计算机软件工程

质量保证与产品可靠性评审测试技术新标准实务全书

◎ 主编 王志锦 罗乔欣

本手册不单独定价出售，随光盘附赠

计算机软件工程质量保证与产品可靠性评审测试技术新标准实务全书

主编　王志锦　罗乔欣

第四卷

第 6 章

管理高速 Web 测试

第一节　项目背景

这个案例分析所用的项目是为公司新开发的一个 Web 站点。文中的“公司”由一些大公司合并而成，是欧洲领先的 IT 咨询公司之一。这个项目的目标是为一个公司发布一条消息。

这个项目是相当简单的：一个用于发布多语言企业信息的工具和一个先

进的搜索工具，不涉及电子商务。要测试的特性是发布工具和搜索工具的功能性。这必须在 InternetExplorer 和 Netscape 4.0 版本或以上版本中测试，而且浏览器运行于 Windows 98、Windows 2000 和 Windows NT 上。

Web 开发是由这个公司的一个子公司（即开发者）完成的，它们对质量和软件测试的态度是很有趣的："我们是伙伴，我们会把这个整理出来，不需要质量保证和测试。我们是相互帮助的。"

几个月之后，公司的项目经理开始变得忧虑了；并没有任何迹象表明开发者计划了质量控制和软件测试配置控制。因此，项目经理开始着手写一份测试计划审查报告（根据 1999 年 Pol 等人记录的 TPI）。

这份审查报告证实了项目经理的忧虑。接下来他决定开始用户验收测试。用于准备、做文档和计划的时间是非常有限的，但问题是如何能在用户验收之前发现错误呢？

第二节　"高速"方法和准备

作一个小的类比，当我还是小孩子的时候，有时我在就寝后半个小时还不睡觉，每次只是为了看看我的父母能忍受我多久才会发怒。我发现母亲总是先于父亲命令我上床睡觉，那就意味着这两个"系统"有不同的忍耐度。我只是想看看会发生什么。我认为这是我做的第一次探索性测试。有一两次，是我和我的兄弟一起这样做的，我们两个人在一起有好处，会感到更加有力。虽然结局通常是一样的，但是这使得我们把忍耐度变大了一些。这与软件测试有很多相同之处。您知道系统会阻止您访问受限制的信息，但是真的会那样吗？两人联合经常会带给您更多的方法。

当然这是一个再简单不过的例子，但它仍然强调出了要点。探索性测试和双人测试都不是新的革命性的方法。尽管如此，但是把它们应用于一个系统，在计划的和结构化的软件测试会话中应用它们，这些都证明了它们会产生令人惊奇的结果。

一、探索性测试

应用双人和探索性方法的思想来源于 James Bach 和他在 www.satisfice.com 上发表的论文。就像 James Bach 提到的那样，探索性测试实际上是一个可传授的方法，测试的成功取决于测试人员和测试经理的技能。因为“临时技巧”这个词看上去没有使大家理解这一点，所以他用了“探索性”这个术语。探索性测试依靠测试人员的知识水平和创造力。没有这些，测试就变得毫无意义。探索性测试的最简单的定义是测试设计和测试执行同时进行。前一次做的测试结果会影响到我们下一次要做的测试，在这种意义上来讲，我们就是在做探索性测试。当我们不能断定在测试周期之前我们应该进行什么测试，或者当我们还没有机会进行那些测试的时候，我们更具有探索性。

在计划、写脚本和执行过程中，探索性技术的应用要比大多数人所认识的还要频繁。甚至写详细测试脚本的时候，人们经常在写脚本之前采用探索性方法以使他们自己熟悉这套软件。完全脚本化测试和完全探索性测试（“故障追查”）之间的差异可以在图 7-6-1 中形象地表示出来，在图中，两端代表两种应用的极端，在中间，您会发现探索性测试的不同级别。

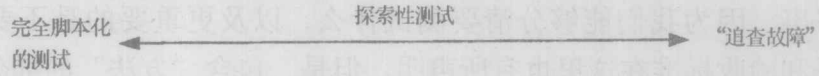

图 7-6-1

二、双人测试

就像这个术语所暗示的那样，双人测试是两个人共同在一台计算机上工作。这样做主要的好处就是测试人员变得更加有信心和有创造力。有信心是因为他们能在测试中讨论一些想法和观点，并且可以相互支持。有创造力是因为两个人思考胜过一个人。一个人的建议可能会使另外一个人产生新的想法，反之亦然。用 Cem Kaner 的话来说就是：“我们注意到了一些高生产力、高创造性工作的例子，这样的工作中把测试人员组织在一起，来分析一个产

品，或策划一个测试，或运行一系列的测试。我们也把它看作或用作一种有效的训练方法。”

第三节　“高速”的测试计划

在挪威，我们有这样一条格言：“好的开始是成功的一半”。但我们的工作一开始就遇到了困难。系统缺少文档，而且可用的文档仅有用户手册和“白皮书”。此外，熟悉系统和开发的人员难于接近。项目经理似乎也意识到了这一点，这是他倡导我们做验收测试的原因之一。我们开始时先用标准的、非探索性的方法进行计划和脚本编写。在制定实际的测试执行计划之前，大量的时间用在了“试验”门户和发布工具上。实际上这个系统并不是十分庞大，因此对我们来说这似乎是一个可行的任务。但在这个阶段中，开发人员一直在别人不知道的情况下对代码做一些突然的改变，这导致了许多问题。过了一段时间这就变成了办公室里的笑料，我们每天早上都会不停地猜测系统今天将怎样工作。

在非常了解这个系统之前，我们在第一周就已经制定了验收测试计划。这是件好事，因为我们能够分清要测试什么，以及更重要的是不要测试什么。责任和验收标准在这里也有所声明。但是，包含“方法”的部分描述得相当肤浅。在经过一段“试验”系统之后，我们就开始写测试脚本，但是不久就意识到这样下去不会有好的结果。我们没有足够的信息，而且系统总是在不断地变化。从项目经理那里我们获取了一些关于新方法的信息，新方法被称为探索性测试和双人测试（www.satisfice.com，2001）。在制定执行计划或“脚本”时和在实际的测试执行过程中，我们变得更加灵活，因此这看上去像是一个有趣的方法。而且，我们也可以在测试中使用每个测试人员的经验和创造力。但是，我们到底应该如何在两天内建立这样一个测试，如何执行它并且保证面面俱到呢？

在许多方面这是非常特殊的测试任务，不久我们意识到我们管理测试的方法将会是最重要的成功因素之一。也许比实际使用的方法还重要。令我们为难的管理问题之一是报告和文档。我们的经理和顾问都关注这个问题，并

且都提前强调了这个问题。我们提出的是一种简化的偏离形式，这种形式易于记录。此外，我们对每一个双人组进行密切监督，以确保我们真正地写出了报告。每一个阶段之后，我们都要在测试人员休息时收集和阅读报告，这是为了趁着这一阶段仍然“在头脑中印象深刻”的时候提出报告中的问题。

我们还缺乏时间，所以为了能够及时做完，我们不得不高效地利用每一分钟。我们在完成计划书之后，就会按约定得到一个测试试验室，在这个实验室中一切都应该“准备就绪”。但是我们仍然需要随身携带所有浏览器的安装文件，包括备份计划 B、C 和 D，以防万一。而且我们确实用到过其中的一些。至少这教会了我们一件事——不要完全相信电话里的人。如果您想用某种方式做事，那么您最好亲自检查一下。我们让自己早上用两个小时来检查测试环境，而且必须利用好这两个小时。但是充分的准备和风险分析却节约了时间。例如，我们需要 Internet Explorer（IE）4.0，但是 Microson 的软件都是升级的，而不是降级，因此，如果每台计算机都安装了 IE 5.0，而我们想要用 IE 4.0，那么最好知道如何绕过错误的消息。

“在观察领域，机遇只垂青那些有准备的头脑。”

这是路易斯 . 巴斯德写下的话，与医学中一样，这句话同样适用于软件测试。甚至在进行探索性测试的时候，一个人需要计划、激励、引导和监督。因此要认真对待您的准备工作，而且要记住“好的开始是成功的一半”。

第四节　“高速”的测试执行

“高速”，我们指的是测试的速度，而不是性能测试。高速是必要的。在两天之内，我们要对一个事先没有测试过的系统进行验收测试。当我们到达实验室的时候，准备好了的计算机即便有，也不会很多。

我们拿着备份计划和安装盘，到了上午 10 点，准备好了测试环境，测试人员也赶到了。这些测试人员主要是此软件的最终用户，在测试前他们进行了一两天的培训。这在后来证明在进行探索性测试时是非常有用的。

组建双人组的时候，一个人应该对工作伙伴和他的性格比较了解。在这种情况下这是不可能的。我们必须根据测试者的专业和个人意愿来编组。来

了14个测试人员，这意味着有7对，我们为每个组都准备了记有最重要信息的文件夹。在20分钟的摘要说明阶段，我们说明了那些文件夹、测试计划中一些重要的地方（什么要测试，什么不要测试等等），还有在接下来的两天内我们应该做什么。

在第一天，我们执行发布工具的功能性测试和如何在Web门户处查看发布的页面。为了覆盖到所有的浏览器、操作系统（operating system，简称OS）和用户群（发布信息的内容编写者以及发布之前批准信息发布的项目经理），我们开发了一系列的模型。这些模型说明了每个小组应该在何时用某种浏览器和OS，以及他们应该以什么身份登录到系统。这试图验证用Internet Explorer 5.5发布的图片和文本能否用Netscape 4查看等。文件夹解释了页面应如何被发布，　以及图片和文件嵌入在结构中的哪一层。除去这些指导原则，测试者可以用他们自己选择的文本大小、字体类型、框架、html和链接来进行实验。不久我们就清楚地知道，发布工具在发布信息之前并不控制任何给定的输入。在Web门户处产生了一些相当奇怪的结果。这也就引发了各个组之间的讨论和意见交换，这一讨论大大超出了我们的预料。测试人员经常会忘记测试指导原则，取而代之的是把注意力放在仅仅发现“故障”上。很遗憾的是，他们由结构化的探索性方法转变为“故障追查”。因此我们面临着再次让测试人员回到结构化方法的任务。为了确保所有的功能和浏览器选项都被覆盖到，第一天除了测试管理的探索性方法之外，我们还把重点放在所制定的测试指导原则和模型上。考虑如何在测试摘要报告中写测试文档时，这些指导原则显得很重要。

由于测试的时间很短，所以任何类型的工具都不可用。取而代之，我们用Excel表格把发现的偏差记录在日志中。

在第二天，我们开始应用一种结构化较弱的探索性方法，在这个过程中我们更多地依靠每个测试人员的知识和在第一天获得的经验。第一天发布的页面用于在Web门户测试搜索功能。我们还做了一个更加广泛的浏览器测试，在测试中门户的每一个功能块都在所有浏览器中测试过，这样产生了相当有趣的结果，页面布局和框架设置有时会在Netscape 4.6和Netscape 6.0中有所不同，而且用Internet Explorer 5.5的时候颜色也发生了变化。这再次证明了在进行Web测试中，使用不同的浏览器是多么地重要。只因为他们使用了Netscape浏览器而无法浏览您的页面，就导致20－30%的市场损失，这

是多么地令人沮丧。

依据计划的测试原则探查完功能之后，我们就可以用文档记录要测试的特性的覆盖率。因此我们尝试着把测试会话的最后几个小时用于纯粹的“故障追查”——同样也是基于每个测试人员的理解和经验。把探索性测试进行到了极端！

第五节　Web 测试经验

这个测试项目记录了包括重复在内的 200 个偏差。大约有 150 个偏差报告给了开发人员，其中 65 个被接受，剩下的偏差不得不推迟到“下一个版本”，因此系统必须变成功能有限的产品——推迟了 3 个月。因此项目小组把测试看作是成功的。我们没有让一个有太多“故障”的系统变成产品而把问题留给客户。

我们在前面提到的类比意思是说每个人在某个时候或其他时候都用过某种形式的探索性或双人组技术。但是要把事情做对需要一定的技巧。那么它需要什么技巧？我们获得了什么样的有用经验呢？

有些人会说什么也没有，他们会争辩说这个项目太小了，而且太“紧迫”。我们很少有时间为双人组做准备。我们并没有一个定义好的、能衡量覆盖率的方法。我们把探索性技术当作一条“出路”，因为我们并没有足够的信息来用传统的测试脚本。因此从这个项目中并不能够得出结论。

其他一些人说我们确实学到了很多：探索性测试、双人测试和管理高速的缺乏文档的 Web 项目。很不幸，这种情况出现得比较频繁，因为上市时间是所有客户都关心的问题。这也是我们第一次尝试测试的一种概念，在这个测试中，双人组和探索性技术被“正式化”了，不仅仅是项目的附属，或临时的测试方法。在很多方面它给予我们的比我们想要的还要多。

就像 James Bach 陈述的那样，在进行探索性测试的时候，测试人员需要提前进行探索性测试的“培训”。幸运的是，我们的测试人员在测试之前就已经完成了为期两天的学习指导课程。为了确保尽可能全面，在第一天我们还或多或少地进行了正式的功能性测试。这不仅使测试人员对软件有了更好

的理解，而且还向他们指出了系统中错误可能发生的地方。这是很重要的，因为探索性测试就是使用您所知道的东西。确实会出现的一个问题是当在软件的一小部分中发现错误的时候，所有的双人组都开始检查那个部分。由于双人组没有进行过探索性测试的培训，因此我们需要在不同的方向上指引他们以提高测试的覆盖率。也许在计划阶段就应该考虑到这个问题。

同事提出了许多问题，他们不知道我们如何控制这样一个非结构化的测试。在发现“故障”的时候，我们如何保证填写了偏差报告？如何能知道测试人员是否坚持了测试计划？如何确保他们是完全具有工作能力的？我们是否会不断地在房间里转来转去？我们不是在一个房间里转，而是在三个不同的房间里“跑”（因为我们把 Windows95、2000 和 NT 放在不同的房间里了）。困难的事情不是让测试人员进行测试，而是让他们不要测试计划以外的内容。当我们在几个房间里穿梭的时候，我们脑子里会得到一幅测试人员工作的画面（只有 14 个人，因此并不那么令人疲惫不堪）。只要我们看到有一组没有在计划区域内工作或是出现了别人已经报告过的错误，我们一般会给他们一些小的暗示，比如“您试过这个了吗”或“如果您那样做会发生什么？”为了使每个人都不脱离“正轨”，我们这样做就可以了。在分组工作时，其中一个测试人员总是负责填写偏差报表。这是一个很容易填写的表，但是使故障重现已经足够了。在每个暂停阶段（在每次 90 分钟的测试后），如果可能的话我们与项目经理一起通读所有的报告，以确保我们理解所记录的东西，以及确保我们有足够的信息来重现故障。此外，在每个暂停阶段，我们常与所有人交流，并且一天三次地把所有人聚集到一个房间里做简短的总结。

我们运气不错，碰上了这些双人组。通常，我们应该提前了解测试人员，并且具有一点“人文知识”才能使分组得当。但我们没有时间做这个，因此我们依据职业（在一个双人组中不要两个都是开发人员）、国籍（为了增进交流）和他们自己的意愿去分组。这样做相当不错。但是我们也有过两天后双人组就需要改变的经历。这是容易和快速的，而且后来所有事情都进行得很顺利。这也许是一个明智的办法，在您找到高效的合作伙伴之前，您经常会不得不去“试验”测试人员。

对使用探索性双人测试来说，其中一个挑战是确保测试执行过程中功能性的覆盖率。双人组的跟踪检查需要广泛的资源和来自测试管理者的实际管

理。更加合适的测试管理工具和测试技术可能会解决这个问题。我们知道很少的工具能支持这种测试计划和执行，但是现在一种称为“以测试会话为基础的测试管理”的技术已经开发出来，这是解决衡量覆盖率和在不破坏创造性的情况下提高效率的方法。

在寻找故障的过程中，我们确实发现了很多偏差和故障，但是其中很多是重复的。我们让测试人员对整个系统进行无秩序的搜寻，很多双人组发现了相同的故障。下一次我们将让每个双人组关注一个特定的部分。过一段时间，我们将把双人组和测试部分进行交换。

第六节　警　告

既要进行探索性测试，又要写脚本，平衡两者是一种挑战。

我们并没有说探索性测试可以用于所有类型的软件系统和任何形式的测试。但是，在更仔细地观察探索性测试（以及双人测试）之后，我们发现探索性测试通常是我们所做项目中的一个重要部分，只是它还没有被称为探索性测试。

我们认为在任何项目中为这类测试作计划是重要的。如果您没有具有计划技能的测试人员来把测试做好，那么最好是做“有计划的临时测试”，而不做没有任何计划的临时测试。

第7章 软件质量控制中的统计方法

第一节　质量控制

“质量控制”一词在本章是指一种在现代制造工业中广泛使用的确保产品质量的强大工具。它是一种在二战期间发展起来的基于统计技术的工具。这些统计技术包括随机取样技术、测试、测量、审查、缺陷原因查找、改进、统计推理和抽样验收。二战期间，军事采购合同要求所有供货商在他们

的过程中采用统计过程控制规程，并且提交已完工产品的随机抽样结果供质量符合性审查使用。从那时起，该工具在美国和海外获得了广泛的使用，同时也成为联邦政府机构和客户群推动高质量产品生产的重要手段。它还是制造业组织在现代市场中生存的有力武器。质量控制手段应用于软件开发中的软件质量控制只是其应用的一个简单扩展。

根据 Cho 的研究，制造工业中的质量控制活动可以总结如下：

1. 建立质量标准。对每个质量特征，应建立一个作为产品规约的标准。完成的产品将与此标准进行比较。例如，应规定滚珠轴承中的钢珠的直径和容许公差，比如 0.25 +/- 0.001 英寸。如果完工的钢珠直径在 0.249 和 0.251 英寸之间，那么这些钢珠的直径就是合格的，否则视为不合格品。

2. 制定要达到质量标准的计划。产品质量目标的实现需要对制造过程和设备进行仔细的计划和安排、采购质量好的原材料和培训操作员等等。

3. 确定控制生产过程的预防措施。古语“防患于未然”在质量控制中同样适用。在制造过程中，所有可能影响产品质量特征的因素都应得到谨慎的控制。例如，原材料的纯度、温度及压力控制都可以影响钢珠的直径。

4. 确定质量符合性。确定的内容包括：

（1）对产品质量标准的解释。

（2）对产品单元随机抽样以进行审查。

（3）对抽样的产品单元进行审查和测量。

（4）对每个抽样产品单元比较（3）与（1）的值。

（5）对每个产品单元的质量符合性的评价。

（6）通过统计方法决定对该批次产品的接受或拒收。

（7）审查数据形成文档。

软件质量控制流程就是基于以上原则，特别是第 4 项（确定质量符合性）中的 1）到 7）各项。

制造工业的工厂通过一个过程把原材料转变为可用的产品。用数学的方法可以如下表示：

$$U = F(R)$$

其中 R 是一组原材料，而 U 是一组产品单元。

工厂有两种主要特征：

1. 它的原材料被明确定义。原材料的类型和质量特征，例如每种原材

料的特性和化学成分，都已明确定义。

2. 产品单元和产品单元的缺陷、每个产品单元的质量特征也都已明确定义。

这些特征使质量控制得以实施以保证产品的质量。

图7-7-1描述了质量控制和制造过程之间的关系，即质量环。该环从确认人们需要之后的产品概念表达开始，然后明确产品特征；以制造为目的设计产品；然后设计产品生产方法和工程步骤；获取机器和原材料；安装设备和仪器；最后，开始制造。对产品进行审查以验证质量与所需特征的符合性；客户购买并使用产品，并将他们的经验反馈给此环。接着该环又从头开始。

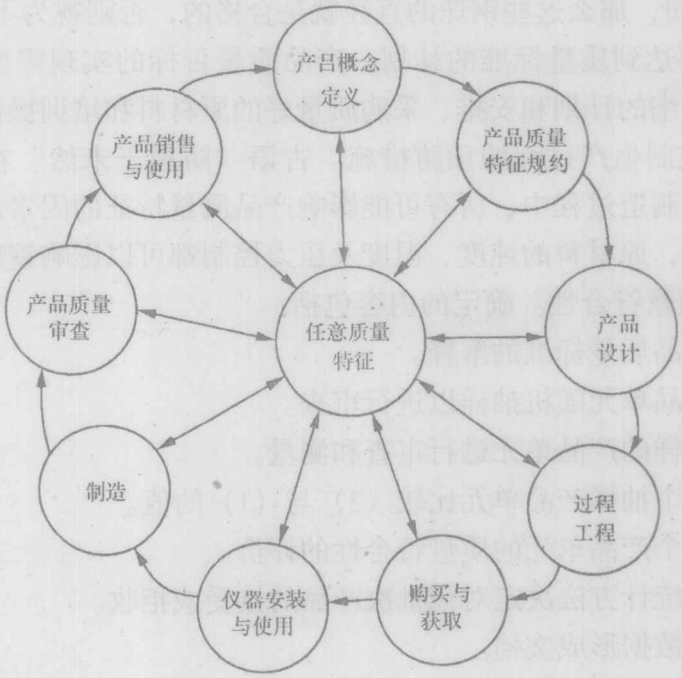

图7-7-1　制造质量环

考虑到产品总体的规模，在确定质量符合性时对产品单元进行大规模审查可能不可行或不经济。因此在这里统计抽样审查扮演了主要角色。抽样审查通常涉及对总体缺陷率θ的估计。缺陷率定义为θ=D/N，其中D是产品总体中存在缺陷的单元数，N是产品单元的总数。对θ的估计可以简单地通过使用二项式分布得到，该分布在概率、统计和质量控制的研究中都很重

要。这种分布的使用通常出现在对包含有限个单元的有限总体的重复抽样中，或对包含无限个单元的无限总体的重复抽样或不重复抽样中。

在任何一个工厂里，生产一批没有缺陷的产品都几乎是不可能的，因而测量产品质量的符合性通常是看缺陷率是否低于一个可接受值，例如 $\theta < 0.01$。通过统计抽样方法，缺陷率的结论可以达到诸如95%的置信率。

还有另一种方法可用来确定产品总体的可接受度，它称为验收抽样(acceptance sampling)。此方法中，选取 K 个单元的样本进行审查。如果样本中有缺陷的单元数低于可接受的数值 c，那么该总体是可接受的，否则为不可接受的。这种抽样方法中，在得出结论过程中存在两类风险：生产者的风险和用户的风险。前者是生产者承担的高质量产品批次被抽样方法拒收的风险。后者是用户承担的接受了低质量产品批次的风险。

抽样技术的详细信息可以在 Cho. 和 Juran 的文献中找到。根据本章的目的，这里只举例讨论用于评估总体缺陷率的抽样审查技术。

第二节　软件质量控制

我们可以在制造业的工厂和一个软件之间找到类比关系。软件的输入和输出就像是工厂的原材料和完成的货品。

如果软件输出用“产品单元”定义，那么输出就是一组被称为软件“输出总体”（output population）的产品单元集合。对任何一个不太小的程序，总体都包含大量的单元。软件测试的目的是找出总体的某些特征，例如总体中有缺陷的单元数和总体中所有单元数的比值。该比值可以叫做总体缺陷率，并可以作为软件的“软件质量指数”（software quality index）。显然对总体进行大规模的审查来得到该比值是过于昂贵的。一种高效的方法是利用统计随机抽样。通过采用统计质量控制，我们可以得到数值形式的可信的质量总体性表示。

基于这种观念，数据质量必须从制造的开始直至提交给用户的过程中都得到保证。类似于制造过程，在质量控制和软件开发之间存在一种密切的关系。这种关系称为“软件质量环”（software quality cycle），如图 7-7-2 所

示。比较图7-7-1和7-7-2就可以发现它们十分相似。

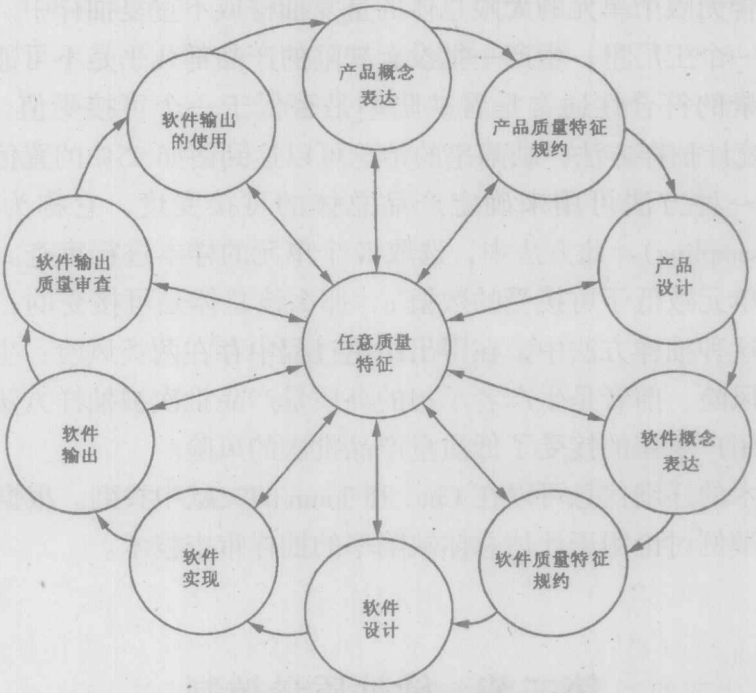

图7-7-2　软件质量环

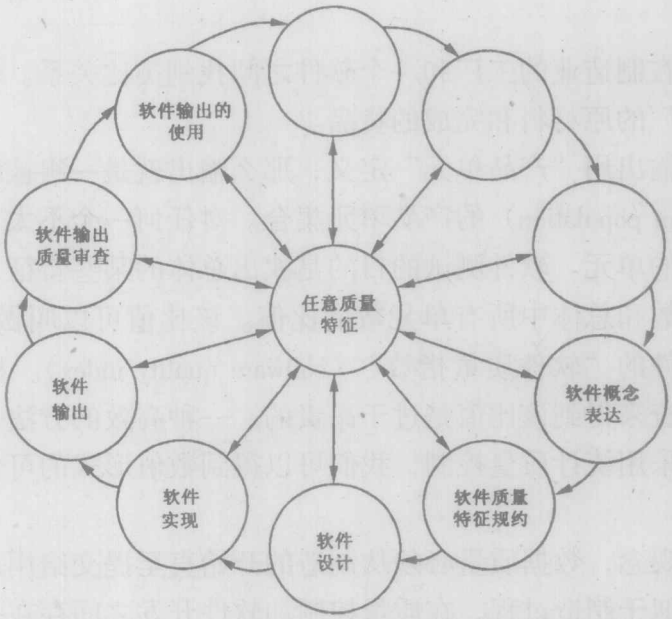

图7-7-3　不完整的软件质量环