HZ BOOKS
华章教育

国家示范性软件学院系列教材

韩万江 姜立新 等编著 宋茂强 审

软件工程案例教程

软件项目开发实践

第2版

Software Engineering
A Case Study Approach

机械工业出版社
China Machine Press

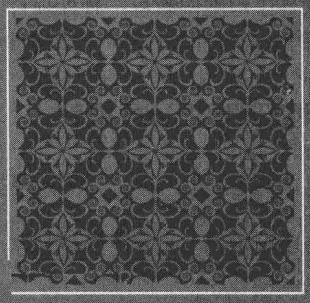
国家示范性软件学院系列教材

韩万江 姜立新 等编著 宋茂强 审

软件工程案例教程

软件项目开发实践

第2版



Software Engineering
A Case Study Approach

机械工业出版社
China Machine Press

本教程以案例的形式讲述了软件工程中软件项目开发的实践过程，全面涵盖软件项目开发中需求分析、概要设计、详细设计、编码、测试、提交以及运行维护等过程中涉及的理论、方法、技术、提交的产品和文档等。本书注重实效，系统、全面，通过贯穿始终的案例的讲述，让学习者在短时间内掌握软件项目开发的基本知识、基本过程，并有效提高实践能力。

本书共分九章，第 1～2 章介绍软件工程的基本概念以及软件工程的主要技术，第 3～9 章系统地讲述软件项目开发的各个过程。本书注重理论与实际的结合，引导学生通过软件开发理论和案例的学习，深刻理解软件工程的实质，为以后的软件工程实践打下基础。

本书既适合作为高等院校计算机及相关专业软件工程、软件测试课程的教材，也适合作为广大软件技术人员的培训教程，同时可以作为软件开发人员在工作及学习中的技术参考书。

图书在版编目（CIP）数据

软件工程案例教程：软件项目开发实践 / 韩万江等编著. —2 版. —北京：机械工业出版社，2011.7
（国家示范性软件学院系列教材）

ISBN 978-7-111-35318-8

Ⅰ. 软…　Ⅱ. 韩…　Ⅲ. 软件工程—案例—高等学校—教材　Ⅳ. TP311.5

中国版本图书馆 CIP 数据核字（2011）第 138411 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
北京京师印务有限公司印刷
2011 年 10 月第 2 版第 1 次印刷
185mm×260mm · 17.75 印张
标准书号：ISBN 978-7-111-35318-8
定价：35.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88378991；88361066
购书热线：（010）68326294；88379649；68995259
投稿热线：（010）88379604
读者信箱：hzjsj@hzbook.com

前 言

本书第 1 版出版后深受广大读者的好评，同时也收到了很多读者建议，再加上本人多年教学和项目实践的新经验，感到有必要对第 1 版教程进行升级改进。本书在第 1 版的基础上，根据软件工程新技术的发展，总结了软件开发实践过程和教学过程的经验教训，经过 2 年的酝酿和更新，最后完成了第 2 版的修订。

本书第 2 版不仅进一步完善了很多软件开发技术和技巧，而且更换了第 1 版的所有案例说明，是一本系统的、有针对性的、实效性强的教材，对于从事软件项目开发以及希望学习软件开发的人员都会起到非常好的借鉴作用。

参与本书编写的有韩万江、姜立新、郑伟、陈力、王晓琼、杨元民、岳鹏、郭士容、岳好等。宋茂强教授对本书进行了多次审核，在此表示感谢！

由于作者水平有限，书中难免有疏漏之处，诚请各位读者批评指正，并希望将你们在实际工作中如何运用本书的体会告诉我，以便我在以后的版本修订中进一步完善。我的 Email 是：casey_han@263.net。

韩万江

2011.1 于北京

目 录

第 1 章

■ 软件工程概述

软件（software）是计算机系统中与硬件（hardware）相互依存的另一部分，它包括程序（program）、相关数据（data）及其说明文档（document）。其中程序是按照事先设计的功能和性能要求执行的指令序列；数据是程序能正常操纵信息的数据结构；文档是与程序开发维护和使用有关的各种图文资料。

软件工程（Software Engineering，SE）是针对软件这一具有特殊性质的产品的工程化方法，它涵盖了软件生存周期的所有阶段，并提供了一整套工程化的方法来指导软件人员的工作。

1.1 软件工程的背景

近年来，计算机软件已经成为现代科学研究和解决工程问题的基础，是管理部门、生产部门、服务行业中的关键因素，它已渗透到了各个领域，成为当今世界不可缺少的一部分。展望将来，软件仍将是驱动事情取得新进展的动力。学习研究工程化的软件开发方法，使开发过程更加规范，变得越来越重要。

20 世纪中期软件产业从零开始起步，并迅速发展成为推动人类社会发展的龙头产业。随着信息产业的发展，软件对人类社会越来越重要，人们对软件的认识也经历了一个由浅到深的过程。

第一个写软件的人是 Ada（Augusta Ada Lovelace），在 19 世纪 60 年代她尝试为 Babbage（Charles Babbage）的机械式计算机写软件，尽管失败了，但她永远载入了计算机发展的史册。20 世纪 50 年代，软件伴随着第一台电子计算机的问世诞生了，以写软件为职业的人也开始出现，他们多是经过训练的数学家和电子工程师。20 世纪 60 年代美国大学里开始出现计算机专业，教人们写软件。

软件发展的历史大致可以分为如下的三个阶段：

第一个阶段是 20 世纪 50～60 年代，即程序设计阶段，基本采用个体手工劳动的生产方式。这个时期的程序是为特定目的而编制的，软件的通用性很有限，往往带有强烈的个人色彩。早期的软件开发也没有什么系统的方法可以遵循，软件设计是在某个人的头脑中完成的一个隐藏的过程，而且，除了源代码往往没有软件说明书等文档。因此这个时期尚无软件的概念，基本上只有程序、程序设计概念；不重视程序设计方法；而且设计的程序主要是用于

科学计算，规模很小，采用简单的工具，基本上采用低级语言；硬件的存储容量小，运行可靠性差。

第二阶段是20世纪60～70年代，即软件设计阶段，采用小组合作生产方式。这个时期软件开始作为一种产品被广泛使用，出现了“软件作坊”专职应别人的要求写软件。这个阶段的程序设计基本采用高级语言开发工具，人们开始提出结构化方法；硬件的速度、容量、工作可靠性有明显提高，而且硬件的价格降低；人们开始使用产品软件（可购买），从而建立了软件的概念。但是开发技术没有新的突破，软件开发的方法基本上仍然沿用早期的个体化软件开发方式。随着软件数量的急剧膨胀，软件需求日趋复杂，维护的难度越来越大，开发成本日益高涨，此时的开发技术已不适应规模大、结构复杂的软件开发，失败的项目越来越多。

第三个阶段从20世纪70年代开始，即软件工程时代，采用工程化的生产方式。这个阶段的硬件向超高速、大容量、微型化以及网络化方向发展；第三、四代语言出现；数据库、开发工具、开发环境、网络 、分布式、面向对象技术等工具方法都得到应用；软件开发技术有很大进步，但未能获得突破性进展，软件开发技术的进步一直未能满足发展的要求。这个时期很多的软件项目开发时间大大超出了规划的时间表，一些项目导致了财产的流失，甚至导致了人员伤亡。同时，一些复杂的、大型的软件开发项目提出来了，软件开发的难度越来越大，在软件开发中遇到的问题找不到解决的办法，使问题积累起来，形成了尖锐的矛盾，失败的软件开发项目屡见不鲜，因而导致了软件危机。

软件危机指的是计算机软件开发和维护过程中所遇到的一系列严重问题。概括来说，软件危机包含两方面的问题：一是如何开发软件，以满足不断增长、日趋复杂的需求；二是如何维护数量不断膨胀的软件产品。落后的软件生产方式无法满足迅速增长的计算机软件需求，从而导致软件开发与维护过程中出现一系列严重问题。

最为突出的例子是美国IBM公司于1963～1966年开发的IBM360系列机的操作系统。该项目的负责人Fred Brooks在总结该项目时无比沉痛地说：“……正像一只逃亡的野兽落到泥潭中做垂死挣扎，越是挣扎，陷得越深，最后无法逃脱灭顶的灾难……程序设计工作正像这样一个泥潭……一批批程序员被迫在泥潭中拼命挣扎……谁也没有料到问题竟会陷入这样的困境……”IBM360操作系统的历史教训已成为软件开发项目中的典型事例被记入史册。

具体地说，软件危机主要有以下表现：

1）对软件开发成本和进度的估计常常不准确，开发成本超出预算，项目经常延期，无法按时完成任务。

2）开发的软件不能满足用户要求。

3）软件产品的质量低。

4）开发的软件可维护性差。

5）软件通常没有适当的文档资料。

6）软件的成本不断提高。

7）软件开发生产率的提高赶不上硬件的发展和人们需求的增长。

软件危机的原因，一方面与软件本身的特点有关；另一方面与软件开发和维护的方法不正确有关。软件危机的产生，迫使人们不得不研究、改变软件开发的技术手段和管理方法。从此软件生产进入软件工程时代。

1968年北大西洋公约组织的计算机科学家在联邦德国召开的国际学术会议上第一次提出了“软件危机”这个名词，同时讨论和制定了摆脱“软件危机”的对策。在那次会议上第

一次提出了软件工程（Software Engineering）这个概念，从此一门新兴的工程学科——软件工程学——应运而生。

“软件工程”的概念是为了有效地控制软件危机的发生而提出来的，它的中心目标就是把软件作为一种物理的工业产品来开发，要求“采用工程化的原理与方法对软件进行计划、开发和维护”。软件工程是一门旨在开发满足用户需求、及时交付、不超过预算和无故障的软件的学科，它的主要对象是大型软件，最终目的是摆脱手工生产软件的状况，逐步实现软件开发和维护的自动化。

从微观上看，软件危机的特征表现在完工日期一再拖后、经费一再超支，甚至工程最终宣告失败等方面；而从宏观上看，软件危机的实质是软件产品的供应赶不上需求的增长。

虽然“软件危机”还没得到彻底解决，但自从软件工程概念提出以来，经过几十年的研究与实践，在软件开发方法和技术方面已经有了很大的进步。尤其应该指出的是，人们逐渐认识到，在软件开发中最关键的问题是软件开发组织不能很好地定义和管理其软件过程，从而使一些好的开发方法和技术都起不到应有的作用。也就是说，在没有很好定义和管理软件过程的软件开发中，开发组织不可能在好的软件方法和工具中获益。

1.2 软件工程知识体系

“工程”是科学和数学的某种应用，通过这一应用，使自然界的物质和能源的特性能够通过各种结构、机器、产品、系统和过程，成为对人类有用的东西。因而，“软件工程”就是科学和数学的某种应用，通过这一应用，使计算机设备的能力借助于计算机程序、过程和有关文档成为对人类有用的东西。软件工程是运用现代科学技术知识来设计并构造计算机程序及开发、运行和维护这些程序所必需的相关文件资料。

软件工程的成果是为软件设计和开发人员提供思想方法和工具，而软件开发是一项需要良好组织、严密管理且各方面人员配合协作的复杂工作。软件工程正是指导这项工作的一门科学。软件工程在过去一段时间内已经取得了长足的进展，在软件的开发和应用中起到了积极的作用。随着软件开发的深入以及各种技术的不断创新和软件产业的形成，人们越来越意识到软件过程管理的重要性，并且传统的软件工程理论也随着人们的开发实践不断完善发展。

高质量的软件工程可以保证生产出高质量的、用户满意的软件产品。但是，对软件工程的界定，总是存在一定的差异。软件工程应该包括哪些知识？这里我们引用 IEEE 在软件工程知识体系指南（Guide to the Software Engineering Body of Knowledge，SWEBOK）中对软件工程的定义：1）软件开发、实施、维护的系统化、规范化、质量化方法的应用，也就是软件的应用工程；2）对上述方法的研究。

1998 年，美国联邦航空管理局在启动一个旨在提高技术和管理人员的软件工程能力的项目时发现，他们找不到软件工程师应该具备的公认的知识结构，于是他们向美国联邦政府提出了关于开发“软件工程知识体系指南”的项目建议。美国 Embry-Riddle 航空大学计算与数学系的 Thomas B. Hilburn 教授接手了该研究项目，并于 1999 年 4 月完成了《软件工程知识本体结构》的报告。该报告发布后迅速引起软件工程界、教育界和一些政府对建立软件工程本体知识结构的兴趣。很快人们普遍接受了这样的认识：建立软件工程知识体系的结构是确立软件工程专业至关重要的一步，如果没有一个得到共识的软件工程知识结构，将无法验证软件工程师的资格，无法设置相应的课程，或者无法建立对相应课程进行认可的判断准则。

对建立权威的软件工程知识结构的需求迅速在世界各地反映出来。1999 年 5 月，ISO 和 IEC 的第一联合技术委员会（ISO/IEC JTC1）为顺应这种需求，立即启动了标准化项目——“软件工程知识体系指南”（http : // www. swebok. org/）。美国电子电气工程师学会与美国计算机联合会联合建立的软件工程协调委员会（SECC）、加拿大魁北克大学以及美国 MITRE 公司（与美国 SEI 共同开发 SW-CMM 的软件工程咨询公司）等共同承担了 ISO/IEC JTC1 “SWEBOK 指南”项目任务。

IEEE 的 SWEBOK 中界定了软件工程的 10 个知识领域（Knowledge Area，KS）：软件需求（software requirements）、软件设计（software design）、软件构建（software construction）、软件测试（software testing）、软件维护（software maintenance）、软件配置管理（software configuration management）、软件工程管理（software engineering management）、软件工程过程（software engineering process）、软件工程工具和方法（software engineering tools and methods）、软件质量（software quality）。这 10 个知识领域的每个知识领域还包括很多子领域。

1.3 软件工程的三段论

在不断探索软件工程的原理、技术和方法的过程中，人们研究和借鉴了工程学的某些原理和方法，并形成了软件工程学。软件工程的目标是提高软件的质量与生产率，最终实现软件的工业化生产。既然软件工程是“工程”，那么我们从工程的角度看一下软件项目的实施过程，如图 1-1 所示。

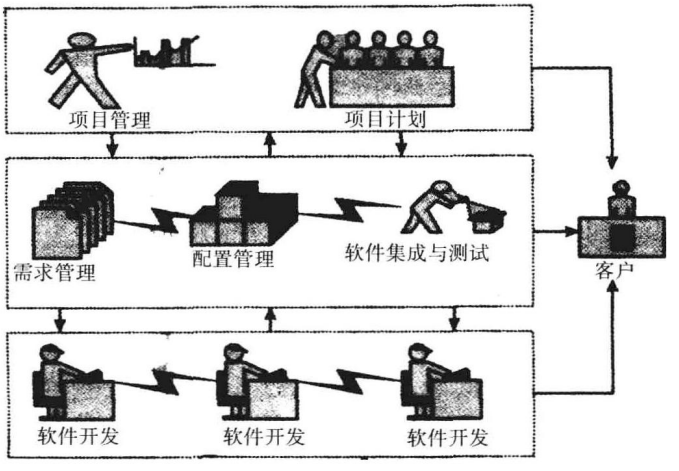
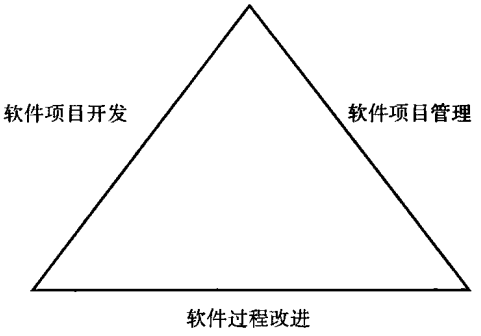


图 1-1 工程化软件开发

客户的需求启动了一个软件项目，为此我们需要先规划这个项目，即完成项目计划，然后根据这个项目计划实施项目。项目实施的依据是需求，这个需求类似工程项目的图纸，开发人员按照这个图纸生产软件，即设计、编码。在开发生产线上，将开发过程的半成品通过配置管理来存储和管理，然后进行必要的集成和测试，直到最后提交给客户。在整个开发过程中需要进行项目跟踪管理。软件工程活动是“生产一个最终满足需求且达到工程目标的软件产品所需要的步骤”。这些活动主要包括开发类活动、管理类活动和过程改进类活动，这里将它定义为“软件工程的三段论”，或者“软件工程的三线索”。一段论是“软件项目管理”，

二段论是“软件项目开发”，三段论是“软件过程改进”。这个三段论可以用一个三角形表示，如图 1-2 所示，它们类似于相互支撑的三角形的三条边。我们知道三角形是最稳定的，要保证三角形的稳定性，三角形的三条边必不可少，而且要保持一定的相互关系。



其中：

“软件项目开发”是软件人员生产软件的过程，例如需求分析、设计、编码、测试等，相当于生产线上的生产过程。

“软件项目管理”是项目管理者规划软件开发、控制软件开发的过程，相当于生产线上的管理过程，管理过程是伴随开发过程进行的过程。

“软件过程改进”相当于对软件开发过程和软件管理过程的“工艺流程”进行管理和改进，如果没有好的工艺则生产不出好的产品，它包括对开发过程和管理过程的定义和改进。

图 1-2 软件工程的三个线索

为了保证软件管理、软件开发过程的有效性，应该保证上述过程的高质量和过程的持续改进。

让软件工程成为真正的工程，就需要软件项目的开发、管理、过程改进等方面规范化、工程化、工艺化、机械化。

软件开发过程中脑力活动的“不可见性”大大增加了过程管理的困难。因此软件工程管理中的一项指导思想就是千方百计地使这些过程变为“可见的”、事后可查的记录。只有从一开始就在开发过程中严格贯彻质量管理，软件产品的质量才会有保证。否则，开发工作一旦进行到后期，无论怎样测试和补漏洞，都无济于事。

1.4 软件工程模型

一个软件项目的基本流程和关联关系如图 1-3 所示。

按照项目的初始、计划、执行、控制、结束五个阶段，可以总结出软件工程的相关过程如下：

1）初始阶段的过程。包括：立项，供应商选择，合同签署。

2）计划阶段的过程。包括：范围计划，时间计划，成本计划，质量计划，风险计划，沟通计划，人力资源计划，合同计划，配置管理计划。

3）执行阶段的过程。包括：需求分析，概要设计，详细设计，编码，单元测试，集成测试，系统测试，项目验收，项目维护。

4）控制阶段的过程。包括：范围计划控制，时间计划控制，成本计划控制，质量计划控制，风险计划控制，沟通计划控制，人力资源计划控制，合同计划控制，配置管理计划控制。

5）结束阶段的过程。包括：合同结束，项目总结。

这些过程活动分布在软件工程的软件项目管理、软件项目开发、软件过程改进三条线索中。

“软件工程”与其他行业的工程有所区别，其模式或者标准很难统一为一个模型，所以，软件工程的模型是弹性的，标准是一个相对的标准。按照软件项目的初始、计划，执行、控制、结束五个阶段，我们建立一个基于过程元素的软件工程模型，如图 1-4 所示。模型用虚

线分割成两部分，第一部分是过程构建和过程改进，其中的过程库是软件项目的标准过程积累；第二部分为基于过程的软件项目实施过程。

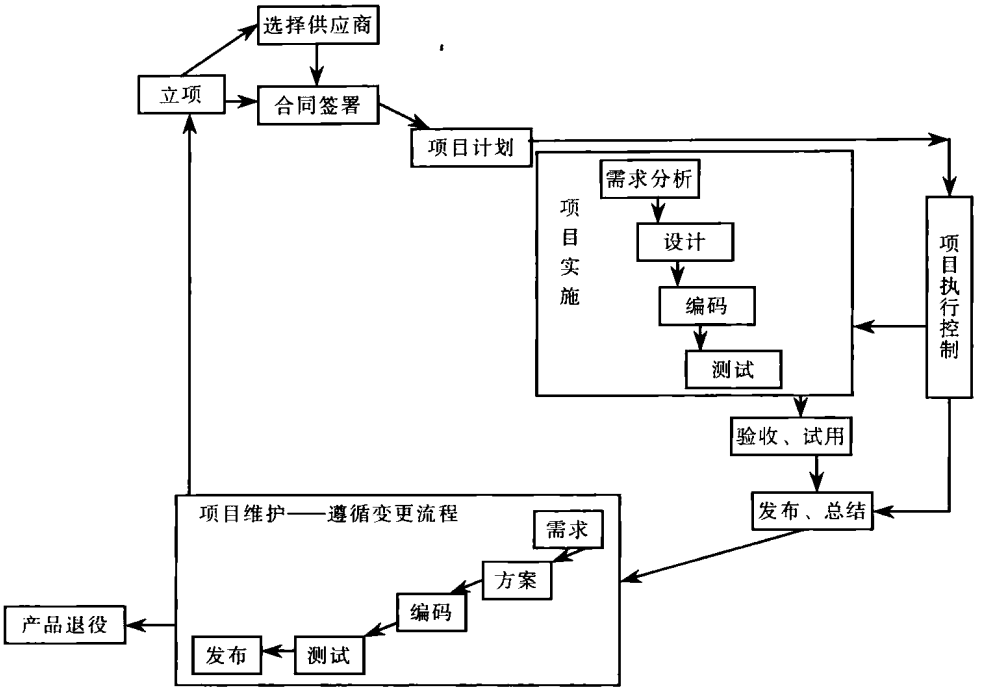


图 1-3 软件工程各个阶段过程之间的关系

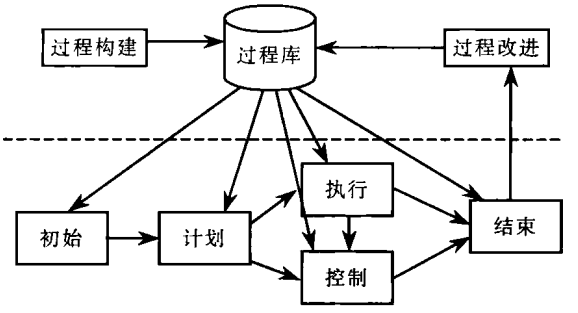


图 1-4 软件工程模型

我们将图 1-4 中的第二部分展开为含有过程的流程模式，其中，五个阶段中的每个过程用“⬭”表示，类似于 J2EE 中的 Java Bean。这些过程元素存储在过程库中，这样就形成了一个基于过程的弹性软件工程模型，如图 1-5 所示。

这个弹性软件工程模型包含了软件工程的开发、管理、过程改进三个方面，对于一个具体的项目，可以选择软件项目开发过程组和软件项目管理过程组中的过程进行组合来完成项目。这里的“弹性”是指可以根据项目的需要选择过程，而软件过程又可以根据需要进行组合。也就是说一个项目可以根据具体情况进行排列和取舍，形成特定项目的模型。

本书的重点在软件开发环节，该环节中的活动也是参与软件项目的软件人员的主要活动，

即主要讲述执行阶段的过程。

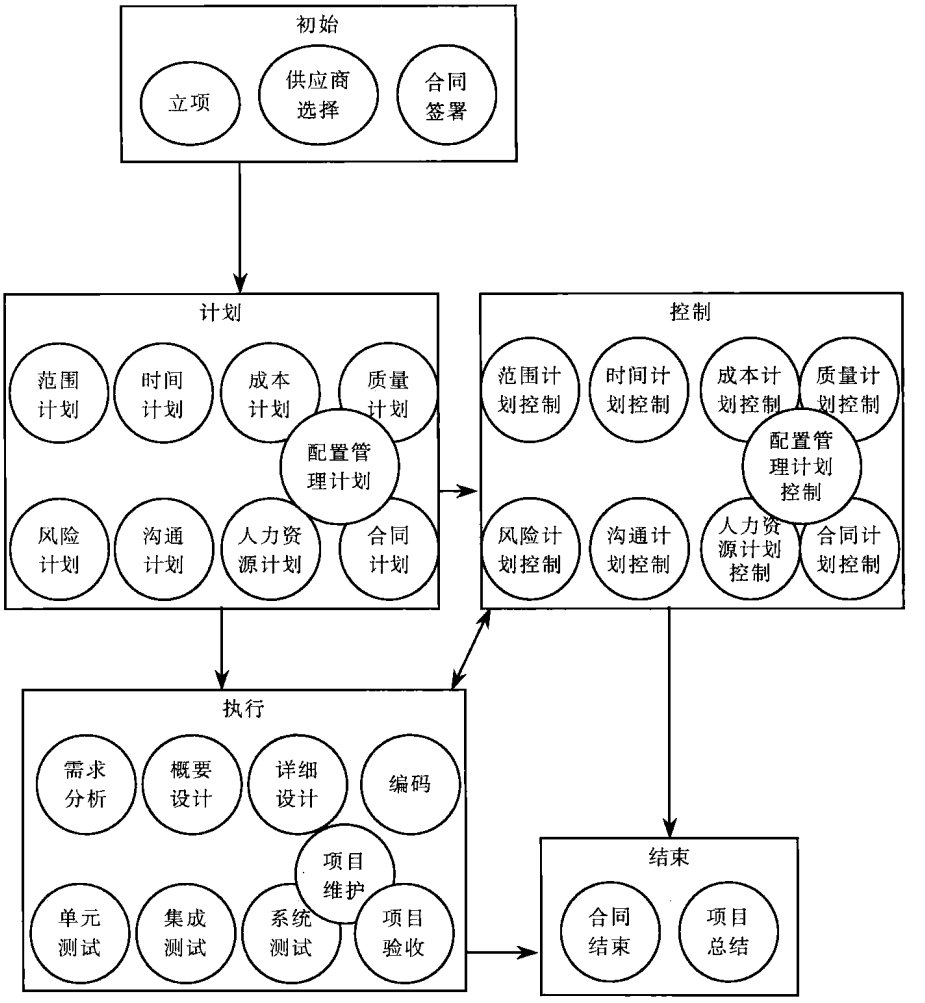


图 1-5 基于过程元素的弹性软件工程模型

1.4.1 软件项目开发路线图

软件项目开发过程是软件工程的核心过程，通过这个生产线可以生产出用户满意的产品。软件工程提供了一整套工程化的方法来指导软件人员的工作。

图 1-6 是软件项目开发的路线图，这个路线图展示了从需求开始的软件开发的基本工艺流程。需求分析是项目开发的基础；概要设计为软件需求提供实施方案；详细设计是对概要设计的细化，它为编码提供依据；编码是软件的具体实现；测试是验证这个软件的正确性；提交（发布）是将软件提交给使用者；维护指软件在使用过程中需要维护。

图 1-6 软件项目开发路线图

如同传统工程的生产线上有很多工序（每道工序都有明确的规程），软件生产线上的工序主要包括需求分析、概要设计、详细设计、编码、测试、提交、维护等。采用一定的流程将各个环节连接起来，并用规范的方式操作全过程，如同工厂的生产线，这样就形成了软件工程模型，也称为软件开发生存期模型，即软件工程模型。

软件开发过程是随着开发技术的演化而改进的。从早期的瀑布（Waterfall）开发模型到后来出现的螺旋（Spiral）迭代开发模型，再到最近开始兴起的敏捷（Agile）开发方法，展示了不同的时代软件产业对于开发过程的不同认识，以及对于不同类型项目的理解方法。

没有规则的软件开发过程带来的只可能是无法预料的结果，这是我们在经历了一次次的项目失败之后逐渐领悟到的道理。随着软件项目的规模不断加大，参与人员不断增多，对规范性的要求愈加严格，基于软件项目管理的、工程化的软件开发时代已经来临。

1.4.2 软件项目管理路线图

美国项目管理专家 James P. Lewis 说：项目是一次性、多任务的工作，具有明确规定的开始和结束日期，特定的工作范围和预算，以及要达到的特定性能水平。

项目经理首先必须弄明白什么是项目。项目涉及 4 个要素：预期的绩效，费用（成本），时间进度，工作范围。这 4 个要素相互关联、相互影响。

例如你去采购商品，原来想好要采购很多东西，回来却发现很多东西忘了买，为避免这种问题，当你再出去采购的时候会在一张纸上记录下所有需要购买的东西，即采购清单，你可以“完成一个采购项，在采购清单上打一个勾”，如果清单中每项都打勾了，就表示所有的采购任务完成了，这个采购清单就是你的计划，你通过不断在采购清单上打勾来控制“采购”这个项目很好地完成。再举一个我们熟悉的例子，假如让你负责一个聚会活动，那么你就是这个“聚会活动”的项目经理，如何使这个项目成功就是你的任务。为了很好地完成这个任务，你需要知道聚会中有哪些活动、费用如何、如何安排时间等，在聚会进行过程中，你需要控制哪些活动完成了，哪些没有完成，进度进展的如何、费用花费的如何等。如果经历几次没有计划的聚会后，你觉得必须要事前制定好一个节目单——相当于一个计划，记录有哪些活动，安排时间，控制花费等。

同理，软件项目管理也是这样，软件项目管理就是如何管理好软件项目的范围、时间、成本，也就是管理好项目的内容、花费的时间（进度）以及花费的代价（规模成本），其他相关的事情都是围绕这三件事情进行的。为此需要制定一个好的项目计划，然后控制好这个计划，即软件项目管理的实质是软件项目计划的编制和项目计划的跟踪控制，如图 1-7 所示。计划与跟踪控制是相辅相成的关系：计划是项目成功实施的指南和跟踪控制的依据；而跟踪控制又用来保证项目计划的成功执行。

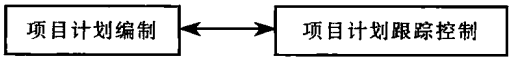


图 1-7 软件项目管理的实质

实际上，要做到项目计划切合实际是一个非常高的要求，需要对项目的需求进行详细分析，根据项目的实际规模制订合理的计划。计划的内容包括进度安排、资源调配、经费使用等，为了降低风险，还要进行必要的风险分析与制定风险管理计划，同时要对自己的开发能力有非常准确的了解，制定切实可行的质量计划和配置管理计划等。这来源于项目经理的职业技能和实践经验的持续积累。

制定了合适的项目计划之后，才能进行有效的跟踪与监督。当发现项目计划的实际执行情况与计划不符的时候要进行适当、及时的调整，确保项目按期、按预算、高质量地完成。

项目成功与否的关键是能不能成功地实施项目管理[⊖]。图 1-8 便是软件项目管理的路线图。

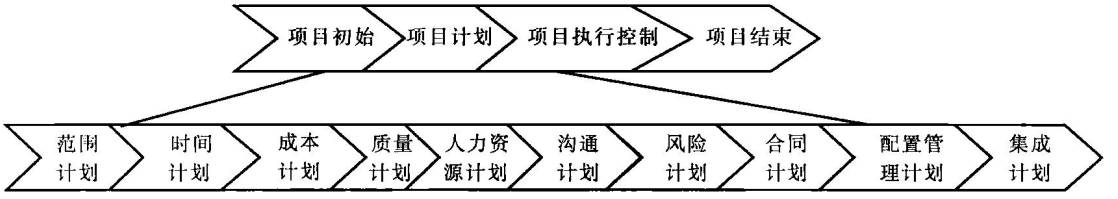


图 1-8 软件项目管理路线图

1.4.3 软件过程改进路线图

自 20 世纪 70 年代软件危机以来，人们不断地展开新方法和新技术的研究与应用，但未取得突破性的进展。直到 20 世纪 80 年代末，人们得出这样一个结论：一个软件组织的软件能力取决于该组织的过程能力。一个软件组织的过程能力越成熟，该组织的软件生产能力就越有保证。

所谓过程，简单来说就是我们做事情的一种固有的方式，我们做任何事情都有过程存在，小到日常生活中的琐事，大到工程项目。对于做一件事，有过经验的人对完成这件事的过程会很了解，他会知道完成这件事需要经历几个步骤，每个步骤都完成什么事，需要什么样的资源、什么样的技术等，因而可以顺利地完成工作；没有经验的人对过程不了解，就会有无从下手的感觉。图 1-9 和图 1-10 可以形象地说明过程在软件开发中的地位。如果项目人员将关注点只放在最终的产品上，如图 1-9 所示，不关注期间的开发过程，那么不同的开发队伍或者个人可能就会采用不同的开发过程，结果导致开发的产品有的质量高，有的质量差，完全依赖个人的素质和能力。

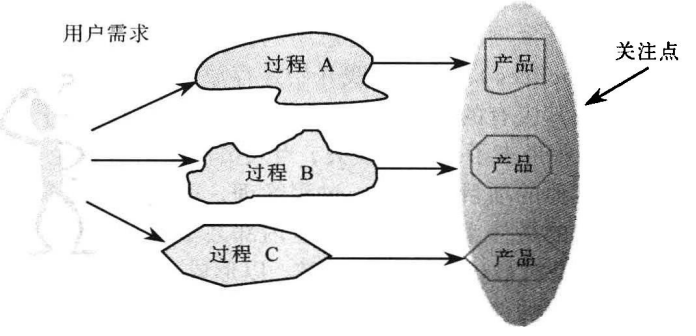


图 1-9 关注开发的结果

反之，如果将项目的关注点放在项目的开发过程，如图 1-10 所示，则不管谁来做，都采用统一的开发过程，也就是说，企业的关注点在过程。经过统一开发过程开发的软件，产品的质量是一样的。可以通过不断提高过程的质量来提高产品的质量，这个过程是公司能力的体现，而不是依赖于个人的。也就是说，产品的质量依赖于企业的过程能力，不依赖于个人能力。

⊖ 有关软件项目管理的详细内容可参考本人所写的《软件项目管理案例教程》(机械工业出版社出版，书号 ISBN978-7-111-26753-9)。

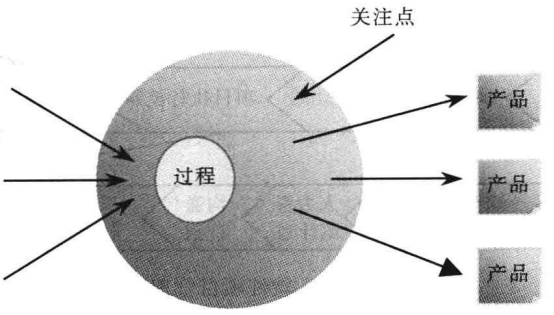


图 1-10 关注开发的过程

对于软件过程的理解，绝对不能简单地理解为软件产品的开发流程，因为我们要管理的并不只是软件产品开发的活动序列，而是软件开发的最佳实践，它包括流程、技术、产品、活动间关系、角色、工具等，是软件开发过程中的各个方面因素的有机结合。因此，在软件过程管理中，首先要进行过程定义，将过程以一种合理的方式描述出来，并建立起企业内部的过程库，使过程成为企业内部可以重用（也称复用）的共享资源。对于过程，要不断地进行改进，以不断地改善和规范过程，帮助提高企业的生产力。

软件过程是极其复杂的过程。软件是由需求驱动的，有了用户的实际需求才会引发一个软件产品的开发。软件产品从需求的出现到最终的产品出现要经历一个复杂的开发过程，软件产品在使用时要根据需求的变更进行不断的修改（这称为软件维护），我们把用于从事软件开发及维护的全部技术、方法、活动、工具以及它们之间的相互变换统称为软件过程。由此可见，软件过程的外延非常大，包含的内容非常多。对于一个软件开发机构来说，做过一个软件项目，无论成功与否，都能够或多或少地从中总结出一些经验。做过的项目越多，经验越丰富，特别是一个成功的开发项目是很值得总结的，从中可以总结出一些完善的过程，我们称之为最佳实践（Best Practices）。最佳实践开始是存放在成功者的头脑中的，很难在企业内部共享和重复利用，发挥其应有的效能。长期以来，这些本应从属于企业的巨大的财富被人们所忽视，这无形中给企业带来了巨大的损失，当人员流动时这种企业的财富也随之流失，并且也使这种财富无法被其他的项目再利用。过程管理，就是对最佳实践进行有效的积累，形成可重复的过程，使我们的最佳实践可以在企业内部共享。过程管理的主要内容包括过程定义与过程改进。过程定义是对最佳实践加以总结，以形成一套稳定的、可重复的软件过程。过程改进是根据实践中对过程的使用情况，对过程中有偏差或不够切合实际的地方进行优化的活动。通过实施过程管理，软件开发机构可以逐步提高其软件过程能力，从根本上提高软件生产能力。

美国卡内基-梅隆大学软件工程研究所（CMU/SEI）主持研究与开发的 CMM/PSP/TSP 技术，为软件工程管理开辟了一条新的途径。PSP（个人软件过程）、TSP（团队软件过程）和 CMM（能力成熟度模型）为软件产业提供了一个集成化的、三维的软件过程改进框架，它们提供了一系列的标准和策略来指导软件组织如何提升软件开发过程的质量和软件组织的能力，而不是给出具体的开发过程的定义，如图 1-11 所示。PSP 注重个人的技能，能够指导软件工程师保证自己的工作质量；TSP 注重团队的高效工作和产品交付能力；CMM 注重组织能力和高质量的产品，它提供了评价组织的能力、识别优先需求改进和追踪改进进展的管理