



实用软件 测试过程

Testing IT

An Off-the-Shelf Software Testing Process

(英) John Watkins 著

贺红卫 杨芳 等译



机械工业出版社
China Machine Press



实用软件 测试过程

Testing IT

An Off-the-Shelf Software Testing Process

(英) John Watkins 著 贺红卫 杨芳 等译



机械工业出版社
China Machine Press

本书介绍了一种实用的软件测试过程，内容涉及测试的技术和方法，项目的计划和管理，测试的角色和职责，测试过程的改进和度量的作用及使用，以及各个测试阶段的全面信息。书中还给出实例分析来说明如何在实际工作中实现这一过程。本书附录提供了一套全面的标准测试文档模板、预制表格和检查列表，方便读者使用。

本书内容全面，注重实际，可作为软件测试技术培训教材，也可作为大专院校计算机有关专业师生的教学参考书，并可以为管理和计划测试项目的管理人员、从事测试的人员以及需要了解测试过程的所有人员提供有价值的参考。

John Watkins: Testing IT: An Off-the-Shelf Software Testing Process (ISBN: 0-521-79546-x).

Authorized translation from the English language edition published by Cambridge University Press.

Copyright © 2001 by Cambridge University Press.

All rights reserved.

Simplified Chinese language edition copyright © 2003 by China Machine Press.

This edition is authorized for sale only in the People's Republic of China (excluding the Special Administrative Regions of Hong Kong and Macau).

本书中文简体字版由英国剑桥大学出版社授权机械工业出版社出版，在中国大陆地区发行。未经出版者书面许可，不得以任何方式复制或抄袭本书内容。

版权所有，侵权必究。

本书版权登记号：图字：01-2002-1179

图书在版编目 (CIP) 数据

实用软件测试过程 / (英) 沃特金斯 (Watkins, J.) 著；贺红卫等译. - 北京：机械工业出版社，2004.1

(软件工程技术丛书 测试系列)

书名原文：Testing IT: An Off-the-Shelf Software Testing Process

ISBN 7-111-12642-4

I. 实… II. ①沃… ②贺… III. 软件-测试 IV. TP311.5

中国版本图书馆CIP数据核字 (2003) 第062104号

机械工业出版社 (北京市西城区百万庄大街22号 邮政编码 100037)

责任编辑：朱 劭

北京昌平奔腾印刷厂印刷·新华书店北京发行所发行

2004年1月第1版第1次印刷

787mm × 1092mm 1/16 · 16.5印张

印数：0 001-5 000册

定价：30.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换
本社购书热线电话：(010) 68326294

前言

为什么天文学是一门科学而占星术却被认为是伪科学呢？换言之，怎样才能证明一个理论如实地描述了现实，然后用这个理论预测未知事实呢？著名的哲学家波普尔研究了这些问题并用一句话总结了结论：“衡量一种理论的科学地位的标准是其可证伪性、可反驳性或可检验性。”^①波普尔认为，“进行确证的证明，除非是真正检验一项理论的结果，否则是不算数的。”

对一个科学理论进行测试的过程非常类似于为肯定某个大胆的预测或试图否定那个理论而提供证明的过程。测试是一项复杂的活动，必须同时考虑理论和外部现实。测试必须为与我们设想的合理现实相关的复杂问题提供客观的答案。

自从建立了一个介于外部世界与用户之间的抽象模型以来，在开发软件时我们就遵循相同的思考过程。在软件中，我们定义将指导自己行为的严格的过程，并建立想在复杂的数据库和模板中操纵的数据。

我们能用波普尔的原理来测试软件吗？答案是肯定的，因为软件测试不应该只是确认应用能正确地工作，还应该确定在发生意外情况时应用能做出正确反应。介于被测试的软件和外部现实之间的这种不变而复杂的关系应该用于指导测试者的日常工作。

尽管通常认为测试是软件开发中必需的步骤，但很少有人将测试作为一项必须进行的活动。在许多项目中，测试被忽略；在其他情况下，执行测试只是为了证明应用能在测试条件下正确地执行。

在阅读本书后，你会发现测试不像想像中那样复杂，它可以像其他开发活动一样被管理。

在阅读本书时，你首先会发现作者把重点放在测试过程和现实世界的案例研究上，据我看来这些是软件测试最重要的方面，它们应用了波普尔的结论。

本书在逻辑上分为三个独立的部分。第一部分重点介绍传统测试过程。尽管技术以闪电般的速度在更新，但过程依旧。因为过程是任何活动的核心，所以它们甚至变得越来越重要。你会发现这一部分非常有用，因为它按照所有项目执行测试的顺序描述从单元测试直到回归测试的各个测试阶段。为便于访问这些信息，各个阶段以同样的结构来进行介绍。

第二部分给出实际的案例研究。其中介绍五个项目，每个项目强调一个不同的测试方面。我们再次确定成功总是和在现实中采用正确的过程有关。

第三部分介绍在需要时可以直接采用的模板和参考文档，你将发现这些模板在你的日常测试工作中非常有用。

本书不只是一本关于测试的书，它还是有助于测试者了解测试真谛的指南。它再一次证明应用易于理解的过程是成功的关键，用一个词来说就是必不可少。

我确信你将喜欢这本书，和我一样，你会把这本书放在你的案头，将作者的经验应用于你的日常工作中。

Maurice Rozenberg, 巴黎

《Test Logiciel》的作者（1998, Eyrolles）

① Karl Popper, *Conjectures and Refutations*. London: Routledge and Kegan Paul, 1963.

目 录

前言

第1章 概述	1
1.1 本书的目的	1
1.2 读者	1
1.3 如何阅读本书	2
1.4 本书的结构和内容	2

第一部分 传统测试过程

第2章 测试概述	7
2.1 概述	7
2.2 测试的挑战性	7
2.3 什么是测试	8
2.4 验证和确认	9
2.5 不进行测试的后果	10
2.6 测试——底线	10
2.7 附加信息	11
第3章 测试技术	13
3.1 概述	13
3.2 通用测试技术	14
3.2.1 肯定测试和否定测试	14
3.2.2 白盒测试和黑盒测试	15
3.2.3 错误猜测	15
3.2.4 自动化软件测试	16
3.3 功能测试技术	17
3.3.1 等价类划分	17
3.3.2 边界分析	17
3.3.3 侵入测试	17
3.3.4 随机测试	18
3.3.5 状态转换分析	18
3.3.6 静态测试	19
3.3.7 线索测试	19
3.4 非功能测试技术	19
3.4.1 配置/安装测试	19

3.4.2 兼容性或互操作性测试	20
3.4.3 文档和帮助测试	20
3.4.4 错误恢复测试	20
3.4.5 性能测试	20
3.4.6 可靠性测试	21
3.4.7 保密性测试	21
3.4.8 压力测试	21
3.4.9 可用性测试	22
3.4.10 容量测试	22
3.5 有关测试技术的参考读物	22

第4章 管理和计划测试	25
4.1 概述	25
4.2 测试的组织	25
4.3 角色和职责	26
4.3.1 概述	26
4.3.2 测试主管	27
4.3.3 测试组组长	27
4.3.4 测试分析员	28
4.3.5 测试者	28
4.3.6 独立测试观察员	29
4.3.7 补充的测试角色	29
4.4 测试阶段	30
4.4.1 概述	30
4.4.2 单元测试	31
4.4.3 集成测试	31
4.4.4 系统测试	32
4.4.5 系统集成测试	32
4.4.6 验收测试	33
4.4.7 回归测试	33
4.4.8 停止测试的时机	33
4.5 V模型在计划中的作用	34
4.6 测试需求管理	35
4.7 配置管理的作用和使用	36
4.8 缺陷跟踪的作用和使用	36
4.9 风险在测试计划和管理中的作用	37

第5章 单元测试	39	9.3 用户验收测试的数据需求	64
5.1 概述	39	9.4 角色和职责	65
5.2 单元测试方法	40	9.5 计划和资源	66
5.3 单元测试的数据需求	41	9.6 输入	66
5.4 角色和职责	41	9.7 用户验收测试的测试技术	67
5.5 计划和资源	42	9.8 输出	67
5.6 输入	42	第10章 操作验收测试	69
5.7 单元测试的测试技术	43	10.1 概述	69
5.8 输出	43	10.2 操作验收测试的测试方法	70
第6章 集成测试	45	10.3 操作验收测试的数据需求	70
6.1 概述	45	10.4 角色和职责	71
6.2 集成测试方法	46	10.5 计划和资源	72
6.3 集成测试数据需求	46	10.6 输入	72
6.4 角色和职责	47	10.7 操作验收测试的测试技术	73
6.5 计划和资源	47	10.8 输出	73
6.6 输入	48	第11章 回归测试	75
6.7 集成测试的测试技术	48	11.1 概述	75
6.8 输出	48	11.2 回归测试方法	76
第7章 系统测试	51	11.3 回归测试的数据需求	77
7.1 概述	51	11.4 角色和职责	77
7.2 系统测试方法	52	11.5 计划和资源	78
7.3 系统测试数据需求	52	11.6 输入	78
7.4 角色和职责	53	11.7 回归测试的测试技术	79
7.5 计划和资源	53	11.8 输出	79
7.6 输入	54	第12章 测试过程改进	81
7.7 系统测试的测试技术	54	12.1 概述	81
7.8 输出	54	12.2 度量的作用及其使用概述	82
第8章 系统集成测试	57	12.3 测试过程中常用的度量	82
8.1 概述	57	12.4 建立和管理度量程序	85
8.2 系统集成测试的方法	58	12.5 建立简单、有效度量集的建议	86
8.3 系统集成测试的数据需求	58	12.6 参考文献	88
8.4 角色和职责	58	第13章 测试过程的引进、采用和维护	89
8.5 计划和资源	59	13.1 概述	89
8.6 输入	60	13.2 测试过程的引进和采用	89
8.7 系统集成测试的测试技术	60	13.2.1 概述	89
8.8 输出	60	13.2.2 建立需求	89
第9章 用户验收测试	63	13.2.3 引进和采用的策略	90
9.1 概述	63	13.3 测试过程的维护	92
9.2 用户验收测试的方法	64		

第二部分 现实世界的测试过程：说明性的案例研究

第14章 案例研究1：大英图书馆	99
14.1 公司概况	99
14.2 测试需求的特征	99
14.3 管理和计划测试	100
14.4 角色和职责	101
14.5 测试阶段	104
14.6 制品	104
14.7 过程改进	105
第15章 案例研究2：路透社产品验收小组	107
15.1 公司概况	107
15.2 测试需求	107
15.3 管理和计划测试	109
15.4 角色和职责	110
15.5 测试阶段	113
15.6 制品	115
15.7 过程改进	116
第16章 案例研究3：皇冠质量保证小组	117
16.1 公司概况	117
16.2 测试需求	117
16.3 管理和计划测试	118
16.4 角色和职责	120
16.5 测试阶段	122
16.6 制品	124
16.7 过程改进	125
第17章 案例研究4：葡萄酒协会	127
17.1 公司概况	127
17.2 测试需求	128
17.3 管理和计划测试	129
17.4 角色和职责	130
17.5 测试阶段	132
17.6 制品	133
17.7 过程改进	134
第18章 案例研究5：自动化数据处理有限公司	135

18.1 公司概况	135
18.2 测试需求的特征	136
18.3 管理和计划测试	137
18.4 角色和职责	138
18.5 测试阶段	140
18.6 制品	143
18.7 过程改进	144

第三部分 附录

附录A 测试人员的参考术语	147
附录B 测试指南	157
附录C 测试计划文档模板	167
附录D 测试规格说明文档模板	175
附录E 测试脚本模板	185
附录F 测试结果记录表格模板	189
附录G 测试日志模板	193
附录H 测试证明模板	197
附录I 重用包检查列表	199
附录J 测试总结报告模板	201
附录K 等价类划分实例	207
附录L 边界分析实例	209
附录M 状态转换实例	211
附录N 自动化测试工具选择标准	213
附录O 可用性测试概述	225
附录P 测试过程健康检查	229
附录Q 面向对象软件的测试	235
参考文献	241
术语表	243
译后记	251

第 1 章

概 述

嗯，你所说的轮子！当然，我并不希望轻视你的成就，但是我到处旅行，在许多地方看到很多人已发明了这样的东西！

Big Ugg, Neander山谷，公元前35 000年

1.1 本书的目的

本书将为在软件测试领域中进行研究或建立公司内的软件测试过程的工作者提供全面而严格的指导。

本书针对测试过程的所有方面提出了建议和指导，其中包括：

- 测试软件的需求以及测试的方法。
- 测试技术的具体细节，包括加工过的例子。
- 测试项目的计划和管理。
- 测试的角色和职责。
- 测试的各个阶段的详细介绍。
- 大量的测试文档模板、预制表格和检查列表。
- 有关测试过程改进的建议以及度量的作用及其应用。
- 开发面向对象和基于组件的系统的人员面临的测试问题。

本书介绍了对大量不同来源的软件进行的测试，这些软件包括自主开发或修改的软件，对已有软件系统所做修改或扩充的软件以及由第三方为某一公司开发的软件。

本书还介绍了对某个公司得到的现成的商业现用软件（commercial off-the-shelf, 简称COTS）或者是公司自主开发的或由第三方代表某个公司开发的COTS软件的验收测试。

使用本书应该讲究实效，实际上它提供了一个测试框架，公司中的软件开发和测试所涉及的所有人员都可以使用这个框架，从而提高所交付的软件的质量，并减少测试的时间、人力和费用。

另外，本书介绍的测试过程可以被定制，以符合某个特定公司特殊的测试需求，并且在本书中给出了一系列的案例研究来说明如何定制测试过程。

1.2 读者

本书适合以下读者阅读：

- 需要改进公司内部软件测试过程的技术领导/主管（就过程的质量、生产率、成本和/或可重复性而言）。

- 需要提出一个在公司范围内使用的正规的软件测试方法的质量保证（QA）专家（如公司的QA领导或主管）。
- 需要节省时间、人力和金钱并通过采用一套完整、标准和现成的方案来改进测试需求的质量的项目主管/领导。
- 为客户提供有关其软件测试过程的建议和指导的独立信息技术（IT）顾问、QA顾问或管理顾问，本书将为这些顾问提供有用的指导。
- 希望采用预先定义好的制品（如测试脚本、测试计划和测试规格说明文档的标准模板）来节省时间和人力的测试/QA专家（如测试分析员、测试者或QA代表）。
- 需要理解软件测试过程（如单元或集成测试所涉及的开发者）的IT专家。
- 强烈希望通过提出一套完整的满足其公司的软件测试需求的测试方案，来改善其职业前景的任何职员，尤其是当需要改进质量或节省时间、人力和成本时。
- 编写或修订测试培训教材的培训主管/培训讲师，他们需要掌握测试过程及其应用的实用观点。
- 需要掌握测试理论和原则的现实观点来组织软件测试需求的学生，或者是对测试过程改进以及度量的作用及应用感兴趣的学生。

1.3 如何阅读本书

本书分为三部分，各部分内容紧密衔接，但是每一部分都可以单独阅读和使用。

第一部分（第2章~第13章）记录了构成一个软件测试过程的各个部分的“传统观点”。第一部分提供了可以用来建立一个测试过程框架的详细信息，进行软件测试的任何公司都可以在这个框架上定制各自的需求。

第二部分（第14章~第18章）提供了一系列的案例研究，说明了许多公司如何利用第一部分介绍的“传统观点”实现他们自己的测试过程。这些案例研究可以帮助一个独立的公司构建一个测试框架以满足它自己的测试需求。

第三部分（附录）包括一套标准的测试文档模板、预制表格和检查列表，以及大量的附属材料，补充介绍了贯穿全书的主题。标准测试文档模板、预制表格和检查列表也可以通过访问 us.cambridge.org/titles/052179546X 得到。它们无需修改就可以直接使用，也可以定制这些表格和模型来反映某个公司的特殊需求（如公司风格、商标或文档标准）。

本书出现的重要术语在词汇表中进行了定义。

1.4 本书的结构和内容

本书各章节及附录的内容概述如下：

- 第2章讨论即使对最简单的软件系统进行充分的测试也是颇具挑战性，介绍了许多测试的定义，给出了软件测试方法的简单概述，为进一步阅读列出了明确的测试参考文献。
- 第3章介绍为测试软件系统设计有效和高效的测试而使用的主要技术（在附录中也提供了

相应的例子以供参考)。

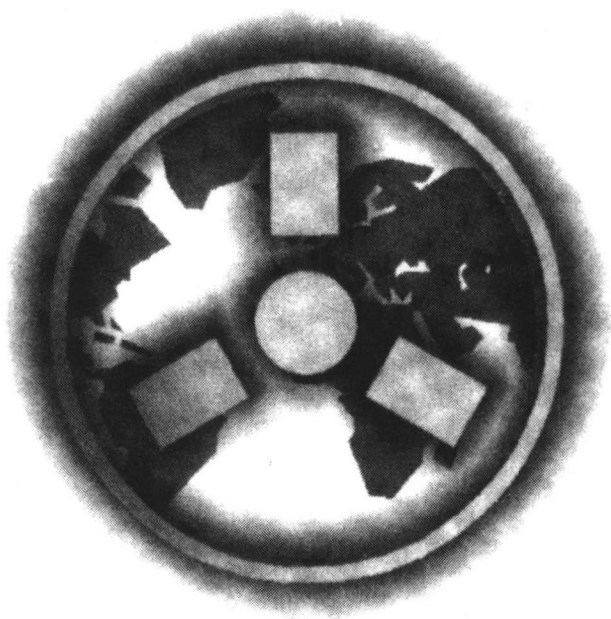
- 第4章解决与管理 and 计划测试过程有关的问题，给出关于组织测试和测试项目以及充分计划的需求方面的指导，描述大量支持计划过程的技术。
- 第5章~第11章详细介绍各个测试阶段（从单元测试到验收测试以及到回归测试^①）以及它们之间的关系。每一章都以标准结构进行介绍，其主要内容包括：
 - 该阶段的总体测试方法。
 - 该阶段的测试数据需求。
 - 与该阶段相关的角色和职责。
 - 该阶段的特殊计划和资源分配问题。
 - 该阶段的输入和输出。
 - 该阶段适用的具体测试技术回顾。
- 第12章考虑在测试过程中进行过程改进的需求，并介绍度量的作用（提出一个实用的度量集，可以有效地用于测试项目内部以及测试各个项目），同时给出有关测试过程改进的进一步的参考资料。
- 第13章是为采用本书中所介绍的测试过程的公司或者在本书介绍的测试过程的基础上建立自己的测试过程框架的公司所写的。讨论一个公司如何引入测试并成功地管理其采用的测试的方法，评审维护测试过程的需求，并提出满足该需求的方法。
- 第14章~第18章给出一系列的案例研究，介绍许多公司如何利用第2章~第13章介绍的测试过程观点来实现它们自己的定制的测试过程。案例研究中介绍的具体公司有：
 - 大英图书馆。
 - 路透社产品验收小组。
 - 皇冠质量保证小组。
 - 葡萄酒协会。
 - 自动数据处理（ADP）有限公司。
- 附录A~J给出一套测试文档模板、预制表格和检查列表：
 - 测试人员的参考术语。
 - 各个测试阶段概括性测试指南。
 - 测试计划文档模板。
 - 测试规格说明文档模板。
 - 测试脚本模板。
 - 测试结果记录表格模板。
 - 测试日志模板。
 - 测试证明模板。
 - 重用包检查列表。
 - 测试总结报告模板。
- 附录K~M给出了使用第3章介绍的测试技术的一系列例子。

① 严格地说，回归测试不是一个独立的测试阶段，但为了保持完整性，也单独用一章介绍回归测试。

- 附录N~Q详述本书正文中介绍的主题，包括：
- 评估商业上可用的自动化软件测试工具优点的方案和一套标准。
- 可用性测试过程及其应用的概述。
 - 进行测试过程审查的方案和一套标准。
 - 讨论面向对象和基于组件应用的测试中涉及到的问题。
 - 本书中引用的参考文献。
- 本书中所用术语的术语表。

第一部分

传统测试过程



第2章

测试概述

毫无疑问，当我们试图在应用中实现一些新特性时，同时也引入了新的缺陷。

2.1 概述

为了说明什么是测试以及为什么测试如此具有挑战性，而且为了强调只要使用测试软件就必须使测试过程尽可能地高效和有效，本章给出测试的概念。

熟悉高效和有效测试的必要性的读者可以跳过本章。

2.2 测试的挑战性

测试有多么困难呢？为了回答这个问题，让我们考虑下面的例子。

假设我们要测试一个简单函数，该函数将两个32位数相加并返回结果。如果每秒钟可以执行1000个测试用例，那么彻底测试完这个函数将占用多长时间呢？

如果你猜测是用几秒钟，那么这个答案差得太远了。如果你猜测是用几分钟，这个答案还是差得太远。即使你猜测是几小时、几天甚至是几个星期，这个答案仍然差得很远。实际的数字是58亿5千万年^①。

确实，一一测试每种可能的情况是不现实的。任何头脑清醒的人都不会试图用每一个可能的值来测试这个函数！实际上，我们会用一些形式化测试设计技术如边界分析和等价类划分等来帮助我们在测试用例中选择样本数据（测试设计技术的详细内容见第3章）。使用这些测试数据，可以假设如果函数执行满足这些样本值，那么它也将满足所有类似的值，从而将测试该函数所需的时间减少到可接受的时间范围。

而作为测试者，我们不应该这么快就充满信心，因为仍然有许多问题会使这个“简单的”函数的测试复杂化。例如：

- 如果函数需要与同一应用中的其他函数互操作会怎么样？
- 如果计算数据要通过复杂的客户端/服务器系统得到和/或需要通过客户端/服务器系统返回结果会怎么样？
- 如果计算是通过复杂的图形用户界面驱动的，用户可以以任何顺序在相应字段中键入加数并点击相应按钮来执行计算会怎么样？
- 如果这个函数要提交到许多不同的操作系统，每个操作系统的特性又稍有不同时会怎样？如果各个用户可以定制重要的操作系统特性会怎么样？

① 用计算器计算一下，可以很容易地得到这个问题的答案： $2^{(32 \times 32)} / 1000 / 60 / 60 / 24 / 365.25 = 584\,542\,046$ 年。

- 如果这个函数要提交到许多配置不同的硬件平台上会怎么样？
- 如果这个函数所属的应用要和其他应用交互会怎样？如果用户可以同时运行任意个其他应用（如电子邮件或日记软件）会怎么样？

这些都是许多测试人员在他们每天的测试工作中要面临的软件系统的常见需求，这些需求使得软件系统高度复杂化，同时也使测试面临着巨大的挑战！

2.3 什么是测试

测试过程并不是一个新事物。牛津英语大词典中记载“test”一词来源于拉丁语 *testum*，原是罗马人使用的一种陶罐，在当时它用来评估像稀有金属矿石这样的材料的质量。

几乎只要是软件开发出来的计算机程序都要经过测试。在软件开发的早期几乎没有正规的测试，而调试被视为开发软件过程中的重要步骤。

随着软件开发过程采用正规方法而逐渐成熟（如参考文献6），测试方法也随着测试专家采用了正规的测试方法和技术（见参考文献8）而日趋成熟。

现代软件开发领域的大多数工作者都对测试及其目的有直观的认识，最常见的看法包括：

- 保证程序和相应的规范说明一致。
- 发现软件中的缺陷。
- 确保软件不做不必要的事情。
- 确保系统合理地执行。
- 明确在系统失败之前可以让系统正常运行到何种程度。
- 明确发布给用户的系统中有哪些风险

下面给出测试的更为正规的定义：

测试是一种旨在评估一个程序或系统的属性或能力，确定它是否符合其所需结果的活动（参考文献1）。

这个定义阐述传统的测试方法，即看系统是否符合其陈述的需求。这是对测试的一种直观的看法，即我们知道系统应该执行的一些状态，然后我们确定是否满足这些需求。这种方法也就是所说的肯定测试（positive testing）。

还有对测试的另一种看法：

测试是为了发现错误而执行一个程序或系统的过程（参考文献2）。

这个定义不太直观，而且严格地说，并没有考虑系统的需求^①。相反，它引入在软件需求范围外积极查找错误的概念。实际上系统中可能会有问题或错误。这种方法也就是所谓的否定测试（negative testing）。

实际上，测试将合并肯定测试和否定测试这两类测试的元素，即检查一个系统是否符合其

① 尽管“错误”可以被看作一个支持特定需求的系统的失败。

需求,同时也试图发现可能会损害系统的成功操作或有用性的错误^①。

最近,用风险来定义测试的观念越来越流行。在这种用法中,“风险”一词与被测应用(application under test, AUT)通不过可靠性或健壮性测试,可能会对用户带来商业上的损害的可能性有关。下面是用风险对测试进行的一个定义:

测试是我们考察并理解与发布的软件系统有关的利益和风险状况的过程(参考文献28)。

在这个测试的定义中,测试者的作用是管理或转移系统失败的风险,以及可能会给用户带来的不良影响。

用风险来定义测试给测试人员提供了另外一种处理系统测试的策略。使用基于风险的方法,测试人员应注重软件分析,以便识别出需要进行充分测试的高风险的区域,确保用户在运行系统时不会出现危险。此外,在项目管理环境中,风险的概念已经众所周知且被大家所了解,大量的工具和技术可以应用到测试过程中(见参考文献28、29和30)。

对于注重计划和设计测试的人员来说识别一个特定的AUT的具体风险可能会很困难(尤其是当他们不太熟悉软件的运行领域时)。在估计风险的时候,考虑下面这些问题是非常重要的:

- AUT的业务、安全或保密性的重要程度。
- AUT的商业/公众的可视性。
- 测试类似或相关系统的经验。
- 测试同一AUT的早期版本的经验。
- AUT用户的看法。
- AUT的分析员、设计者和实现者的看法。

在测试过程中分析风险的必要性将在第4章中阐述。

2.4 验证和确认

另外,要介绍两个经常使用却又经常混淆的测试术语,即验证(verification)和确认(validation)。在参考文献40中给出这两个术语的正规的定义:

验证是通过检查并提供(在AUT开发过程中)履行具体需求的客观证据来进行证实的过程。

确认是证实(AUT的)对一个具体的目标用法的特殊需求被履行的过程。

参考文献26中给出了这两个术语的简明的更易于记忆的定义:

验证:我们正确地构造了产品吗?

确认:我们构造了正确的产品吗?

从本质上看,验证证明在AUT的开发过程中采用了好的经验,比如,遵循规范的开发过程(见参考文献8)。

① 在需求完整并有精确的规范的理想情况下,不需要进行否定测试,因为指定了被测应用(AUT)的每个方面,在这一点上人们还有所争议。但是,我们并不能处于完全理想的情况下,所以在进行测试时还应包括否定测试。

确认证明AUT必须符合它的正式需求,在这一方面它和本章前面讨论的测试的Hetzel定义很相近(见参考文献1)。

验证和确认(也简写为V&V)都是保证AUT质量的关键,必须结合需求管理的严格方法进行验证和确认。第4章将介绍需求管理的作用,并说明了它在V&V中的作用。

2.5 不进行测试的后果

对于那些以安全应用为主、以业务为主或以保密性为主的系统来说,以及一旦系统失败(或者诉诸法律或者失去公众信任),最终都将导致该软件失去市场的领域都有很多例子。

即使对于不是处理关键应用的系统,情况明确的系统的失败,如一个公司的网站、共享软件或演示软件的失败,仍然会对该公司有严重的商业影响,可能会失去公众的信任,有损声望。

一些缺陷很微小且难以发现,但是它们却可能会对一个公司的业务产生重大的影响。例如,在缺陷被发现之前的某一天系统失败或无效了,那么这个公司就会损失每个相关人员的一天的人力。如果一个未被发现的缺陷只是会引起系统性能的降低,那么用户可能根本不会意识到问题的存在。然而,如果该缺陷导致了每天30分钟的生产率的损失,那么这个公司每年就会损耗每个人20天的人力。

2.6 测试——底线

像“零缺陷软件”或“无缺陷系统”这样的说法都是夸张的说法,至多只能被看做是一个可望而不可及的目标^①。

实际上,由于计算机系统的复杂性,以及开发过程和这一过程涉及的人员易于出错,所以即使是保证一个相对简单的程序没有缺陷也是不可能的。

简言之,对一个给定系统进行充分地测试以确定其没有缺陷是不可能实现的。当这个问题与测试资源的有限和(更典型地是)供应短缺这样的事实合并在一起时,那么充分的测试就显得有问题了。为了尽可能多地发现并改正缺陷,测试者必须集中精力使测试过程尽可能地高效和有效。

最终,测试只会使大家确信某个给定软件系统在它的特定目标方面是可接受的。这个级别的信任必须在决定发布或接受软件之前在系统预期的作用(如安全要求、业务要求、保密性要求、机密或情况明确的应用)和系统运行失败的风险之间进行权衡。

有效测试的关键是使过程尽可能地高效:必须采用严格的计划和项目管理;测试者必须利用他们测试技术的知识和经验指导他们策划有效的测试;在测试过程的所有阶段中必须引入和管理重用;公司要仔细考虑自动化软件测试工具的好处和作用。

这些问题在后面的章节中会详细介绍。

^① 即使使用严格的数学方法(如Z和VDM),也不可能说任何软件是无缺陷的(除了软件中最简单的部分以外)。