重点大学计算机教材

HZ BOOKS
华章教育

软件测试教程

Software Testing

宫云战 主编
赵瑞莲 张威 赵会群 等编著

为教师配有电子课件

机械工业出版社
China Machine Press

重点大学计算机教材

软件测试教程

Software Testing

宫云战　主编
赵瑞莲　张威　赵会群　等编著

机械工业出版社
China Machine Press

本书系统介绍了软件测试的基本原理和常用方法，同时阐述了近几年出现的一些新的软件测试方法，基本上涵盖了当今软件测试技术的全部内容。本书共分8章，内容包括：软件测试概述、黑盒测试、白盒测试、基于缺陷模式的软件测试、集成测试、系统测试、软件评审和测试管理。本书结合实例，介绍了多种目前比较流行的软件测试工具，并将它们合理地融合在每一章中。

本书可作为高等院校计算机专业本科生、研究生的教材，也可以作为从事软件测试与软件质量保障工程师的参考书。

图书在版编目（CIP）数据

软件测试教程 / 宫云战主编. —北京：机械工业出版社，2008.9
（重点大学计算机教材）

ISBN 978-7-111-24897-2

I. 软… II. 宫… III. 软件－测试－高等学校－教材 IV. TP311.5

中国版本图书馆CIP数据核字（2008）第123401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：周茂辉
三河市明辉印装有限公司印刷·新华书店北京发行所发行
2008年9月第 1 版第 1 次印刷
184mm × 260mm · 15.75印张
标准书号：ISBN 978-7-111-24897-2
定价：29.00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：（010）68326294

序

在过去的几十年中，软件技术得到了快速发展，软件系统的应用已经遍布社会的各个领域，成为人类改造自然不可或缺的重要组成部分。

一方面，软件的应用给社会带来了巨大的进步，大大提升了人们改造自然的能力。另一方面，由于软件的故障、漏洞等因素导致的软件不可信的程度也变得更加突出，这在很大程度上制约了软件技术的发展和软件系统的使用。软件的不可信问题正由一个纯粹的技术问题向社会问题转变，已到了非解决不可的地步。在诸如航空、航天、电信、医疗、金融等众多安全第一的领域，软件错误造成的危害是触目惊心的。

软件是一个逻辑体，软件中的错误都是由人类自己造成的。由于软件规模、复杂性等因素，使得难以证明软件是正确的。软件中的错误是不可避免的，人们只能根据需要尽可能地减少软件中的错误。

软件测试是发现软件缺陷，提高软件可信性的重要手段。在过去的三十年中，随着社会对软件测试需求的增加，软件测试理论和技术得到了较快的发展。特别是近十年来，国际上一些著名的学术机构，以及微软、IBM等众多国际IT巨头的参与，使得软件测试理论正在走向成熟，软件测试对错误与缺陷的发现能力、软件测试工具的自动化程度都得到了大幅度的提升。以软件测试工具、软件测试服务为主导的软件测试产业正在兴起，目前在全国已经形成近2 000家的软件评测企业、数十万人的软件测试队伍。

本书作者长期以来一直从事软件测试技术的研究和教学，对软件测试技术有比较深刻的理解，对软件测试的教学有比较好的把握。希望这本书为大家学习、理解软件测试技术提供有益的参考。

中国科学院
中国工程院 院士
北京邮电大学 教授

2008年7月

前　言

4年前，宫云战教授和赵瑞莲教授分别写过一本名为《软件测试》的书。同4年前相比，软件测试技术与软件产业得到了快速发展，主要表现在：社会对其认识更加深刻、需求增大；我国的软件评测企业大幅度增加，目前已有近2 000家，各个行业、各个省、发达地区的各个市都建立了软件评测中心；软件测试从业人员已达10数万人，我国软件测试产业产值已经达到上百亿元；国际上的IT巨头，如IBM、微软等，都在从事与软件测试相关的工作，众多的IT企业都在中国建立了软件测试外包为主导的软件企业；以软件测试工具和软件测试服务为核心的软件测试产业每年都在以超过20%速度递增；软件测试学术活动异常活跃，新的测试方法和测试工具不断出现。相比之下，原来书中有些内容虽然理论性强，但实用价值不大，而有些内容则处于被淘汰阶段。所有这些因素都促使我们认为有必要重新撰写本书。

本书叙述软件测试的一般原理和各种基本方法，包括基本的白盒测试、黑盒测试和集成测试方法，并结合近几年软件测试技术的发展，重点介绍了目前国际上一些比较流行的软件测试方法与软件测试工具，包括：

1) 面向缺陷模式的软件测试技术：该技术以其缺陷检测效率高、准确，自动化程度高、易学等特点，在过去的几年中得到迅速发展，目前大约有80多个与该技术相关的工具。在美国，以该工具为基础的软件测试服务取得了很大的成功，成为美国一种主流软件测试技术。目前，随着缺陷模式的不断增加，该技术将有更广阔的应用前景。本书叙述了该技术的一般方法以及作者应用该技术开发的一款软件测试系统——缺陷测试系统（DTS）。

2) 软件评审：软件评审发现缺陷的效率高且比较经济，是目前常用的提高软件质量的方法，已在许多大型软件开发中得到了印证。本书详细叙述了软件评审的内容及如何组织软件评审。

3) 随着软件开发规模的扩大、复杂程度的增加，为了尽可能多地找出程序中的故障，开发出高质量的软件产品，必须对测试工作进行组织策划和有效管理，并采取系统的方法建立起软件测试管理体系。通过它们对测试活动进行监管和控制，以确保软件测试在软件质量保证中发挥应有的关键作用。

4) 软件测试工具是提高软件测试效率与质量的重要手段，在过去的几年中，在软件开发过程的各个阶段，产生了大量的软件测试工具，一些新技术的使用，也使得软件测试工具的自动化程度得到了大幅度的提高。本书介绍了目前多种主流的软件测试工具。

5) 近几年来，随着IT技术的发展，与软件系统交互的相关技术也越来越多，包括网络、协议、安全性、界面等，所有这些方面都需要测试，而这些测试和基本的软件测试是不同的。本书全面论述了软件系统以及与此相关的系统测试。

赵瑞莲教授编写了本书的第1、2、8章，赵会群教授编写了第6章，张威教授、万琳副教授编写了第3、5、7章，杨朝红博士编写了第4章，全书由宫云战教授统稿、审查。

限于作者的水平，书中对某些问题的论述可能是肤浅的，也可能存在错误，恳请读者批评指正。

宫云战

2008年5月4日于北京

教学建议

教学内容	学习要点及教学要求	课时安排（学时）
第1章　软件测试概述	• 了解软件可靠性问题 • 了解软件测试的目的和意义 • 掌握软件测试过程 • 掌握软件测试与软件开发的关系 • 了解软件测试的发展历程和现状 • 了解我国软件测试产业现状 • 了解软件测试工具	2
第2章　黑盒测试	• 了解黑盒测试的基本概念 • 掌握等价类划分方法及其在测试中的应用 • 掌握边界值分析法及其在测试中的应用 • 掌握因果图法及其在测试中的应用 • 了解决策表法及其在测试中的应用 • 了解常用的黑盒测试工具	6
第3章　白盒测试	• 掌握控制流测试方法 • 掌握数据流测试方法 • 了解程序插装的概念及方法 • 掌握程序变异测试方法 • 了解常用的白盒测试工具 • 掌握常用的软件缺陷分析方法	8
第4章　基于缺陷模式的软件测试	• 了解基于缺陷模式的软件测试的概念 • 了解基于缺陷模式的软件测试指标 • 掌握常用的缺陷模式，包括故障模式、安全漏洞模式、缺陷模式和规则模式 • 熟悉基于缺陷模式的软件测试系统（DTS）	8
第5章　集成测试	• 了解集成测试的概念、层次及原则 • 掌握常用的集成测试策略 • 掌握集成测试用例设计方法 • 了解集成测试过程 • 了解面向对象的集成测试	4
第6章　系统测试	• 掌握性能测试的概念及方法 • 掌握压力测试的概念及方法 • 了解容量测试的概念及方法 • 掌握健壮性测试的概念及方法 • 掌握安全性测试的概念及方法 • 掌握可靠性测试的基本概念及模型 • 了解恢复性测试与备份测试的概念 • 了解协议一致性测试的概念及方法 • 了解兼容性测试的概念 • 了解安装性测试的概念 • 掌握可用性测试的概念及方法 • 了解配置性测试的概念及方法	8

（续）

教学内容	学习要点及教学要求	课时安排（学时）
第6章　系统测试	• 了解文档性测试的概念及方法 • 了解GUI测试的概念及方法 • 掌握验收测试的内容、策略及方法 • 了解回归测试的概念及方法 • 了解系统测试工具及其应用	8
第7章　软件评审	• 了解软件评审的目的、组织及管理 • 掌握需求评审的概念及评审细则 • 掌握概要设计评审的概念及评审细则 • 掌握详细设计评审的概念及评审细则 • 了解数据库设计评审的概念及评审细则 • 掌握测试评审的概念及评审细则	2
第8章　测试管理	• 了解测试管理体系 • 掌握测试的组织管理、过程管理、资源和配置管理及文档管理 • 掌握测试管理的原则 • 了解常用的测试管理工具	2

目　录

第 1 章　软件测试概述

随着计算机技术的飞速发展，人们对计算机的需求和依赖与日俱增。随之而来的是计算机系统的规模和复杂性急剧增加，其软件开发成本以及由于软件故障而造成的经济损失也正在增加，软件质量问题已成为人们关注的焦点。因此，许多科学家在展望21世纪计算机科学发展方向和策略时，把软件质量放在优先于提高软件功能和性能的位置上。

软件测试是对软件需求分析、设计规格说明和编码的最终复审，是软件质量保证的关键步骤。随着软件系统规模和复杂性的增加，进行专业化高效软件测试的要求越来越严格，软件测试职业的价值逐步得到了认可，软件测试从业人员急剧增加，软件测试评测中心如雨后春笋般成长起来。软件测试技术已经作为一门新兴产业而快速发展起来了。

1.1　计算机系统的软件可靠性问题

随着人们对计算机需求和依赖的与日俱增，计算机系统的规模和复杂性急剧增加，使得计算机软件的数量以惊人的速度急剧膨胀。与此同时，计算机出现故障引起系统失效的可能性也逐渐增加。由于计算机硬件技术的进步，元器件可靠性的提高，硬件设计和验证技术的成熟，硬件故障相对显得次要了，软件故障正逐渐成为导致计算机系统失效和停机的主要因素。

下面介绍几个实例，以说明软件故障可能造成的损失和灾难。

1. 千年虫问题

千年虫问题是一个众所周知的软件故障。20世纪70年代，人们所使用的计算机存储空间很小，这就迫使程序员在开发工资系统时尽量节省存储空间，一个简单的方法是在存储日期时，只存储2位，如1974存储为74。工资系统常依赖于日期的处理，因此他们节省了大量的存储空间。他们知道在2000年到来时，会出现问题，比如银行在计算利息时，是用当前的日期（如“2000年1月1日”）减去客户的存款日期（如“1974年1月1日”），如果年利息为3%，那么每100元银行应付给客户78元的利息。如果年份存储问题没有得到纠正，其存款年数就变为－74年，客户反而应该付给银行利息了，这显然是不合理的。但他们认为在20多年内程序肯定会更新或升级，而且眼前的任务比计划遥不可及的未来更加重要。为此，全世界付出了数亿美元的代价来更换或升级类似程序以解决千年虫问题，特别是金融、保险、军事、科学、商务等领域，花费了大量的人力、物力对已有的各种各样程序进行检查、修改和更新。

2. 爱国者导弹防御系统

1991年，美国爱国者导弹防御系统首次应用在海湾战争中对抗伊拉克飞毛腿导弹。尽管人们对该系统的赞誉不绝于耳，但是它确实在几次对抗导弹战役中出现了失误，其中一枚在沙特阿拉伯的多哈误杀了28名美国士兵。分析专家发现症结在于一个软件故障。一个很小的系统时钟错误积累起来就可能拖延14小时，造成跟踪系统失去准确度。在多哈袭击战中，这样一个小故障造成系统被拖延100多个小时。

3. 美国火星登陆事故

1999年，美国宇航局火星极地登陆飞船在试图登陆火星表面时突然坠毁失踪。故障评测委员会调查分析了这一故障，认定出现该故障的原因可能是由于某一数据位被更改，并认为该问题在内部测试时应该能够解决。

为什么会这样呢？简单而言，火星登陆过程计划是：飞船在火星表面降落时，着陆伞自动打开以减缓飞船的下降速度。当飞船距离火星表面1 800米时，丢弃着陆伞，点燃登陆推进器，缓缓降落到地面。然而，美国宇航局为了节省开销，简化了关闭着陆推进器的装置，在飞船的支撑脚部安装了一个触点开关，在计算机中设置一个数据位来控制触点，以关闭飞船燃料。显然，飞船没着陆以前，推进器就应该一直处于着火工作状态。不幸的是，在许多情况下，当飞船的支撑脚迅速打开准备着陆时，机械震动也会触发触点开关，导致设置了错误的数据位，关闭了登陆推进器的燃料，使飞船加速下降1 800米后撞向地面，撞成碎片。

结果是灾难性的，但原因很简单。事实上，飞船发射之前，经过了多个小组的测试，其中一个小组负责测试飞船支撑脚的落地打开过程，另一个小组负责测试此后的着陆过程。前一个小组没有检测触点开关数据位，那不是他们的职责；后一个小组总是在测试之前重置计算机，清除数据位。两个小组工作得都很好，但从未在一起进行过集成/系统测试，接口错误没有被检测出，从而导致了这一灾难性的事故。

4. Intel奔腾处理器芯片缺陷

在PC机的“计算器”中输入以下算式：

$$(4\ 195\ 835 / 3\ 145\ 727) \times 3\ 145\ 727 - 4\ 195\ 835$$

如果答案不为0，就说明该计算机使用的是带有浮点除法软件缺陷的老式Intel奔腾处理器。

1994年，美国弗吉利亚州Lynchburg学院的一位博士在用奔腾PC机解决一个除法问题时，发现了这个问题。他将发现的问题放在Internet上，引发了一场风暴，成千上万的人发现了同样的问题，以及其他得出错误结果的情形。万幸的是，这种情况很少出现，仅在精度要求很高的数学、科学和工程计算中才会出现。

这个事件引起人们关注的原因并不是这个软件缺陷，而是Intel公司解决问题的态度。

- Intel公司的测试工程师在芯片发布之前已经发现了这个问题，但管理层认为还没严重到一定要修正，甚至公开的程度。
- 当这个软件缺陷被发现时，Intel公司通过新闻发布和公开声明试图弱化问题的严重性。
- 当压力增大时，Intel承诺可以更换有问题的芯片，但要求用户必须证明自己受到缺陷的影响。

结果舆论哗然。Internet上充斥着愤怒的客户要求Intel公司解决问题的呼声，新闻报道将Intel公司描绘成不诚信者。最后，Intel公司为自己处理软件缺陷的行为道歉并拿出4亿多美元来支付更换芯片的费用。由此可见，一个小小软件缺陷造成的损失可能有多大。

5. Windows 2000安全漏洞

微软曾经承认，Windows 2000操作系统远程服务软件中存在安全漏洞，这些漏洞可能导致3种不同的安全隐患——拒绝服务、权限滥用和信息泄漏，并发布了相应的补丁软件进行修补。拒绝服务隐患可能导致DOS受到攻击，使得合法用户无法远程登录系统；而权限滥用和信息泄漏隐患则涉及系统权限管理，有可能使攻击者控制Windows 2000系统，从而在计算机上添加用户，删除或安装组件，破坏数据或执行其他操作。

据美国军方证实，一名联机攻击者利用微软网络软件中的一个缺陷控制了其国防部的一

个服务器接口。尽管美国陆军网络技术事业部称受到攻击的军事网站不属于军方，但他们强调陆军很认真地对待这个事件。这个缺陷也使微软的安全团队大吃一惊，因为没有一名安全研究人员发现这个问题，从而造成了恶劣的影响。

巨额的财产损失和生命的代价让人们开始重视软件质量。类似的例子，国内也可举出很多。大大小小的软件故障几乎每天甚至每时每刻都在发生，只不过有些问题不那么严重罢了。

随着信息技术的飞速发展，软件产品已应用到社会的各个领域，软件质量问题已成为人们关注的焦点。软件开发商为了占有市场，必须把软件质量作为企业的重要目标之一，以免在激烈的竞争中被淘汰。用户为了保证自己业务的顺利完成，当然希望选用优质的软件。质量欠佳的软件产品不仅会使开发商的维护费用和用户的使用成本大幅增加，还可能产生其他的责任风险，造成公司信誉下降。在一些关键领域的应用系统，如民航订票系统、银行结算系统、证券交易系统、自动飞行控制软件、军事防御系统和核电站安全控制系统中，对软件质量提出了更高的要求。使用质量欠佳的软件，还可能造成灾难性的后果。勿庸置疑，如何提高软件质量，如同如何提高软件生产率一样，已成为整个软件开发过程中必须始终关心和设法解决的问题。

1.2 软件测试的目的和意义

由于人的主观认识常常难以完全符合客观现实，与工程密切相关的各类人员之间的通信和配合也不可能完美无缺。因此，对于软件来讲，不论采用什么样的技术和方法，软件中都会存在故障。即使标准商业软件里也存在故障，只是严重程度不同而已。采用新的编程语言、先进的开发方式、完善的开发过程可以减少故障的引入，但是我们无法完全杜绝软件中的故障。这就需要用测试来发现软件故障，软件中的故障密度也需要通过测试来估计。

软件测试是对软件需求分析、设计规格说明和编码的终审，是软件质量保证的关键步骤。但对于什么是软件测试，一直未达成共识，根据侧重点不同，主要有三种描述：

定义1.1 1983年，IEEE（国际电子电气工程师协会）提出的软件工程标准术语中给软件测试下的定义是：

“使用人工或自动手段来运行或测定某个系统的过程，其目的在于检验它是否满足规定的需求或是弄清预期结果与实际结果之间的差别。”

该定义包含了两方面的含义：

1) 是否满足规定的需求。

2) 是否有差别。

如果有差别，说明设计或实现中存在故障，自然不满足规定的需求。因此，这一定义非常明确地提出了软件测试以检验软件是否满足需求为目标。

定义1.2 软件测试是根据软件开发各阶段的规格说明和程序的内部结构而精心设计一批测试用例，并利用这些测试用例去执行程序，以发现软件故障的过程。该定义强调寻找故障是测试的目的。

定义1.3 软件测试是一种软件质量保证活动，其动机是通过一些经济有效的方法，发现软件中存在的缺陷，从而保证软件质量。

上述三种观点实际上是从不同角度理解软件测试，但不论从哪种观点出发，都可以认为软件测试是在一个可控的环境中分析或执行程序的过程，其根本目的是以尽可能少的时间和人力发现并改正软件中潜在的各种故障及缺陷，提高软件的质量。

测试目的决定了测试方案的设计，如果我们的目的是要证明程序中没有隐藏的故障存在，

那就会不自觉地回避可能出现故障的地方，设计出一些不易暴露故障的测试方案，从而使程序的可靠性受到极大的影响。相反，如果测试的目标是要证明程序中有故障存在，那就会力求设计出最能暴露故障的测试方案。软件测试是一项花费昂贵的活动，测试者希望通过软件测试来提高软件的质量或可靠性。这就意味着要发现并改正程序中的错误。所以，进行测试时不应该为了显示程序是没有问题的，而应该从软件中含有故障这个假定出发去测试程序，从中发现尽可能多的软件故障。因此，“一个好的测试用例在于发现至今尚未被发现的故障”，“一个成功的测试是发现了至今未被发现的故障的测试”。

1.3 软件测试过程

软件测试是软件开发过程的一个重要环节，是在软件投入运行前，对软件需求分析、设计规格说明和编码实现的最终审定，贯穿于软件定义与开发的整个过程中。

软件项目一旦开始，软件测试也随之开始。从单元测试到最终的验收测试，其整个测试过程如图1-1所示。

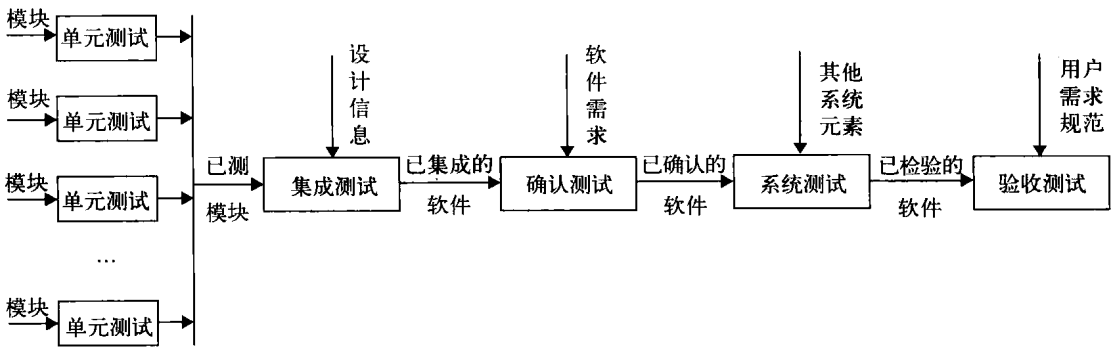

图1-1 软件测试过程

从测试过程可以看出，软件测试由一系列不同的测试阶段组成，即**单元测试**、**集成测试**、**确认测试**、**系统测试**和**验收测试**。软件开发是一个自顶向下逐步细化的过程。软件测试则是自底向上逐步集成的过程。低一级的测试为上一级的测试准备条件。

单元测试是测试执行的开始阶段，即首先对每一个程序模块进行单元测试，以确保每个模块能正常工作。单元测试大多采用白盒测试方法，尽可能发现并消除模块内部在逻辑和功能上的故障及缺陷。然后，把已测试过的模块组装起来，形成一个完整的软件后进行集成测试，以检测和排除与软件设计相关的程序结构问题。集成测试大多采用黑盒测试方法来设计测试用例。确认测试以规格说明书规定的需求为尺度，检验开发的软件能否满足所有的功能和性能要求。确认测试完成以后，给出的应该是合格的软件产品。但为了检验开发的软件是否能与系统的其他部分（如硬件、数据库及操作人员）协调工作，还需进行系统测试。最后进行验收测试，以解决开发的软件产品是否符合预期要求、用户是否接受等问题。

1.3.1 单元测试

单元测试是在软件开发过程中进行的最低级别的测试活动，其目的是要检测程序模块中有无故障存在。也就是说，一开始不是把程序作为一个整体来测试，而是首先集中注意力来测试程序中较小的结构块，以便发现并纠正模块内部的故障。

单元测试又称为模块测试。模块并没有严格的定义，按照一般的理解，模块应具有一些基本属性，如名字、明确规定的功能、内部使用的数据（或称局部数据）、与其他模块或外界的数据联系、实现其特定功能的算法，模块可被其上层模块调用，也可调用其下属模块进行协同工作等。

在传统的结构化编程语言中（比如C语言），单元测试的对象一般是函数或子过程。在像C++这样的面向对象语言中，单元测试的对象可以是类，也可以是类的成员函数。对Ada语言而言，单元测试可以在独立的过程和函数上进行，也可以在Ada包的级别上进行。单元测试的原则同样也可以扩展到第四代语言（4GL）中，这时单元被典型地定义为一个菜单或显示界面。

单元测试的对象是软件设计的最小单位，与程序设计和编程实现关系密切，因此，单元测试一般由测试人员和编程人员共同完成。测试人员可通过模块详细设计说明书和源程序代码清楚地了解模块的内部逻辑结构和I/O条件，常采用白盒测试方法设计测试用例。

在实际软件开发工作中，单元测试和代码编写所花费的精力大致相同。经验表明：单元测试可以发现很多的软件故障，并且修改它们的成本也很低。在软件开发的后期阶段，发现并修复故障将变得更加困难，将花费大量的时间和费用。因此，有效的单元测试是保证全局质量的一个重要部分。在经过测试单元后，系统集成过程将会大大地简化，开发人员可以将精力集中在单元之间的交互作用和全局的功能实现上，而不是陷入充满故障的单元之中不能自拔。

1.3.2 集成测试

集成测试，又称组装测试、子系统测试，是在单元测试基础之上将各个模块组装起来进行的测试，其主要目的是发现与接口有关的模块之间的问题。这是因为时常有这样的情况发生：每个模块都能单独工作，但这些模块组装起来之后却不能正常工作。程序在某些局部反映不出的问题，在全局上很可能就暴露出来，影响功能的正常发挥。可能的原因有：

- 模块相互调用时引入了新的问题。例如，数据可能丢失，一个模块对另一模块可能有不良影响等。
- 几个子功能组合起来不能实现主功能。
- 误差不断积累达到不可接受的程度。
- 全局数据结构出现错误等。

因此，在每个模块完成单元测试以后，需要按照设计的程序结构图，将它们组合起来，进行集成测试。集成测试是按设计要求把通过单元测试的各个模块组装在一起，检测与接口有关的各种故障。那么，如何组织集成测试呢？是独立地测试程序的每个模块，然后再把它们组合成一个整体进行测试好呢？还是先把下一个待测模块组合到已经测试过的模块上去，再进行测试，逐步完成集成好呢？前一种方法称为**非增量式集成测试法**。后一种方法称为**增量式集成测试法**。

非增量式测试法的集成过程是：先对每一个模块进行单元测试，我们可以同时测试或是逐个地测试各个模块，这主要由测试环境（如所用计算机是交互式的还是批处理式的）和参加测试的人数等情况来决定。然后，在此基础上按程序结构图将各模块连接起来，把连接后的程序当作一个整体进行测试。这种集成测试方法容易造成混乱。因为测试时可能发现很多错误，定位和纠正每个故障非常困难，并且在修复一个故障的同时又可能引入新的故障，新旧故障混杂，很难断定出错的原因和位置。

增量式集成测试不是孤立地测试每一个模块，而是将待测模块与已测模块集合连接起来进行测试。在这一过程中，不断地把待测模块连接到已测模块集（或其子集）上，对待测模块进行测试，直到最后一个模块测试完毕为止。

在软件集成阶段，测试的复杂程度远远超过单元测试的复杂程度。可以类比一下，假设要清洗一台已经完全装配好的食物加工机器，无论你喷了多少水和清洁剂，一些食物的小碎片还是会粘在机器的一些死角上，只有任其腐烂并等待以后再想办法。但如果这台机器是拆开的，这些死角也许就不存在或者更容易接触到，并且每一部分都可以毫不费力地进行清洗。

1.3.3 确认测试

确认测试是对照软件需求规格说明，对软件产品进行评估以确定其是否满足软件需求的过程。比如，编写出的程序是否符合软件需求规格说明的要求？程序输出的信息是否是用户所要求的信息？程序在整个系统的环境中是否能够正确稳定地运行？等等。该测试包含了对软件需求满足程度的评价。

集成测试完成以后，分散开发的模块已经按照设计要求组装成一个完整的软件系统，各模块之间存在的种种问题都已基本排除。为进一步验证软件的有效性，对它在功能、性能、接口以及限制条件等方面做出更切实的评价，就应进行确认测试。在开发的初期，软件需求规格说明中可能明确地规定了确认标准，但在测试阶段需要更详细、更具体地在测试规格说明中加以体现。除了考虑功能、性能以外，还需检验其他方面的要求。例如，可移植性、兼容性、可维护性、人机接口以及开发的文档资料是否符合要求等。

经过确认测试，应该为已开发的软件给出结论性的评价。这包括两种情况：

1) 经过检验，软件功能、性能及其他方面的要求都已满足需求规格说明的规定，该软件是一个合格的软件。

2) 经过检验，发现与软件需求规格说明有些偏离，于是得到一个缺陷清单，由开发部门和用户进行协商，找出解决的办法。

1.3.4 系统测试

软件只是计算机系统的一个重要组成部分，软件开发完成以后，还应与系统中其他部分配合起来，进行一系列系统集成和测试，以保证系统各组成部分能够协调地工作。这里所说的系统组成部分除软件外，还包括计算机硬件以及相关的外围设备、数据及采集和传输机构、计算机系统操作人员等。**系统测试**实际上是针对系统中各个组成部分进行的综合性检验，很接近日常测试实践。例如，在购买二手车时要进行系统测试，在订购在线网络时要进行系统测试等。系统测试的目标不是要找出软件故障，而是要证明系统的性能。比如，确定系统是否满足其性能需求；确定系统的峰值负载条件及在此条件下程序能否在要求的时间间隔内处理要求的负载；确定系统使用资源（存储器、磁盘空间等）是否会超界；确定安装过程中是否会导致不正确的方式；确定系统或程序出现故障之后能否满足恢复性需求；确定系统是否满足可靠性需求等。

系统测试很困难，需要很大的创造性。那么，系统测试应该由谁来进行？可以肯定，以下人员、机构不能进行系统测试：

- 系统开发人员不能进行系统测试。
- 系统开发组织不能负责系统测试。

之所以如此，第一个原因是，进行系统测试的人必须善于从用户的角度考虑问题。他最

好能彻底地了解用户的看法和环境，了解软件的使用。显然，最好的人选就是一个或多个用户。然而，一般的用户没有前面所说的各类测试的能力和专业知识，所以理想的系统测试小组应由如下一些人组成：几个职业的系统测试专家、1～2个用户代表，1～2个软件设计者或分析者等。第二个原因是系统测试没有清规戒律的约束、灵活性很强。而开发机构对自己程序的心理状态往往与这类测试活动不相适应。大部分开发软件机构最关心的是让系统测试能按时圆满地完成，并不真正想说明系统与其目标是否一致。一般认为，独立测试机构在测试过程中查错积极性高并且有解决问题的专业知识。因此，系统测试最好由独立的测试机构完成。关于系统测试，有很多种类型，我们将在第6章进一步讨论。

1.3.5 验收测试

验收测试的目的是向用户表明所开发的软件系统能够像用户所预期的那样工作，可以类比为建筑的使用者来对建筑进行的验收。首先，他认为这个建筑是满足规定的工程质量的，这是由建筑的质检人员来保证的。使用者关注的重点是住在这个建筑中的感受，包括建筑的外观是否美观、各个房间的大小是否合适、窗户的位置是否合适、是否能够满足家庭的需要等。这里，建筑的使用者执行的就是验收测试。验收测试是将最终产品与最终用户的当前需求进行比较的过程，是软件开发结束后软件产品向用户交付之前进行的最后一次质量检验活动，它解决开发的软件产品是否符合预期的各项要求，用户是否接受等问题。验收测试不只检验软件某方面的质量，还要进行全面的质量检验并决定软件是否合格。因此，验收测试是一项严格的、正规的测试活动，并且应该在生产环境中而不是开发环境中进行。

验收测试的主要任务包括：

- 明确规定验收测试通过的标准。
- 确定验收测试方法。
- 确定验收测试的组织和可利用的资源。
- 确定测试结果的分析方法。
- 制定验收测试计划并进行评审。
- 设计验收测试的测试用例。
- 审查验收测试的准备工作。
- 执行验收测试。
- 分析测试结果，决定是否通过验收。

验收测试关系到软件产品的命运，因此应对软件产品做出负责任的、符合实际情况的客观评价。制定验收测试计划是做好验收测试的关键一步。验收测试计划应为验收测试的设计、执行、监督、检查和分析提供全面而充分的说明，规定验收测试的责任者、管理方式、评审机构以及所用资源、进度安排、对测试数据的要求、所需的软件工具、人员培训以及其他的特殊要求等。总之，在进行验收测试时，应尽可能去掉一些人为的模拟条件，去掉一些开发者的主观因素，使得验收测试能够得到真实、客观的结论。

1.4 软件测试与软件开发的关系

软件开发过程是软件工程的重要内容，也是进行软件测试的基础。近年来，软件工程界普遍认为，对软件生命周期的每一阶段都应进行测试，以检查本阶段的工作成果是否接近预期的目标，尽可能早地发现并改正错误。因此，软件测试贯穿于软件开发的整个生命周期。