软件缺陷模式与测试

宫云战　杨朝红　金大海
肖　庆　王雅文　著

科学出版社

内 容 简 介

基于缺陷模式的软件测试是21世纪初发展起来的一种新型软件测试技术，是高可信、大型及基础软件测试必备的方法之一，有强烈的工程需求，它以缺陷检测效率高、缺陷定位准确、自动化程度高、易学易用、与其他软件测试技术具有很好的互补性等特点，目前已逐步成为国际上主流的软件测试技术。本书全面论述了基于缺陷模式软件测试的一般方法，包括软件缺陷的综合论述、面向C/C++/Java的软件缺陷模式的分类、各种软件缺陷模式的定义、基于缺陷模式的软件测试原理、提高测试精度的区间运算技术、敏感路径分析技术、函数间分析技术等。

本书是软件测试领域的专业书籍，可供从事软件测试技术工作的研究人员学习和参考。

图书在版编目(CIP)数据

软件缺陷模式与测试 / 宫云战等著. —北京：科学出版社，2011

ISBN 978-7-03-031726-1

Ⅰ. 软… Ⅱ. 宫… Ⅲ. 软件-测试 Ⅳ. TP311.5

中国版本图书馆CIP数据核字(2011)第121184号

责任编辑：张艳芬 / 责任校对：林青梅

责任印制：赵 博 / 封面设计：耕者设计工作室

科学出版社出版

北京东黄城根北街16号

邮政编码：100717

http://www.sciencep.com

新蕾印刷厂印刷

科学出版社发行 各地新华书店经销

*

2011年7月第 一 版 开本：B5(720×1000)

2011年7月第一次印刷 印张：17 1/4

印数：1—3 500 字数：338 000

定价：56.00元

(如有印装质量问题，我社负责调换)

前　言

软件测试目前有数十种方法。首先，软件测试是分步进行的，不同的步骤具有不同的测试方法；其次，软件测试与软件的类型和应用有关，测试的侧重点是不同的；最后，软件测试与用户是否提供源代码有关。虽然目前有很多测试方法，但软件的质量仍难以保障，高可信软件的测试成本巨大，已经占据软件成本的80%甚至更高。

软件中有些类型的故障是非常难测的，其中包括各种小概率故障、需要大量测试用例才能检测到的故障等。这些故障利用传统的软件测试方法一般是无法检测出来的，但故障一旦被激活，往往会造成灾难性后果。因此，基于缺陷模式的检测在高可信应用领域、基础软件、大型软件中有着非常重要的作用，是软件测试不可或缺的重要方法。

基于缺陷模式的软件测试是20世纪初从美国发展起来的一种新型软件测试方法，和传统的软件测试方法不同，该方法首先定义软件的各种缺陷模式 $M=\{M_1,M_2,\cdots,M_n\}$，测试时根据 M_i，从源代码中计算出与 M_i 相匹配的程序元素 $IP_1,IP_2,\cdots,IP_L$，即可疑故障点。

基于缺陷模式的软件测试包括两部分主要内容：一是缺陷模式的研究；二是检测精度问题。对于第一方面，本团队的研究成果是将软件中的缺陷分为四个层次，即故障模式、安全漏洞模式、疑问代码模式和规则模式。这种分类是基于缺陷的后果，对缺陷模式的研究主要是基于理论和实践提出更多种类的模式，更多的缺陷模式可以使系统检测出更多的缺陷。本书全面论述了软件的各种缺陷模式，并以C/C++/Java为例，给出了数百种缺陷模式。

对于第二方面，当称IP是可疑故障点时，尚需要人工进一步确认其是否是缺陷。检测精度包括两方面内容：一是误报问题，高误报率会使系统的实用性受限；二是漏报问题，高漏报率会使系统失去价值。本书全面研究了提高检测精度的方法，包括区间运算技术、敏感路径分析技术和函数间分析技术等。

在过去的几年中，产生了众多的基于缺陷模式的软件测试工具，这些工具以缺陷检测效率高、定位准确、可以检测传统软件方法难以检测的缺陷等优点而风靡全球，目前已成为软件测试的主流技术之一，以销售软件测试工具和软件测试服务在市场上取得了极大的成功。基于上述技术，作者在多个国家“863”项目和国家自然科学基金项目的资助下，开发了软件缺陷测试系统(DTS)，本书全面论述了DTS测试技术的一般原理以及所涉及的关键技术。

全书共 8 章，第 1 章全面论述与软件缺陷相关的概念和理论；第 2～4 章分别论述了软件的故障模式、漏洞模式、疑问模式和规则的缺陷模式，这些模式有的是作者参考国际上的相关文献、工具，有的是国内外相关工厂企业提供，并经过作者理论加工得来的；第 5 章论述了基于缺陷模式的软件测试的一般方法；第 6～8 章论述了提高测试精度的关键技术。

本书所给出的内容取自作者自身的研究，并参考了国际上的相关论文，国际上目前尚无此类软件测试的书籍，本书是在该研究领域的首次尝试。本书主要由北京邮电大学宫云战、杨朝红、金大海、肖庆和王雅文撰写，在撰写和出版过程中得到了国家“863”项目(2009AA012404)和国家自然科学基金项目(91018002)的资助，在此一并表示感谢。

书中错误之处在所难免，敬请读者批评指正。

作　者

2011 年 4 月

目　录

第 1 章　软件缺陷与缺陷模式

1.1　软件缺陷的概念

缺陷是一个系统的基本属性，无论何种系统都会存在缺陷。有些缺陷来自于设计，有的则来自于制造，还有的来自于外部环境，很多正常运行的系统事实上都会存在缺陷，只不过是这些缺陷当时没有起作用，或者不那么重要。

软件缺陷都是人为造成的，这和目前已有的其他系统都不相同。由于软件的规模及逻辑上的复杂性远远超出了人类的能力，因此人类无法证明程序的正确性，也无法检测出软件中的所有缺陷。可以预测，在未来很长一段时间，软件缺陷必然伴随软件而存在。

软件缺陷是计算机程序的错误(error，flaw，mistake，failure，fault，bug)，会引起与预期不一致的行为。大多数缺陷是由于人为的程序源代码或设计引起的，少数是由于编译器产生的错误代码引起的。软件缺陷是多种多样的，不同的软件阶段、不同的开发语言和环境、不同的开发人员、不同的测试工具、不同的软件类型、不同的软件规模等，是不大可能用一个统一的模型来描述的。软件缺陷的影响按照不同程序具有多样性。有些缺陷可能仅仅影响很小的功能，也许很久才被发现。更严重的一些缺陷会导致程序崩溃，如有些缺陷会导致灾难性的损失或者无授权的入侵。

从不同的角度考虑，软件缺陷的概念具有不同的定义，可以从程序的角度进行定义，如变异测试方法；也可以从功能的角度进行定义，即凡是不符合预期软件功能的都是缺陷；也可以从用户的角度进行定义，即用户不满意的都是缺陷。下面是从用户的角度给出的定义。

软件失效(failure)：交付的服务偏离正确的服务称为软件失效。

软件错误(error)：至少一个系统的外部状态偏离正确的服务状态称为软件错误。

软件故障(fault)：错误的原因称为软件故障。

软件缺陷(defect)：对软件产品预期属性的偏离现象称为软件缺陷。

下列几种情况都可以归结为软件缺陷：

(1) 产品说明书中规定要做的事情，而软件没有实现。例如，产品说明书要求计算器要实现加、减、乘、除功能，而研发出来的计算器不能进行除运算，这就是一

个缺陷。

(2) 产品说明书中规定禁止做的事情,而软件却实现了。例如,产品说明书要求计算器只需实现加、减、乘、除功能,其他的功能不要实现,而研发出来的计算器不仅能进行加、减、乘、除运算,还能进行乘方或三角函数运算,这也是一个缺陷。

(3) 产品说明书没有提到的事情,而软件却实现了。例如,产品说明书要求计算器要实现加、减、乘、除功能,而研发出来的计算器还能进行乘方运算,这也是一个缺陷。

(4) 产品说明书中没有提到但是必须要做的事情,软件却没有实现。例如,产品说明书要求计算器要实现加、减、乘、除功能,但是没有提到在电量很低情况下也能正常使用,而研发出来的计算器在电量很低的情况下计算时出现错误,这也是一个缺陷。

(5) 软件很难理解,很难投入使用,速度非常慢,测试人员站在最终用户的角度看到的问题是常见的但不是正确的。例如,产品说明书要求计算器要实现加、减、乘、除功能,但是按键使用的文字或标识不清楚,如加按键用和表示,或者计算1+1需要1分钟甚至更长时间,这也是一个缺陷。

软件缺陷是影响软件质量的重要因素和关键因素之一,发现和排除软件缺陷是软件生命周期中的重要工作之一。每一个软件组织都必须妥善处理软件中的缺陷,这是关系软件组织生存、发展的质量根本。

发现和排除软件缺陷需要花费大量的成本。2006年,美国花费与软件缺陷相关的费用大约为780亿美元。美国国防部的数据表明,在IT产品中,大约42%的资金是用于与软件缺陷相关的工作上。目前在美国,软件测试的花费占整个软件费用的53%~87%,因此,对软件缺陷及其相关问题进行研究是极有价值的。我国目前尚无这样的统计报告,但我国目前的软件测试费用一般要占整个软件费用的30%以上。

软件缺陷密度的定义是千行源代码包含的缺陷数量,用 X 个缺陷/KLOC或者 X 个缺陷/KSLOC来表示,缺陷密度是软件质量的一个重要指标,在很多高可靠领域,都将缺陷密度作为软件验收的一个重要指标。

软件工程师在工作中一般会引入大量的缺陷。统计表明,有经验的软件工程师的缺陷引入率一般是50~250个缺陷/KLOC,平均的缺陷引入率为100个缺陷/KLOC左右。即使软件工程师学过软件缺陷管理之后,平均的缺陷引入率也为50个缺陷/KLOC左右。

目前,高水平的软件组织所生产的软件可以达到的缺陷密度为2~4个缺陷/KLOC,一般的软件组织所研发的软件其缺陷密度为4~40个缺陷/KLOC,美国国家航空航天局(NASA)研发的软件其缺陷密度可以达到0.1个缺陷/KLOC,我国某型号软件要求的缺陷密度为1个缺陷/30KLOC。

开发低缺陷密度的软件需要花费大量的成本。20世纪90年代，NASA的软件平均1行代码需要1000美元，测试费用占据了整个软件开发费用的90%以上。一般CMM5的软件开发成本是CMM1的几倍、几十倍甚至上百倍。

要想开发高质量的软件，必须降低软件的缺陷密度，下列几种技术是常用的。

(1) 教育与培训：通过教育与培训，使得软件开发者能对语言的使用比较熟练或非常熟练，可以大大减少缺陷。

(2) 使用先进的管理工具。

(3) 严格管理。

(4) 严格测试：通过相关的测试工具、验证工具、代码审查等各种软件进行测试。

1.2　软件缺陷的来源

虽然软件的缺陷是多种多样的，但从理论上看，软件中的任何一个部分都可能产生缺陷，而这些缺陷的来源不外乎下列四个方面：

(1) 疏忽造成的错误(carelessness defect，CD)。

(2) 不理解造成的错误(misapprehend defect，MD)。

(3) 二义性造成的错误(ambiguity defect，AD)。

(4) 遗漏造成的错误(skip defect，SD)。

MD、AD、SD三类缺陷主要存在于软件开发的前期阶段，如需求分析阶段、设计阶段、编码阶段，而在实施第三方测试时，一般不会存在这三类缺陷，其原因是这三类缺陷的检测概率都比较大，一般很容易测试。在这里所分析的多个例子中，第三方测试所测试出来的95个缺陷，只有1个缺陷是AD类缺陷，其他都是CD类缺陷。

疏忽造成的缺陷是必然的，因为疏忽是人类的固有本性，只不过是有些人做事比较认真，犯错误的概率小，反之亦然，但不可能不犯错误。人类软件开发造成的缺陷是多种多样的，此类缺陷难以准确预测和估计，因为不同的人、不同的软件、同一个人在不同的时刻都难说会犯何种错误。虽然人们不可能发现所有的缺陷，但根据语言的特点，可以总结出下列几种类型。

1. 显式约束造成的缺陷

设A是程序中的一个元素(一条语句或语句的一部分，或者是语句的集合)，由于A的存在，在A之前或之后必须要跟另外一个动作B，则称其为显式约束。如果B不存在或者B不是A所要求的，则都是缺陷，如switch语句后必须跟default语句等。

2. 隐式约束造成的缺陷

设 A 是程序中的一个元素(一条语句或语句的一部分,或者是语句的集合),根据程序的语义,A 必须满足某些约束,否则就是错误,如非法计算类错误、空指针使用错误、数组越界错误、指针使用错误等。

3. 路径造成的缺陷

当程序结束时,需要对 A 进行某些操作,若忘记了这些操作就是缺陷。如存储器泄露故障——在某个路径上忘记了释放内存,资源泄露错误——在某个路径上忘记了释放资源等。

程序是软件的载体,软件中的缺陷都可以归结为源代码的缺陷,源代码缺陷表现为语法、语义或者其他形式的缺陷,无论是哪种缺陷,都可以通过修改代码来更正。探究软件缺陷来源的目的是试图从上述原因中找出规律,以便为形成缺陷模式奠定基础。

1.3 软件缺陷的严重性和优先级

严重性和优先级是表征软件测试缺陷的两个重要因素,它影响软件缺陷的统计结果和修正缺陷的优先顺序,特别是在软件测试的后期,它将影响软件是否能够按期发布。

对于软件测试初学者或者没有软件开发经验的测试工程师而言,对这两个概念的作用和处理方式往往理解得不够彻底,在实际测试工作中不能正确表示缺陷的严重性和优先级。这将影响软件缺陷报告的质量,不利于尽早处理严重的软件缺陷,可能影响软件缺陷的处理时机。

1. 缺陷严重性和优先级的概念

严重性是指软件缺陷对软件质量的破坏程度,即此软件缺陷的存在将对软件的功能和性能产生怎样的影响。

在软件测试中,对于软件缺陷的严重性应该从软件最终用户的观点做出判断,即判断缺陷的严重性要为用户考虑,考虑缺陷对用户使用造成的恶劣后果的严重性。

优先级是指处理和修正软件缺陷的先后顺序的指标,即哪些缺陷需要优先修正,哪些缺陷可以稍后修正。

确定软件缺陷优先级,更多的是站在软件开发工程师的角度考虑问题,因为缺陷的修正顺序是个复杂的过程,有些不是纯粹的技术问题,而且开发人员更熟悉软

件代码，能够比测试工程师更清楚修正缺陷的难度和风险。

2. 缺陷的严重性和优先级的关系

缺陷的严重性和优先级是含义不同但相互联系密切的两个概念。它们都从不同的侧面描述了软件缺陷对软件质量和最终用户的影响程度和处理方式。

一般地，严重性程度高的软件缺陷具有较高的优先级。严重性高说明缺陷对软件造成的质量危害性大，需要优先处理；而严重性低的缺陷可能只是软件不太尽善尽美，可以稍后处理。

但是，严重性和优先级并不总是一一对应的。有时严重性高的软件缺陷优先级不一定高，甚至不需要处理，而一些严重性低的缺陷却需要及时处理，具有较高的优先级。

一方面，修正软件缺陷不是一件纯技术问题，有时需要综合考虑市场发布和质量风险等因素。例如，若某个严重的软件缺陷只在非常极端的条件下产生，则没有必要马上解决。另外，若修正一个软件缺陷，则需要重新修改软件的整体架构，这可能会产生更多潜在的缺陷，而且软件由于市场的压力必须尽快发布，此时即使缺陷的严重性很高，是否需要修正，也需要全盘考虑。

另一方面，如果软件缺陷的严重性很低，如界面单词拼写错误，此时可以稍后修正。但是如果是软件名称或公司名称的拼写错误，则必须尽快修正，因为这关系到软件和公司的市场形象。

对于比较规范的软件测试，一般使用软件缺陷管理数据库进行缺陷报告和处理，这需要在测试项目开始前对全体测试人员和开发人员进行培训，对缺陷严重性和优先级的表示和划分方法进行统一规定和遵守。

在测试项目进行过程中和项目接收后，充分利用统计功能统计缺陷的严重性，确定软件模块的开发质量，评估软件项目的实施进度。统计优先级的分布情况，控制开发进度，使开发按照项目尽快进行，有效处理缺陷，降低风险和成本。

为了保证报告缺陷的严重性和优先级的一致性，质量保证人员需要经常检查测试和开发人员对于这两个指标的分配和处理情况，发现问题时及时反馈给项目负责人，以便及时解决。

对于测试人员而言，通常经验丰富的人员可以正确地表示缺陷的严重性和优先级，为缺陷的及时处理提供准确的信息。对于开发人员而言，开发经验丰富的人员其严重缺陷的错误较少，但是不能将缺陷的严重性作为衡量其开发水平高低的主要判断指标，因为软件模块的开发难度不同，各个模块的质量要求也有所差异。

1.4 软件缺陷的发现、排除及效率

1. 软件测试的检测效率分析

根据软件的开发过程，软件可以分为需求、概要设计、详细设计、编码、测试和维护六个阶段，由于软件缺陷可能存在于软件开发的每个阶段，因此软件具有不同的测试方法，每种测试方法检测缺陷的能力是不同的，表 1.1 给出了几种软件测试方法的测试能力。

表 1.1 几种软件测试方法的测试能力

软件测试阶段	测试能力/%
非形式化的设计检查	25～40
形式化的设计检查	45～65
非形式化的代码检查	20～35
形式化的代码检查	45～70
单元测试	15～50
新功能测试	20～35
回归测试	15～30
集成测试	25～40
系统测试	25～55
低强度的 β 测试(<10 个客户)	20～40
高强度的 β 测试(>1000 个客户)	60～85

2. 影响软件测试的效率

1) 人为因素

在软件测试缺少自动化程度高的测试工具的前提下，软件测试的许多工作是由人来完成的。因此，人的因素是影响测试效率的重要方面。Basili 和 Selby 的实验数据清楚地揭示了人为因素的作用。表 1.2 揭示了不同水平层次的测试人员在发现软件错误的数量和测试效率的差异。这里，阶段 1、2 和 3 分别对应单元测试、集成测试和系统测试。

表 1.2　人为因素对测试效率的影响

检测结果	阶段	测试者水平层次					
		初级		中级		高级	
		平均值	标准差	平均值	标准差	平均值	标准差
发现错误的个数	1	3.88	1.89	4.07	1.69	—	—
	2	3.04	2.07	3.83	1.64	—	—
	3	3.90	1.83	4.18	1.99	5.00	1.53
发现错误的效率	1	1.36	0.97	2.22	1.66	—	—
	2	1.00	0.85	0.96	0.74	—	—
	3	2.14	2.48	2.53	2.48	2.36	1.61

2）软件类型

软件类型也是影响测试效率的一个重要因素，表 1.3 是 Basili 和 Selby 的实验数据。$P_1 \sim P_4$ 代表了四个不同的程序。

表 1.3　软件类型对测试效率的影响

检测结果	阶段	程序							
		P_1		P_2		P_3		P_4	
		平均值	标准差	平均值	标准差	平均值	标准差	平均值	标准差
发现错误的个数	1	4.07	1.62	3.48	1.45	4.28	2.25	—	—
	2	3.23	2.20	3.31	1.97	—	—	3.31	1.84
	3	4.19	1.73	—	—	5.22	1.75	3.41	1.66
发现错误的效率	1	1.60	1.39	1.19	0.83	2.09	1.42	—	—
	2	0.98	0.67	0.71	0.71	—	—	1.05	1.04
	3	2.15	1.10	—	—	3.70	3.26	1.14	0.79

3）缺陷类型

不同的测试方法检测不同类型错误的能力也是不同的。错误的分类方法目前有很多种，前边已经给出了比较详细的分类方法，为了分析方便，下面给出其中一种简单的分类方法。

（1）初始化错误：初始化代码中的错误。

（2）控制错误：控制的转移条件或转移地址错误。

（3）数据错误：包括程序中的常数、数据库中的数据错误等。

（4）计算错误：计算代码错误。

（5）集成错误：各模块之间、软件与环境之间的错误。

（6）容貌错误：人机界面、打印格式等错误。

图 1.1 是根据 Basili 和 Selby 的实验数据整理的。

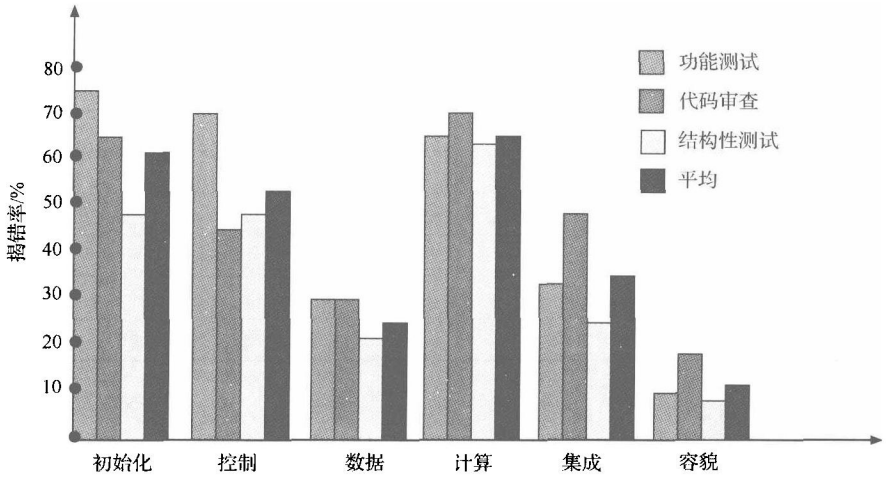

图 1.1 各种测试方法检测软件不同类型错误的能力

1.5 软件缺陷数据库

软件缺陷数据库是软件质量和软件企业的一个重要资源。国际上著名的软件企业、很多高可信需求企业都建立了自己的缺陷数据库，目前这些缺陷数据库的规模有的可达千万级。我国很多行业或者大的 IT 企业也或多或少建立了软件缺陷数据库，但规模一般比较小，从数百、数千到数万不等。百万级以上的软件缺陷数据库对软件测试、软件缺陷预防并最终提高软件质量具有重要作用。

软件缺陷数据库是管理软件测试缺陷的专用数据库系统，可以高效率地完成软件缺陷的报告、验证、修改、查询、统计、存储等任务。尤其适用于大型多语言软件的测试管理。

1. 提高软件缺陷的报告效率和质量

1）保持高效率的测试过程

由于测试缺陷数据库通过测试组内部局域网运行，因此打开和操作速度快。测试工程师可随时向内部数据库添加新发现的缺陷，而且如果遗漏某项缺陷的内容，数据库系统将会及时给出提示，保证软件缺陷报告的完整性和一致性。软件缺陷验证工程师将主要精力放在验证数据库中新报告的缺陷上，因此保证了效率。

2）提高软件缺陷报告的质量

软件缺陷报告的一致性和正确性是衡量软件测试公司测试专业程度的指标之

一。通过正确和完整地填写软件缺陷数据库的各项内容，可以保证不同测试工程师的缺陷报告格式统一。为了提高报告的效率，缺陷数据库的很多字段内容可以直接选择，而不必每次都手工输入。

3）实时管理、安全控制

软件缺陷查询、筛选、排序、添加、修改、保存、权限控制是数据库管理的基本功能和主要优势。通过方便的数据库查询和分类筛选，便于迅速定位缺陷和统计缺陷的类型。通过权限控制，保证只有适当权限的人才能修改或删除软件缺陷。

2. 满足大型多语言软件测试的需要

软件测试缺陷数据库可以满足大型软件测试项目的需要。大型多语言软件测试具有以下特征：

（1）测试周期较长：通常要测试多个版本(builds)，跨度达3～6个月或更长。

（2）存在较多软件缺陷：由于软件功能丰富，软件设计结构复杂，规模庞大，需要本地化的资源文件数量多，类型复杂，因此可能存在数百甚至上千个软件缺陷。

（3）测试范围广：从测试内容来看，其包括功能测试、性能测试、安装/卸载测试、用户界面测试、语言质量测试等。从测试方法来看，其包括手工测试和自动脚本测试。

（4）对测试质量要求较高：要求发现重要的缺陷，方便跟踪和及时处理测试发现的缺陷。

（5）多语言软件同时测试：可能需要同时测试多种语言的软件。

满足如此高质量要求的软件测试，如果项目测试组内部没有高效的软件缺陷管理和控制方式，是很难保证测试质量和测试进度的。测试实践证明，在测试组织不完善的新型测试机构的测试初期，引入内部软件缺陷数据库是很有必要的。

另外，测试人员的不确定性难以保证新加入的测试成员能够尽快适应实际测试项目的需要。为了保证测试软件缺陷报告的质量，人们引入内部测试缺陷数据库，从测试工具和测试流程上，保证不同测试技术背景的测试成员书写结构一致的测试报告。

引入内部软件测试缺陷数据库属于软件公司创建测试组织的基础性工作，可以满足现在和今后软件测试业务不断发展的需要。这种基础工作做好了，可以使初期的测试项目顺利实施，也为今后大型测试项目的实施打下了良好的基础。