



软件开发与测试丛书

# 软件测试 实用方法与技术



北京跟踪与通信技术研究 所

刘文红 张卫祥 司倩然 齐玉华 陈青 马贤颖 社会森 编著

清华大学出版社

软件开发与测试丛书

# 软件测试 实用方法与技术

刘文红 张卫祥 司倩然 齐玉华 陈青 马贤颖 编著



清华大学出版社  
北京

## 内 容 简 介

本书全面系统地介绍了软件测试的方法与技术。书中结合实例,详细介绍了动态测试和静态测试中的典型技术方法,比较了各种方法的不同之处并分析了它们的优缺点;紧扣软件测试实际和标准规范要求,从测试原则、测试环境、测试策略、测试内容、测试方法、测试过程等不同方面分别介绍了单元测试、集成测试、配置项测试和系统测试等不同测试级别中实用的测试方法与技术;此外还介绍了常用的软件测试工具,软件测试文档的编写,以及回归测试、面向对象软件测试、FPGA 测试等专门测试。

本书定位于一本软件测试方法和技术的实用指南,适用于软件从业人员了解软件测试的基础知识、一般流程、实用技术方法和常用测试工具,帮助软件从业人员提高技术能力和过程能力水平,也适用于软件测试机构建立测试能力体系,规范软件测试管理。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

### 图书在版编目(CIP)数据

软件测试实用方法与技术/刘文红等编著. —北京:清华大学出版社,2017

(软件开发与测试丛书)

ISBN 978-7-302-48066-2

I. ①软… II. ①刘… III. ①软件—测试 IV. ①TP311.55

中国版本图书馆 CIP 数据核字(2017)第 207759 号

责任编辑:石磊

封面设计:常雪影

责任校对:赵丽敏

责任印制:宋林

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座 邮 编:100084

社总机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, [c-service@tup.tsinghua.edu.cn](mailto:c-service@tup.tsinghua.edu.cn)

质量反馈:010-62772015, [zhiliang@tup.tsinghua.edu.cn](mailto:zhiliang@tup.tsinghua.edu.cn)

印 装 者:三河市金元印装有限公司

经 销:全国新华书店

开 本:185mm×260mm

印 张:19.75

字 数:480千字

版 次:2017年8月第1版

印 次:2017年8月第1次印刷

印 数:1~3000

定 价:69.00元

产品编号:072908-01

# “软件开发与测试丛书”

## 编审委员会

主任委员：董光亮

副主任委员：匡乃雪 吴正容 赵 辉

委 员：孙 威 马 岩 杜会森 许聚常 鲍忠贵

王占武 尹 平 闫国英 董 锐

主 编：刘文红

副 主 编：张卫祥

秘 书：韩晓亚

## “软件开发与测试”丛书序

为应对“软件危机”的挑战,人们在 20 世纪 60 年代末提出借鉴传统行业在质量管理方面的经验,用工程化的思想来管理软件,以提高复杂软件系统的质量和开发效率,即软件工程化。40 多年以来,软件已广泛应用到各个工程领域乃至生活的各个方面,极大地提高了社会信息化水平,软件工程也早已深入人心。

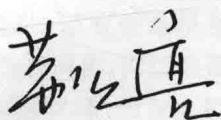
质量是产品的生命,对软件尤其如此。软件的直观性远不及硬件,软件的质量管理相对困难得多;但与传统行业类似,大型复杂软件的质量在很大程度上取决于软件过程质量。质量评估是质量管理的关键,没有科学的评估标准和方法,就无从有效地管理质量,软件评测是质量评估的最有效和最重要的手段之一。

北京跟踪与通信技术研究所软件评测中心是从事软件评测与工程化管理的专业机构,是在我国大力发展航天事业的背景下,为保障载人航天工程软件质量,经原国防科工委批准,国内最早成立的第三方软件评测与工程化管理的技术实体组织之一。自成立以来,软件评测中心出色地完成了以载人航天工程、探月工程为代表的数百项重大工程关键软件评测项目,自主研发了测试仿真软件系统、测试辅助设计工具、评测项目与过程管理软件等一系列软件测试工具,为主制订了 GB/T 15532—2008《计算机软件测试规范》、GB/T 9386—2008《计算机软件测试文档编制规范》、GJB 141《军用软件测试指南》等软件测试标准,深入研究了软件测试自动化、缺陷分析与预测、可信性分析与评估、测试用例复用等软件测试技术,在嵌入式软件、非嵌入式软件和可编程逻辑器件软件等不同类型软件测试领域,积累了丰富的测试经验和强大的技术实力。

为进一步促进技术积累和对外交流,北京跟踪与通信技术研究所组织编写了本套丛书。本丛书是软件评测中心多年来技术经验的结晶,致力于以资深软件从业者和工程一线技术人员的视角,融会贯通软件工程特别是软件测试、质量评估与过程管理等领域相关的知识、技术和方法。本丛书的特色是重点突出、实用性强,每本书针对不同方向,着重介绍实践中常用的、好用的技术内容,并配以相应的范例、模板、算法或工具,具有很高的参考价值。

本丛书将为具有一定知识基础和工作经验、想要实现快速进阶的从业者提供一套内容丰富的实践指南。对于要对工作经验较少的初入职人员进行技术培训、快速提高其动手能力的单位或机构,本丛书也是一套难得的参考资料。

丛书编审委员会



2015 年 5 月 6 日

# 前 言

软件测试是保障软件质量的重要手段,是构建高可信软件的关键环节。随着软件应用的日益广泛,人们对软件测试重要性的认识越来越深刻。20世纪80年代以来,特别是在最近二三十年间,软件测试无论是作为一项技术、一门学科还是作为一个行业,都得到了快速蓬勃的发展。

本书定位于一本软件测试方法和技术的实用指南,紧扣软件测试实际和标准规范要求,结合行业内软件测试现状,系统地介绍软件测试相关的知识、方法、技术和软件工具,给出较为详细的软件测试过程技术文档模板。

本书旨在帮助软件从业人员了解软件测试的基础知识、一般流程、实用技术方法和常用测试工具,提高技术能力和过程能力水平,以及帮助软件测试机构建立测试能力体系,规范软件测试过程管理。

本书是作者多年从事软件测试工程实践和技术方法研究的经验总结,与其他公开教材相比,主要特色有:实用性强,本书紧扣软件测试实际和标准规范要求,着重介绍测试实践中常用和好用的知识、技术和方法;视角特殊,本书从第三方测评的角度,系统地阐述适用于工程实践的测试方法和技术,反映行业实际需求和技术发展动态。

本书共有12章,可分为5大部分。

第一部分(第1章)是软件测试概述,简要介绍软件测试发展历程、软件测试典型定义、软件测试一般原则、常用软件测试模型、常见软件测试级别与测试类型、软件测试相关的标准规范等基本内容。

第二部分(第2、3章)是软件测试技术,结合程序实例,分别介绍静态测试技术和动态测试技术,并对典型技术进行比较分析。

第三部分(第4~7章)按照不同的测试级别,从测试原则、测试环境、测试策略、测试内容、测试方法、测试过程等方面分别介绍单元测试、集成测试、配置项测试和系统测试中的实用测试方法与技术。

第四部分(第8~10章)以独立章节分别介绍回归测试、面向对象软件测试和FPGA测试等专门测试类别中的实用测试方法与技术。

第五部分(第11、12章)主要内容是测试实践,第11章分别介绍了在静态测试、动态测试和测试管理中常用的软件工具。第12章着重介绍测试策划、测试设计、测试实施和测试总结等测试过程中常用的技术文档,给出通用技术文档模板。

软件测试包含大量相关的活动,有些是技术性的,有些是管理性的,还有些是相互交织的。例如,单元测试、集成测试、配置项测试、系统测试、回归测试中测试用例、测试数据和测试期望结果的设计是典型的技术性活动;人员计划、成本预算以及配置管理、项目监控中的大部分内容是典型的管理性活动。如前所述,本书偏重于讲述软件测试的技术性活动,软件

测试的管理性活动将在本套丛书的另一本书《软件测试管理》中重点介绍。

本书第1~3章由张卫祥编写,第4~7章和第12章由刘文红编写,第8章由陈青编写,第9章由齐玉华编写,第10章由司倩然编写,第11章由马贤颖和司倩然编写。全书由刘文红、杜会森统稿。赵辉、张卫民、杨宝明、李国华、牛胜芬等专家审阅了初稿并提出了许多宝贵意见。

在本书编写过程中,得到了北京跟踪与通信技术研究所,特别是软件评测中心的大力支持,还得到了编者同事、朋友和家人的关心与帮助,在此一并表示感谢!

由于水平有限,本书肯定还存在不少问题,敬请大家批评指正。

编 者  
2017年4月

# 目 录

<b>第 1 章 软件测试概述</b> .....	1
1.1 软件测试简史 .....	1
1.2 软件测试定义 .....	3
1.3 软件测试原则 .....	4
1.4 软件的可测试性 .....	6
1.4.1 可测试性定义与内涵.....	6
1.4.2 可测试性设计与实现.....	7
1.4.3 可测试性度量与评估.....	9
1.5 软件测试模型 .....	9
1.5.1 V 模型 .....	9
1.5.2 W 模型 .....	10
1.5.3 H 模型 .....	11
1.6 软件测试级别与测试类型.....	12
1.6.1 软件测试级别 .....	12
1.6.2 软件测试类型 .....	13
1.6.3 软件关键等级 .....	16
1.7 软件测试标准规范.....	18
1.7.1 相关标准概述 .....	18
1.7.2 GB/T 9386—2008《计算机软件测试文档编制规范》 .....	21
1.7.3 GB/T 15532—2008《计算机软件测试规范》.....	21
1.7.4 GB/T 25000.51—2010《软件工程软件产品质量要求和评价 (SQuaRE)商业现货(COTS)软件产品的质量要求和测试细则》 .....	21
1.7.5 ISO/IEC 29119 <i>Software Testing</i> .....	22
1.8 软件测试人员能力素质要求.....	26
1.9 术语与缩略语.....	27
<b>第 2 章 静态测试技术</b> .....	29
2.1 文档审查.....	29
2.1.1 实施要点 .....	30
2.1.2 组织与流程 .....	30
2.1.3 成果形式 .....	31



2.2	代码审查	34
2.2.1	实施要点	34
2.2.2	组织与流程	35
2.2.3	成果形式	35
2.3	静态分析	38
2.3.1	实施要点	38
2.3.2	组织与流程	39
2.3.3	成果形式	40
2.4	代码走查	40
2.4.1	实施要点	40
2.4.2	组织与流程	40
2.4.3	成果形式	41
2.5	静态测试技术分析	41
<b>第3章</b>	<b>动态测试技术</b>	<b>43</b>
3.1	白盒测试	43
3.1.1	概述	43
3.1.2	白盒测试基础	46
3.1.3	基本路径测试	51
3.1.4	控制结构测试	53
3.1.5	其他白盒测试技术	61
3.2	黑盒测试	64
3.2.1	概述	64
3.2.2	等价类划分	65
3.2.3	边界值分析	68
3.2.4	因果图与决策表法	70
3.2.5	组合测试	75
3.2.6	基于场景测试	77
3.2.7	错误推测法	81
3.2.8	黑盒测试技术分析	84
3.3	灰盒测试	85
3.3.1	概述	85
3.3.2	实施步骤	86
3.3.3	灰盒测试技术分析	86
3.4	动态测试技术分析	87
<b>第4章</b>	<b>单元测试</b>	<b>89</b>
4.1	概述	89
4.1.1	单元测试的定义	89

4.1.2	单元测试的目的	89
4.1.3	单元测试的重要性	90
4.2	单元测试原则	92
4.3	单元测试环境	92
4.4	单元测试策略	94
4.4.1	自顶向下	94
4.4.2	自底向上	94
4.4.3	独立单元	95
4.5	单元测试内容	95
4.5.1	功能测试	96
4.5.2	性能测试	96
4.5.3	接口测试	96
4.5.4	局部数据结构测试	96
4.5.5	边界条件测试	97
4.5.6	独立执行路径测试	97
4.5.7	错误处理测试	97
4.6	单元测试方法	98
4.6.1	静态测试	98
4.6.2	动态测试	99
4.7	单元测试用例设计	100
4.8	单元测试过程	101
4.8.1	测试策划	102
4.8.2	静态测试	103
4.8.3	动态测试	111
4.8.4	测试总结	113
<b>第5章</b>	<b>集成测试</b>	<b>114</b>
5.1	概述	114
5.1.1	集成测试的定义	114
5.1.2	集成测试的目的	115
5.1.3	集成测试的重要性	116
5.2	集成测试原则	117
5.3	集成测试环境	117
5.4	集成测试策略	118
5.4.1	大爆炸式集成	119
5.4.2	自顶向下集成	120
5.4.3	自底向上集成	122
5.4.4	三明治式集成	123
5.4.5	核心系统先行集成	124

5.4.6	分层集成	125
5.4.7	基于功能的集成	126
5.4.8	高频集成	127
5.4.9	基于进度的集成	128
5.4.10	基于使用的集成	128
5.4.11	基于风险的集成	129
5.4.12	客户/服务器系统的集成	129
5.5	集成测试内容	130
5.6	测试方法	131
5.6.1	体系结构分析	131
5.6.2	模块分析	131
5.6.3	接口分析	132
5.6.4	可测试性分析	133
5.6.5	集成测试策略分析	133
5.7	集成测试用例设计	133
5.8	集成测试过程	134
5.8.1	测试策划	135
5.8.2	测试设计与实现	137
5.8.3	测试执行	138
5.8.4	测试总结	138
<b>第6章</b>	<b>配置项测试</b>	<b>140</b>
6.1	概述	140
6.1.1	配置项测试的定义	140
6.1.2	配置项测试的目的	140
6.1.3	配置项测试的重要性	141
6.2	配置项测试原则	141
6.3	配置项测试环境	142
6.4	配置项测试策略	143
6.5	配置项测试内容	143
6.6	配置项测试方法	144
6.6.1	功能测试	145
6.6.2	性能测试	146
6.6.3	接口测试	147
6.6.4	人机交互界面测试	148
6.6.5	强度测试	149
6.6.6	余量测试	149
6.6.7	安全性测试	150
6.6.8	恢复性测试	151

6.6.9	边界测试	152
6.6.10	数据处理测试	152
6.6.11	安装性测试	153
6.6.12	容量测试	154
6.7	配置项测试用例设计	154
6.7.1	概述	154
6.7.2	SFME&FTA 综合分析	155
6.7.3	建立软件测试用例设计模式	158
6.7.4	应用实例	159
6.8	配置项测试过程	161
6.8.1	测试策划	162
6.8.2	测试设计与实现	163
6.8.3	测试执行	164
6.8.4	测试总结	164
<b>第7章</b>	<b>系统测试</b>	<b>166</b>
7.1	概述	166
7.1.1	系统测试的定义	166
7.1.2	系统测试的目的	166
7.1.3	系统测试的重要性	167
7.2	系统测试原则	167
7.3	系统测试环境	168
7.4	系统测试策略	168
7.5	系统测试内容	169
7.6	系统测试方法	169
7.6.1	可靠性测试	169
7.6.2	互操作性测试	172
7.6.3	兼容性测试	173
7.7	系统测试用例设计	174
7.7.1	概述	174
7.7.2	系统形式化模型	175
7.7.3	基于模型的系统测试	178
7.7.4	实例	182
7.8	系统测试过程	184
<b>第8章</b>	<b>回归测试</b>	<b>185</b>
8.1	概述	185
8.1.1	回归测试的定义	185
8.1.2	回归测试的目的	186

8.1.3	回归测试的重要性	186
8.2	回归测试策略	187
8.3	软件更动影响域分析方法	187
8.3.1	黑盒测试更动影响域分析	187
8.3.2	白盒测试更动影响域分析	191
8.4	回归测试用例设计	195
8.4.1	回归测试用例设计原则	195
8.4.2	已有测试用例的选取	195
8.5	回归测试过程	198
<b>第9章</b>	<b>面向对象软件测试</b>	<b>200</b>
9.1	面向对象软件简介	200
9.2	面向对象软件测试概述	203
9.2.1	面向对象软件的特点对测试的影响	203
9.2.2	面向对象软件测试和传统测试的不同	204
9.2.3	面向对象软件测试分类	205
9.3	面向对象软件测试模型	205
9.3.1	面向对象分析测试	206
9.3.2	面向对象设计测试	208
9.3.3	面向对象编程测试	208
9.3.4	面向对象单元测试	209
9.3.5	面向对象集成测试	212
9.3.6	面向对象系统测试	215
<b>第10章</b>	<b>FPGA 测试</b>	<b>218</b>
10.1	FPGA 测试概述	218
10.1.1	可编程逻辑器件的基本概念	218
10.1.2	硬件描述语言的发展历程	219
10.1.3	VHDL 语言	219
10.1.4	Verilog HDL 语言	220
10.1.5	面向可编程逻辑器件的开发过程	220
10.1.6	可编程逻辑器件软件与传统软件的不同	222
10.1.7	全过程域的可编程逻辑器件测试框架	223
10.2	静态测试	223
10.2.1	文档审查	224
10.2.2	代码审查	227
10.2.3	编码规则检查	229
10.2.4	跨时钟域分析	230
10.2.5	等效性验证	235

10.2.6	静态时序分析	239
10.3	仿真测试	243
10.3.1	仿真测试的特点	244
10.3.2	仿真测试平台的组成	245
10.3.3	仿真测试的流程	245
10.3.4	功能仿真测试	247
10.3.5	门级仿真测试	248
10.3.6	时序仿真测试	248
10.3.7	仿真测试支持工具	249
10.4	软硬协同验证	250
10.4.1	验证环境构成	250
10.4.2	支持工具	251
10.5	板级验证	251
10.5.1	作用	251
10.5.2	板级验证的典型环境	252
10.5.3	板级验证的流程	252
<b>第 11 章</b>	<b>测试工具</b>	<b>254</b>
11.1	概述	254
11.2	静态测试工具	255
11.2.1	Logiscope	255
11.2.2	PRQA	257
11.2.3	SpyGlass	259
11.2.4	PrimeTime	261
11.2.5	Formalpro	261
11.2.6	其他静态测试工具	262
11.3	动态测试工具	262
11.3.1	QACenter	262
11.3.2	WinRunner	265
11.3.3	JUnit	266
11.3.4	Testbed	268
11.3.5	CodeTest	270
11.3.6	QuestaSim	271
11.3.7	其他动态测试工具	272
11.4	测试管理工具	272
11.4.1	TestCenter	272
11.4.2	TP-Manager	274
11.4.3	其他测试工具	278

第 12 章 软件测试文档 .....	279
12.1 概述 .....	279
12.2 制定测试计划 .....	280
12.2.1 测试计划内容 .....	280
12.2.2 测试计划模板 .....	281
12.2.3 测试计划常见问题 .....	284
12.3 测试设计与实现 .....	285
12.3.1 测试设计与实现的内容 .....	286
12.3.2 测试说明模板 .....	287
12.3.3 测试设计与实现常见问题 .....	288
12.4 测试执行 .....	290
12.4.1 测试执行的内容 .....	290
12.4.2 测试执行模板 .....	290
12.4.3 测试实施常见问题 .....	292
12.5 测试总结 .....	293
12.5.1 测试总结的内容 .....	293
12.5.2 测试总结模板 .....	294
12.5.3 测试总结常见问题 .....	296
参考文献 .....	298

## 软件测试概述

随着科学技术的迅速发展,人类已经进入信息社会。信息的获取、处理、交流和决策都需要大量的软件,软件成为了人们工作和生活中不可或缺的工具。软件的应用日益广泛,软件的质量受到人们越来越多的关注。特别是在航空航天、金融保险、交通通信、工业控制等重要领域,软件一旦失效将造成重大损失,因此对软件质量提出了更高的要求。

1996年6月,阿丽亚娜5(ARIANE 5)火箭在历时十年研制后的首次发射中,由于软件故障造成火箭升空40s后即发生爆炸,直接经济损失达到5亿美元,还使得耗资80亿美元的开发计划推迟了近3年。2003年5月,俄罗斯TMA1号宇宙飞船,由于软件错误导致导航系统故障,造成飞船从国际空间站返回地面时与飞行控制中心失去联系长达11min,飞船最终降落在与预定降落点偏差460多千米的地方。2003年8月,由于FirstEnergy公司电力监测与控制管理系统的预警服务出现软件错误,导致多个重要设备出现故障,而操作员在一个小时后才得到控制站的指示,造成美国及加拿大部分地区史上最大停电事故。

软件测试是保障软件质量的重要手段,是构建高可信软件的关键环节。随着人们对软件测试重要性的认识越来越深刻,软件测试阶段在整个软件开发周期中所占的比重日益增大。统计数据表明,软件测试占软件开发总成本的比例一般达到50%以上。现在有些软件开发机构将40%以上的研制力量投入到软件测试之中;对于某些性命攸关的软件,其测试费用甚至高达所有其他软件工程阶段费用总和的3~5倍。尽管人们在软件开发过程中也采用形式化方法描述和证明软件规约,并采用程序正确性证明、模型检验等方法保证软件质量,但是这些方法都存在一定的局限性,尚未达到广泛实用阶段。软件测试在今后较长时间内仍将是保证软件质量的重要手段。

### 1.1 软件测试简史

人们对软件测试的认识是逐步发展起来的,如图1-1所示。在20世纪50年代,计算机技术发展初期,软件规模都很小,复杂度相对较低,软件错误大部分在开发人员的调试阶段就发现解决了,软件测试被定义为“程序员为了在他们的程序中找到bug所做的事情”。在这个阶段,测试和调试是等同的,都是由开发人员自己完成。

在20世纪60年代早期,软件测试与调试区分开来,测试主要在程序编写后进行。人们开始考虑以遍历代码的可能路径或可能的输入变量的方式,对软件进行“彻底”的测试。现



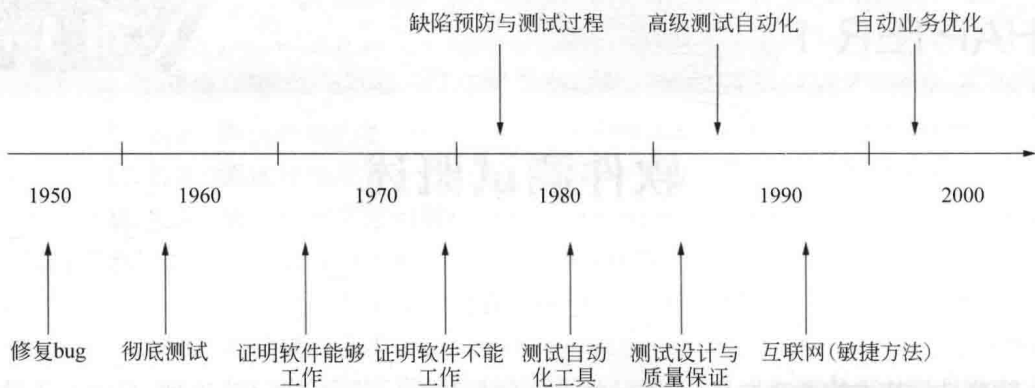


图 1-1 软件测试的发展

在我们知道,由于程序的输入域太大、有太多可能的输入路径以及设计和规约等问题很难测试,完全彻底地测试一个程序是不可能的。

在20世纪70年代早期,随着软件开发的成熟,人们开始提出软件开发的工程化思想,软件测试得到发展,软件测试的地位得到确认。软件测试被定义为“证明一个程序的正确性而要做的事情”,即证明软件能够工作。这期间提出的“正确性证明技术”,建议在软件分析、设计和实现过程中通过证明的方式进行软件正确性验证,在理论上很有前途,但在实践中过于耗时、效率极低。

在20世纪70年代后期,测试被定义为带着找到错误的意图执行程序的过程,而不是证明程序能够运行的过程。与上一观点相反,这种观点强调了好的测试用例能够有更大几率去发现尚未发现的错误,成功的测试是发现了尚未发现的错误的测试。

20世纪80年代,随着软件行业的飞速发展,软件测试的理论和技術得到了快速的发展,人们开始把软件测试作为软件质量保证的重要手段,认识到软件测试不是一次性在开发后期完成的,软件测试也不再仅限于程序本身,软件测试的定义被扩展到缺陷预防,软件测试活动被扩充到对需求、设计、编码、测试等整个软件开发生存周期的过程控制中。IEEE对软件测试定义为“使用人工或自动的手段来运行或测定某个软件系统的过程,其目的在于检验它是否满足规定的需求或弄清楚预期结果与实际结果之间的差别”,并发布了软件测试文档的国际标准。

在20世纪80年代中期,自动化测试工具出现了。相比于手动测试,自动化测试工具的引入大幅提高了测试的效率和質量。测试工具起初还是相当原始的,不具有高级脚本语言工具。

在20世纪90年代早期,从质量保证的观点,测试的含义被定义为“策划、设计、建造、维护和执行测试及测试环境”,软件测试的过程应是受管理的,其自身也存在一个生存周期。更多的录制/回放测试工具提供了丰富的脚本语言和报告功能,测试管理工具可帮助管理从测试需求分析和设计到测试脚本和缺陷的所有内容,也有一些商用的性能工具来测试系统的性能,能够进行压力和负载测试等。

在20世纪90年代中期,人们仍认为测试应是一个贯穿整个软件开发生存周期的过程,但是随着互联网的流行,软件的生存周期模型和开发模式发生了很大的变化,使得软件测试变得越来越困难。测试有时在没有明确预先定义所有测试方向的情况下进行,这