CMM 定量过程管理 KPA 实践分析中的软件过程的定量控制*

胡凤根, 张可汇
(中山大学软件研究所, 广东 广州 510275)

摘 要: 首先阐述软件能力成熟度模型 SW-CMM 与 CMM 中的定量过程管理关键过程域, 然后讨论有关定量控制的若干经验公式和理论模型, 再从软件开发过程的三个方面出发, 分别利用这些经验公式和理论模型来分析定量过程管理在其中的实践应用。

关键词: 软件能力成熟度模型; 定量过程管理; 软件过程; 定量分析
中图分类号: TP311.5 文献标识码: A 文章编号: 0529-6579 (2003) S2-0120-04

1 引 言

软件产业部门逐渐成为代表一个国际高新技术水平的信息技术产业中新兴的、发展最快的和潜力巨大的产业部门, 但现实情况是, 软件企业经常不能够在给定的预算和给定的时间里完成复杂的大型的软件产品。即使完成了, 软件中也留有很多隐患。这些问题长期以来困扰着软件业的发展[1]。

针对这种情况, 卡耐基·梅隆大学 (CMU) 的软件工程研究所 (SEI) 自 1987 年开始, 研制出一套软件生产过程成熟度的理论框架——能力成熟度模型 (CMM, capability maturity model)。CMM 通过指导软件开发人员的活动来改进软件过程, 以达到软件过程可复用性、可定量管理、可有效控制的目的[6]。

2 CMM 定量过程管理 KPA 阐述

定量过程管理 (QPM, quantitative process management) 处于软件能力成熟度模型 (SW-CMM, the capability maturity model for software) 的第四级——定量管理级。处于该级别的软件过程具有如下基本特征[2]:

(1) 设置了定量的质量目标
(2) 项目的产品质量和过程是受控的和稳定的
(3) 组织的软件过程能力是可定量预测的
(4) 开发新领域软件的风险是可定量估计的

定量管理级的两个关键过程域 (KPA, key process area) 中的其中之一——定量过程管理的目的, 是为了定量地控制软件项目的过程性能[1]。定量过程管理包括建立集成软件管理关键过程域所描述的项目定义的软件过程的性能目标, 测量过程性能, 分析这些测量结果并适当调整过程, 以保持过程性能在可接受的范围之内。当过程性能稳定在可接受范围内时, 项目定义的软件过程、相关的测量数据和测量的可接受范围等都被作为基线建立起来, 并被用来定量地控制过程性能[2]。

定量过程管理在 CMM 中有着重要的地位。它使得软件过程具有精确定义的、一致的评价方法, 这些评价方法为评估项目的软件产品和质量奠定了一个量化的基础, 使软件开发真正变成一种工业生产活动。同时定量过程管理也为软件过程的可持续的改进与优化打下坚实的基础。

2.1 软件过程的定量控制

由定量管理级的 4 个基本特征可以看出, 软件产品的如下 3 个属性需要用到定量管理: 软件产品质量, 软件产品生产过程, 软件产品开发过程中的风险 (可预测的软件过程能力是渗透在以上 3 个属性中的估算过程, 故不作为一个属性单独存在)。

另一方面, 参考文献[4]中提到, 软件项目的关键数据是项目特征, 项目进度, 项目工作量, 规模和缺陷。此 5 个属性中, 项目特征是指项目名称、应用领域, 开发语言等描述性的数据, 与定量分析无关; 项目进度由项目工作量和规模联合决定; 缺陷清除率是度量软件质量的常用标准[4]。

故定量过程管理的实践可以从以下 3 个方面展开, 即: 软件过程进度管理, 软件过程质量管理和软件过程风险管理。每个方面又分别按照测量、估算两个步骤进行定量分析。

在进行定量分析之前, 首先分别介绍一下测量、估算的概念。

* 收稿日期: 2003-07-15
作者简介: 胡凤根 (1980 年生), 男, 硕士研究生; E-mail: ahgenhu@sina.com

(C)1994-2020 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

2.2　相关概念

测量：根据机构的测量标准对软件开发过程中所产生的数据进行收集和整理，如代码行（LOC）和功能点（FP）的数量。

测量作用：测量为估计不准、进度缓慢、可见性差等普遍问题提供了矫正的方法。实际上，任何机构任何项目在不同的程度都能够从测量活动中获益。测量的最终目的则是建立过程能力基线（process capability base-line）。

过程能力基线代表的是按量化术语描述的过程能力。过程能力实质上是遵照该过程所能预期得到的结果范围。换言之，如果项目遵循某个过程，过程能力可用来决定项目可能的输出结果范围。基线化是采集和分析贯穿于项目的数据，并从这些数据中得到参考点。

基线所关注的是质量和生产率。可以由如下方程定义。

如果我们有：

F＝用功能点描述的软件规模

E＝项目花费的所有工作量

D_1＝在开发过程（提交之前）中发现的所有缺陷数

D_2＝提交后发现的缺陷数

$D=D_1+D_2$

对于一个项目，则有如下定义：

$$\text{生产率}=E/F \quad (1)$$

$$\text{质量}=D_2/F \quad (2)$$

$$\text{缺陷注入率}=D/F \quad (3)$$

$$\text{整体缺陷清除率}=D_1/D \quad (4)$$

估算：估算通常的定义是，对未来事实非零可能性的最乐观的预测。

3　定量分析

3.1　软件进度管理的定量分析

3.1.1　测量　每个机构都会根据自己的优先级别决定测量什么，但是大多数软件项目最少要保留项目规模、计划、资源要求和质量指标等数据。

下面是针对进度管理所应该建立的一些数据元素：

(1) 已经评估过的历史记录与一个项目实际还需要的天数（可以是百分比）的对比（帮助追踪和提高评估的准确度）。

(2) 按编程语言统计每个员工每月的平均代码行数（有助于规划编程活动）。

(3) 按编程语言统计每个员工每月的平均功能点数（有助于规划编程活动）。

(4) 产生每页文档的平均小时数（有助于规划项目的文档活动）。

3.1.2　估算　创建一个准确的开发进度的过程包含以下 3 个估算步骤：①估算产品规模（代码行或功能点）；②估算工作量（人月）；③估算进度（日历月份）。

(1) 规模估算。规模估算可以用以下几种方法：①用估算算法进行估算。如功能点估算是从程序功能来估算程序规模；②用规模估算软件。这种方法是按照程序功能（如屏幕、对话框、文件、数据库表等等）的描述对项目规模进行估算；③基于旧系统相似部分，按百分比形式估算新系统每个主要部分的规模。然后把每个部分估算规模加起来就可以得到新系统总的估算规模。

功能点估算是对程序规模的一个综合量度，经常用于项目早期阶段。从需求说明书确定功能点比确定代码行容易。以下方法最接近“1984IBM 方法”，是 IBM 和国际功能点用户团体（IFPUG's）现有准则的基础。

程序中功能点的数量建立在以下各项的数量和复杂度之上：

输入、输出、查询、内部逻辑文件和外部接口文件。它们的系数如表 1 所示。

表 1　功能点系数

Tab. 1　The coefficients of function point

程序功能	功能点		
	一般复杂	中等复杂	很复杂
输入数量	×3	×4	×6
输出数量	×4	×5	×7
查询	×3	×4	×6
内部逻辑	×7	×10	×15
外部接口	×5	×7	×10

(2) 工作量估算。下面几个方法可以用来把规模估算转换成工作量估算：①使用估算软件直接从规模估算得出工作量估算；②使用相应的进度表把代码行形式的规模估算转换成工作量估算；③使用组织中的历史数据确定具有相近规模的先前的项目花了多少工作量；④使用工作量估算模型，如自顶向下的方法，自底向上的方法[4]。

(3) 进度估算。进度估算是软件项目估算的第三步。一旦估算了规模和工作量，估算进度变得近乎微不足道。下面的方程 5 是以经验得出的按照工作量的大小估算进度的一种方式[5]。

$$\text{月进度}=3.0\times\text{人月}^{1/3} \quad (5)$$

(C)1994-2020 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

以下一些方法是根据工作量来估算软件进度的：①根据规模和工作量，用估算软件估算进度；②利用组织的历史数据；③利用进度表查找基于团队规模估算的进度估算；④利用某一算法（例如，COCOMO）的进度估算步骤，提供一种调整得更好（比方程 5 的估算）的估算。

这三步是最一般的步骤，后续步骤是：提供某一范围内的估算，并且随着项目的进行，定期改进范围，以提供更高的精确度。

下面给出 Jones 的一阶估算准则

假如有了功能点总数，就能够根据该数使用 Capers Jones 描述的“一阶估算”直接计算进度。要使用它，先取功能点的总和，并从下表中选取合适的幂次将它升幂。表中的幂次是 Jones 根据数千个项目的基本数据分析而得到的。

表 2 由功能点计算进度的幂次

Tab. 2 The power of schedule computed by function point

软件类型	最优级	平均	最差级
系统软件	0.43	0.45	0.48
商业软件	0.41	0.43	0.46
封装商品软件	0.39	0.42	0.45

3.2 软件质量管理的定量分析

软件质量是软件产品满足规定的、隐含的、需求的能力和有关特性的集合，是描述所有计算机软件优秀特性程度的组合。

在实践中，质量管理经常是围绕着缺陷的。因此，人们往往使用提交的缺陷密度——即，在提交的软件中每单元规模的缺陷数——作为质量的定义。这个定义成为事实上的工业标准，用来表明软件项目的目标是提交缺陷尽可能少的软件。

所以，软件质量管理的定量分析，就是对软件中存在的缺陷的测量和估算。

3.2.1 测量 针对缺陷所应该建立的一些数据元素：①公开的缺陷数与报告总缺陷数的对比（预测项目发布时间）；②审查发现的缺陷数量和运行测试发现的缺陷数的对比（可帮助你规划质量保证措施）；③产品发布前清除的缺陷数在总缺陷数中所占的百分比（有助于评估产品的质量）；④按严重缺陷、子系统缺陷，分类统计平均修复时间（有助于规划纠正缺陷的工作）。

3.2.2 估算 前面提到，项目的质量目标是项目预期交付的缺陷数。即，质量目标是在验收测试阶段预期发现的缺陷数，所以软件质量估算就是要估计验收测试阶段预期发现的缺陷数：

(1) 如果使用类似项目的数据，那么可以估计当前项目在验收测试时发现缺陷数，它等于在类似项目的验收测试阶段发现的缺陷数和这个项目估计的工作量与类似的总工作量比率的乘积。用如下公式表示：

$$\text{验收测试缺陷的估计(P)} = \text{验收测试缺陷数(SP)} \times \frac{\text{工作量估计(P)}}{\text{实际工作量(SP)}}$$

(2) 使用过程能力基线中的数据，那么可以使用几种方法来计算这个值：①质量目标设定为每功能单元的缺陷数，那么功能点规模按前面讨论的方式进行估计，预期的缺陷数是质量数据和估计规模的乘积。

②质量目标设定为过程缺陷清除率。在这种情形下，在验收测试阶段预期存在的缺陷数可以由缺陷注入率、过程中的清除率目标和估计的规模一起来决定。

3.3 软件风险管理的定量分析

软件风险管理工作就是在风险成为影响软件项目成功的威胁之前，识别、着手处理并消除风险的源头。风险管理的第一步就是要识别那些可能将风险带到项目计划中的因素，也就是对风险进行分类。

(1) 技术风险。如果项目采用了复杂或高新技术，或采取了非常规方法，就有潜在问题。另外，如技术目标过高、技术标准发生变化等也可造成技术风险。

(2) 管理风险。比如进度和资源配置不合理、计划草率且质量差、项目管理的基本原则使用不当等就可能造成管理风险。

(3) 组织风险。常见的是组织内部对目标未达成一致、高层对项目不重视、资金不足或与其他项目有资源冲突等都是潜在的组织风险。

(4) 外部风险。比如法律法规变化、项目相关接口方的情况发生变化，这些事件往往是不可控制的。但注意，一般将不可控制的“不可抗力”不作为风险，这些事件往往作灾难防御。

一种很有用的风险分析方法就是确定每个风险的“风险暴露量”。风险暴露量常备缩写为“RE”(risk exposure)。风险的一个定义就是“不希望的损失”。风险暴露量就等于“不希望的损失”的概率乘以损失的程度。

下面建立一个由风险、损失概率、损失大小和风险暴露量组成的风险评估表，见表 3。

在上述所示的风险评估表中，列出了可能给该假设项目造成项目进度延长 1 周到 20 周的潜在风险，风险发生的概率范围在 5% ~50%之间。

(C)1994-2020 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

表 3　进度风险评估表

Tab. 3　Schedule risk of evaluating table

风险	发生的概率/%	损失的大小 周	风险暴露 周
根据市场要求增加功能(未知的特别功能)	35	8	2.8
计划过于乐观	50	5	2.5
设计低劣—要求重新设计	15	15	2.25
新的开发工具不能节省预期的时间	30	5	1.5

4　定量管理过程控制

在定量管理过程中，控制有进度控制、质量控制和风险控制等几种。

其中，进度控制是比较实际状态和计划之间的差异，并作出必要的调整使项目向有利的方向发展。进度控制可以分成 4 个步骤：Plan(计划)、Do(执行)、Check(检查)和 Action(行动)，简称 PDCA。[7]

质量控制是为了保证软件的质量。软件的质量是软件的生命，保证软件质量就是保证软件项目的成功。“软件设计评审”和“软件测试验证”是软件质量控制两大关键环节。

风险控制的目的有三个：①监视风险的状况，例如风险是已经发生、仍然存在还是已经消失；②检查风险的对策是否有效，监控机制是否在运行；③不断识别新的风险并制定对策。风险控制常用的方法有：

(1) 风险审计：专人检查监控机制是否得到执行。并定期作风险审核，例如在大的阶段点重新识别风险并进行分析，对没有预计到的风险制定新的应对计划。

(2) 偏差分析：与基准计划比较，分析成本和时间上的偏差。例如，未能按期完工、超出预算等都是潜在的问题。

(3) 技术指标：比较原定技术指标和实际技术指标差异。例如，测试未能达到性能要求，缺陷数大大超过预期等。

5　结　语

在软件开发过程中，对软件过程的监控往往是不透明的，管理人员要想对过程进行透明的监控，定量的管理就在过程管理中有着非常重要的作用。只有对过程进行定量的管理，对过程的结果进行量化，才能衡量过程的效能。因此，对软件过程的量化管理在软件过程中就有着非常重要的作用。

参考文献：

[1] 杨一平. 软件能力成熟度模型 CMM 方法及其应用[M]. 北京：人民邮电出版社, 2001.

[2] 何新贵, 王纬, 王方德, 等. 软件能力成熟度模型[M]. 北京：清华大学出版社, 2000.

[3] Joseph Raynus. Software Process Improvement with CMM. USA: Artech House, 1999.

[4] Pankaj Jalote. CMM in Practice: Processes for Executing Software Projects at Infosys[M]. India: Pearson Education, Inc. 2000.

[5] Steve McConnell. Rapid Development: Taming Wild Software Schedules[M]. USA: Microsoft Press. 1996.

[6] 韩杰, 顾庆, 陈道蓄, 等. 基于 CMM 的软件配置管理模型 CSCM 研究[J]. 计算机工程与应用, 2001(5): 32—35.

[7] 佚名. 项目管理过程之进度控制[J]. 计算机产品与流通, 2002(11). http://www.zdnet.com.cn/biztech/enterprise-manage/story/0,2000096154,39097763,00.htm.

Quantitative Control on Software Process in the Analysis on QPM Practice of SW-CMM

HU Feng-gen, ZHANG Ke-hui

(The Institute of Software Research, Sun Yat-sen University, Guangzhou 510275, China)

Abstract: Based on the analysis on the practice of Quantitative Process Management, three aspects of software process are discussed. And some experience formulas and academic models are given. Then some quantitative control methods are applied to the aspects.

Key words: SW-CMM; QPM; software process; quantitative analysis

(C)1994-2020 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net