

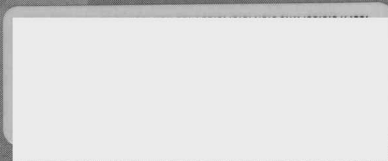
现代软件测试技术 与管理研究

赵仕波 魏生斌 罗耀华 编著

XIANDAI RUANJIAN CESHI JISHU
YU GUANLI YANJIU



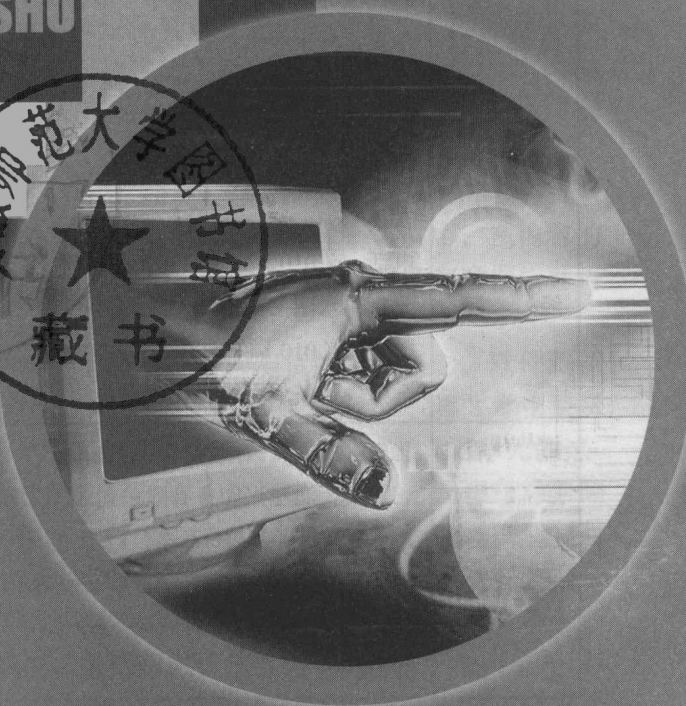
中国水利水电出版社
www.waterpub.com.cn



现代软件测试技术 与管理研究

赵仕波 魏生斌 罗耀华 编著

XIANDAI RUANJIAN CESHI JISHU
YU GUANLI YANJIU



TP311.55
8320



中国水利水电出版社
www.waterpub.com.cn

内 容 提 要

全书共 10 章，以软件测试技术和管理为主要研究对象，介绍了软件测试的相关技术和策略，包括黑盒测试、白盒测试、单元测试、集成测试、系统测试、验收测试，以及面向对象测试、软件自动化测试、国际化测试和本地化测试、测试计划、测试文档、缺陷测试、测试评估，并对软件测试管理、软件质量保证与过程改进等知识点也进行了相关阐述。

本书内容丰富、取材先进、文字表述简单扼要，是一本比较适合软件测试爱好者的实用性强的学术著作类图书，对相关领域的研究人员也是一本颇为有益的参考书。

图书在版编目（C I P）数据

现代软件测试技术与管理研究 / 赵仕波，魏生斌，
罗耀华编著. — 北京：中国水利水电出版社，2014.7
ISBN 978-7-5170-2281-7

I. ①现… II. ①赵… ②魏… ③罗… III. ①软件—
测试—研究 IV. ①TP311.5

中国版本图书馆CIP数据核字(2014)第155649号

策划编辑：杨庆川 责任编辑：杨元泓 封面设计：马静静

书 名	现代软件测试技术与管理研究
作 者	赵仕波 魏生斌 罗耀华 编著
出版发行	中国水利水电出版社 (北京市海淀区玉渊潭南路 1 号 D 座 100038) 网址：www.waterpub.com.cn E-mail: mchannel@263.net (万水) sales@waterpub.com.cn
经 售	电话：(010) 68367658 (发行部)、82562819 (万水) 北京科水图书销售中心 (零售) 电话：(010) 88383994、63202643、68545874 全国各地新华书店和相关出版物销售网点
排 版	北京鑫海胜蓝数码科技有限公司
印 刷	三河市天润建兴印务有限公司
规 格	185mm×260mm 16 开本 17.75 印张 432 千字
版 次	2015 年 1 月第 1 版 2015 年 1 月第 1 次印刷
印 数	0001—3000 册
定 价	62.00 元

凡购买我社图书，如有缺页、倒页、脱页的，本社发行部负责调换

版权所有·侵权必究

前言

软件产业的发展关系到一个国家经济发展和文化安全，体现的是整个国家的综合实力，可以说是决定 21 世纪国际竞争地位的战略性的产业。随着信息技术的普及，人们对软件质量的要求愈来愈高，而且对软件产品质量的稳定性和可靠性也十分重视，如何处理好软件的质量问题一直都是所有软件开发人员工作的重心。早期软件由于规模较小，相对应的 Bug 也较少，但是随着软件规模的增大，因为质量问题将上百万行源代码推倒重来的事例愈演愈烈。在此情况下，软件工程和测试技术应运而生并快速发展起来。

软件测试是软件开发过程中的重要环节，在提高软件质量方面具有不可替代的作用。随着软件产业的迅速发展，市场对于进行专业化、高效化软件测试的需求愈来愈强烈。在软件测试研究和应用方面我国起步较晚，另外，相对于一些发达国家而言，我国在软件测试理论研究、软件测试工具及框架研发、软件测试过程管理、软件质量保证、软件测试工具等方面还比较落后。随着软件测试的新理论与新技术的不断发展，我国的软件从业人员和测试从业人员都将面临严峻的考验。为适应当前形势的需求，我们特此提出并编撰了《现代软件测试技术与管理研究》一书。

本书较为系统地对软件测试技术及相关管理进行了研究，力求做到逻辑严谨、简明易懂、内容新颖。在编撰本书时，我们将软件测试的新概念、新方法、新技术融入其中，在内容的安排上注重由易到难、深入浅出，以便软件测试的基本知识能够更好地被理解和掌握，并将其迅速地运用到实际的测试工作中去。

本书内容分为 10 章。第 1 章主要就软件测试的背景、软件测试的基本概念、软件测试与软件开发、软件测试人员素质，以及软件测试的发展进行了探讨；第 2 章、第 3 章主要研究了软件测试的基本技术及测试策略与过程，包括白盒测试技术、黑盒测试技术、单元测试、集成测试、系统测试、验收测试，以及测试后的调试；第 4 章～第 6 章主要就面向对象测试的方法、模型、用例设计，自动化测试的框架、技术、工具、生存周期方法，以及国际化测试与本地化测试展开研究；第 7 章主要探讨了软件测试计划的制定和测试文档的撰写与管理；第 8 章主要研究了软件缺陷的危害，软件缺陷的生命周期，软件缺陷的跟踪、管理与评估；第 9 章主要对软件测试的相关管理展开研究，包括测试计划管理、测试组织及人员管理、测试进度与成本管理、测试配置管理、测试风险管理；第 10 章从软件质量出发，研究了质量保证、软件测试度量的相关知识，同时根据作者多年的测试经验，对测试过程的改进

提出了一些参考建议，最后对软件维护及再工程技术展开了探讨。

在本书的编撰过程中，参考或引用了有关专家的相关著作，在此表示衷心的感谢。由于作者水平所限，加之时间仓促，书中疏漏和讹误在所难免，恳请各位专家同仁予以批评指正。

作者
2014年1月

前言

随着计算机技术的飞速发展，软件在国民经济和社会生活中扮演着越来越重要的角色。软件维护及再工程作为软件生命周期中不可或缺的一环，其重要性日益凸显。本书旨在系统阐述软件维护及再工程的基本理论、方法和实践，为从事相关工作的工程技术人员提供理论指导和实践参考。

本书共分八章。第一章介绍软件维护及再工程的基本概念、分类及重要性；第二章至第四章分别详细论述了软件需求维护、软件代码维护及软件测试维护；第五章至第七章分别详细论述了软件配置管理、软件文档管理及软件项目管理；第八章总结全文，并对软件维护及再工程的发展趋势进行了展望。

本书可作为高等院校计算机专业及相关专业的教材，也可供从事软件维护及再工程工作的工程技术人员参考。

本书由XXX编写，XXX审核。在编写过程中，得到了XXX、XXX等人的大力支持，特此致谢。

由于时间仓促，书中难免存在不足之处，恳请各位专家同仁批评指正。

目 录

前言	1
第 1 章 导 论	1
1.1 软件测试的背景	1
1.2 软件测试的基本概念	4
1.3 软件测试与软件开发	8
1.4 软件测试人员的素质	14
1.5 软件测试的发展	16
第 2 章 软件测试基本技术	19
2.1 软件测试技术分类	19
2.2 黑盒测试技术	21
2.3 白盒测试技术	29
第 3 章 软件测试策略与过程	38
3.1 软件测试策略概述	38
3.2 单元测试	41
3.3 集成测试	47
3.4 系统测试	53
3.5 验收测试	55
3.6 测试后的调试	60
第 4 章 面向对象测试	65
4.1 面向对象方法	65
4.2 面向对象测试概述	69

4.3	面向对象测试模型	77
4.4	面向对象测试用例设计	86
第5章	软件自动化测试	95
5.1	自动化测试概述	95
5.2	自动化测试框架	102
5.3	自动化测试技术	109
5.4	自动化测试工具	116
5.5	自动化测试生存周期方法	124
第6章	国际化与本地化测试	132
6.1	国际化测试	132
6.2	本地化测试	139
6.3	常用测试工具	150
第7章	测试计划与测试文档	153
7.1	软件测试计划	153
7.2	软件测试文档	169
第8章	软件缺陷测试与测试评估	180
8.1	软件缺陷概述	180
8.2	软件缺陷的生命周期	187
8.3	软件缺陷的跟踪与管理	191
8.4	软件缺陷管理工具	198
8.5	软件测试的评估与总结报告	202
第9章	软件测试管理	208
9.1	测试管理概述	208
9.2	测试计划管理	212
9.3	测试组织及人员管理	218
9.4	测试进度与成本管理	229
9.5	测试配置管理	235
9.6	测试风险管理	243

第 10 章 软件质量保证与过程改进.....249

10.1 软件质量保证..... 249

10.2 软件质量度量..... 253

10.3 软件测试过程改进..... 256

10.4 软件维护与再工程..... 259

参考文献.....276

1.1 软件测试的背景

1.1.1 软件缺陷

1. 造成软件缺陷的原因

产业界（如Nippon Electric TRW）研究表明，软件故障不一定是由编写所引起的大部分是因为在详细设计阶段、概要设计阶段甚至是在需求分析阶段存在的问题引起的。如果软件需求说明书写得不够全面、清楚，在开发过程中经常被更改，或开发组的成员之间没有很好地进行交流和沟通，都会导致软件缺陷。如图 1-1 所示，为软件缺陷的原因分布图。软件需求说明书产生的缺陷最大，其次是设计阶段产生的软件缺陷，由源代码引起的软件缺陷只占 7%，其他原因引起的软件缺陷占 10%。

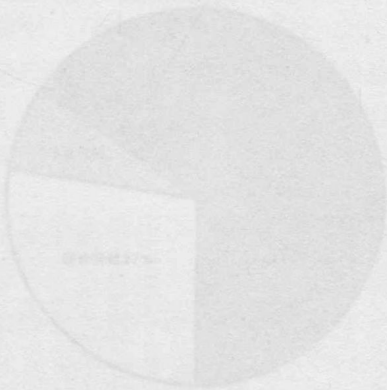


图 1-1 软件缺陷的原因分布图

第 1 章 导 论

软件无处不在，人们在不同的场合都有可能会在不知不觉中使用软件，如日常生活中的手机、智能冰箱、新一代的数字电视等，软件越来越多地影响和改变人类生活的各个方面。然而，软件构成及开发的日益复杂、软件应用领域的日益拓宽也使得人们常常受到有缺陷的软件的影响，软件缺陷给人们带来了许多物质上和精神上的损失。软件质量不断受到人们的重视，为了发现软件中的缺陷，保证软件质量，软件测试应运而生。

1.1 软件测试的背景

1.1.1 软件缺陷

1. 造成软件缺陷的原因

产业界（如 Nippon Electric、TRW）研究表明，软件故障不一定是由编码所引起的，大部分是因为在详细设计阶段、概要设计阶段甚至是在需求分析阶段存在的问题引起的。如果软件需求说明书写得不够全面、清楚，在开发过程中经常被更改，或开发组的成员之间没有很好地进行交流和沟通，都会导致软件缺陷。如图 1-1 所示，为软件缺陷的原因分布图。软件需求说明书产生的缺陷最大，其次是设计阶段产生的软件缺陷，由源代码引起的软件缺陷只占 7%，其他原因引起的软件缺陷占 10%。

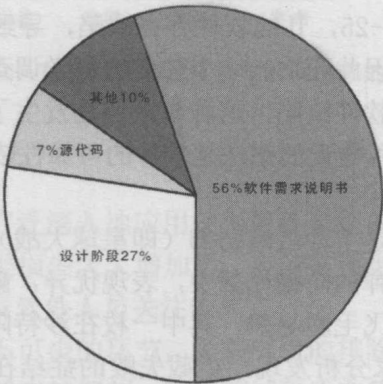


图 1-1 软件缺陷的原因分布图

软件从设计、编写、测试，直到用户公开使用的过程中，都有可能产生软件缺陷，随着软件整个开发过程的推移，软件修正的费用呈几何倍数增长，如图 1-2 所示。

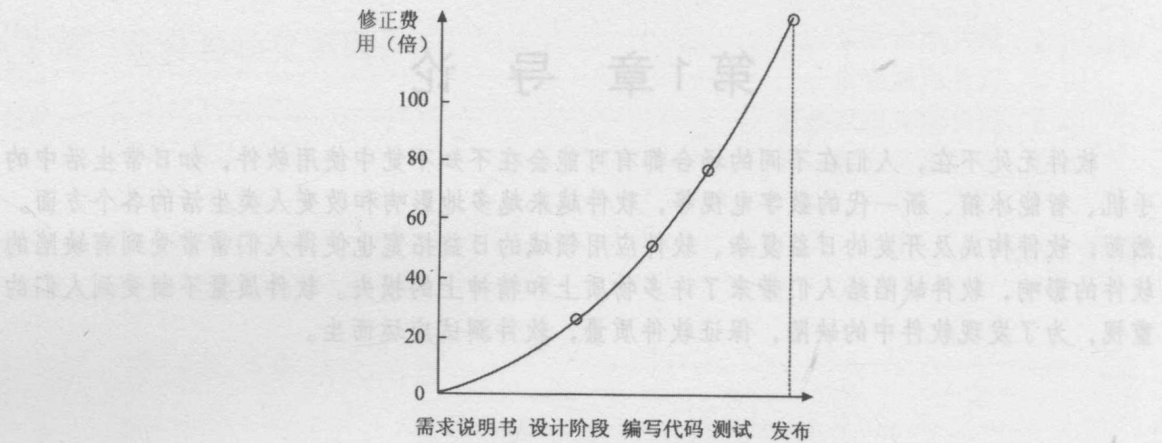


图 1-2 不同阶段的软件缺陷修正费用

IBM 公司的研究结果表明，软件缺陷存在放大的趋势。如果在需求阶段漏过一个错误，则该错误可能会引起 K 个设计错误， K 称为放大系数。不同阶段的 K 的数值不同。经验表明，从概要设计到详细设计阶段的缺陷放大系数约为 1.5，从详细设计到编码阶段的缺陷放大系数约为 3。如图 1-3 所示即为缺陷放大的大致状况。

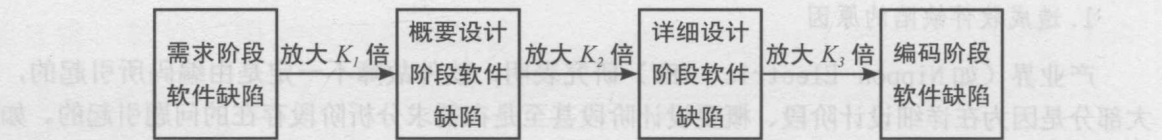


图 1-3 缺陷放大模型图

2. 软件缺陷带来的困扰

(1) 放疗设备致死案

由于放射性治疗仪 Therac-25, II 的软件存在缺陷，导致几个癌症病人受到非常严重的过量放射性治疗，其中 4 个人因此死亡。一个独立的科学调查报告显示：即使在加拿大原子能公司已经处理了几个特定的软件缺陷，这种事故还是发生了。造成这种低级但致命错误的原因是缺乏软件工程实践，一种错误的想法是软件的可靠性依赖于用户的安全操作。

(2) 爱国者导弹防御系统

美国爱国者导弹防御系统是主动战略防御（即星球大战）系统的简化版本，它在第一次海湾战争对抗伊拉克飞毛腿导弹的防御作战中，表现优异，赢得各界的赞誉。但它还是有几次失利，没有成功拦截伊拉克飞毛腿导弹，其中一枚在沙特阿拉伯的多哈爆炸的飞毛腿导弹造成 28 名美军士兵死亡。专家分析发现，拦截失败的症结在于爱国者导弹防御系统的一个软件缺陷，当爱国者导弹防御系统的时钟累计运行超过 14 小时后，系统的跟踪系统就会不

准确。在多哈袭击战中，爱国者导弹防御系统运行时间已经累计超过 100 多个小时，显然那时系统的跟踪系统已经很不准确，从而造成这种结果。

（3）丹佛新机场推迟启用

丹佛新国际机场希望被建成现代的（state-of-the-art）机场，它将拥有复杂的、计算机控制的、自动化的包裹处理系统，而且，还有 5300 英里长的光纤网络。不幸的是，在这包裹处理系统中存在一个严重的程序缺陷，导致行李箱被绞碎，居然还开着自动包裹车往墙里面钻。结果，机场启用推迟 16 个月，使得预算超过 32 亿美元，并且废弃这个自动化的包裹处理系统，使用手工处理包裹系统。

3. 软件缺陷修复的代价

缺陷被发现之后，要尽快修复这些被发现的缺陷。错误并不只是在编程阶段产生，需求和设计阶段同样会产生错误。也许一开始，只是一个较小范围内的潜在错误，但随着产品开发工作的进行，小错误会扩散成大错误，为了修改后期的错误所做的工作要大得多，即越到后来往前返工也越复杂。如果错误不能及早发现，那只能造成越来越严重的后果。缺陷发现或解决的越迟，成本就越高。

平均而言，如果在需求阶段修正一个错误的代价是 1，那么，在设计阶段就是它的 3~6 倍，在编程阶段是它的 10 倍，在内部测试阶段是它的 20~40 倍，在外部测试阶段是它的 30~70 倍，而到了产品发布出去时，这个数字就是 40~1000 倍。修正错误的代价不是随时间线性增长，而几乎是呈指数增长的，如图 1-4 所示。

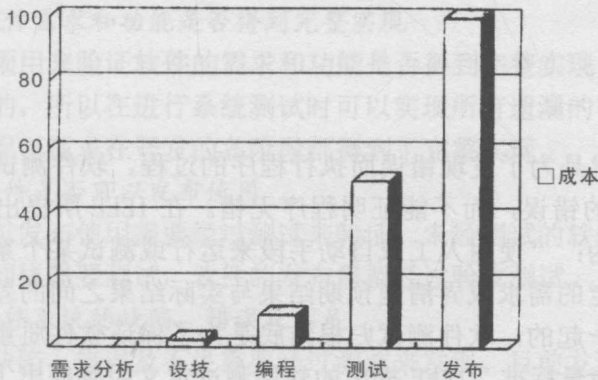


图 1-4 软件缺陷修复代价与时间推移的关系

1.1.2 软件可靠性

计算机技术的迅速发展和广泛深入地应用使得软件系统的规模和复杂性与日俱增，软件中存在的缺陷与故障造成的各类损失大大增加了，有的甚至带来了灾难性的后果。软件质量问题成为了所有使用软件和开发软件人员关注的焦点。

软件测试是软件开发中必不可少的环节，是最有效的排除和防治软件缺陷的手段。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。大量测试文献表明,通常花费在软件测试和排错上的代价大约占软件开发总代价的 50% 以上。现在有些软件开发机构将研制力量的 40% 以上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。

在已投入运用的软件质量中,软件可靠性是其中一个重要标志。从实验系统所获得的统计数据表明,运行软件的驻留故障密度各不相同,与生命攸关的关键软件为每千行代码 0.01~1 个故障,与财务(财产)有关的关键软件为每千行代码 1~10 个故障,其他对可靠性要求相对较低的软件系统故障就更多了。然而,正是由于软件可靠性的大幅度提高才使得计算机得以广泛应用于社会的各个方面。

一个可靠的软件应该是正确的、完整的、一致的和健壮的。美国电气和电子工程师协会(IEEE)将软件可靠性定义为:系统在特定的环境下,在给定的时间内无故障地运行的概率。软件可靠性牵涉到软件的性能、功能性、可用性、可服务性、可安装性、可维护性以及文档等多方面特性,是对软件在设计、生产以及在它所预定环境中具有所需功能的置信度的一个度量,是衡量软件质量的主要参数之一。软件测试则是保证软件质量,提高软件可靠性的最重要手段。

1.2 软件测试的基本概念

1.2.1 软件测试的定义

软件测试,简单来说就是为了发现错误而执行程序的过程。软件测试是一个找错的过程,测试只能找出程序中的错误,而不能证明程序无错。在 IEEE 所提出的软件工程标准术语中,软件测试被定义为:“使用人工或自动手段来运行或测试某个系统的过程,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。”软件测试是与软件质量密切联系在一起的,软件测试归根结底是为了保证软件质量。通常软件质量是以“满足需求”为基本衡量标准,IEEE 提出的软件测试定义明确提出了软件测试以检验是否满足需求为目标。

著名软件测试专家 Glen Myers 认为“软件测试是为了发现错误而执行程序的过程”。根据这个定义,软件测试是根据软件开发各个阶段的规格说明和程序的内部结构而精心设计的一批测试用例,并利用这些测试用例运行程序以及发现错误的过程,即执行测试步骤。测试是采用测试用例执行软件的活动,它有两个显著目标:找出失效或演示正确的执行。其中,测试用例是为特定的目的而设计的一维输入输出、执行条件和预期结果,测试用例是执行测试的最小实体。

测试步骤详细规定了如何设置、执行、评估特定的测试用例。除此之外,Glen Myers 在他关于软件测试的著作中陈述了一系列可以服务于测试目标的规则,这些规则也是被广泛接

受的:

- 测试是为了证明程序有错,而不是证明程序无错误。
- 一个好的测试用例是在于它能发现至今未发现的错误。
- 一个成功的测试是发现了至今未发现的错误的测试。

在这一测试定义中,明确指出“寻找错误”是测试的目的,相对于“程序测试是证明程序中不存在错误的过程”,Myers的定义是对的。因为把证明程序无错当作测试的目的不仅是不正确的、完全做不到的,而且对于做好测试工作没有任何益处,甚至是十分有害的。因此从这方面讲,可以接受Myers的定义以及它所蕴含的方法观和观点。不过,这个定义也有其局限性。它将测试定义规定的范围限制得过于狭窄,测试工作似乎只有在编码完成以后才能开始。更多专家认为软件测试的范围应当更为广泛,除了要考虑测试结果的正确性以外,还应关心程序的效率、可适用性、维护性、可扩充性、安全性、可靠性,系统性能、系统容量、可伸缩性、服务可管理性、兼容性等因素。随着人们对软件测试更广泛、深刻的认识,可以说对软件质量的判断决不只限于程序本身,而是整个软件研制过程。

对上述内容进行分析总计,可以对软件测试作出如下定义:软件测试是为了尽快尽早地发现在软件产品中所存在的各种软件缺陷而展开的贯穿整个软件开发生命周期,对软件产品(包括阶段性产品)进行验证和确认的活动过程。

1.2.2 软件测试的目的

软件测试的目的,可以归纳为以下几个方面。

(1) 验证软件需求和功能是否得到完整实现

测试首先必须用来验证软件的需求和功能是否得到完整实现。系统测试是根据软件产品需求规格来进行的,所以在进行系统测试时可以实现所有遗漏的需求。但在开发时还是需要通过需求跟踪来保证需求在开发的各阶段都得到了完整实现。

(2) 验证软件是否可以发布使用

软件是否可以发布使用需要经过测试来验证,未经测试的软件是不能发布的。即使是内部使用的软件也同样需要测试,软件的发布需要经过验收测试。

(3) 发现软件系统的缺陷、错误及不足

软件系统的缺陷、错误及不足需要经过测试来发现。目前发现软件系统的缺陷、错误及不足的主要手段有评审、检视、走读、单元测试、集成测试、系统测试等。

(4) 获取软件产品的质量信息

软件产品的质量信息也必须通过测试才能获取,没有经过测试的软件,软件质量的好坏是无从知道的,最多只能根据开发人员的水平进行推测。经过测试后,就可以得到开发各阶段发现的缺陷数,进而可以较为准确地推测出软件潜在的缺陷数。

(5) 预防下一版本可能出现的问题

测试不仅可以用来发现当前版本的问题,还可以根据目前发现的问题进行分析,找出当前版本出现的问题有哪些类型,产生这些类型问题的根源是什么。

(6) 预防用户使用软件时可能出现的问题

把没有经过测试的软件提供给用户使用，将会使用户在使用过程中遭受大量挫折，大大降低了愉快的用户体验。测试可以有效地发现大部分影响使用的错误，经修正后软件预防了用户使用软件时可能出现的问题。

(7) 提前发现开发过程中的问题和风险

测试还能提前发现开发过程中的问题和风险。写系统测试用例时可以发现需求中的问题和遗漏，写集成测试用例时可以发现高层设计中的问题，写单元测试用例则可以发现详细设计和编码中的问题。通过测试，可以在早期阶段就发现这些错误，降低开发的风险。

(8) 提供可以用以分析的测试结果数据

测试还能提供用以分析的测试结果数据、测试问题记录表等数据。在测试完后进行分析，可以了解主要有哪些类型的缺陷，进而分析产生这些类型缺陷的原因。还可以分析开发各阶段发现的问题数，把他们与以前的经验数据进行对比分析，从而确定在开发阶段中哪个阶段是薄弱环节，进而对薄弱环节进行加强。

1.2.3 软件测试的原则

软件测试从不同的角度会有不同的测试原则。软件测试的基本原则是站在用户的角度，对产品进行全面测试，尽早、尽可能多地发现缺陷，并负责跟踪和分析产品中的问题，对不足之处提出质疑和改进意见。零缺陷只是一种理想，足够好是测试的原则。根据测试目的，软件测试的基本原则可归纳如下：

- 1) 应当把“尽早地和不断地进行软件测试”作为软件开发者的座右铭。
- 2) 程序员应避免检查自己的程序（不是指对程序的调试），测试工作应该由独立、专业的软件测试机构来完成。
- 3) 在设计测试用例时，应当包括合理的输入条件和不合理的输入条件。不合理的输入条件是指异常的、临界的、可能引起问题异变的输入条件。
- 4) 测试用例应由测试输入数据和与之对应的预期输出结果两部分组成。
- 5) 充分注意测试中的群集现象。经验表明，测试后程序残存的错误数目与该程序中已发现的错误数目或检错率成正比。应该对错误群集的程序段进行重点测试。
- 6) 严格执行测试计划，排除测试的随意性。测试计划应包括：所测软件的功能，输入和输出，测试内容，各项测试的进度安排，资源要求，测试资料，测试工具，测试用例的选择，测试的控制方法和过程，系统的组装方式，跟踪规则，调试规则，回归测试的规定等等以及评价标准。
- 7) 应当对每一个测试结果做全面的检查。测试时间应当尽量宽松，不要希望在极短的时间内完成一个高水平的测试。
- 8) 妥善保存测试计划，测试用例，出错统计和最终分析报告，为维护提供方便。
- 9) 全面彻底地检查每一个测试结果，避免不可再现的测试。
- 10) 在某一程序片段中发现的错误越多，则这个程序段所隐含的尚未发现错误的可能性就越大。这就是测试中的群集现象。经验表明，测试后程序中残留的错误数目与程序中已发现的错误数目成正比。根据这一规律，应该对出现错误群集的程序段进行重点测试，以提高

测试投资的效益。

- 11) 让最好的程序员去进行测试的工作，不要为使测试变得容易而更改程序。
- 12) 设计软件系统要保证将要集成到系统中的每个模块仅集成一次，注意确保软件的可测试性。

1.2.4 软件测试信息流

软件测试信息流的示意图如图 1-5 所示。

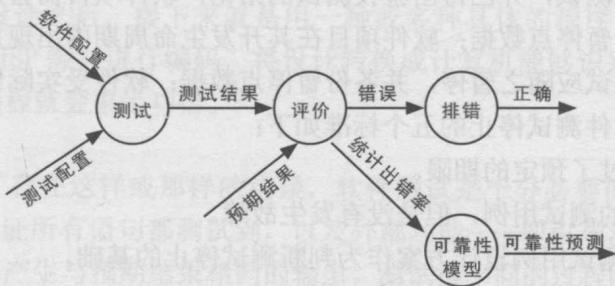


图 1-5 测试信息流

一般来说，实施测试应包括三类信息。

- (1) 软件配置
软件配置指的是测试的对象，包括软件需求规格说明书、设计规格说明书和被测试的源程序。

- (2) 测试配置
测试配置通常包括测试计划、测试步骤、测试用例或测试数据，以及具体实施测试的测试程序等。实际上，在整个软件工程中，测试配置只是软件配置的一个子集。

- (3) 测试工具
为提高软件测试效率，可以使用测试工具支持测试工作，其作用就是为测试的实施提供某种服务，以减轻测试任务中的手工劳动。例如，测试数据自动生成程序、静态分析程序、动态分析程序、测试结果分析程序以及驱动测试的测试数据库等。

测试之后，要对所有测试结果进行分析，即将实测的结果与预期的结果进行比较。如果发现出错的数据，就意味着软件有错误，就需要开始调试排错。即对已经发现的错误进行错误定位，确定出错性质，并改正这些错误，同时修改相关的文档。修正后的文档一般都要经过再次测试，直到通过测试为止。

排错的过程是测试过程中最不可预知的部分，即使是一个微小的错误，也可能需要花上很长的时间去查找原因并改正错误。也正是因为排错中的这种固有的不确定性，使得我们很难确定可靠的测试进度。

通过收集和分析测试结果数据，即可针对软件建立可靠模型。如果经常出现需要修改设计的严重错误，那么软件质量和可靠性就值得怀疑，同时也表明需要进一步测试。反之，若软件功能能够正确完成，出现的错误易于修改，那么就可以断定，或者是软件的质量和

可靠性达到了可以接受的程度，或者是所做的测试不足以发现严重的错误。如果测试发现不了错误，那么几乎可以肯定，测试配置考虑得不够细致充分。错误仍然潜伏在软件中。这些错误最终不得不由用户在使用过程中发现，并在维护时由软件开发人员去改正。但那时改正错误的费用将远远大于在开发阶段的改正。

1.2.5 软件测试停止标准

软件系统经过单元、集成、系统等测试，分别达到单元、集成、系统等测试停止的标准；软件系统已经过验收测试，并已得出验收测试的结论；软件项目需暂停以进行调整时，测试应随之暂停，并备份暂停点数据；软件项目在其开发生命周期内出现重大估算、进度偏差，需暂停或中止时，测试应随之暂停，并备份暂停点数据；软件受实际情况的制约，软件测试最终是要停止的。软件测试停止的五个标准如下：

- 1) 测试时间超过了预定的期限。
- 2) 执行了所有的测试用例，但是没有发生故障。
- 3) 使用特定的测试用例设计方案作为判断测试停止的基础。
- 4) 正面指出了停止测试的具体要求。
- 5) 根据单位时间内查出的缺陷的数量判断是否停止测试。

1.3 软件测试与软件开发

1.3.1 软件产品的组成

软件产品一般需要客户需求说明书、软件产品说明书、软件设计文档（常用的软件设计文档的内容包括：构架即描述软件整体设计的文档、状态变化示意图、数据流示意图、流程图、注释代码等）、软件开发进度表、软件测试文档、软件产品组成部分（主要包括：帮助文件、用户手册、样本和示例、产品和支持信息、标签、图表、错误提示、广告宣传材料、软件安装说明书、说明软件文件、测试错误提示信息）。

1.3.2 软件开发的基本过程

软件开发的基本过程，可以被简单地分为需求分析、概要设计和详细设计、编程、测试和维护等几个阶段。

(1) 需求分析

需求分析是根据客户的要求，清楚地了解客户需求中的产品功能、特性、性能、界面和

具体规格等，然后进行分析，确定软件产品所能达到的目标。软件产品需求分析是软件开发过程的第一个环节，也是至关重要的环节。如果需求分析做的有瑕疵，后面的设计、编程做得再好，客户（用户）也不可能对开发出来的软件产品感到满意。

(2) 设计

软件设计是根据需求分析的结果，考虑如何在逻辑、程序上去实现所定义的产品功能、特性等。可以分为概要设计和详细设计，也可以分为数据结构设计、软件体系结构设计、应用接口设计、模块设计、算法设计、界面设计等。

(3) 编程

经过需求分析、设计之后，接下来就是用一种或多种具体的程序语言（如 C / C++、Java、PHP / ASP / JSP 等）进行编码，将设计转换成计算机能够识别的形式。如果设计做得好、做得仔细，编程就会事半功倍。

(4) 测试

任何编程，免不了存在这样或那样的错误，软件测试是十分必要的。测试过程集中于软件的内部逻辑——保证所有语句都测试到，以及外部功能——即引导测试去发现错误，并保证定义好的输入能够产生与预期结果相同的输出。测试按不同的过程阶段分为单元测试、集成测试、功能测试、系统测试、验证测试等。

(5) 维护

从理论上，软件测试的覆盖率不可能做到百分之百，所以软件在交付给用户之后有可能出现其他问题，而且用户的需求会发生变化，特别是开始使用产品之后，对计算机系统有了真正的认识 and 了解，会提出适用性更好的、功能增强的要求。因此，软件交付之后不可避免地要进行修改、升级等。

1.3.3 软件开发模式

软件开发过程包含各种复杂的风险因素，为了解决由这些风险带来的种种问题，软件开发人员经过多年的摸索，总结出了许多软件工程的实现方式——软件过程模型。目前主要有以下几种模型：

1. 瀑布模型

瀑布模型将软件生命周期的各项活动规定为按照固定的顺序相连的若干阶段性的工作，形如瀑布流水，最终得到软件产品，这种开发模式就叫做瀑布模式。如图 1-6 所示，为瀑布开发模式示意图。

瀑布开发模式具有的特点是易于理解；强调早期计划及需求调查；调研开发的阶段性；确定何时能够交付产品及何时进行评审与测试。但是该开发模式需求调查分析只进行一次，不能适应需求变化；顺序的开发流程使开发中的经验教训不能反馈到修改项目的开发中；不

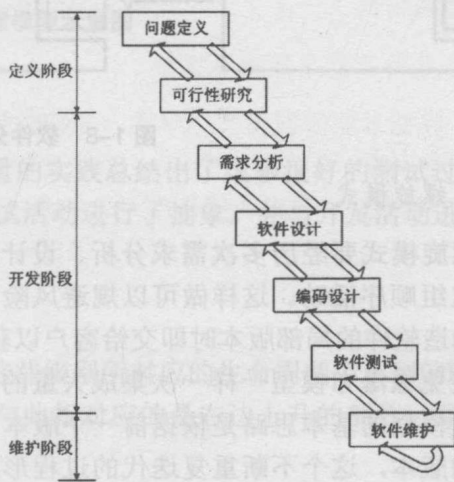


图 1-6 瀑布开发模型