CMM：软件过程的管理与改进

雷剑文
陈振冲
李明树
著

清华大学出版社

CMM：软件过程的管理与改进

雷剑文 陈振冲 李明树 著

清华大学出版社

(京)新登字 158 号

内 容 简 介

本书将 CMM 中的一些抽象的概念融于实际生活之中，让读者透过生活中熟悉的实例，对 CMM 有一个全面的认识。然后以不同的层面对 CMM 作进一步深入的探讨。并从不同的角度解释了 CMM，即：(1)哲学与实际生活；(2)软件投资者和 CMM 的关系；(3)管理框架；(4)CMM 的数学表达；(5)KPA 与其应用上的理解；(6)介绍国外 CMM 的辅助软件；(7)国外 CMM 网上参考资料。本书的特点在于让不了解软件的人很快看懂，让已经了解 CMM 的读者对软件改进过程有更深一层的理解。为了让读者在很快读完某一章节的同时，得到某一角度的全面的认识，本书加入大量的图表，使每一章节简短、易懂。本书没有假设读者有任何的开发经验，因而无论是公司的投资者、管理人员或编程人员都能很容易透过书中的例子清楚地了解 CMM 的哲学。

本书特别适合于作为软件改进过程、软件工程、信息技术管理、MBA 等专业的教科书，也可以作为改进软件过程的人员，包括评估组成员、软件工程过程组成员以及软件从业人员的参考书。

书　　名：CMM：软件过程的管理与改进
作　　者：雷剑文　陈振冲　李明树　著
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
责任编辑：薛　慧
印 刷 者：清华大学印刷厂
发 行 者：新华书店总店北京发行所
开　　本：787×1092 1/16　**印张**：8.75　**插页**：2　**字数**：162 千字
版　　次：2002 年 9 月第 1 版　2002 年 9 月第 1 次印刷
书　　号：ISBN 7-302-05739-7/TP・3389
印　　数：0001～6000
定　　价：15.00 元

CMM 软件过程的管理与改进

序

CMM(软件能力成熟度模型,Capability Maturity Model)是由美国 SEI (Software Engineering Institute)提出的一套对软件过程的管理、改进与评估的模式。软件项目有别于其他的工程项目,软件是无形的知识产品,不像制造业的产品可以直接地进行检测评估。因而,软件项目的管理相比其他项目的管理有很大的差别。再者,尽管软件是无形的,但它的开发投资费用却往往很高。同时软件项目的失败可能会造成所有资金的损失。为了使得项目成功,就必须改善整个软件组织开发软件的能力,而 CMM 可以在这方面给予指导。

本书不仅是一本 CMM 的基础课程,也不只是 CMM 的技术报告,而是面向不同层次读者的有关 CMM 的实际应用和学术理论。本书在介绍各种概念的同时,还附有详尽的图例和练习,便于读者阅读和理解。

为了方便读者很快掌握 CMM 的理念,本书首先将一些核心的概念融于实际生活之中,从而让我们透过熟悉的实例,对 CMM 有一个全面的认识。然后再对 CMM 作进一步的探讨,并从不同的角度解释了 CMM。编写本书的目的是让不了解软件的人很快看懂,也让已经了解 CMM 的读者对软件改进过程有更深一层的理解。书中加入大量的图表,使每一章节简短、易懂。我们没有假设读者有丰富的开发经验,因此无论是公司的投资者、管理层或编程人员都能很容易透过书中的例子清楚地了解 CMM 的哲学。基于以上原因,我们建议不同背景的读者根据自己的实际情况和需要,按下面的指导来阅读:

(1) 项目投资者:第 1 章;第 2 章

（2）非计算机专业的管理者：第1章；第2章；第3章；第4章；第5章；第11章

（3）对CMM报告已经有一定了解的读者：第2章；第5章；第6章

（4）已了解CMM的观念却想清楚每个KPA的读者：第5章；第7章；第8章；第9章；第10章

（5）准备通过CMM认证的企业、计算机专业的学生或攻读MBA的人士：本书所有章节

最后，由衷地感谢高爱章为本书的排版以及Neuper Burkhard和Manuela Stimpfl对本书第十一章的帮助。

作　者

2002年5月10日

Preface

CMM (Capability Maturity Model) was developed by the Software Engineering Institute at Carnegie Mellon University around 1991. Software is an artifact that is produced without any obvious physical laws governing its production, unlike the manufacture of a vehicle or the construction of a subway system. Thus, the software process cannot be easily measured, because the progress of the project is extremely difficult to visualize. Although software represents an abstraction of the real world applications that it serves, it becomes very real in terms of the resources needed for its development, and the trend has been towards greater investment in software projects. Higher investment brings much higher risks; the failure of a software project might cause a total loss because an incomplete software product has practically zero sales value, which is unlike other engineering projects. CMM provides a model of best management practices, applying engineering discipline to manage and continue improving the software process.

This book is not an introduction to CMM, nor is it a technical report on CMM. Instead, we attempt to present specific material on CMM so that we can help our readers apply the model in reality and advance their studies in software process improvement. Therefore, this book presents CMM from

different perspectives. Many of the chapters are completely self-contained, and our readers can, therefore, choose to read these in any sequence they prefer.

Practical knowledge and experiences of managing the software process are not easily acquired or appreciated only by reading. Thus, we endeavor to associate the concepts we present with more familiar activities, such as shopping. Moreover, looking at something from different perspectives provides a broader view, and this is true of CMM in application. As a result, those who do not yet understand CMM will be able to grasp the main concepts and KPAs (key process areas); whereas those who have already understood CMM will be able to gain deeper insights into the model. Our target audience for this book include all software project investors, decision-makers, managers, CMM project leaders, project team members, computer science and engineering students, and MBA students. Here, we suggest how you might begin to read this book according to your interests and background.

(1) Software investors: 1, and 2.

(2) Managers without a computing background: 1,2, 3, 4, 5, and 11.

(3) Readers with a basic understanding of CMM: 2, 5, and 6.

(4) Readers with an understanding of KPAs: 5, 7, 8, 9, and 10

(5) Readers working on CMM certification, computer science students, MBA students: all chapters.

Finally, we would like to thank GAO Aizhan for typing, and Neuper Burkhard and Manuela Stimpfl for helping us in Chapter 11. And we feel very fortunate to have Ms XUE Hui to review our book.

CMM 软件过程的管理与改进

目 录

第1章 导 论

一位有天赋的程序员，不管他编写程序有多快，但仅依靠个人的力量，仍很难完成一个大型的软件系统。因为对于大规模的软件开发项目，参与其开发的人数可能是百余人甚至上千人不等，所花的时间可能是半年甚至是几年。不管他对编程有多擅长，也不可能代替这么多人的工作，因而还是无法开发一套大型系统。这位优秀的开发员意识到，要使项目成功，必定要与他人配合工作，让每个参与开发的成员都能相互了解各自的工作范围以及自己的职责。但很快出现了另一个典型的软件开发问题：两个程序员以完全不同的方式编写两个相似的功能。此问题看来没什么大不了的，但当系统运用一年后，需要修改时，负责维护的工程师要多花一倍的时间来学习一个同类的问题。更糟糕的是在系统开发进入尾声后，客户突然说系统需求被错误理解。此时那位杰出的程序员开始认真地思索：怎样管理才可使软件项目取得真正的开发(包括维护等)成功呢？其实早在20世纪80年代末，很多富有经验的开发主管就已经提出了这个问题。

一位资深的经理，接任某大型复杂项目的执行经理，其项目的进展速度比预定的计划还要快。不久猎头公司便与他联系，这使他联想到：若他离开这个岗位，会给项目带来什么样的影响？他这才注意到整个项目管理中隐藏了很多地雷，而且看来只有他才知道这些地雷所分布的位置，所以他可以避开或拆除。首先，项目里有很多问题或很多工作是变相重复的，原因是已往工作的经验都没有任何文档记录，也就是说很多工作过程都不能被追踪，加上工程师之间欠缺相互沟通，使得他们只关注所分配的工作，他们甚至不了解其他成员工

作过程的重要性。若他们遇上问题，也不清楚其他同事是否也曾碰过同样的问题。但他们会主动向经理汇报，因为经理个人的记忆力和分析能力强，很快便有应对的方法。基本上，执行经理起着中枢控制的作用，这就是我们常说的“英雄式管理”。再者，由于项目计划和评估只基于他个人以往的经验，当然不是否定其重要性，但因为没有数据的基准，当项目进度与计划有差异时，除他以外，却没有人能作出正确的决策。因为没有衡量标准和健全的文档，别人根本无法弄清问题是计划本身的错误，还是执行上的错误，更别说什么决策。他想到这里，明白了以模式为基础的重要性，决心改变管理方法。此时他想知道软件世界是否已有一完整模式供他参考呢？

一位有远见的投资者，要挑选一两家软件公司进行某项目投资，项目的回报是可观的，并且涉及的投资金额也很大。但怎样去评估一个软件组织的能力是否可以承担该项目呢？当多家公司规模相似且项目也类似时，他自然会去了解各公司的技术、组织和管理。假如不是软件公司而是生产企业，通过参观它们的工厂，看看厂房规划、半成品、机器的品牌和保养，就可以粗略地了解公司的技术、组织和管理。但软件工业不容易从工作环境中了解其真实水平，除非他是资深的软件工作者，否则参观软件组织及其外部环境是毫无意义的。这时他会问：有没有什么认可的标准来衡量软件公司的技术、组织和管理等一系列的问题呢？

本书将会讨论以上三种不同身份的人所关心的问题。其实，早在 20 世纪 90 年代初，这些问题就已开始受到重视，并诞生了专门研究此类大型软件项目所面临问题的学科，统称为软件工程过程（Software Engineering Process）。软件工程过程并不是以工程数学来探讨软件过程，而是应用工程式的纪律来管理和组织软件开发或维护过程，CMM 就是这学科中的一个管理和改进软件过程的模式。前面所说的他们面临的问题，其答案就是 CMM。

本书着重从管理的角度，深入浅出地介绍 CMM 和利用其不断优化软件过程的原理。与其他大部分行业的产品不同的是：软件产品虽然也可以说是有形的、可视的和可感觉的，但产品里面的东西却是无形的、抽象的和高数字组合的。因此要管理这样性质的事物，肯定要比传统工业的生产管理更抽象，CMM 也不例外。本书会尽可能简单、具体地讲述 CMM，让以上三类人都能明白 CMM 便是他们所需要的东西。

最后要说明的是，每个成功的项目都需要高层管理者的支持，同样实施

CMM 也要有高级管理层支持。只有让他们了解 CMM 以及实施 CMM 后会给企业带来怎样的利益，他们才会全面支持和参与 CMM 的实施。对某些高层管理者来说，软件过程是高深的。笔者看过国外有关 CMM 的书籍，它们都假定了读者有软件开发的经验，这对高层管理者或投资者来说，是比较困难的。有鉴于此，本书将把 CMM 一些抽象的哲学和专业名词，以简单、清楚的方式说明，让不同层次的人都能明白 CMM 的核心。

1.1 什么是软件工程过程

软件工程(Software Engineering) 一词最早出现在 1968 年的欧洲研讨会。当时“软件工程”被定义为：

建立健全的工程的法则，以便经济有效地开发可靠和高性能的软件

在此之后，从编程技术到质量管理，软件工程的研究不断被延伸，软件工程的定义也因此而不断被修正。经过 30 多年工业和学术界的共同努力，到目前为止软件工程已有五个主导方向。表 1.1 分别说明这五个方向在技术、组织和管理这三个层面上的不同。

表 1.1 软件工程的五个发展方向

项目	方向	说明	技术	组织	管理
1	编程方法 (Programming Methodologies)	着重于问题分析和在程序上的实现。例如： - 结构化编程 - 面向对象编程	高	低	低
2	形式化方法 (Formal Methods)	以数学方法来分析编程。例如： - Z 语言 - CSP - SDL	高	低	低
3	计算机辅助软件工程 (Automated Software Engineering)	协助设计和自动生成基本程序的工具。例如： - CASE 工具 - UML 工具	高	低	低

续表

项目	方　向	说　明	技术	组织	管理
4	软件开发模型 (Software Development Models)	着重探讨以组织为单元的系统开发模型。例如： - 瀑布模型(Waterfall Model) - 螺旋模型(Spiral Model)	高	中	低
5	软件工程过程 (Software Engineering Process)	探讨软件过程(包括开发过程、软件支持过程或维护过程等)的管理和改进。例如： - CMM - SPICE - BOOTSTRAP - SPERM	高	高	高

表 1.1 简要说明软件工程的五个发展方向，其中软件工程过程是最全面的，本书将着重描述组织和管理这两个方面。因为不管企业的技术有多强，要使项目取得预期的成功，必须要有良好的组织结构和项目管理。前四个方式都没有告诉我们怎样确保大型软件项目成功以及怎么评估产品质量等方面的问题。历史告诉我们，20 世纪 80 年代末和 90 年代初很多项目都有不同程度的失败，大部分原因是项目管理中存在一些问题。从此软件的专门管理开始受到重视。

最后补充一点：CMM、SPICE、BOOTSTRAP 和 SPERM 都是软件过程的管理模式，只要明白其中的某一模式，其他便可融会贯通。

1.2 CMM

CMM 是由美国卡内基·梅隆大学(Carnegie-Mellon University)的软件工程研究所(Software Engineering Institute)提出的一套对软件过程的管理、改进与评估的模式。CMM 最早被应用于美国国防部(DoD，Department of Defense)那些外部企业承接的军事软件项目，这些项目都涉及金额巨大的投资，而 CMM 是用来评估那些有兴趣的软件商是否有能力承担项目的工作，也就是说评估他们的软件过程能力。

在此之后，CMM 被作为一认可标准，并且建立了 CMM 认证体系，用来衡量组织或企业的成熟度等级或软件过程的能力。CMM 分成五个成熟度等级。凡是具有开发软件能力的组织，都可以申请 CMM 第一等级的认证。若该组织具有基本的项目管理能力，对于软件工程和 CMM 有清晰的概念，并能具体地应用以往开发项目的成功经验，这样便达到申请第二等级认证的条件。其他等级认证的申请，依此类推。CMM 的每一级有着不同的评估标准，尽管 CMM 技术报告没有完全否定跳级的可能性，但 CMM 认证不能越级评估。表 1.2 简单地描述了 CMM 各级的标准。

表 1.2 CMM 各级的标准

成熟度等级（能力等级）	名 称	描 述	性能评定
2	重复级	在该等级，通过制定基本项目管理过程，来追踪成本、进度和功能，使以前项目的成功经验在今后的类似项目得以应用。	以往的成绩得以重复利用，项目计划和跟踪管理过程是稳定的。
3	定义级	在该等级，针对管理和工程行为的软件过程被文档化、标准化以及综合化成一个组织标准软件过程。所有项目运用一个经核准和修改的组织标准软件过程版本来开发和维护软件。	性能随着明确定义的过程提高。
4	管理级	在该等级，收集了软件过程和产品质量的详细估量标准。软件过程和产品被定量理解和控制。	性能在过程和产品被定量理解的基础上持续提高。
5	优化级	在该等级，从过程引导创新的理念及技术的反馈导致了过程持续提高。	性能持续提高以满足过程效率的提高，避免了昂贵的重做，并缩短了开发时间。

申请 CMM 的主任评估员不需要有什么特别的先决条件，但首先要经过严格（或昂贵）的培训。与其他计算机专业资格的评定一样，CMM 评估员资

格也不是终身制的。CMM 评估员每年要评估两家以上的软件企业，评估的过程和结果也必须提交给卡内基·梅隆大学软件工程研究所的数据库，以便对其评估报告的公正性进行调查。

CMM 并没有强调所有的软件组织或企业都应采用统一的管理模式和规范，CMM 只是提供一系列评估的指标或改进的指南，协助组织进一步实现软件规范化管理，比如项目的文档间是否保持一致性、软件开发人员的管理是否严格、开发过程是否有清楚的定义和衡量标准、对使用的软件新技术是否能进行有效的分析等。

在此我们要特别说明，很多企业为了 CMM 认证而进行改进，从而加强了软件企业的专业形象。这只是实行 CMM 的一个很小的好处，而改善组织对软件开发或服务的能力，使每一项目的成功都能依据健全的管理模式，才是 CMM 真正的价值。CMM 主要作者之一 Mark Pault 在 2001 年欧洲研讨会（EuroSPI 2001）强调 CMM 是目前可帮助不断改进软件过程的一个最好的模式，而认证只是 CMM 的副产品。

当我们开始关注 CMM 或软件过程的改进时，会发现很多大的软件企业如微软、Oracle 和 Sybase 等都没有进行 CMM 认证，便会疑惑为什么如此成功的软件公司，并没有通过 CMM 的认证，却仍然特别出色。其实在上文中已经提到，你可以参考及应用别人已发展的模式，在一般情况下这是最经济的，但亦可以像微软和 Oracle 等公司一样投资去发展自己企业的软件过程模式。如果你的企业没有很多资源来研究软件工程过程，那就应该利用别人已建立的好模式，而 CMM 会是你最好的选择。（你不会因开发某一商业系统而不用现有的 Windows 或 UNIX，除非你有很特别的理由要自行重新编写整个的作业系统。）

另一点要补充的是随着技术进步，传统的组织工作结构变成了虚拟的形式（vitural），CMM 也不能完全通用，所以不断有新的模式或是 CMM 的改进版本（CMM－I）出现，这也是 Mark Pault 说“在软件界里没有一种模式能包括所有”的原因。关于这些问题，由于超出了本书范围，在此不多探讨。需要强调的是：目前 85％的组织都是以传统工作方式来运作的，因此 CMM 能广泛应用于国内的软件组织和企业中。

1.3 CMM的益处

CMM把软件组织描述为五个阶梯式的成熟层次，它们也代表着不同的软件过程能力。在这里，过程能力是根据过去的项目是否有良好的软件过程的管理模式来决定的。一个不成熟软件组织可能有着相当成功项目的经验，但如果其成功不是用模式来指引，我们还是怀疑该组织的能力，因为任何的变异（特别是人员的流动）都可能会使正在进行的或未来的项目完全失败。相反，在成熟组织里，过去的成功经验有了系统的记录，并作为参考和改进的依据。由于项目的执行是建立在软件过程模式上，成功机会和产品质量都是可以预测的。

在不成熟的软件组织中，不存在判断产品质量或者解决产品或过程问题的客观基础。因此，产品质量或项目成功率难以预测。当进度落后时，常缩短甚至取消软件测试和评审等过程，这样做只会影响质量。

在成熟的软件组织中，经理监控产品的质量和顾客的满意程度，并以定量的方法，客观地判断产品质量及过程问题。由于进度和预算是基于以往的数据，而非凭空设想，因此通常在成本、进度和质量等方面都能达到预期的结果。成熟组织的软件开发是有纪律的过程，组织的员工都认同其价值并遵循其模式，管理层和工程师们都能支持软件过程模式，使成熟组织走向成功。说到这里，有些读者（特别是高级管理层）会觉很CMM的益处很抽象。下面以具体的实例加以说明。

Atos Origin India（AOI）为印度的软件企业，成功地达到CMM第五级认证[1]。AOI是Atos Origin的子公司，约有400位技术人员，客户主要来自欧美等地。AOI的服务范围包括：软件工程技术支持和他们的EPR系统。早在1994年，该公司就已达到ISO 9000质量体系标准。但他们并不满足，原因是该公司相信市场的竞争不完全取决于价格，还有项目执行的时间表、满足客户的要求、产品稳定和无缺陷等。所以他们选择了CMM为努力的目标。为什么会是CMM？因为CMM强调不断改进和优化过程以达到优质的哲学思想与公司管理层的信念是一致的，即品质是没有终结点的，而只有不断地改进。此理念清楚地指出ISO 9000和CMM的分别：ISO 9000是说明优质标准的最低要求，属静态；而CMM是主动的不断改进，属动态。