第28卷 第11期 计算机工程与设计 2007年6月
Vol. 28 No. 11 Computer Engineering and Design June 2007

基于CMMI的软件测试过程度量研究

万邦睿[1,2]， 丁晓明[1,2]
(1. 西南大学 计算机与信息科学学院，重庆 400715；2. 重庆市智能软件与软件工程重点实验室，重庆 400715)

摘 要：对软件测试过程的度量能够提高软件测试的有效性，保证软件的质量。通过CMMI与软件测试、度量的分析，提出了基于CMMI的软件测试过程模型，并以此为基础对软件测试过程度量元的有效选取进行了研究。建立了CMMI与GQM的映射关系模型，提出了度量元的选取原则，以一个应用案例证明了研究结果的可行性。

关键词：集成能力成熟度模型；度量；软件测试过程；度量元；有效性
中图法分类号：TP311.5 **文献标识码**：A **文章编号**：1000-7024 (2007) 11-2530-03

Research of software test process measurement based on CMMI

WAN Bang-rui[1,2], DING Xiao-ming[1,2]
(1. College of Computer and Informance Scicence, Southwest China University, Chongqing 400715, China;
2. Intelligent Software and Software Engineering Research Institution, Chongqing 400715, China)

Abstract: The measurement of the software test process can improve software testing efficiency and software quality. After the introduction and analysis for the measurement and test of CMMI, a software test process model is proposed based on CMMI. On the basis of the understanding, the analysis of effective metrics of software test process is performed. Then an interrelation between the CMMI and the GQM is guided, and some selected rules of metrics is suggested. Finally, the feasibility of research on an application is proved.

Key words: CMMI; measurement; software test process; metrics; validity

0 引 言

软件测试是软件质量保证的重要手段，有研究表明：越早发现软件中存在的问题，开发费用就越低，软件质量越高，软件发布后的维护费用越低[1]。一个好的、成熟的软件测试过程能够最大限度的保证软件测试的质量和有效性。目前，对软件测试的研究工作主要集中在测试技术和测试工具上，而对测试过程的改进研究工作还需进一步加强。

度量是改进过程的有效途径之一[2]。通过对测试过程的度量，可以使测试过程规范化、可视化；对度量数据的分析，可以测量出测试过程的有效性及存在的问题，明确测试过程的改进方向，从而保证软件的质量。因此，对软件测试过程的度量研究具有十分重要的意义。

本文以集成能力成熟度模型[3](CMMI)为基础，对软件测试过程的度量进行了研究，着重于度量元的选取，并以一个应用案例证明了研究结果的可行性。

1 研究背景

1.1 CMMI与软件测试

CMMI是美国卡耐基梅隆大学的软件工程研究所针对软件质量保证的核心——软件过程改进所提出的，它是一个成功的、广泛使用的过程改进模型[4]。众所周知，软件测试是软件过程不可分割的一部分，所以CMMI实际上也为软件测试过程改进提供了一系列的指导方针和关键原则。在CMMI连续表示中，验证(VER)和确认(VAL)过程域紧密的与软件测试联系在一起；而诸如配置管理(CM)、风险分析(RSKM)、量化项目管理(QPM)等过程域也同样包含了对测试过程改进的有力支持。

1.2 CMMI与度量

CMMI提供了度量和分析(MA)过程域，其目的是开发和维持用于支持管理信息需要的度量能力[5]。MA实际包含了度量和分析两个步骤，度量在前，是为获得过程或产品的表征数据；分析在后，是对数据进行分析，发现不一致、发现趋势和发现问题。MA的执行流程如图1所示。

从图1中可以得出，CMMI的度量和分析流程主要分为3个部分：计划、收集和分析。在计划阶段，主要任务是度量目标的确定和目标的细化；在收集阶段，主要任务是按数据采集和存储规程进行数据的收集、数据完整检查；在分析阶段，主要任务是按分析规程进行数据分析、存储数据和结果，以及报告结果。MA过程域定义严格，流程清楚，目标明确，可以用它

收稿日期：2006-08-31 **E-mail**：miller530@sina.com
基金项目：重庆市科委自然科学基金项目 (CSTC, 2004BB0146)。
作者简介：万邦睿 (1981－)，男，重庆人，硕士研究生，研究方向为软件工程、软件测试；丁晓明，男，副教授，硕士生导师，研究方向为软件工程、软件测试、数据库和人工智能。

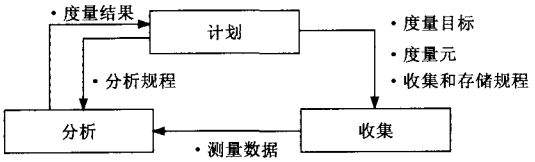

图 1 度量和分析(MA)执行流程

来指导软件测试过程的度量活动。

2 软件测试过程

2.1 过程的定义

按照 MA 的指导，任何对过程的度量都是建立在清晰的过程定义上的，因此，必须对软件测试过程有一个准确的定义。传统的软件测试过程包括：测试计划、测试设计、测试开发和测试执行等阶段，每个阶段都有一系列的任务。但是传统的软件测试过程定义无论是从测试范围还是测试工作方式上都存在着明显的不足，不利于进行过程度量，而CMMI则对传统的软件测试过程进行了扩充。因此，我们提出了一个基于 CMMI 的软件测试过程模型 (software test process model, STPM)，如图 2 所示。

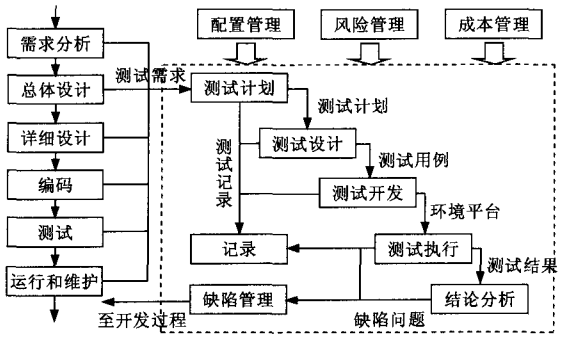

图 2 软件测试过程模型

STPM 有如下几点改进：

(1)软件测试不再只是对程序代码的测试工作，而是包含了所有在软件开发过程阶段产出物的测试工作，这扩展了测试工作的范围，更有利于发现软件开发中的缺陷；

(2)测试过程与整个软件开发过程是并行的，扩展了测试过程的工作方式；

(3)清晰的定义了过程的输入输出流，为整个过程的度量奠定了基础；

(4)将测试的结果纳入到了软件开发的缺陷管理机制中，扩展了测试的作用。

2.2 度量目标

通过对 STPM 的分析[6]，可以确定 3 类度量目标：①对测试产品的度量，这主要是对应 STPM 的输入，包括需求规格说明、详细设计说明、系统设计说明以及程序代码等的度量；②对测试过程产品的度量，这主要是对应 STPM 的输出，包括测试用例、测试报告等的度量；③对测试过程本身的度量，主要是对应 STPM 的执行状态，包括时间状态、环境状态等的度量。

3 度量元的选取和细化

3.1 选取模型

CMMI为过程改进提供了足够多的实践指导，但是，它只阐述了该做什么，而没有阐述该如何做，这一点也在MA中体现了出来。所以，在CMMI提供的要求和原则下，具体的度量和分析工作需要我们自己来定义。在此，我们引入 GQM 方法来保证度量元选取和细化的有效性。GQM(goal-question-metric)方法源于软件行业，是一种系统地对软件及其开发过程实施定量化的度量方法。GQM 引入了目标驱动的度量概念，在软件开发过程中已经取得了很好的效果[7]。GQM 的基本思想是：先确定一组目标；再针对各个目标，提出可能会遇到的问题，来定义这个目标；最后，针对每一个问题再给出一组测量方法，并用这一组测量方法测量出来的数据(度量元)就是对这个问题的回答。

GQM 方法在实施过程中最重要的就是要保证 G-Q-M 之间问题转化的完整性和匹配性。而对 CMMI 的一个单独的过程域而言，可以这样来描述：首先，对该过程域定义一个目的；然后，为了达到这个目的，给出了一系列的关键目标(包含特定目标和共性目标)；最后，针对每一个目标，细分关键实践来实现，而每个关键实践又可再分为具体的子实践。即是说，CMMI 的过程域从定义到实践是一个严格意义上的完整性和匹配性转化。如果从每个过程域的角度来进行软件测试过程的度量研究，那么就可以建立起CMMI模型与GQM方法的映射关系，如图 3 所示。

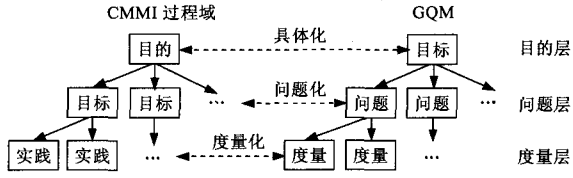

图 3 CMMI 与 GQM 的映射关系模型

上面的映射关系分为目的层、问题层和度量层，这样，CMMI 也是目标驱动的。由于 GQM 方法是一套已经被证明成功的度量元选取方法，所以在生成了 CMMI 到 GQM 的映射后，由 CMMI 导出度量元是可行的。将 CMMI 与 GQM 的映射关系定义为 C-G 模型，并将应用到 C-G 模型中的 CMMI 相关软件测试过程域定义为 STPA(software test process area)，C-G 模型和 STPA 为度量元选取和细化提供了理论依据。

3.2 选取原则

CMMI 不是专门为改进软件测试过程而出现的，所以将 STPA 引入进入 C-G 模型中时，需要有一定的剪裁和适用的选取原则[8,9]，才能保证度量元的有效性。本文从过程域、实践、文档、公共特性和相似结果 5 个方面，根据STPM，提出度量元的选取和细化遵循原则。

原则 1 过程域剪裁。每个 STPA 都有几个相关过程域，如果几个STPA同时都包含一个确定的过程域，那么该过程域应该保留并合并；如果STPA涉及的相关过程域与软件测试相关性不强，那么该过程域应该完全删除不予考虑。这样既可以保证 STPA 引申出来的软件测试实践的完整，又可以抓住

STPA 的核心。

原则 2 实践剪裁。CMMI 面向的是整个软件过程改进，所以在分析 STPA 时：①某些关注整个软件过程的实践，关注重点都要置换成软件测试过程，并要以特定的测试输入输出作为基础；②某些与软件测试无关的实践，或者是涉及到过程域的定义、维护以及审查等实践，都可以被剪裁掉；③对某些零散的实践，如果表示的都是同一个目标，那么应该将其合并；④对子实践，如果直接度量元仍然无法确定，应允许再进行拆分。

原则 3 文档剪裁。每个 STPA 都包含了大量繁琐的文档，不利于管理和维护。因此，应该只保留与测试输入输出有关的计划、规程、标准和报告等文档，这也是将整个度量和分析过程开发成自动化系统时所期望得到的文档数据。在保留的文档中，还应该注意相关文档的合并消除，降低文档之间的冗余关系。

原则 4 公共特性剪裁。每个 STPA 都有许多执行能力公共特性，这些内容在企业进行全面过程改进时或许有用，但是在进行软件测试过程度量时，有些显得无关和没有必要。对此，可以采取以下 3 种方法：①纳入到相关 STPA 中，例如人员、工具等资源的管理，可以在配置管理中统一度量；②直接作为该过程域的度量目标，例如过程状态管理、维护，可以作为对测试过程本身的度量；③完全放弃，例如人员培训、角色分配等。

原则 5 相似结果的合并。通过对不同的 STPA 分析，可能会对软件测试过程的某个属性重复度量，这时就必须涉及到相似度量元的合并。主要原则有 3 点：①如果度量元完全重复，那么应该只保留一个；②如果度量元属于同一度量目标，那么两者都应该保留并归为一类；③如果这个度量元是直接度量和间接度量的关系，那么应该将间接度量置于直接度量之下，并与该直接度量的其它间接度量进行合并。

4 应用案例

现以 STPA 中与软件测试联系最为紧密的确认(VAL)过程域为例，分析软件测试过程的度量。VAL 的目的是证实产品或产品构件置于预期的环境时满足预期的用途。在 STPM 中，产品或产品构件就是被测试的产品，预期的环境是搭建的测试平台，而用途则是产品的需求。因此，VAL 在 C-G 模型的目的层是：证实被测试的产品置于搭建好的测试平台下是否满足产品预期的需求。

按照原则 1 和原则 2，集中分析 VAL 的两个特定目标：确认准备和确认产品。确认准备在 STPM 中对应的是测试计划和测试设计部分，输入主要有测试产品和产品需求等，输出主要有需求确认计划和产品确认计划等。确认产品在 STPM 中对应的是测试执行和结果分析部分，输入主要来源与确认准备目标的输出，而输出主要有确认结果、确认报告和对查表等。因此，VAL 在 C-G 模型的问题层是：Q1 是否准备好了确认产品和确认环境？Q2 确认的过程和结果是否有效？

在明确了 VAL 的目标以及输入输出后，可以根据 VAL 的具体实践以及实践说明展开其在 C-G 模型中的问题层，结合原则 2、原则 3 和原则 4，将上述的 VAL 目标映射到 C-G 模型的度量层：M1 产品规模度量；M2 需求规模度量；M3 测试用例规模度量；M4 测试用例的有效性度量；M5 测试结果的度量；M6 测试工具使用度量；M7 测试过程进度度量；M8 测试充分性度量；M9 测试平台稳定性度量。

其中 M1 和 M2 属于对测试产品的度量，M3、M4、M5 和 M6 属于对对测试过程产品的度量，M7、M8 和 M9 属于对测试过程本身的度量。进一步分析可以得到 VAL 对于 STPM 的直接度量元，如表 1 所示。

表 1 所呈现的都是直接度量元，事实上，在 VAL 的分析过程中，一些直接度量元的导出是以间接度量元为基础的，这些信息应该与度量元的数据收集方法、数据单位等一起写入度量元的属性定义中。

5 结束语

通过对软件测试过程的度量和分析能够提高软件测试的有效性，保证软件的质量。本文在清晰的软件测试过程定义下，以 CMMI 这个成功的过程改进模型为基础，对度量目标和度量元的有效选取进行了研究。下一步的工作重点，是定义数据采集和存储规程进行度量数据的收集，以及定义分析规程来分析数据，提供度量结果和改进措施。最终，还需要开发一个自动化系统来支持整个度量和分析过程。而所有研究的基础，则是对软件测试过程度量元的选取，这也是本文研究的意义所在。

表 1 确认过程域的软件测试过程度量

度量类别	度量元	度量类别	度量元
产品规模	有效的代码行数	测试用例规模	测试用例的总数
	总的功能点数	测试过程进度	测试过程持续时间
需求规模	需求文档页数		测试工作时间
	需求覆盖的功能点数		单元时间工作完成量
测试用例的有效性	测试用例覆盖的功能数	测试结果	未发现错误的功能点数
	成功的测试用例数		满足需求的功能点数
	发现缺陷的测试用例数		预期的缺陷数
测试充分性	测试用例覆盖的总功能数		测试过程中发现缺陷的总数
	代码审查覆盖的功能点数		测试用例发现的缺陷数
	动态测试覆盖的功能点数		开发方证实的缺陷总数
测试平台稳定性	测试平台支持的测试用例数	测试工具使用	测试工具支持的测试用例数
	测试平台故障时间		测试工具支持的测试用例发现缺陷数

(下转第 2546 页)

表3 采购计划用例属性

编号	用例名称	ARU	MS	RMS
1	月计划	4	24	24.4%
2	待料需求	4	20	20.3%
3	临时需求	4	16	16.1%
4	计划变更	3	18	18.2%
5	计划撤销	3	14	14.1%
6	计划批准	3	7	6.9%

表4 采购计划用例关系矩阵

	月计划	待料需求	临时需求	计划变更	计划撤销	计划批准
月计划	0	0	0	0	0	0
待料需求	Ext	0	0	0	0	0
临时需求	Ext	0	0	0	0	0
计划变更	0	0	0	0	0	0
计划撤销	0	0	0	0	Inc	0
计划批准	0	0	0	0	0	0

结果都比较小，用例加权关系数(相对加权关系数)的值也较小，用例之间的关系简单。对于大型项目，研究该用例自身复杂度及(相对)消息维尺寸更能有效估算项目复杂度。如果用例的相关执行者数目超过5(这一判断，应随着实践数据的积累不断巩固或修正)，则应考虑将用例进行细分，防止用例过于复杂。对于用例复杂度CU的获得，采用常用的Rational-Rose作为用例建模工具，通过UMTSA(UML based measurement tool of software artifacts)调用RationalRose的插件能较方便的获取模型数据信息[10]，经过算法分析，对以度量指标进行计算，得出结果并存入数据库。由于篇幅有限，其它度量结果不再一一列出。

5 结束语

对用例相关的度量的研究特殊意义在于：

(1)预防性：针对需求分析模型和设计模型进行的度量。将问题发现并杜绝在软件开发的早期阶段，会大大减少了由于错误或不合理而导致的花费；

(2)事前估计性：提出的度量指标能够对项目进行有效的估计，帮助开发部门合理调度资源，做好软件开发计划；

(3)评估性：为比较开发过程提供了一定的量化依据。对于软件过程改进开始时基线的识别、改进进行到一定阶段后改进效果的衡量，提供切实可行的操作方法；

(4)实用性：研究是针对目前的主流开发过程RUP——以用例驱动为中心，涉及的建模语言UML也是在软件工程界中占据着主导地位，因此对此的研究具有积极的意义。

随着用例技术的更普遍使用，基于用例的相关度量分析必将更加深入而广泛。

参考文献：

[1] Jones C.Software assessments, benchmarks, and best practices [M].Beijing: China Machine Press, 2003.

[2] Ivar Jacobson,Grady Booch,James Rumbaugh.The unified software development process[M].Beijing: China Machine Press, 2002.

[3] James Rumbaugh, Ivar Jacobson, Grady Booch. The unified modeling language reference manual[M].USA: Addison Wesley Longman Inc, 1999.

[4] 曾实,罗燕京.基于用例的软件需求开发与管理平台的研究[J].计算机工程与设计, 2006,27(8):1311-1313.

[5] Geri Schneider Jason P Winters.Applying use cases[M].Second Edition.Beijing: China Machine Press, 2002.

[6] Stephen H Kan.Software quality engineering: measurement and model[M].Second Edition.Beijing: Publishing House of Electronics Industry, 2004.

[7] John McGarry,David Card,Cheryl Jones.Practical software measurement: Objective information for decision makers[M].Beijing: China Machine Press, 2003.

[8] IEEE Std 982.21992.IEEE standard for a software quality metrics methodology[S].

[9] SEI Software.Engineering measurement and analysis[EB/OL]. http://www.sei.cmu.edu/sema/welcome.html, 2001.

[10] PAN Qiu-ling, LIU Zong-tian, JIA liang. UML-based software process engineering environment[J].Wuhan University Journal of Natural Sciences, 2001,6(1-2):524-530.

(上接第2532页)

参考文献：

[1] Roger S Pressman. Software engineering: A practitioner's approach [M]. Fifth Edition.北京:机械工业出版社,2002.

[2] Edward Kit. Software testing in the real world improving the process[M]. 北京:机械工业出版社,2003.

[3] CMMISM for software engineering version 1.1 continuous representation[S]. Camegie Mellon, Software Engineering Institute, 2002.

[4] Dennis M Ahern, Aaron Clouse, Richard Turner. CMMI Distilled: A practical introduction to integrated process improvement [M]. 北京:机械工业出版社,2002.

[5] 黄锡伟. CMMI解析与实践[M]. 北京:人民邮电出版社,2004.

[6] 郑人杰,王纬,王方德,等.基于软件能力成熟度模型(CMM)的软件过程改进——方法与实践[M]. 北京: 清华大学出版社, 2003.

[7] 陈祖荫,刘建丽.GQM软件度量模式的某些决策问题[J]. 北京工业大学学报,2000,26(2):45-48.

[8] 龚波,于自跃,何新贵.小型软件企业实施CMMI过程改进研究和分析[J]. 计算机应用研究,2004,21(8):64-67.

[9] 周金陵,张鹏. 基于CMMI的软件过程改进研究[J]. 计算机工程与设计,2003,24(11):60-62.