文章编号:1007 – 5429(2002)01 – 0009 – 05

微软 WORD 格式中的软件质量保证

[美] 刘国柱
(美国 Rogerson Kratos 公司)
缪卓群 译 林益耀 校

摘要:“在廉价的甜味逐渐减弱消失后,就会倍尝长期持续的苦味。”高质量软件的生命周期成本只是低质量软件的生命周期成本的几分之一。在一些情况下,质量保证的软件对于使用该软件的客户意味着生命悠关。例如,飞行员读了令人误解的荧屏显示就有可能以付出他(她)以及乘客的生命作为代价。软件质量保证方法或过程必须标准化并用于确保软件产品的安全以及成本效果。软件质量保证应占项目总成本的 6%。

关键词:软件质量保证;软件;质量保证
中图分类号:TP311.1 文献标识码:A

Software Quality Assurance in Microsoft Word Format

Joe LIU
(Rogerson Kratos Co., USA)

Abstract: “The taste of the bitterness lasted long after the sweet taste of the cheap price is worn off.” The life cycle cost of quality software is many times cheaper than poor quality software. In some cases, quality assured software means life and death to the client that uses the software. For example, a pilot reading a misleading display may cost his or her life and the passengers. Software quality assurance method/process must be standardized and applied to ensure the safeties and cost effectiveness of a software product. Software Quality Assurance should account for 6% of the total project cost.

Key words: software quality assurance; software; quality assurance

1 引言

为筹备奥运会的游览观光,亚特兰大市引入现代的行李传送系统使机场升级。在系统试运行时,行李从起运点到它的终点需要花费数个小时。另外,为了旧金山大桥的管理,当地市政府定合同包出了一个交通控制系统。数年过去了,花费了数百万的资金,系统还未能交付使用。这些是由于低质量软件的开发过程所导致的项目失败的几个事例。事实上,软件开发过程日益复杂,它正在阻碍着美国高科技的进步。另有一例是石油加工厂控制软件,该软件的编制和维护均很差,它已花费数百万美元来修补,而且还不包括检修时设备的停工损失(石油加工厂的停工与再启动就要花费数百万美元)。软件的开发费用是很昂贵的。微软对每个软件产品开发费用达数十个亿,尽管如此,使用者还会不时地遇到微软产品的‘应用出错’的问题。

软件开发有时候可以用一座冰山来描述。当一个缺乏经验的软件开发员或一个用户着手开发一个软件项目时,他首先想到的仅是编码/编程序(冰山的山顶)或最后的软件产品。他没想到最终的软件包仅是整个开发结果的一小部分,而整个软件开发结果包括下列各项:

——主计算机;
——接口设备;
——软件开发工具(软件和硬件);
——仿真器(软件和硬件);
——测试工具;
——文档(用户手册)。

工业软件开发成本见表 1。

注:本文系作者直接投稿,原文为英文。
收稿日期:2001 – 07 – 10
作者简介:刘国柱(1958 –),男,美籍华人,美国 Rogerson Kratos 公司项目工程师。

表 1　工业软件开发成本

项目	成本百分比
需求	12%
体系结构	8%
详细的设计	10%
编码	14%
工具开发	8%
测试	28%
质量保证	6%
管理	8%
其他	6%

注 这是一个高效的管理项目，而一些军事/政府的项目则需要 30% 的管理成本。

在今天的环境下，一个百万美元的项目已被认为是个小项目。大多数项目要高得多，因此，从上述例子中可以看到，为软件开发失败所付的代价是昂贵的。大多数软件的获取和管理是否成功，是以成本、时间进度、性能、可支持性，以及最重要的质量等因素来衡量的。

软件质量可用结构复杂度（Structure Complexity）和模块复杂度（Module Complexity）来衡量。结构复杂度是衡量线性独立路径的，也即衡量对模块而言必须经过检测的那些路径的最小数目（模块被定义为最小的软件单位——子例行程序、功能）。模块复杂度是衡量决策结构的，后者系控制模块中的直接辅助模块的调用。如果结构复杂度和模块复杂度数处于高位，那么所形成的软件将会有如下的问题：

——错综复杂的逻辑；
——矛盾的决策；
——不可行的测试路径；
——对体系结构改进的约束；
——将知识传给新同事有困难。

结构复杂度数应该低于 20。美国石油加工厂控制软件的结构复杂度数超过了 2000。

2　美国的软件质量保证标准

软件开发过程是一个“NP Complete”问题，而“NP Complete”是一个软件术语，它表示计算机逻辑执行的无限组合。由于软件的执行依赖于时间、物理环境（温度）、人机交互以及不可控制事件等因素，因此不可能完全地校验软件的完整性。美国的各个署基于这一原因，制定了 DO-178B（用于运输部门）和 MIL-STD2167（用于军事部门）标准来保证高质量软件产品的开发。

这两个标准，除了 MIL-STD2167标准在文档格式上更严格以外，是相类似的。它们的生命周期数据文档格式规定由国防部开发软件的承包人员填写。DO-178B 标准开发过程是较困难的，因为它需经指定的工程代表（Designated Engineering Representative，DER）审计和联邦航空署（Federal Aviation Agency，FAA）代理的批准而不是客户和软件开发人员的认可。

软件质量保证标准涉及了以下事项：

——涉及软件开发的系统方面问题；
——软件生命周期过程、定义和过程之间的转换准则；
——软件规划过程；
——软件开发过程；
——软件验证过程；
——软件测试过程；
——软件配置管理过程；
——软件质量保证过程；
——认证联络过程；
——软件生命周期数据；
——额外需要考虑的事项；
——前期开发软件的使用；
——工具合格性；
——替换方法。

表 2 对一些标准进行了对比。

表 2　各标准特点比较

标准	概要
ISO 9X（ISO 9001） 质量系统：设计/开发、生产、安装和服务方面的质量保证模型。	最常用的国际标准。有宽广的包含性。过程最少。主观的。被广泛应用，但软件的过程/质量较差。
IEC 1508 功能性安全：与系统相关的安全。	用于中度高可靠性系统的一般标准。坚实的基础，强有力的测试。在过程验证（人为因素）和先验法的建立方面能力差。
EN 5018 铁道系统应用：铁道管理和保护系统的软件。	应用于铁道工业。适用但能力差。
FDA QSM 美国食品和药品管理署质量系统	“Good Manufacturing Process”的改进产品标准较新，但在客观的软件准则方面能力差。
IEC 880 核电站安全系统中的计算机软件	技术性。关于编过程序和冗余度和规划等方面很好的标准
RTCA/DO 178B 空运系统及其设备认证方面软件的依据	美国商务航空工业的主要指导文件。在过程与生命周期方面能力强。有些解释内容，但属泛述。
MIL-STD 2167 用于国防部（军事）的标准	种类广。文档和格式性强。高的开销，合同为主。

表 3 给出了软件成本因素的比较。

表 3 软件成本因素的比较

属性	商用软件	信息系统	国防	电讯	医疗仪器	航空电子
产品规模(LOC)	100k – 10M	1M	50k – 5M	500k	10k – 50k	10k – 1M
产品生命期	1 年	4 年	15 年	3 – 5 年	3 年	7 – 10 年
升级频率	6 月	2 年	8 年	3 年	2 年	3 – 4 年
维护	10%	100%	50%	200%	100%	200%
上市时间	8 月	1.5 年	6 年	1 年	1.5 年	2 – 3 年
LOC 生产率	100/天	30/天	2/天	10/天	15/天	4 – 5/天
微小缺陷的成本	$0	$10k	$100k – 500k	0 – $100k	0 – $100k	$100k – $500k
重大缺陷的成本	极小量	$10k – $500k	0 – $20M	$500k	$100k – $5M	$1M – $500M

注释:LOC = 代码行数;M = 1000000;k = 1000;$ = 美元

DO-178B 认证的成本高达 $500k – $1M 的理由是因为软件中的重大缺陷可能会导致 5 亿美元以上的损失。试想象一下,如果满载乘客的航班突然失事会导致多大的损失?

3 软件质量保证工程师

软件质量保证(SQA)在于验证并确保与合同或公司的需求相一致,其基础是将基本质量要求概念标准化,并确保软件的开发符合被认可的标准和程序。

所包含的标准由合同以及具体的一系列条款和采购订单所定义。这里包括所有必需的政府标准、或最优的商务惯例、以及与所确定的软件标准和规划相一致;而质量可作为独立的款项加入到所制定的软件质量指令中。

软件质量保证(SQA)不可加入任何不属于质量标准惯例内的方法、程序、指令或需求。然而,这并不限制软件质量包含能增强全面质量效果的新方法和程序。

软件质量保证工程师(SQAE),精通软件质量方法和程序并在这方面有经验的将对 SQA 的功能负责。SQAE 人员的资格应基于(1)教育程度;(2)工作经历;(3)实际工作能力等职务说明指标来确定。SQAE 应能读懂软件编码以及了解软件开发方法知识。SQAE 应向独立于软件开发部门的质量保证部门报告。

SQAE 并不限于任何计划项目的传统检查/审计活动。SQAE 将参与并使用校验审计法来获取客观的证据,并且在所有相关计划/项目的整个生命周期中成为一名软件开发团队中能起作用的一员。

4 工作方法

SQAE 应参与软件开发的每一个阶段(假定是以下的"瀑布模式"——一个阶段顺序地接着另一个阶段):

(1) 规划;
(2) 要求设计;
(3) 设计;
(4) 编码;
(5) 集成软件;
(6) 确认和验证。

在批准阶段转换时,SQAE 也是一个关键的人物。项目的三个主要里程碑是:计划设计审查(PDR)、关键设计审查(CDR)、测试准备审查(TRR)。以上 PDR 是批准规划,CDR 是批准设计,TRR 是批准软件。

4.1 规划阶段

在项目被启动后,SQAE 要参加项目领导,建立规划文档。SQAE 同时促成软件验证计划与软件认证计划,并作出软件质量保证计划。所有这些计划首先需要被 DER 而不是 FAA 的批准。

4.2 设计阶段

在设计阶段,为要与标准及程序相符,SQAE 将参与 PDR 和 CDR。

4.3 文档

SQAE 应按照所确定的产品规范审查所有的软件生命周期数据/文档(程序、计划、标准等),以求内容和技术参数完整。应建立起项目档案库(可以保持在正规备份的服务器内),以记录项目所有的活动。在档案库中应包含但不仅限于以下各项的内容:

——项目生命周期数据;
——软件各项确认的规划;
——软件开发计划;
——软件质量保证计划;

——软件配置管理计划；
——软件开发标准；
——软件验证计划；
——系统要求规范；
——软件需求文件(高级别软件需求)；
——设计记述文件(低级别软件需求)；
——软件源代码；
——软件测试步骤(闭箱测试步骤)；
——软件增补性测试步骤(开箱测试步骤)；
——软件验证报告；
——测试有效范围分析；
——软件配置索引；
——软件完成工作概要；
——会议记录；
——通讯—电话转换,各办公室间的备忘录；
——实际测试报告,审查校验表,审计报告,问题报告；
——所有硬拷贝资料。

4.4 软件开发文件夹(Sofware Development Folder，SDF)

在软件开发期间,应在计划/项目中保持软件开发文件夹。该文件夹连封面一起用 SQI-003 的格式。每一个项目的计算机软件配置项(Computer Software Configuration Item,CSCI)都应建立起文件夹。创建/生成每个项目 CSCI 的 SDF,并在整个软件开发生命周期内保持 SDF 的内容是软件工程师本身的职责。软件工程项目领导的责任是检查和审核每一个 SDF。在检查文本基线和测试准备之前,SQAE 不介入检查和审核。TDR 必须在确认和验证阶段启动前完成。SDF 应包括但不仅限于以下各项：
——SDF 包含校验单；
——设计注释；
——审查注释；
——在软件生命周期以外的所有其他数据。

在 MIL－STD2167 中,每一个编程员都被要求保持他(她)自己的 SDF。任何注释(手写或打字的)都必须被保存在 SDF 内。实验室笔记本是 SDF 中主要的一项。因为军事产品含有机密信息,SDF 本身可分类,而不可以从保密区域内复制或取出。

4.5 代码检查

软件工程师要进行代码检查,并提交给 SQAE 审查和批准。SQAE 应随机地对校验单和软件模块样品进行独立的审查。代码检查应该核查但不仅限于以下的对象：
——风格/格式符合软件开发标准以建立软件工程；
——源代码可追踪到低级需求；
——软件代码编译及链接无错误和警告；
——软件代码按单元测试；
——代码设计的客观证据存在于 SDF 中。

代码检查校验单作为历史证据保存在 SDF 中。该信息应包括所有工程注释,并由软件组和质量工程组作为工作工具使用。随着开发的进展,每一个增加到 SDF 中的新的代码检查校验单将有助于提高它们的审计质量。

4.6 基线/软件的质量担保/软件生命周期

被认可的提供给工程服务的软件/固件的基线应该有质量担保。在媒体(磁盘,H/W,ROM/PROM/EPROM 等)的显眼的地方应有质量担保的封印和 SQA 的工程标记。用这样的媒体才能进行正式测试,从而对客户确保固件的质量控制基线的完整性。作为工程服务担保一部分的所有文档和列表被归入质量控制并遵守同样的担保需求。

SQA 部门进行的软件生命周期监控是通过正在进行的检查和分析,从软件的需求过程开始,经软件设计过程,再经过编码过程,最后以测试和集成过程软件需求的方法结束。

4.7 测试见证人及参与

SQA 部门应该是批准对生产性验证和确认(V&V)、以及验收测试程序(ATP)的所有正式测试的联络点和权威。SQA 部门应根据被认可并已颁发的测试程序参与这些测试。另外,SQA 部门有权委派见证人作为 SQA 部门的代表。在与客户一起作任何正式的测试前,应按被批准的程序先做预测试。对于认证测试,SQA 必须通过每个方块图验证符合度验证测试设备是否满足软件认证计划的需求。在测试期间,SQA 参与作出任何克服故障的决策。这包括测试期间的客户信息以及在任何测试前和测试过程中的会议资料/数据的检查。

在测试期间,软件工程师应实施测试任务。SQA 部门应监测并在测试数据纸上记录测试结果。测试的执行应直至完成所有的测试程序为止。如果某测试项失败,软件工程师要进行分析以确定失败的实际原因。如果失败是功能性的(不符合需求),那么整个测试被认为是失败的;必须报告所存在的问题,并从软件开发过程返回到编码阶段。如果问题的性质是程序方面的,或仅是排印上的错误,则应校正测试程序,并重复该项测试。

4.8 校正工作

校正工作在问题报告中(在生产软件发布前的工程阶段)或在系统问题报告中(在生产软件发布后)要有资料证明。在生产软件发布后,校正工作必须经处理,并且得到 SQA 部门的批准。未完成的校正工作或对有关原因和校正工作所作的未获批准的处理将导致有条件质量的验收(这意味着可交付产品的正式质量验收因解决矛盾而成为悬案)。

注释:不是所有的校正工作都会导致软件的修改。校正工作也有可能是需求的印刷错误或对需求的澄清。

4.9 对基线的修正

任何对于已制定的软件—固件基线的修正应由软件质量来校验。软件—固件审计跟踪应在开发周期尽早建立起来,以便对于软件—固件基线的修正进行跟踪。所有对于基线的配置的修改应按工程指令单(Engineering Orders, EO)实施,并经由软件修改管理委员会(Software Change Control Board, SCCB)提交。

软件配置管理(Software Configuration Management, SCM)通过控制对文档及代码文件的修改保持源代码和文档的完整性,修改请求被跟踪,并经正式批准过程批准。项目修改仅在收到相关的正式批准单后才作出。具体的软件配置管理过程的描述包含在软件配置管理计划中。

对于每个软件修改版本,SQA 部门检查以下的客观证据:

——问题报告(工程)或系统问题报告(产品)已被批准和处理;

——校验表(代码检查,文档检查)已完成;

——测试(单元或集成)记录;

——软件配置索引已更新;

——必需的硬件标记已建立。

4.10 符合度

软件的符合度由 SQAE 或有资格的代表处理。符合度的检查是通过一系列的目视以及校验表单的检查来实施。由上述活动产生的客观证据将决定所规划的软件—固件生命周期过程,包括软件生命周期数据的生成在内,是否完成并与下述各项相符:

——软件规划文件、软件需求文件、以及软件测试程序已被批准和颁布;

——问题报告已被处理;

——异常和偏差作了记录并被认可;

——经代码检查并在配置受控情况下颁布了软件代码;

——软件需求可跟踪到测试程序。

用于执行高级别需求硬件—软件集成测试的符合度校验表,应符合所附的最高级别的条款验证校验单的标准。符合度的陈述应由质量保证部门产生及签署,而验收书由各客户的代表签署。符合度所有的客观证据应被纳入 SDF 内,并在采购订单合同期间存放在 SQA 的文档中。

在符合度的检查期间,是由 FAA 还是 DER 参与,这始终是一个由 SQA 支持的选项。

4.11 销售/客户验收/认证书

SQA 应参与合同中的出售活动和验收阶段。在交付产品的同时形成一个经认可的签名单是恰当的,这应包括计划办公室、合同、测试、工程和质量各部门以及客户。