Unified Process Elaboration Phase	统一过程最佳实践：细化阶段	Scott Ambler
1.1.2	The Unified Process Construction Phase	统一过程最佳实践：构造阶段	Scott Ambler
1.1.2	The Unified Process Transition & Production Phase	统一过程最佳实践：移交和产品化阶段	Scott Ambler
1.1.3	Managing Global Software Projects	全球化软件项目管理	Gopalaswamy Ramesh
1.1.3	Software Project Management: A Unified Framework	软件项目管理:一个统一的框架	Walker Royce
1.1.3	How to Run Successful Projects III: The Silver Bullet	成功的软件项目管理:银弹方案(原书第3版)	Fergus O'Connell
1.1.3	Successful Software Development, 2E	成功的软件开发(原书第2版)	Scott E. Donaldson
1.1.3	Six Sigma Software Development	六西格码软件开发	Christine B. Tayntor
1.1.3	IT Project Management: On Track from Start to Finish, 2E	实用IT项目管理(原书第2版)	Joseph Phillips
1.1.3	Successful IT Project Delivery: Learning the lessons of Project Failure	IT项目成功交付的秘诀	David Yardley
1.1.3	Software Project Management, 3E	软件项目管理(原书第3版)	Bob Hughes
1.1.3	Architecture-Centric Software Project Management	软件项目管理实用指南:以体系结构为中心	Daniel J. Paulish
1.1.3	Mentoring Object Technology Projects	对象技术项目管理	Richard T. Dué
1.1.3	Virtual Project Management	虚拟项目管理	Paul E. Mcmahon
1.1.3	AntiPatterns and Patterns in Software Configuration Management	软件配置管理中的模式与反模式	William J. Brown
1.1.4	Handbook of Software Quality Assurance, 3E	软件质量保证(原书第3版)	Gordon G. Schulmeyer
1.1.4	Software Reliability Engineering	软件可靠性工程	John Musa
1.1.4	Implementing ISO 9001:2000 The Journey from Conformance to Performance	2000版ISO 9001标准实施指南:从符合性到业绩改进	Tom Taormina

丛书编号	英文书名	中文书名	作　者
1.1.4	CMMI Distilled: A Practical Introduction to Integrated Process Improvement	CMMI精粹:集成化过程改进实用导论	Dennis M. Ahern
1.1.4	CMM Implementation Guide	CMM实施与软件过程改进	Kim Caputo
1.1.4	Implementing the Capability Maturity Model	CMM实施指南	James R.Persse
1.1.4	Object-Oriented Defect Management of Software	面向对象的软件缺陷管理	Houman Younessi
1.1.4	Metrics and Models in Software Quality Engineering	软件质量工程:度量与模型	Stephen H. Kan
1.1.4	Performance Solutions	软件性能工程	Connie U. Smith
1.1.4	Peer Reviews in Software: A Practical Guide	软件同级评审	Karl E. Wiegers
1.1.4	Software Assessments, Benchmarks, and Best Practices	软件评估、基准测试与最佳实践	Capers Jones
1.1.5	Practical Software Measurement	实用软件度量	John McGarry
1.1.5	Software Metrics: A Rigorous and Practical Approach, 2E	软件度量(原书第2版)	Norman E. Fenton
1.2.1	The Object Primer, 3E	Object Primer中文版(原书第3版)	Scott W. Ambler
1.2.1	Object-Oriented Methods: Principles & Practices, 3E	面向对象方法:原理与实践(原书第3版)	Ian Graham
1.2.1	Principles of Object-Oriented Software Development, 2E	面向对象软件开发原理(原书第2版)	Anton Eliëns
1.2.1	Object Solutions: Managing the Object-Oriented Project	面向对象项目的解决方案	Grady Booch
1.2.1	An Introduction To Object-Oriented Programming, 3E	面向对象编程导论(原书第3版)	Timothy Budd
1.2.1	The Unified Modeling Language User Guide	UML用户指南	Booch/Rumbaugh/Jacobson
1.2.1	The Unified Modeling Language Reference Manual, 2E	UML参考手册(原书第2版)	Rumbaugh/Jacobson/Booch
1.2.1	Applying UML and Patterns , 3E	UML与模式应用(原书第3版)	Craig Larman
1.2.1	Object-Oriented Analysis and Design with Applications, 2E	面向对象分析与设计(原书第2版)	Grady Booch
1.2.1	Business Modeling with UML: Business Patterns at work	UML业务建模	Hans-Erik Eriksson
1.2.2	Software Reuse: Architecture, Process and Organization for Business Success	软件复用:结构、过程和组成	Ivar Jacobson
1.2.2	Software Reuse Techniques: Adding Reuse to the Systems Development Process	软件复用技术:在系统开发过程中考虑复用	Carma McClure
1.2.2	Practical Software Reuse: Strategies for Introducing Reuse Concepts in Your Organization	软件复用实践	Donald J. Reifer
1.2.2	Large-Scale Component-Based Development	大规模基于构件的软件开发	Alan W. Brown
1.2.2	Component-based Product Line Engineering with UML	基于构件的产品线工程:UML方法	Colin Atkinson
1.2.2	Business Component Factory	基于构件的企业级开发	Peter Herzum
1.4.1	Object-Oriented Analysis & Design	面向对象的分析与设计	Andrew Haigh
1.4.1	Analysis Patterns: Reusable Object Models	分析模式:可复用的对象模型	Martin Fowler
1.4.1	Requirements Analysis and System Design: Developing Information Systems with UML	需求分析与系统设计	Leeszek A. Maciaszek
1.4.1	Systems Analysis and Design in a Changing World	系统分析与设计	John W. Satzinger
1.4.1	Advanced Use Case Modeling, Vol I: Software Systems	高级用例建模 卷I:软件系统	Frank Armour
1.4.1	Requirements Engineering: A Good Practice Guide	需求工程	Ian Sommerville
1.4.1	Software Requirements and Estimation	软件需求与估算	Swapna Kishore
1.4.1	Effective Requirements Practices	有效需求实践	Ralph R.Young
1.4.1	Applying Use Cases: A Practical Guide, 2E	用例分析技术(原书第2版)	Geri Schneider
1.4.1	Managing Software Requirements	软件需求管理:统一方法	Dean Leffingwell
1.4.1	Writing Effective Use Cases	编写有效用例	Alistair Cockburn
1.4.1	Problem Frames Analyzing and Structuring Software Development Problems	软件开发问题框架: 现实世界问题的结构化分析	Michael Jackson
1.4.1	Use Cases: Requirements in Context	用例:通过背景环境获取需求(原书第2版)	Daryl Kulak
1.4.1	Practical Software Requirements	实用软件需求	Benjamin L. Kovitz

丛书编号	英文书名	中文书名	作　者
1.4.2	Pattern-Oriented Software Architecture, Vol I: A System of Patterns	面向模式的软件体系结构 卷1:模式系统	Frank Buschmann
1.4.2	Pattern-Oriented Software Architecture, Vol II: Patterns for Concurrent and Networked Objects	面向模式的软件体系结构 卷2:用于并发和网络化对象的模式	Douglas Schmidt
1.4.2	Pattern-Oriented Software Architecture, Vol III: Patterns for Resource Management	面向模式的体系结构 卷3: 资源管理模式	Michael Kircher
1.4.2	Server Component Patterns	服务器组件模式——EJB描述的组件结构	Markus Volter
1.4.2	DesignPatterns: Elements of Reusable Object-Oriented Software	设计模式:可复用面向对象软件的基础	Gamma/Helm/Johnson /Vlissides
1.4.2	Patterns of Enterprise Application Architecture	企业应用架构模式	Martin Fowler
1.4.2	Software Architecture: Organizational Principles and Patterns	软件架构: 组织原则与模式	David Dikel
1.4.2	The UML Profile for Framework Architectures	框架体系结构的UML档案	Marcus Fontoura
1.4.2	Software Architect's Profession: An Introduction	软件架构师职业导读	Marc Sewell
1.4.2	Aspect-Oriented Programming With AspectJ	面向方面编程	Ivan Kiselev
1.4.2	Aspect-Oriented Software Development	面向方面的软件开发	Robert E. Filman
1.4.2	The Art of Software Architecture	软件体系结构的艺术	Stephen T. Albin
1.4.2	Object-Oriented Reengineering Patterns	软件再造: 面向对象的软件再工程模式	Serge Demeyer
1.4.2	Model Driven Architecture with Executable UML	MDA与可执行UML	Chris Raistrick
1.4.3	Building J2EE Applications With The Rational Unified Process	用RUP构建J2EE 应用程序	Peter Eeles
1.4.3	Programming from Specifications	从规范出发的程序设计	Carroll Morgan
1.4.4	Testing IT: An Off-the-Shelf SoftwareTesting Process	实用软件测试过程	John Watkins
1.4.4	Lessons Learned in Software Testing	软件测试经验与教训	Cem Kaner
1.4.4	Testing Computer Software: The Bestselling Software Testing Book Of All Time, 2E	计算机软件测试(原书第2版)	Cem Kaner
1.4.4	Software Testing in the Real World: Improving the Process	软件测试过程改进	Edward Kit
1.4.4	Effective Methods for Software Testing, 2E	软件测试的有效方法(原书第2版)	William E. Perry
1.4.4	Beta Testing for Better Software	软件Beta测试	Michael R. Fine
1.4.4	A Practical Guide to Testing Object-Oriented Software	面向对象的软件测试	John D. McGregor
1.4.4	Managing the Testing Process, 2E	软件测试过程管理(原书第2版)	Rex Black
1.4.4	Software Testing: A Craftsman's Approach, 2E	软件测试(原书第2版)	Paul C. Jorgensen
1.4.4	Just Enough Software Test Automation	软件测试自动化	Daniel J. Mosley
1.4.4	The Craft of Software Testing: Subsystem Testing, Including Object-Based and Object-Oriented Testing	软件子系统测试	Brian Marick
1.4.4	The Web Testing Companion: The Insider's Guide to Efficient and Effective Tests	Web测试指南	Lydia Ash
1.4.4	The Art of Software Testing , 2E	软件测试的艺术（原书第2版）	Glenford J. Myers
1.5	Java Tools for Extreme Programming: Mastering Open Source Tools, Including Ant,JUnit, and Cactus	Java极限编程	Richard Hightower
1.5	Agile Software Development Ecosystems	敏捷软件开发生态系统	Tom DeMarco
1.5	Agile Modeling: Effective Practices For eXtreme Programming and The Unified Process	敏捷建模:极限编程和统一过程的有效实践	Scott W. Ambler
1.5	A Practical Guide to Feature-Driven Development	特征驱动开发方法:原理与实践	Steve R. Palmer
1.5	Pair Programming Illuminated	结对编程技术	Laurie Williams
1.5	Agile Management for Software Engineering	软件工程的敏捷管理	David Anderson
1.5	Principles of the Business Rule Approach	业务规则方法原理	Ronald G. Ross

软件工程技术丛书结构图



目 录

第 1 章 导　论

什么是软件过程？软件过程是项目的状态、阶段、方法、技术以及人们用于开发和维护软件相关产品（计划、文档、模型、代码、测试用例、手册等）的实践的集合。考虑图 1-1 的狩猎过程。虽然我确信 X-Files 将最终会提供一个关于为什么穴居人会灭绝的说明，我的理论是，执行不适当的过程，诸如前面所提到的（穴居人狩猎的）过程，是解释他们（穴居人）衰败的一个可能的候选答案。关键在于，人们需要的不仅仅只是一个软件过程，真正需要的是一个被证明能在实践中正常工作、适合精确需要的软件过程。

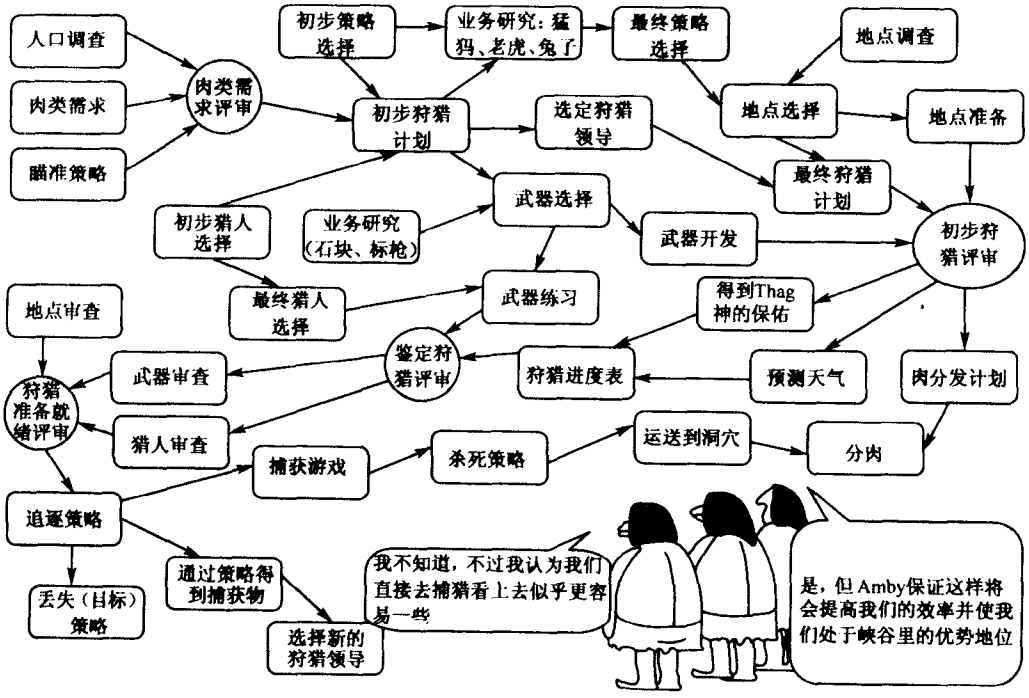


图 1-1　穴居人究竟发生了什么事情（作者并不知道）

为什么需要软件过程？一个有效的软件过程能够增加软件开发的生产率。第一，理解软件开发的基本原理，能够帮助你做出明智的决定。例如，懂得远离 SnakeOil v2.0——一个号称使

软件过程基本部分能自动进行的奇迹工具。第二，能标准化投入，提高复用和项目小组之间的一致性。第三，提供一个向软件组织引入诸如代码审查、配置管理、变更控制、架构建模等业界最佳实践的机会。

在好几个方面，一个有效的软件过程既能提高一个组织的维护和支持工作，也包括产品化工作。首先，它将定义如何管理变更及适当地分配维护变更到将来发布的软件版本，简化变更过程。其次，它将首先定义如何平稳地把软件转移到运行和支持以及运行和支持是如何进行的。没有有效的运行和支持过程，软件将很快变成搁置品。

一个有效的软件过程将会同时考虑开发和生产的需要。

为什么采用现有的软件过程，或者用新技术提高现有的过程？事实是，软件变得越来越复杂，并且没有一个有效的方法进行软件开发和维护，软件达到预期质量的可能性只会更低。不只是软件变得更加复杂，你也被要求同时创建多个软件。大多数机构都普遍有多个项目同时在开发并且同时另有多个在生产阶段——这些项目都需要有效的管理。而且，软件产业处于危机之中；我们仍旧在简单地从两位数年转移到四位数年（y2k）问题，世界范围 6 000 亿美元的价格标签的“小”问题上眩晕不止。我们构建的软件的性质正在发生改变，从简单的 20 世纪 70 年代的用结构技术连接的批处理系统，到以面向对象和基于构件技术为目标的交互的、国际化的、用户友好的、7×24 的、高交易的、高可用性的在线系统。当开发软件的时候，你会被要求增加要交付的系统的质量，并且尽可能地复用以使软件开发更迅速和更廉价。如果不能有效组织和管理你的团队，对于一个高要求的订单，要想做到这点几乎是不可能的。一个软件过程可以提供做到这些的基本帮助。

软件正变得更加复杂，而不是更简单。

1.1 统一过程

统一过程是 Rational 公司（Kruchten，1999）最新的一项努力成果，该公司所引入的统一建模语言（UML）已经成为业界标准的建模符号。统一过程的核心是对象工厂过程，它是 Rational 公司几年前和 Ivar Jacobson 的对象工厂组织合并时获得的产品和服务中的一个。Rational 公司用其自己的过程增强了对象工厂，并形成了于 1998 年 12 月发布的统一过程的最初版本（5.0 版）。

图 1-2 显示了最初的由 4 个连续阶段和 9 个核心工作流所组成的统一过程的生命周期。沿着该图的底部可以看出贯穿整个统一过程的所有的开发周期都应该以迭代的形式组织。基本的思路是：工作组以迭代的方式工作在恰当的工作流中，以便各次迭代结束时都可产生和用户方共同工作的内部可执行产品。这样，通过增加与用户的交流降低了项目的风险。另一个内置于统一过程中降低风险的技术是应该在各阶段末尾做出“继续/中止”的决策，如果项目将会失败，你一定想尽早中止它。当然，最重要的决策点通常是初始和细化阶段的末期（到构造阶段的末期再取消项目就为时太晚了）。对于大型关键项目的失败率达到 80%～90%以上的一个产

业（Jones，1996），具备这一思路是非常重要的。

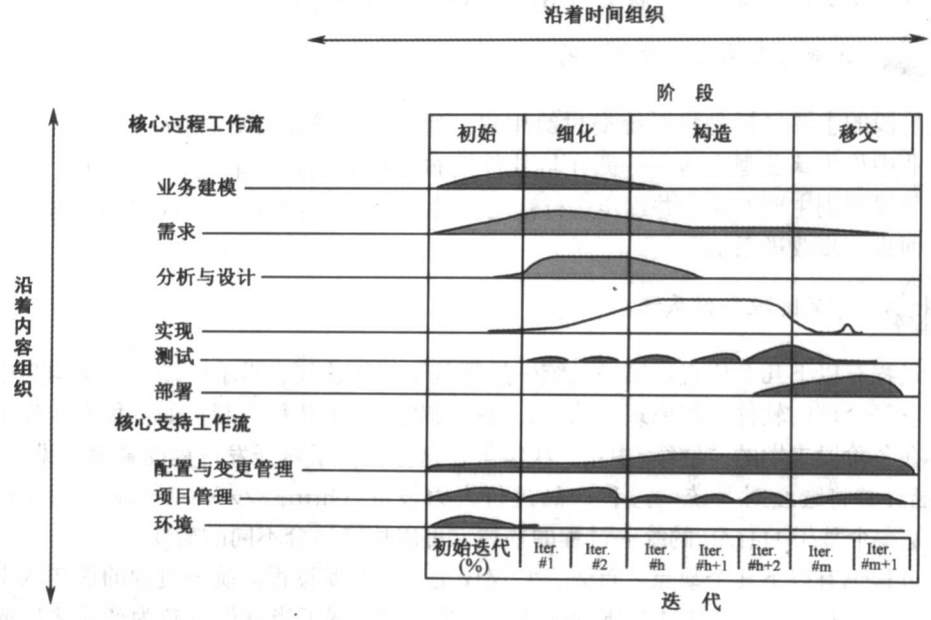


图 1-2 统一过程的初始生命周期

初始阶段主要用于定义系统的项目范围及业务用例。在本阶段要确定软件的最初用例并简要描述关键的用例。用例是业界定义系统功能需求的一项标准技术。由于用例所关注的是系统对于用户的价值而不是对于产品特征的价值，因此，它们比传统的需求文档显著地提高了生产率。基本的项目管理文档是在初始阶段开始的，包括最初的风险评价、估算及项目进度等。正如人们所预想的，本阶段的关键任务涉及业务建模和需求工程以及环境的最初定义，包括工具的选择和过程的定制等。

初始阶段定义项目范围和业务用例。

细化阶段，是本书的主题，关注问题域的详细分析以及项目基础架构的定义。由于用例并不能充分定义所有的需求，所以在本书的第 4 章（就是描述需求工作流的那一章），一个移交的补充规约被定义，该定义用来描述系统的非功能性需求。用于构造阶段的详细的项目计划也是在这个阶段制定的，详细项目计划的基础是初始阶段的管理文档。

细化阶段定义系统基础架构。

构造阶段用于进行系统详细设计并同时生成相应的源代码。此阶段的目标是生产交付给用户的软件及相应的支持文档。项目组所犯的一个最常见的错误就是将重点放在这个阶段，由于机构没有为前两个阶段提供足够的资源投入，缺乏成功开发满足用户需求的软件的基础，因此

通常这个错误对项目有害。在初始和细化阶段，应该为问题域和解决域的理解提供必要的资源。此阶段的主要目标就是生产移交给用户的软件和支持文档。

完成将在构造阶段部署的系统。

移交阶段的主要目标是将系统交付给用户。通常会先给用户软件的一个 beta 版本——在大多数企业中称为试验型发布——即在将软件发布给所有用户之前先由一小部分用户使用系统。在此阶段识别主要的缺陷并逐步进行修复。最后对投入进行评估，确定是否需要下一个开发周期以便进一步改进系统。

移交阶段交付系统。

统一过程有以下几个优点。第一，统一过程基于例如迭代、需求驱动、以架构为中心的增量软件开发等合理的软件工程原理。第二，统一过程提供了几种机制，如：每次迭代末期的工作原型，在各阶段末期的“继续/中止”决策等，这些提供了对开发过程的管理可见度。第三，Rational 已经并将继续对其统一过程产品进行大力投资（http://www.rational.com/products/rup）。这是一个基于 HTML 的统一过程的描述，可以用来适合不同的需要。

统一过程也有以下几个缺点。首先，它仅仅是一个开发过程。统一过程的最初版本并没有覆盖整个软件过程。正如在图 1-2 所看到的，很明显它遗漏了当软件发布为产品之后的运行和支持的概念。其次，统一过程并没有明确地支持多项目基础开发，例如组织、企业级架构建模，该部分在第 5 章讨论（该章描述基础管理工作流），损失了组织内进行大范围复用的机会。最后，生命周期的迭代特性对于许多有经验的开发者来说是全新的，因此接受它们有一定难度，而且图 1-2 所示的生命周期图对此并没有提供帮助。

1.2 超越统一过程

因此，如何增强统一过程以使它适合实际的开发需要？第一，你需要以对过程的需求开始。——一个好的开始是能力成熟度模型（CMM）。第二，你应当寻找其他相竞争的过程，如 OPEN 过程（Graham、Henderson－Sellers 和 Younessi，1997）和面向对象软件过程的过程模式（Ambler 1998b，Ambler 1999），并从这些过程中找到那些能用来复用的特征。最后，应该基于所学表示一个增强的生命周期并且用已经证明的最佳实践来支持该生命周期。

1.2.1 能力成熟度模型

卡耐基梅隆大学（Carnegie Mellon University）软件工程学院（The Software Engineering Institute，SEI）（http://www.sei.cmu.edu）提出了能力成熟度模型——能够为用于大型复杂软件产品的开发过程定义一个框架（SEI，1995）。CMM 定义了 5 个成熟度级别，这是连接在一起的且达到组织能成就的成熟软件过程的几个发展阶梯。根据 CMM，达到一个特定的成熟度级别，一个组织必须满足和制度化该级别和该级别前的所有关键过程域（KPA）。每个关键过程域，即使是高级别的关键过程域，都定义了软件过程必须展示的明确特征。例如，那些关