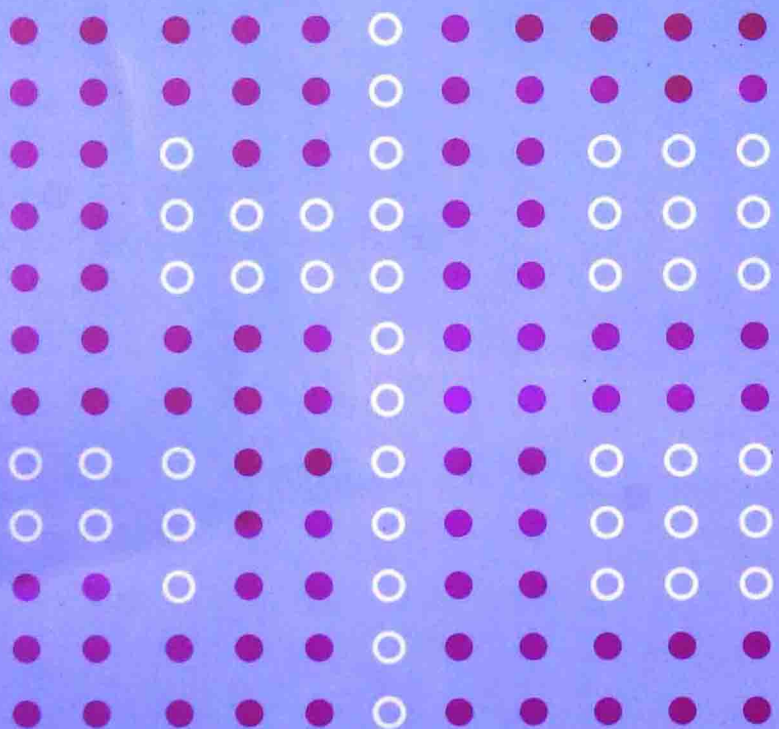
软件工程系列教材

曲朝阳 刘志颖 杨杰明 刘迪 编著

软件测试技术
(第2版)



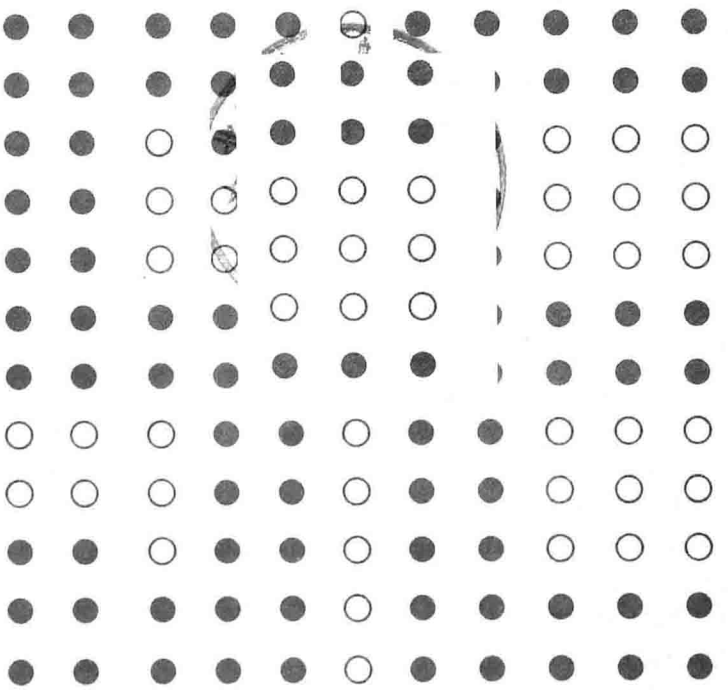
Ruan Jian Ce Shi Ji Shu

清华大学出版社

软件工程系列教材

软件测试技术
(第2版)

曲朝阳 刘志颖 杨杰明 刘迪 编著



清华大学出版社

内 容 简 介

本书详尽地阐述了软件测试领域中的一些基本理论和实用技术。首先从学生需要理解并掌握的软件测试基本概念和基本知识入手，使学生弄清楚为什么要进行软件测试，什么是软件测试？如何运用数学工具进行测试的描述和分析；在此基础上，结合经典案例讨论如何进行黑盒和白盒测试；然后依托实际案例深入讨论如何进行单元测试、集成测试和系统测试，以及具体的测试实施过程。最后，讨论了如何选择和使用各种自动化测试工具提高测试效率，以及如何进行软件缺陷的管理。

本书作为软件测试的实际应用参考书，除了力求突出基本知识和基本概念的表述外，更注重软件测试技术的运用，在介绍诸多知识点的过程当中结合直观形象的图表或实际案例进行深入浅出的分析，从而使读者可以更好地理解掌握软件测试理论知识，并迅速地运用到实际测试工作中去。

本书适合作为各层次高等院校计算机及相关专业的教学用书，也可作为软件测试人员的参考书。

图书在版编目(CIP)数据

软件测试技术/曲朝阳等编著. --2 版. --北京：清华大学出版社，2015
软件工程系列教材
ISBN 978-7-302-38253-9

Ⅰ. ①软…　Ⅱ. ①曲…　Ⅲ. ①软件—测试—高等学校—教材　Ⅳ. ①TP311.5

中国版本图书馆 CIP 数据核字(2014)第 235238 号

责任编辑：白立军　徐跃进
封面设计：傅瑞学
责任校对：李建庄
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
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课件下载：http://www.tup.com.cn，010-62795954
印 刷 者：北京富博印刷有限公司
装 订 者：北京市密云县京文制本装订厂
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：26.25　　**字　　数**：658 千字
版　　次：2006 年 8 月第 1 版　2015 年 2 月第 2 版　　**印　　次**：2015 年 2 月第1 次印刷
印　　数：1～2000
定　　价：49.00 元

产品编号：056489-01

PREFACE

软件工程系列教材

前言

随着信息技术的普及，各种各样的软件已经应用到很多领域，设计的复杂程度逐渐增加，开发周期不断缩短。而用户对软件要求却越来越高，不再仅仅关注软件产品功能的先进性，并且十分重视对产品质量的稳定性和可靠性的考察，这使得软件开发人员和软件测试人员面临着前所未有的挑战。因此，如何保证软件质量将成为软件工程领域深入研究的课题。

毋庸置疑，优化软件开发过程和提高软件测试人员的技术水平，是保证软件质量的最佳途径，这种观念正在被更多的软件行业人士理解、接受和实施。但软件测试在国内仍处于起步阶段，各种软件测试的方法、技术和标准都还在探索阶段。可以认为，当今中国的软件测试行业处于“春秋战国”时期，百家争鸣。一方面，这给行业的创新和发展提供了营养丰富的土壤；另一方面在测试行业一派“欣欣向荣”的气象背后，也隐藏着深深的危机。软件质量和测试观点“良莠不齐”、“泥石俱下”。

在这种情况下，很多高校为了培养更多软件行业急需的软件测试人才，都已开设了软件测试课程，为了适应当前教学的需要，编者在软件测试课程实践的基础上，结合教学和科研成果，以及当前软件测试技术的最新发展动态编写了本书。

本书作为软件测试的实际应用参考书，除了力求突出基本知识和基本概念的表述外，更加注重软件测试技术的运用，在介绍很多知识点的过程中都结合直观形象的图表或实际案例进行了深入浅出的分析，从而使读者可以更好地理解和掌握软件测试技术理论知识，并迅速地运用到实际测试工作中去。

本书参考教学时数为 32～40 学时。全书包括 9 章：第 1 章讨论了软件测试的发展历史、软件测试的定义和基本原则；第 2 章介绍了软件测试过程中需要掌握的离散数学和图论基础知识；第 3、4 章结合经典案例讨论了白盒和黑盒测试技术，第 5～7 章讨论

了单元测试、集成测试和系统测试相关的知识，并且以实际软件系统的测试为例讨论了具体的实施过程；第8章介绍了软件测试自动化，讨论了自动化测试的时机，自动化测试成本的衡量，自动化测试工具的选择和使用；第9章介绍了关于软件bug及其管理方面的知识，讨论了软件bug的分类、提交和管理。为了读者的方便，本书在出版社网站提供了测试案例使用的源代码、配置说明等相关资源。需要的读者可到清华大学出版社网站(www.tup.com.cn)下载。

本书在编写过程中，参阅了很多国内外同行的著作或文章，汲取了该领域最新的研究成果。在此，对这些成果的作者表示深深的感谢！

由于编者水平有限，书中难免存在一些错误和不妥之处，希望有关专家、同行和广大读者批评指正。

编著者

2015年1月

CONTENTS

软件工程系列教材

目 录

CHAPTER

第1章 概　述

【本章要点】

- 软件测试的发展历史；
- 软件测试技术的分类方法；
- 软件测试原则；
- 软件测试的定义；
- 软件测试同软件开发之间的关系；
- 软件测试与开发模型；
- 软件测试工作流程。

【本章目标】

- 了解软件测试的发展历程和行业现状；
- 掌握软件测试技术的分类；
- 理解软件测试的目的和软件测试原则，以及了解人们对软件测试行业的错误认识；
- 掌握软件测试中的基本定义、基本知识；
- 理解软件开发与软件测试的关系。

1.1 软件测试的发展历程及现状

1.1.1 软件测试的发展历程

随着计算机的诞生——在软件行业发展初期就已经开始软件测试，但这一阶段还没有系统意义上的软件测试，更多的是一种类似调试的测试。测试是没有计划和方法的，测试用例的设计和选取也都是根据测试人员的经验随机进行的，大多数测试的目的是为了证明系统可以正常运行。

20 世纪 50 年代后期到 20 世纪 60 年代，各种高级语言相继诞生，测试的重点也逐步转入使用高级语言编写的软件系统中，但程序的复杂性远远超过了以前。尽管如此，由于受到硬件的制约。在计算机系统中，软件仍然处于次要位置。软件正确性的把握仍然主

要依赖于编程人员的技术水平。因此,这一时期测试理论和方法的发展比较缓慢。

20 世纪 70 年代以后,随着计算机处理速度的大幅度提高,存储器容量的快速增加,软件在整个计算机系统中的地位变得越来越重要。随着软件开发技术的成熟和完善,软件的规模也越来越大,复杂度也大大增加。因此,软件的可靠性面临着前所未有的危机,给软件测试工作带来了更大的挑战,很多测试理论和测试方法应运而生,逐渐形成了一套完整的体系,培养和造就了一批批出色的测试人才。

如今在软件产业化发展的大趋势下,人们对软件质量、成本和进度的要求也越来越高,质量的控制已经不仅仅是传统意义上的软件测试。传统的软件测试大多是基于代码运行的,并且常常是在软件开发的后期才开始进行,但大量研究结果表明设计活动引入的错误占软件开发过程中出现的所有错误数量的 50%～65%。因此,越来越多的声音呼吁,软件产业化要求有一个规范的软件开发过程。而在整个软件开发过程中,测试已经不再只是基于程序代码进行的活动,而是一个基于整个软件生命周期的质量控制活动,贯穿着软件开发的各个阶段。

1.1.2 我国软件测试的现状

在我国,软件测试可能还算不上一个真正的产业,许多软件开发企业对软件测试认识淡薄,软件测试人员与软件开发人员往往比例失调,而在发达国家和地区软件测试已经成了一个产业,微软的开发工程师与测试工程师的比例是 1∶2,国内一般公司是 6∶1。很多人认为导致这种现状产生的原因与接受的传统教育和开发习惯有相当大的关系。软件行业相对于其他一些行业来说是相当年轻的,开发工作包含了需求管理、分析、设计、测试和部署等工作,由于软件业的历史年轻,而且一般人认为,开发周期前面的工作没有完善之前,比较难于考虑到稍后的阶段。因此,可以看到软件业大部分的精力都投入在需求管理、分析、设计三个阶段的开发,造成了这些方面软件和方法论的快速发展,而忽视了测试工作。

总之,与一些发达国家相比,国内测试工作还存在一定的差距。主要体现在测试意识以及测试理论的研究、大型测试工具软件的开发以及从业人员数量等方面。其实,这与中国整体软件的发展水平是一致的,因为我国整体的软件产业水平和软件发达国家水平相比有较大的差距,而作为软件产业重要一环的软件测试,必然有不小的差距。但是,我们在软件测试实现方面并不比国外差,国际上优秀的测试工具,我们基本都有,这些工具所体现的思想我们也有深刻的理解,很多大型系统在国内都能得到了很好的测试。

1.2 什么是软件测试

正如食品生产厂家在把产品销售给商家之前要进行合格检验一样,软件企业在把软件提交给客户之前也需要进行严格的测试。如果把所开发出来的软件看作一个企业生产出的产品,那么软件测试就相当于该企业的质量检测部分。简单地说,在编写完一段代码之后,检查其是否如自己所预期的那样运行,这个活动就可以看作是一种软件测试工作。

1.2.1 软件测试的定义

从前面的章节中，已经了解到软件测试的研究可以追溯到 20 世纪 60 年代，至今已有 50 多年的发展历史，但对于什么是软件测试，还一直未能达成共识。根据侧重点的不同，主要有以下三种观点：

- IEEE 在 1983 年将软件测试定义为“使用人工或自动手段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”，该定义明确地提出了软件测试以检验是否满足需求为目标。
- Myers 则认为软件测试“是为了发现错误而执行程序的过程”，明确提出了“寻找错误”是测试目的。
- 从软件质量保证的角度看，软件测试是一种重要的软件质量保证活动，其动机是通过一些经济、高效的方法，捕捉软件中的错误，从而达到保证软件内在质量的目的。

上述三种观点实际上是从不同的角度理解测试，是将测试置于不同的环境下得出的结论。事实上，在公开出版的刊物中，有多种不同的关于软件测试的定义，根据这些定义可以认为软件测试是一个在可控的环境中执行软件的过程，目的就是为了验证软件是否按照预期运行。测试过程中的活动既包括“分析”软件，也包括“运行”软件。常常把与分析软件开发中的各种产品相关的测试活动称为静态测试(static testing)。静态测试包括代码审查、走查和桌面检查。相比之下，把与运行软件有关的测试活动叫做动态测试(dynamic testing)。因此，不能简单地认为软件测试就是程序测试，只有在程序编码结束后才能够进行的工作；而是一项贯穿于整个软件开发过程的工作。测试对象既包括源程序，也包括需求规格说明、概要设计说明、详细设计说明。因此，也有人认为软件测试(software testing)就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的最终复审，是软件质量保证的关键步骤。测试包括寻找缺陷，但不包括跟踪漏洞及其修复。测试的重要性在于，它必须保证所开发的软件达到设计时的需求，免除由于软件自身的“缺陷”带来的“漏洞”，最大限度地降低软件开发的成本。

软件测试有两个基本职责：即验证和确认。Schulmeyer 和 Mackenziee(2000)对验证和确认所做的定义是：

- 验证(verification)，保证开发过程中某一具体阶段的产品与该阶段和前一阶段的需求一致。
- 确认(validation)，保证最终得到的产品满足系统需求。

有时，初学者常常会混淆软件测试和调试。其实二者是不同的，表现在以下几方面：

(1) 调试是一个分析和定位软件 bug 的过程。可以认为它是一种支持测试，但不能完全代替测试的活动。

(2) 调试的目的是为了使软件能够正确运行，而测试的目的是为了发现软件中存在的错误。

(3) 调试的对象主要是源代码，而测试的对象则是软件开发过程中各个阶段所产生的所有产品。

1.2.2 软件测试生命周期

图 1-1 给出了测试生命周期(software testing life cycle)的模型。把测试的生命周期分为几个阶段。前三个阶段就是引入程序错误阶段,也就是开发过程中的需求规格说明、设计、编码阶段,此时极易引入错误或导致开发过程中其他阶段产生错误。然后就是通过测试发现错误的阶段,这需要通过使用一些适当测试技术和方法来共同完成。后三个阶段就是清除程序错误的阶段,其主要任务就是进行缺陷分类、缺陷隔离和解决缺陷。其中在修复旧缺陷的时候很可能引进新的错误,导致原来能够正确执行的程序出现新的缺陷。

在软件测试生命周期的每个阶段都要完成一些确定的任务,在执行每个阶段的任务时,可以采用行之有效的结构分析设计技术和适当的辅助工具;在结束每个阶段的任务时都进行严格的技术审查和管理复审。最后提交最终软件配置的一个或几个成分(文档或程序)。

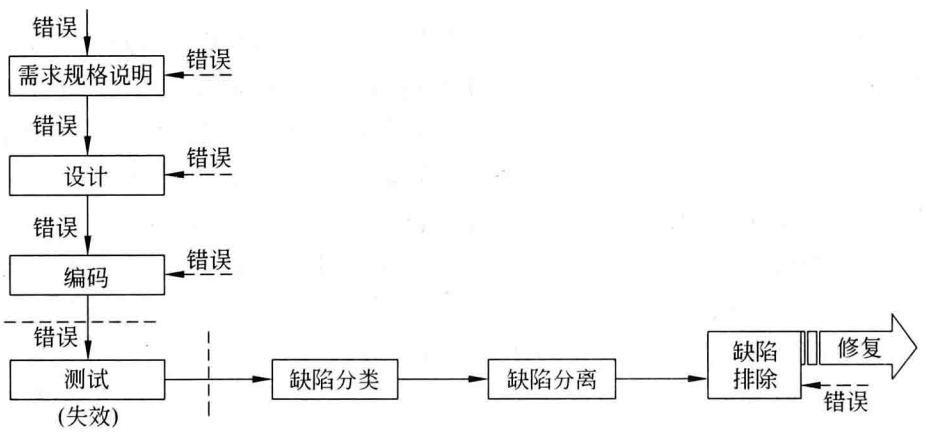


图 1-1 测试生命周期

1.2.3 软件开发与测试模型

通常情况下,测试过程包括确定要测试什么(测试范围和条件)以及产品如何被测试(制作测试用例),建立测试环境,执行测试,最后再评估测试结果,检查是否达到已完成测试的标准,并报告进展情况等活动。由此可以看出,软件测试不仅仅是执行测试,而是一个包含很多复杂活动的过程,并且这些过程应该贯穿于整个软件开发过程。如果把测试设计放在最后阶段,就会错过发现构架设计和业务逻辑设计中存在的严重问题的时机,到时候修复这些缺陷将很不方便,因为缺陷已经扩散到系统中,所以这样的错误将很难寻找和修复,并且代价更高。读者可能会问,那么如何协调软件测试与开发活动之间的关系?在软件开发的过程中,应该什么时候进行测试呢?如何更好地把软件开发和测试活动集成到一起呢?其实这也是软件测试工作人员必须要考虑的几个问题,因为只有这样才能提高软件测试工作的效率,提高软件产品的质量,最大限度地降低软件开发与测试的成本,减少重复劳动。下面将介绍几种典型的软件开发与测试模型,这些模型在不同程度上回答了前面所提出的问题。

1. **软件开发与测试 V 模型**

V 模型如图 1-2 所示。

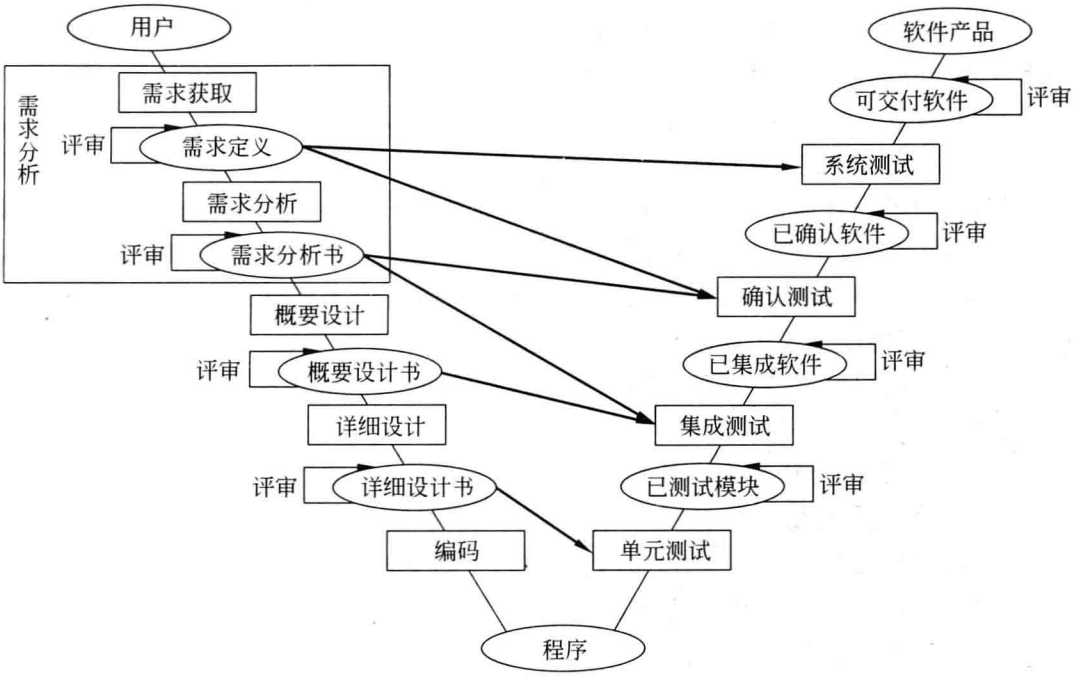


图 1-2 V 模型示意图

事实上，在传统开发过程中，仅仅把测试过程作为在需求分析、概要设计、详细设计及编码之后的一个阶段。如在瀑布模型中，认为测试只是在很多重要开发活动完成后的收尾工作，而不是主要的过程。V 模型对此进行了改进，不再把测试看作是一个事后弥补行为，而是一个同开发过程同样重要的过程。该模型最早由已故的 Paul Rook 在 20 世纪 80 年代后期提出，在英国国家计算中心文献中发布，在欧洲尤其是英国被普遍接受，并把它当作瀑布模型的替代品。

如图 1-2 所示的 V 模型描述了一些不同的测试级别，并说明了这些级别所对应的生命周期中不同的阶段。其中，左边下降的部分是开发过程各阶段，与此相对应的是右边上升的部分是测试过程的各个阶段。读者要注意，在不同的组织中对测试阶段的命名可能有所不同。在模型图中的开发阶段一侧，先从定义业务需求开始，然后要把这些需求不断地转换到概要设计和详细设计中，最后开发程序代码。在测试执行阶段一侧，执行先从单元测试开始，然后是集成测试、确认测试和系统测试。

V 模型的价值主要在于它非常明确地标明了测试过程中存在的不同级别，并且清楚地描述了这些测试阶段和开发过程期间的对应关系：

- 单元测试的主要目的是根据详细设计说明书来验证和确认每个单元模块是否符合预期的要求，发现编码过程中可能存在的各种错误。
- 集成测试主要目的是根据概要设计来验证和确认各个模块是否已正确集成到一起，主要是检查各单元与其他模块之间的接口上可能存在的错误。

- 确认测试主要目的是根据需求分析来验证和确认软件是否符合用户的预期要求。
- 系统测试主要目的是根据需求定义,验证和确认系统作为一个整体是否能够正常有效地运行,例如,判断系统是否达到了用户预期的性能。

在不同的开发阶段,所引入的缺陷和错误类型不同,因此需要使用不同的测试技术和方法来发现这些缺陷。在后面的章节中将对此作具体的介绍。

根据 V 模型的要求,一旦有文档提供,就要及时确定测试条件,编写测试用例,这些工作对每个级别的测试都有意义;当需求提交之后,就需要针对这些需求确定更高级别的测试用例;当概要设计编写完成后,就需要确定测试条件来查找该阶段的设计缺陷。因此如果能尽早提交测试文档,就可以有更多的检查和审阅时间,使测试者可以在项目中能尽早发现规格说明书中存在的问题。这些都说明测试的目的不仅仅是为了评定软件,更重要的是能够尽可能早地找出缺陷所在,从而达到改进项目质量的目的。参与前期工作的测试者可以预先估计可能存在的问题和测试执行的难度,大大减少总体测试时间,提高项目进度。

V 模型常被错误地认为要求开发和测试保持一种线性的前后关系,需要有严格的指令表示上一阶段完全结束,才可正式开始下一个阶段,这样就无法支持迭代,自发性以及变更调整,情况其实并不是这样的。各种模型只是简单地提醒我们,有必要定义一些必须要做什么(需求)——What,然后描述如何做(设计)——How,最后花很多力气来实现(编码)——Do。V 模型所做的是强调每一个开发级别都有一个与之相关联的测试级别,并且建议测试应该在各级别之前进行设计。各模型并没有规定工作量大小,有经验的开发人员通常能够将项目分解为可操作的小阶段,例如在迭代式开发中整个项目被分解为很多小片段,并且忽略各片段的实际大小。此时,模型的 what-how-do 顺序对于按时交付就具有重要意义,而且对于保证每一个阶段目标的实现也非常重要。

V 模型适用于所有类型的开发过程,但并不一定适用于开发和测试过程的所有方面。不管是 GUI 还是批处理、大型机还是 Web、Java 还是 Cobol,都需要单元测试、集成测试、系统测试和验收测试。但是,V 模型本身并不会告诉你如何定义单元测试或集成测试的内容、如何才能使测试工作顺利进行、如何进行具体的测试设计,以及该输入什么样的数据,输出的结果是什么样才正确。

还有一些测试者认为:“是否使用 V 模型要根据项目本身来定。有些项目需要应用 V 模型,而有些项目不需要。那么,可以只在需要 V 模型的项目中采用。”在实际工作中,这样的做法是不对的,应该尽可能地去应用模型中对项目有实用价值的方面,但不要为了使用模型而使用模型,否则便失去了使用模型的实际意义。

正因为 V 模型还存在一些不够完善的地方,随着软件测试技术的不断发展,由 V 模型演化出很多种软件开发与测试模型,下面以图例的形式把另外几种比较典型的模型介绍给大家,这几种测试模型都不同程度地改进了 V 模型的不足之处。

2. 软件开发与测试 W 模型

由于原始问题的复杂性、软件的复杂性和抽象性、软件开发各个阶段工作的多样性以及各种层次人员之间工作的配合关系等因素,使得开发的每一个环节都可能产生错误。如果坚持各个阶段的技术评审,就能够尽早发现和预防错误。图 1-3 为软件开发与测试