

软件质量管理中的统计过程控制^{*}

SPC in Software Quality Management

赖旭芳 王 青

(中科院软件所先进软件技术联合实验室 北京100080)

Abstract SPC (Statistical Process Control) is a process control method which can help the organizations to measure and control the quality of the processes, and then to achieve the goal of improving the process stability. Thereby, the quality of products and service is advanced. SPC is used widely and maturely in manufacturing, but it is inchoative in software engineering. In this paper, we discuss the applying method and foreground of SPC used to manage and control the software quality. Some applying examples are given.

Keywords Statistical process control, SPC, Software quality management, Software Process, Process control

1. 引言

统计过程控制 (SPC, Statistical Process Control) 是一种借助数理统计工具的过程控制方法, 它利用统计工具及技术 (如控制图) 对过程或过程输出进行分析, 找出过程中的不确定因素并及时消除, 控制、管理、改进过程产品的质量或过程能力, 从而达到保证产品质量的目的。它能帮助用户采取适当措施来保证过程处于统计意义的受控状态, 并且帮助用户提高生产能力, 以满足或超越顾客的期望。

SPC 最早是在工业界提出的, 在工业界的成功也促使它被应用于其它许多商业领域。本文主要探讨统计过程控制在软件质量管理中的实施。

2. 统计过程控制 (SPC) 简介

SPC 及其相关的控制图是由美国贝尔实验室的 Walter A. Shewhart 在 20 世纪 20 年代提出的, 其主要目的是控制产品的成本和质量。Shewhart 的技术在二次世界大战时获得了广泛的使用。二战后, 由于国际竞争压力的减小, 许多好的科学管理方法, 例如 SPC, 逐渐从美国工业中消失。与此同时, 日本却通过 W. Edwards Deming 博士将 SPC 的概念和方法引入国内。经过近 30 年的努力, 到 1980 年, 日本已跃居世界质量与生产率的领先地位。鉴于日本的强有力的竞争和 SPC 在日本的成功, 从 80 年代起, 西方国家发起了复兴 SPC 的运动, 并取得了显著成效。

控制图是 SPC 的重要工具, 它通过把过程度量获得的数据以图示的方式表现出来, 让用户能直观地看出过程中存在的波动、异常和过程的趋势。一般它包括以下要素: 中心线 (CL)、上控制界线 (UCL)、下控制界线 (LCL)、样本统计量数值等。目前在实际中大量运用的是基于 Shewhart 原理的传统的控制图, 近年来又发展了许多其它有用的工具。主要有直方图 (histograms)、排列图 (Pareto charts)、原因-影响图 (cause and effect diagrams)、散布图 (scatter charts) 等等。利用控制图实施 SPC 将在本文第 5 节做更详细的介绍。

SPC 的目标是确定过程是否受控、确定过程是否符合要求和识别过程波动的原因。SPC 的主要特点有:

1. 强调贯穿于全过程、全系统的监控, 一旦发现异常, 立即分析原因并及时解决;

2. 实施 SPC 的目的不仅仅是发现问题, 而是用科学方法 (主要是统计技术) 来保证全过程的对问题的预防;

3. SPC 不仅适用于产品质量控制, 更可应用于一切管理过程的改进。

SPC 提出了一种思想, 通过仔细地分析数据来提高过程能力。SPC 与其它传统统计方法的不同之处在于它对消除缺陷原因提供了更有建设性的建议并有助于组织进行质量改进。这些传统方法包括问题管理、零缺陷 (zero defects) 和产品生产后的纠正和返工等。

SPC 发展到现在已经成为一门质量控制科学, 其理论已经非常成熟、完善, 各种 SPC 技术已达近百种之多。这些技术都是基于一个相同的基本原理, 即统计学中的小概率事件原理: “小概率事件在一次试验中是几乎不可能发生的, 一旦发生就可认为系统出现问题”。把此原理应用于工程领域, 可描述为: 假定一个过程处于稳定状态, 则观测结果显示为不稳定状态是一个小概率事件, 一旦出现这样的观测结果, 即可认为过程的稳定性被破坏, 于是需要及时调整。

应用 SPC 可以使组织获得以下收益: 定义企业核心过程; 对过程波动进行分析、监控和消除; 最小化或消除非预期的活动结果; 建立与实现企业目标; 提高产品质量; 改进企业的管理。

3. 软件质量管理

质量是指某一事物的特征或属性。软件质量可以定义为“符合明确表述的功能和性能要求, 符合明确的形成文件的开发标准, 同时包含专业性开发的软件应具有隐含特性”^[3]。该定义强调了三个要点:

1) 软件需求是进行“质量”度量的基础。不符合需求的产品不具有高质量。

2) 软件行业有自己特定的标准, 这些定义了一套开发准则, 用以指导软件开发的行为。如果这些准则没有被遵守, 几乎可以肯定软件缺少高质量。

3) 除用户明确提出的需求外, 还存在一套不被明确提出

^{*} 本文研究课题受国家 863 项目 863-306-ZD12-02-1 和国家自然科学基金 69896250-3 的支持。赖旭芳 硕士生, 主要研究方向为软件工程和软件工程质量管理。王 青 副研究员, 硕士生导师, 主要研究方向为软件工程和软件工程质量管理。

的“隐含需求”应被满足,例如高可维护性。若软件仅符合明确的需求而不符合隐含需求,软件的质量就值得怀疑。

高质量的产品能提高企业的竞争力,因此任何提供产品的企业都应建立质量管理制度,采取质量保证措施。

进行软件质量管理有两种基本途径:质量验证(测试)、过程控制。

软件的质量验证,即测试活动,是软件质量保证的关键步骤。测试的目的是发现软件中存在的缺陷和错误,以便决定过程能否进入下一阶段。但是仅靠测试活动并不足以保证软件的质量,因为测试并不能保证软件中没有错误,事实上,没有任何技术能保证已排除了软件中的所有错误。人是注定会犯错误的,测试活动只能起事后的检验作用。

要获得高质量的软件产品,除了提高软件开发技术外,还必须在开发过程的管理上下功夫,事先预防可能发生的错误。“测试”意味着质量控制,即度量出产品的质量;“质量保证”则还要求能度量出用以创造有质量的产品的流程的质量。如果能找出过程参数与产品的质量参数之间的关系,据此有针对性地改进过程,就能减少错误的发生,提高产品质量。

众所周知,软件质量有一个显著特点,即产品中的缺陷或错误发现得越晚,造成的损失就越大。所以将质量保证措施应用于整个在软件开发周期中具有更重要的意义。也就是说,应对整个软件过程进行控制。

过程控制、质量验证这两种途径不是孤立的,而是相辅相成的。质量验证应贯穿于过程控制,成为过程控制的辅助手段。

4. SPC 能用于软件质量保证领域吗?

乍一看,SPC 似乎无法应用于软件业。首先,软件产品无法精确度量,软件质量的高低难以用量化指标衡量,其产品质量的评价经常是主观的。其次,软件开发过程与其它制造过程也有很大不同。在其它制造行业,同样的产品重复生产,可以有统一的质量标准和完善的质量度量手段,通过对不同阶段的产品进行抽样,可获得各阶段产品质量的量化数据,并据此发现过程中存在的导致质量波动的原因。而在软件业,软件生产的关键过程是从需求分析到编码的所有过程,本身是创造性的生产,每一次生产都是针对不同的需求,所以可能采用不同的开发工具和技术,面向的领域也可能有显著差别,不同产品之间无法采用统一的产品质量验证标准。

然而,随着软件度量技术的发展,尽管还不可能有统一的量化软件质量标准,但是已经能够得到许多有价值的数据。而且,统计过程控制是针对过程而不是最终产品的。每次软件开发的产品虽然不同,但在成熟的软件企业,开发过程却是可重复的。尤其是在一个软件需要不断推出新版本的情况,每次的开发过程都可以在原开发过程的基础上进行改进,这为实施 SPC 提供了条件。在软件开发过程中应用 SPC 技术,可以揭示出过程中的潜在问题,不断改进开发过程,从而达到提高产品质量的目的。

有鉴于此,SEI 的 CMM 模型中第4和第5级成熟度就是建立于统计过程控制的基础上。

5. 软件过程的管理、度量和改进

要实施统计过程控制首先要确定“过程”的含义。在 SEI 的 CMM1.1 版中,过程被定义为“一系列步骤”,通过这些步骤,工作人员在工具和设备的帮助下把原材料转化为产品。这

个定义中未考虑人员、物资等因素,而现在我们要应用度量手段来确定改进产品质量和过程能力的可能性和时机,当寻找造成异常波动的原因时不考虑这些重要因素是一个错误。因此我们可以采用 ISO 9000:2000 的一个含义更广泛的定义:过程(Process)就是使用资源将输入转化为输出的活动的系统。根据这个定义,过程既包括活动,也包括资源。与此相适应,我们可以将软件过程定义为软件项目在生产中或组织生产软件产品时所应用的所有过程及子过程。

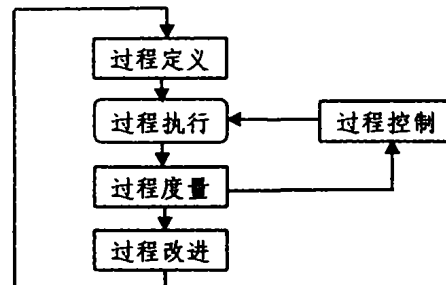


图1 过程管理示意图

定义过程的输入输出,制定详细的过程控制办法并为其分配资源就构成了过程管理。图1中的“过程定义”、“过程度量”、“过程控制”与“过程改进”就是过程管理的组成部分。过程管理的目标是保证组织的过程按预期执行,并实现其预定目标。软件过程管理是指对与软件产品或系统的开发、维护及支持活动相关的工作过程的管理。过程管理是否成功的标准是看过程产生的产品与服务是否完全符合用户的需求及该组织的商业目标。

所以,从软件工程的角度看,SPC 用于软件工程主要度量以下两个重要指标:

(1)过程稳定性,即过程行为和结果是否可预期。从 SPC 的角度来看,一个过程是受控的,并不是指该过程不会产生波动,而是指它能维持一个稳定的波动范围,从而它的过程结果是可预期的。这也被成为统计意义上的受控。这里的“可预期”不是指过程结果是同一的,只是意味着若过程处于统计受控状态,过程结果的波动处于一个可预期的限制范围内。

(2)过程能力,即过程是否能提供符合需求的产品,过程的表现是否符合该组织的商业需要。

“改进任一过程的唯一合理的方法是测量该过程的特定属性,并提出一套基于这些属性的有效的度量方法,然后应用这些度量方法来提供能指示改进策略的指标”^[3]。有效的度量方法能界定过程的重要事件和趋势,从而有效地将有用信号量与杂音分离。它对于指导软件组织作出明智的决定意义重大。

进行软件过程度量,首先要对测量活动作出计划,并把它与软件过程结合起来(测量应贯穿于整个软件过程)。实施过程度量第一步是收集并整理数据。其次需要对得到的数据进行分析。SPC 的方法在这方面扮演了重要的角色。分析数据的工作包括分离信号量与杂音,评估过程的稳定性(使用控制图等方法查明过程中存在的不稳定性和失控的情况),评估过程的能力(包括软件过程是否稳定、预期结果是否能满足客户的需求)。然后根据分析的结果,找出过程不稳定的原因并改正,从而改进软件过程。

6. 实施 SPC

6.1 SPC 基本概念

•波动 (Variation) 过程的波动包括两个部分: 常规波动 (common-cause variation) 和可指定原因波动 (assignable-cause variation)。常规波动又可称为系统波动, 是指当前过程固有的噪音。在一定的测量精度下, 任何过程都存在常规波动。常规波动的大小反映了过程的能力。可指定原因波动则是由非本过程事件引起的, 例如操作失误、环境变化等, 它是过程存在异常的信号量, 可以采取一定措施排除此类波动。当过程排除了可指定原因波动后, 它就处于稳定状态。

•数据 过程度量的数据有两类: (1) 变量; (2) 独立点或属性。变量数据是对连续现象的度量结果, 或记录尺寸、状态等的数值数据; 而独立点数据则有很大不同, 它记录事件出现的次数, 常常是从对数据项是/否符合要求的记录数产生的。独立点数据总是数值型的, 例如发现的缺陷数、每小时审查的源代码行数等等。分析这两类数据, 适用的 SPC 方法是不同的。例如 X-bar 图和 R 图常用于变量数据分析, 而 XmR 图 (独立点和移动范围图) 常用于对依时间顺序排列的一系列独立点数据的分析。

•CL、UCL、LCL 中心线 (CL)、上控制界线 (UCL)、下控制界线 (LCL) 是控制图中的主要组成部分。中心线表示过程的观察平均值, 通常采用度量数据的算术平均值, 但在某些特殊情况下可能采用中位数或波动范围的中值。上下控制界线用来指示过程的稳定域。它们的值是由过程度量数据计算得到。不同的数据采用不同控制图, UCL 和 LCL 的计算公式也不同。例如, 在 XmR 图中, 计算公式分别为:

$$UCL = \bar{X} + 2.66 * mRbar$$

$$CL = \bar{X}$$

$$LCL = \bar{X} - 2.66 * mRbar$$

其中, $\bar{X}$ 代表数据的平均值, $mRbar$ 代表相继数据对的差的绝对值的平均数。这里的 UCL 和 LCL 的公式是根据统计假设推导得到的, 现实中, 它们还应满足实际情况的限制, 如 LCL 可能必须不小于零。

在控制图中, 若存在度量数据点在控制线以外, 或代表数据的点形成了一定的模式, 如连续多个点在中心线同侧, 表明过程中存在不稳定因素。

6.2 实施 SPC 的步骤

图2中列出了实施 SPC 的主要步骤^[1]:

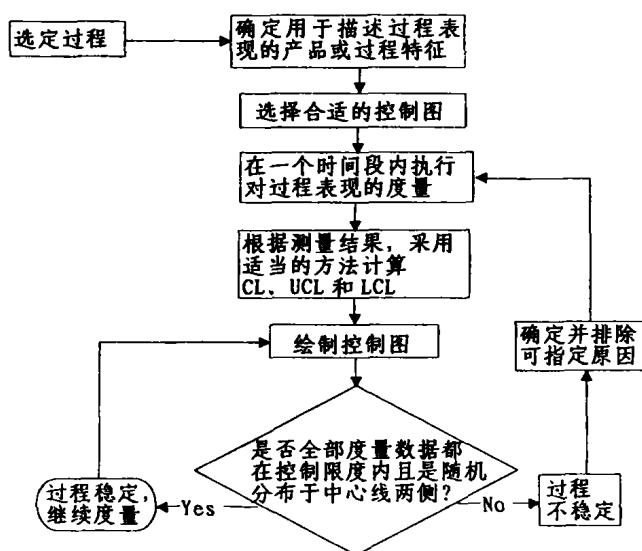


图2 使用控制图评估过程稳定性的步骤

6.3 在软件质量管理中实施 SPC

目前, 已经有不少将 SPC 技术应用于软件行业的例子, 但它们多是集中于比较成熟的软件企业。应用 SPC 对数据进行分析时, 首先应把度量结果数据进行合理分组, 使同组数据具有统一的波动原因。例如, 不同子过程的数据应分别进行分析。如果简单地把所有数据绘制成一张控制图, 很可能掩盖了系统中的异常现象, 或使异常的原因难于分析。只有区分不同来源 (子过程) 的数据, 发掘真正的不稳定因素及其原因, 以得到理想的过程改进方法。在分析过程中, 还可以对数据采样过程进行改进, 以提高将来的过程控制效率。

下面引用的数据来自于美国 United Space Alliance 和 SEI 在航天飞机机载软件项目中实施 SPC 的合作^[2]。所用数据从软件审查 (Inspection) 过程中得到。主要度量的对象是新增或修改过的源代码行数和审查工作量。由此可以计算审查的复查率, 即平均每审查小时所复查的源代码行数。根据度量数据的类型, 这里选用 XmR 图进行分析。

图3中显示了一组对其中一个模块的复审数据。上半部分是相继30轮审查的每小时审查源代码行数, 下半部分是每组数据中的离散情况。仔细分析审查记录, 可以发现图中的那些超出限界的点都是从对各种数据结构的审查得到的, 如数据列表、数据库表和数组。对审查人员和开发人员的调查表明, 对这部分资料的审查工作与事先设计的过程不同, 其中使用了某些辅助的验证工具。因此, 即使审查过程设计者没有意识到, 对这部分数据和对其它数据的审查应是分属不同的过程。分析人员于是把这些数据分离, 它们在图4中表示为“x”。

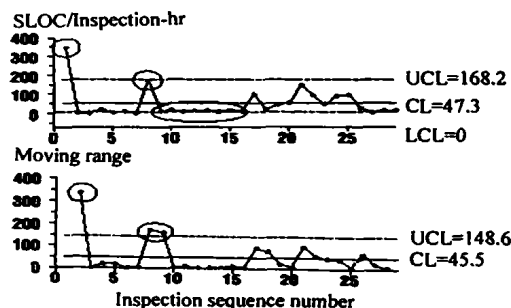


图3 一组审查数据

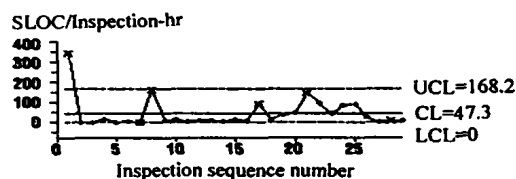


图4 分属不同原因的数据

另外, 注意到图3上半部分中从第9点到第16点有连续8个点处于平均线以下, 这预示着还有未发现的可指定原因, 可对其进行进一步分析。

排除前面指出的存在其它原因的数据, 并增加后续的审查产生的数据, 得到了图5。其中将审查数据按审查小组的规模进行了分离。显然从图5上可以判断过程中存在异常。通过分析越界数据, 发现它们是由再审查 (Reinspection) 过程得到的。这是由于有时设计未能满足需求或设计较复杂, 使审查小组认为应再次审查造成的。由于是再审查, 因此其速度相对较快。图中标注“x”的点就是由再审查得到的数据。排除再审查数据后, 得到图6。从该图可以看出, 这个软件过程是一个相对

稳定的过程。

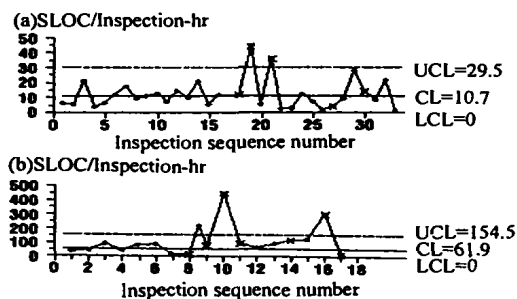


图5 (a)审查率小于60 SLOC 的审查小组的数据
(b)审查率大于60 SLOC 的审查小组的数据

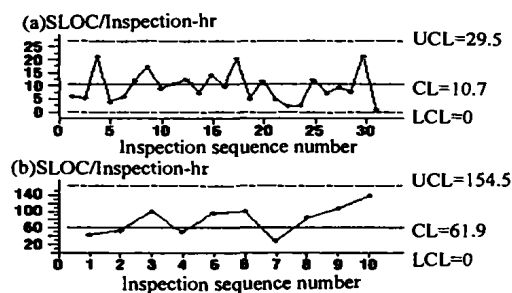


图6 排除再审查数据后控制图

需要注意的一点是,UCL、CL 和 LCL 的值都是通过计算得出的,增加了数据后应重新计算,以获得正确的结果.XmR 图的 UCL、CL、LCL 计算公式参见5.1节。

上文所举的 SPC 的应用实例是针对软件审查过程的,是通过 SPC 的方法判断审查过程的稳定性.这个例子有一定的特殊性.事实上,SPC 在软件质量管理中的应用远不止于此.例如,我们还可以用 SPC 方法来分析稳定的审查和测试过程产生的结果数据(如前文所述,可通过 SPC 方法判断过程的稳定性),以对产品质量进行评估,并可对将来正式发布版本的质量作出一定程度的预测.如果审查和测试活动发现的缺陷率保持稳定的话,说明开发过程应该是稳定的,产品的质量也保持稳定,这时我们可以根据中心线代表的值预测最终产品的质量;如果缺陷率出现异常,说明开发过程存在异常,可通过分析异常原因,对其进行改进。

结论 随着软件工程的发展,软件生产工程化的特点愈

来愈突出,像 SPC 这样的工程质量控制技术将会愈来愈多地用于软件工程领域.有效地实施 SPC 需要注意以下几点:

(1)识别和定义软件工程的关键过程,确定可以实施 SPC 的比较稳定的软件过程,形式化并确定稳定过程的性能参数,坚持收集整理度量数据,理解并明确数据的含义。

(2)数据的质量非常重要.将质量差的数据用于分析,不可能得到正确的结论.这要求度量技术能有进一步的发展.选择什么度量方法,度量时如何采样都关系到 SPC 实施的成效。

(3)应用 SPC 方法需要对各种数据进行分析,虽然可以有相应的软件工具提供帮助,但发掘问题的原因,提出解决方案等工作只能是由有经验的管理人员来做.因此对于管理者的培训是十分重要的。

虽然,现在软件行业中 SPC 的应用仍不普遍,目前还只有有限的在高成熟度组织(CMM4级或5级)中的成功应用实例,但我们相信 SPC 技术可以在所有制定了度量程序的组织(2级以上)中实施并取得成效。

对实施 SPC 的收益应有正确的态度.SPC 只是用于帮助管理者保证当前定义的过程处于受控状态.它并不能保证产品满足顾客需要.如果一个受控过程生产的产品仍不符合顾客的需求,说明该过程定义不当.此时应从过程改进入手,采取更好的生产技术,提高过程的能力。

参考文献

- 1 Florac W A, et al. Practical Software Measurement: Measuring for Process Management and Improvement: [Tech Report CMU/SEI-97-hb003]. Software Eng. Inst., Carnegie Mellon Univ., Pittsburgh, 1997
- 2 Florac W A, et al. Statistical Process Control: Analyzing a Space Shuttle Onboard Software Process. IEEE SOFTWARE July/August 2000
- 3 Pressman R S. Software Engineering A Practitioner's Approach. McGraw-Hill, 1997
- 4 Weller E F. Practical Applications of Statistical Process Control. IEEE Software, May/June 2000
- 5 Paulk M, et al. Capability Maturity ModelSM for Software, Version 1.1: [Tech Report CMU/SEI-93-TR-024]. Software Eng. Inst., Carnegie Mellon Univ., Pittsburgh, 1993
- 6 Pressman R S 著. 黄柏素, 梅宏译. 软件工程——实践者的研究方法. 机械工业出版社, 1999

(上接第119页)

- 55 Li C N, Thompson S A. Mandarin Chinese: A Function Reference Grammar. University of California Press, 1981
- 56 Bai Bo-Ren, Tseng Chiu-Yu, Lee Lin-Shan. A Multi-Phase Approach For Fast Spotting Of Large Vocabulary Chinese Keywords From Mandarin Speech Using Prosodic Information. ICASSP-97. 903~906
- 57 Lin Chia-Hsien, Wang Hsiao-Chuan. Keyword Spotting by Searching the Syllable Lattices. ICSLP-2000
- 58 Lin C H, et al. Frameworks for Recognition of Mandarin Syllables with Tones Using Sub-Syllabic Units. In Speech Communication, Elsevier Science B. V, 1996. 18
- 59 Seide F, Wang N J C. Two-Stream Modeling of Mandarin Tones. ICSLP-2000
- 60 Gao Jianfeng, et al. A Unified Approach to Statistical Language

Modeling for Chinese. ICASSP-2000

- 61 Zhang Hong, Xu Bo, Huang Taiyi. How to Choose Training Set for Language Modeling. ICSLP-2000
- 62 Caminero J, López E, Hernández L. Two-Pass Utterance Verification Algorithm for Long Natural Numbers Recognition. ICSLP-98
- 63 Gu Yong, Thomas T. Competition-based Score Analysis for Utterance Verification in Name Recognition. ICSLP-2000
- 64 Gu Yong, Thomas T. A Hybrid Score Measurement For HMM-Based Speaker Verification. ICASSP-99
- 65 Jouvet D, Bartkova K, Mercier G. Hypothesis Dependent Threshold Setting for Improved Out-of-Vocabulary Data Rejection. ICASSP-99
- 66 Bazzi I, Glass J R. Modeling Out-of-Vocabulary Words for Robust Speech Recognition. ICSLP-2000