软件缺陷原因分析方法

刘 海 郝克刚
(西北大学信息科学与技术学院 西安 710127)

摘 要 软件缺陷原因分析对提高软件质量、保证软件项目顺利进行具有重要的意义。对定性和定量的软件缺陷分类方法进行了系统的总结,分析了它们的优势和弱点,并探讨了将这两类方法相融合以增强其功能和实用性的方法。
关键词 软件缺陷原因,缺陷预防,原因分析,RCA

Cause Analysis Method of Software Defect

LIU Hai HAO Ke-gang
(College of Information Science and Technology, Northwest University, Xi'an 710127, China)

Abstract Cause analysis of software defect is important for the high quality of software and the success of software projects. Qualitative and quantitative cause analysis methods of software defect were summarized systematically, and their advantages and shortcomings were analyzed. The strategies of combining this two kinds of methods to improve their capability and practicability were also introduced.
Keywords Cause of software defect, Defect prevention, Cause analysis, RCA

1 引言

软件缺陷不仅会造成软件质量下降,而且是导致软件项目延期、开发成本超支的重要因素,因此软件组织都采取了各种技术、过程和方法来尽量减少软件中的缺陷。但是,许多软件组织减少软件缺陷的努力通常只集中在缺陷的检测和排除上。事实证明,这种方式的效果是有限的。因为无论是测试还是各种技术审查,都只是一种被动的缺陷检测方法,无法防止缺陷最初的引入,也无法保证能够检测到所有缺陷。此外,检测和排除缺陷的过程会消耗大量成本(特别是在软件项目后期,如系统测试和维护阶段)。因此,为了最大程度地减少缺陷并实现软件项目的效益,还须采取主动的预防措施,分析缺陷产生的根本原因,有针对性地消除这些原因,防止将缺陷引入到软件中,即通常所说的"缺陷预防"(Defect Prevention, DP)[1]。

缺陷预防的核心任务是缺陷原因分析,也就是找到导致软件缺陷产生的共性原因和基本原因,目前国内有关这方面的技术资料较缺乏,使软件组织在应用这一技术时有一定的困难。本文系统地总结和分析了定性和定量的软件缺陷原因分析方法,并探讨了怎样将这两种方法相结合,使之具有更强的功能和更广泛的适用性。

2 基于RCA的定性分析方法

根本原因分析(Root Cause Analysis, RCA)是一种质量管理工具,在工业和医疗领域已有广泛的应用。RCA可以简单地定义为:使用结构化的过程和方法,识别问题产生的根本原因并制订相应的解决方案,使问题不会重复地发生[2,3]。"根本原因"(Root Cause)是指导致问题发生的最基本原因。与直接原因或表面原因不同,根本原因的消除可防止问题的再次发生,且一个根本原因往往与一组或一类问题相关,而不仅仅限于某一个问题[4]。

RCA过程包括收集问题信息、理解问题、分析根本原因和产生解决方案等几个步骤,每个步骤都会根据问题的性质采取特定的方法和工具。例如在分析根本原因时,可采用因果分析、失效模型与影响分析、事件与原因因素分析、人员能力分析等方法,并可使用因果图(也称鱼骨图、Ishikawa图)、关系图(Interrelationship Diagram)、当前现实树(Current Reality Tree)等辅助工具[3,4]。

将RCA技术应用于软件缺陷原因的分析,首先要明确软件缺陷的含义。一个软件缺陷是指软件对其期望属性的偏离,它包含3个层面的信息,即失效(failure)、错误(fault)和差错(error)[5,6]。失效是指软件系统在运行时其行为偏离了用户的需求,即缺陷的外部表现;错误是指存在于软件内部的问题,如设计错误、编码错误等,即缺陷的内部原因;差错是指人在理解和解决问题的思维和行为过程中所出现的问题,即缺陷的产生根源。一个差错可导致多个错误,一个错误又可导致多个失效。软件缺陷原因的分析不能只停留在"错误"这一层面上,而要深入到"差错"层面,才能防止一个缺陷(以及类似缺陷)的重复发生,因此软件缺陷的根本原因往往与过程及人员问题相关。

基于RCA的软件缺陷原因分析过程与传统RCA过程类似,一般包括选择缺陷数据、分析缺陷数据、识别公共原因

到稿日期:2008-02-21 本文受西安市科技计划项目(ZX06028)资助。
刘 海 博士研究生,工程师,主要研究方向为软件质量保证和软件测试,E-mail: lh_tyb@sina.com;郝克刚 教授,博士生导师,主要研究方向为软件工程。

并提出改进措施3个步骤[7,8]。采用该方法的软件组织通常是在软件项目的每个开发阶段结束后，或者定期（如每个月末）进行缺陷原因分析，提出改进措施，从而促进组织的过程改进。基于RCA的软件缺陷原因分析通常是以会议的形式进行，会议人员应包括开发团队中的重要成员，如项目管理人员、过程管理人员、技术人员（特别是那些报告和修复缺陷的人员）等，这样才能更好地理解和分析软件缺陷，并产生最佳的解决方案[10]。这种会议的一个典型范例是IBM的“缺陷预防会议”[8]。

对于中小规模的软件项目，缺陷数据的选择较为简单，可选择某一时期或某一项目阶段所发现和处理的所有缺陷作为待分析的数据集。但对于大型项目来说，由于缺陷数量庞大，而RCA又是一个费时的任务，只能选择一个具有统计显著性的缺陷样本集合。选择时需考虑的因素包括缺陷的严重程度、复杂性、相对于软件模块和开发团队的分布等。在进行缺陷分析之前，还要对所选择的缺陷数据集进行检查和预处理，即修补不完整和不一致的数据，清理错误数据（如误报的缺陷）和重复数据[7]。

对所选择的缺陷数据，要使用RCA方法和工具逐个进行分析，确定缺陷的触发条件、在哪一个开发阶段被引入以及为什么会引入等等[7,8]。随着软件组织经验的增长和数据的积累，可对分析结果进行分类，从而有利于更快速和准确地进行分析。例如文献[7]根据Motorola公司的RCA经验将软件缺陷引入的根本原因分为开发阶段相关（phase-related）、人员相关（human-related）、项目相关（project-related）和复审相关（review-related）4类。其中人员相关问题分为通信障碍、缺少相关知识、协作失误和人为疏忽等类。在这些分类的基础上，可很方便地使用一些RCA工具来确定缺陷产生的根本原因。例如图1所示的因果图可作为辅助工具来分析一个缺陷的根本原因[9]。

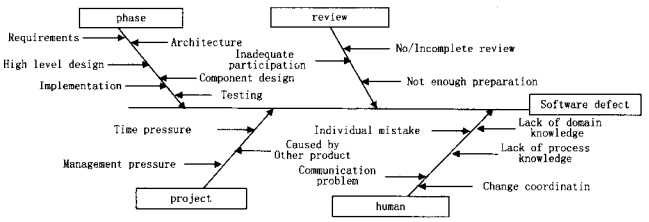

图1　软件缺陷原因分析因果图

在逐个分析了缺陷之后，还要对缺陷数据的总体趋势进行分析，识别出公共问题及其产生原因，并制定有关过程、技术和人员管理方面的改进措施[7,8]。例如，如果经统计发现由人员通信问题所引起的软件缺陷占有较大的比例，就要调整开发团队的结构，改善人员交流的方式；如果由需求问题引起的软件缺陷较多，就要改进需求管理过程和技术。

基于RCA的软件缺陷定性分析方法的优点是可对缺陷进行逐个分析，针对性强，可产生详尽的分析结果和解决方案，因此特别适合于对一些严重程度较高的缺陷进行重点分析。但其缺点是易受分析人员主观因素的影响，资源消耗较大，效率低；在大型项目中，由于必须对缺陷数据进行采样，因此分析覆盖率较低；对缺陷逐个进行详细分析的方法抽象程度太低，不宜于缺陷总体趋势和公共原因的分析；此外，该方法总体上属于一种“事后分析”方法，即在某一项目阶段结束后才对该阶段发现的缺陷进行分析，实时性较差，其分析结果对当前项目的帮助非常有限。针对这些问题，一些软件组织开发了一种基于缺陷分类的定量分析方法。

3　基于缺陷分类的定量分析方法

该类方法的基本思想是[11]：首先对缺陷数据进行合理的定义，使缺陷属性能够充分反映软件产品在开发过程中的进展情况，然后在缺陷跟踪流程中快速提取缺陷属性值。由于缺陷属性捕捉了丰富的有关软件产品和开发过程的信息，因此通过对缺陷数据的统计分析可推断出缺陷被引入的原因。

该类方法的实现需要两个条件：合理的缺陷数据定义（也称为缺陷分类）方法和良好的缺陷管理工具支持[12]。缺陷数据定义不仅要满足缺陷跟踪流程的需要，还要包含若干属性，使其值域能够充分覆盖软件过程中的各种活动，从而反映出缺陷被引入的过程原因。符合这一要求的一种典型的缺陷数据定义方法是IBM开发的“正交缺陷分类法”（Orthogonal Defect Classification，ODC）[13]，该方法的最新版本定义了8个缺陷属性，如表1所示。其中“缺陷类型”（Defect Type）属性不仅反映了缺陷的内在性质，而且反映了软件过程中所出现的问题。因为某种类型的缺限往往产生于特定的项目阶段，例如功能缺陷大多产生于需求分析阶段，而赋值/初始化缺陷大多产生于编码阶段。因此，根据缺陷类型数据对某一开发阶段进行持续改进，将会显著减少相应类型的缺陷的引入。其它属性如“目标”（Target）、“限定”（Qualifier）、“来源”（Source）等也从不同侧面反映了缺陷被引入的原因。对这些属性数据进行综合分析，可以客观地分析出软件过程中出现的问题和缺陷产生的原因。

表1　正交缺陷分类法的缺陷属性定义

属性	说明	取值	输入时机
Activity	检测出缺陷的活动	设计复审、代码审查、单元测试、功能测试、系统测试	报告缺陷
Triggers	使缺陷被激活的触发条件	逻辑/数据流、向前/向后兼容、负载、并发、顺序…	报告缺陷
Impact	缺陷对软件产品造成哪方面的影响	功能需求、可靠性、性能、可移植型、易维护型…	报告缺陷
Target	为了修复缺陷，需修改哪些软件实体	需求规约、设计、代码	修复缺陷
Defect Type	修复缺陷时，对软件做了哪方面的修改	赋值/初始化、算法、功能/类、时序、接口…	修复缺陷
Qualifier	是什么原因导致软件元素产生了缺陷	缺少软件元素、错误的软件元素、多余的软件元素	修复缺陷
Source	缺陷所在的软件产品的来源	内部开发、从构件库中复用、外包、移植	修复缺陷
Age	缺陷被引入的时机	遗留错误、开发新功能时引入、修改旧功能时引入、修复缺陷时引入	修复缺陷

缺陷属性值是在缺陷跟踪流程中的相应时机输入的。为了提高输入的效率和准确性，缺陷数据的定义要实用和精简，避免过多的属性和属性取值，且属性值之间要相互独立，没有重叠（即保持正交性）[12]。

缺陷管理工具不仅要实现缺陷跟踪流程，而且要提供缺

（下转第251页）

[9] Liedtke J. Toward real microkernel. comm. of ACM, 1996
[10] Spencer R, Smalley S, Hibler M, et al. The Flask Security Architecture: System Support for Diverse Security Policies // Proceedings of the 8th USENIX Security Symposium. 1999: 123-139
[11] Chou A, Yang J, Chelf B, et al. An Empirical Study of Operating System Errors // Proc. 18th ACM Symp. on Oper. Syst. Prin. 2001: 73-88
[12] Swift M, Annamalai M, Bershad B, et al. Recovering Device Drivers // Proc. Sixth Symp. On Oper. Syst. Design and Impl. 2004: 1-15
[13] CSC-STD－001-83. Department of Defense Standard, Department of Defense Trusted Computer System Evaluation Criteria, DoD Computer Security Center, Aug. 1983
[14] 石文昌. 安全操作系统研究的发展. 计算机科学, 2002, 29(6): 29(7)
[15] Ford B, Hibler M, Lepreau J, et al. Microkernels meet recursive virtual machines // Proceedings of the Symposium on Operating Systems Design and Implementations. New York: ACM Press, 1996: 137-151
[16] Anderson J P. Computer Security Technology Planning Study Volume II. ESD-TR-73-51, Vol. II, Electronic Systems Division. Air Force Systems Command, Hanscom Field, Bedford, MA, USA, Oct. 1972

(上接第 243 页)

陷数据的统计分析功能。须实现的最基本的统计功能是计算不同缺陷属性值所对应的缺陷数量及其百分比，统计结果要能够以表格和直方图(或饼状图)显示，从而可以明显地展示出缺陷相对于各属性值的分布，这种分布反映了导致软件缺陷的过程问题。这样，在缺陷管理工具的支持下，缺陷数据在软件开发过程中被实时地录入并实时地统计分析，将产品质量情况和软件过程问题快速地反馈给开发人员，使其采取相应的调整措施，减少缺陷的引入。

软件组织在不同的成熟度阶段，以及在开发不同领域的软件产品时，产生的缺陷信息会有所不同，缺陷的原因也会有所差别，这就要求缺陷管理工具具有缺陷属性及其值的自定义功能[12]。

基于缺陷分类的定量分析方法以客观的数据统计为基础，易于实现自动化和定量的控制，实时性好，分析覆盖率高，资源消耗小，适用于大型项目。但该方法的局限性在于缺陷数据的统计结果难以揭示有关人员和管理因素的深层次问题，往往只是这些问题的间接"反映"，因此通常还需结合定性方法作为补充。例如，如果通过统计发现某类型的缺陷相对于各开发阶段有明显异常的数量分布趋势，要对这种情况做进一步的定性分析，发现开发阶段中存在的有关人员、管理或技术上的深层次原因。此外，该方法在提高了分析抽象程度的同时，也容易忽略对一些特殊情况和个别严重缺陷的详细分析，有可能遗漏一些特定的缺陷原因。

结束语 软件工程不同于传统的工程学科，没有任何一种模型可以用来描述所有软件开发过程，也不可能完全实现自动化和定量的工程控制，在很多情况下开发人员必须根据当前项目的实际情况并结合经验数据做出主观判断。因此，软件组织在分析缺陷原因时，应融合定性方法和定量方法，把客观的数据统计与主观判断结合起来，具体方法是：

第一，在做定量分析时，结合定性方法得到有关人员、管理等方面的深层次原因和有效的解决方案；在做定性分析时根据需要收集数据，以量化的统计结果作为分析的依据和结果的验证。

第二，用定量的方法分析缺陷的宏观趋势，由此发现的共性问题再采用定性的方法进行具体分析。此外，对于会造成严重后果的缺陷，应采用定性的方法(如失效模式和影响分析、故障树分析等)进行逐个详细分析，以预防该类缺陷的产生。

在"能力成熟度模型集成(CMMI)"中，缺陷原因分析属于 level 5 的"原因分析与解决(CAR)"关键过程域，这在一定程度上给人们造成了一种印象，即只有成熟度级别很高、已完全实现了量化管理的软件组织才能进行缺陷原因分析和预防，实际上这是一种错误认识。在缺陷管理工具的支持下，将定性与定量方法相结合，处在较低成熟度级别(level 2 以上)的软件组织也可以分析缺陷的产生原因并采取相应的改进和预防措施[8]。在今后的研究工作中，我们将继续对缺陷原因分析的方法及其支持工具进行深入研究，使之更为实用和易用。

参 考 文 献

[1] 卡耐基梅隆大学软件工程研究所. 能力成熟度模型(CMM)：软件过程改进指南. 刘孟仁，等译. 北京：电子工业出版社，2001
[2] Rooney J J, Heuvel L N V. Root Cause Analysis for Beginners. Quality Progress, July 2004
[3] Doggett A M. A Statistical Comparison of Three Root Cause Analysis Tools. Journal of Industrial Technology, 2004, 20(2)
[4] U. S. Department of Energy. Root Cause Analysis Guidance Document. http://www.hss.doe.gov/NuclearSafety/techstds/standard/nst1004/nst1004.pdf, February 1992
[5] Fenton N, Pfleeger S L. 软件度量-严格而实用的方法. 第 2 版. 北京：机械工业出版社，2004
[6] Lanubile F, Shull F, Basili V. Experimenting with Error Abstraction in Requirements Documents // Proceedings of 5th International Symposium on Software Metrics. Bethesda, Maryland, 1998
[7] Leszak M, Perry D E, Stoll D. A Case Study on Root Cause Analysis // Proceedings of the 22nd International Conference on Software Engineering. Limerick, Ireland, 2000
[8] Mays R G, Jones C L, Holloway G J, et al. Experiences with Defect Prevention. IBM Systems Journal, 1990, 29 (1)
[9] Buglione L, Abran A. Introducing Root-cause Analysis and Orthogonal Defect Classification at Lower CMMI Maturity Levels // Proceedings of 1st International Conference on Software Process and Product Measurement. Cadiz, Spain, 2006
[10] Dunn A. Getting Root Cause Analysis to Work for You // Proceedings of International Conference of Maintenance Societies (ICOMS). Sydney, Australia, 2004
[11] Chillarege R. ODC-a 10x for Root Cause Analysis // Proceedings of RAM 2006 Workshop. Berkeley California, 2006
[12] 刘海，郝克刚. 软件缺陷数据的定义. 计算机应用，2008(1)
[13] IBM Research Center for Software Engineering. Orthogonal Defect Classification. http://www.research.ibm.com/softeng/ODC/ODC.HTM, 2002