软件可靠性测试及其测试环境

Software Reliability Test and It's Testing Environment

【摘要】阐述了软件测试,特别是软件可靠性测试的概念,论述了软件运行剖面和测试用例的生成方法。文中给出了一个已经开发成功的嵌入式软件仿真测试环境,包括该测试环境的体系结构与基本功能。

关键词:软件测试,软件可靠性测试,测试环境

Abstract: The concept of software testing, especially the software reliability testing and the method of generating operational profile and test cases are presented. A developed successfully embedded-software simulative test environment with it's hierarchy and fundamental functions is given.

Key words: software test, software reliability test, testing environment

近二、三十年来,随着计算机在军用与民用产品上的应用日益增多,软件缺陷所引发的产品故障,甚至灾难性事故也越来越严重。据美国国家宇航局 NASA 的统计:在 80 年代初,软件引起的故障与硬件引起的故障,其比率约为 1.1∶1.0,到了 80 年代末,这一比率已达到 2.5∶1.0。在我国,这一比率至少已达到 3∶1。随着 21 世纪的来临,信息技术的迅猛发展,计算机已深入到军用、民用的各个领域,甚至居民的日常生活之中,因此,软件故障将日益成为高新技术产品发展的瓶颈。

为了提高软件的质量和可靠性,必须在软件开发的生命周期中,抓紧软件的设计、测试与管理这几个关键环节。鉴于篇幅所限,本文将只阐述软件测试,尤其是软件可靠性测试及其测试的环境。

1 软件可靠性测试的概念

软件测试的目的是为了发现软件中存在的缺陷并予以排除,以确保其功能能满足需求。

软件可靠性测试是为了达到或验证用户对软件的可靠性要求而对软件进行的测试;通过测试发现并纠正软件中的缺陷,提高其可靠性水平,并验证它是否达到了用户的可靠性要求。软件可靠性测试能有效地暴露在实际使用过程中影响可靠性要求的软件缺陷,最先暴露的一般是高发生概率的缺陷,然后是较低发生概率的缺陷。

软件可靠性测试的一般流程如图 1 所示。

软件可靠性测试中最关键的三个环节是:

①根据用户实际使用软件的方式,构造软件运行剖面,生成测试用例;

②开发软件可靠性测试的环境,使被测软件能在该环境中得以测试;

③对测试结果进行分析,并作出软件可靠性的预计。

本文将对前两个环节进行探讨。

2 软件运行剖面构造与测试用例生成方法

2.1 软件运行剖面的构造

软件可靠性测试的主要特点是按照用户实际使用软件的方式来测试软件。软件的运行剖面(operational profile)是定量描述用户实际使用软件方式的有力工

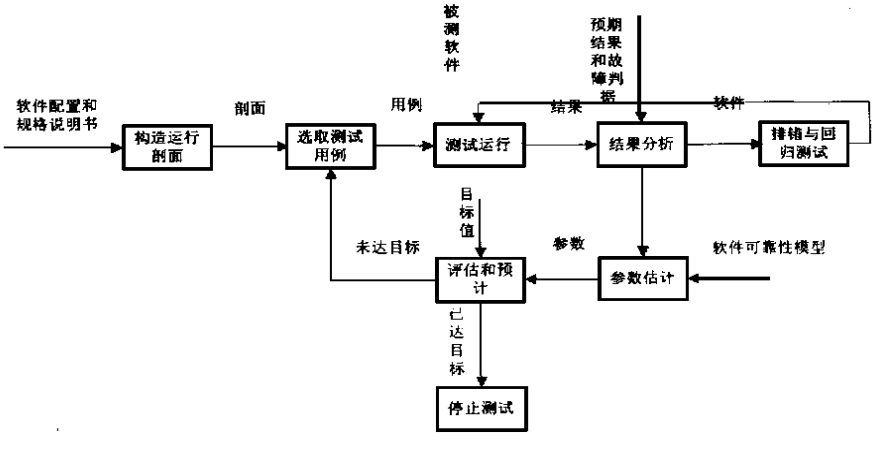

图 1 软件可靠性测试流程图

具。构造软件的运行剖面是实现软件可靠性测试的关键步骤,也是软件可靠性测试最主要的特征。

在构造运行剖面的过程中,需要了解用户是如何使用该软件的。要充分了解用户使用软件的各种模式和各种功能,完成这些功能相应的输入变量。同时,还要了解用户在使用软件时各系统模式和功能发生的概率。这些信息大都来自软件开发的文档、规格说明书和接口文件等资料。这需要用户与测试人员不断地交换信息。系统模式及功能划分得越完整,概率越准确,构造出的运行剖面越能说明软件的实际使用情况。

构造软件的运行剖面的方法是按照一种层次结构,自顶向下地把用户使用软件的输入空间划分为系统模式剖面,把系统模式剖面划分为功能剖面,最后把功能剖面划分为运行剖面。

图 2 展示出系统模式剖面、功能剖面和运行剖面的层次关系。

2.2 软件测试用例生成方法

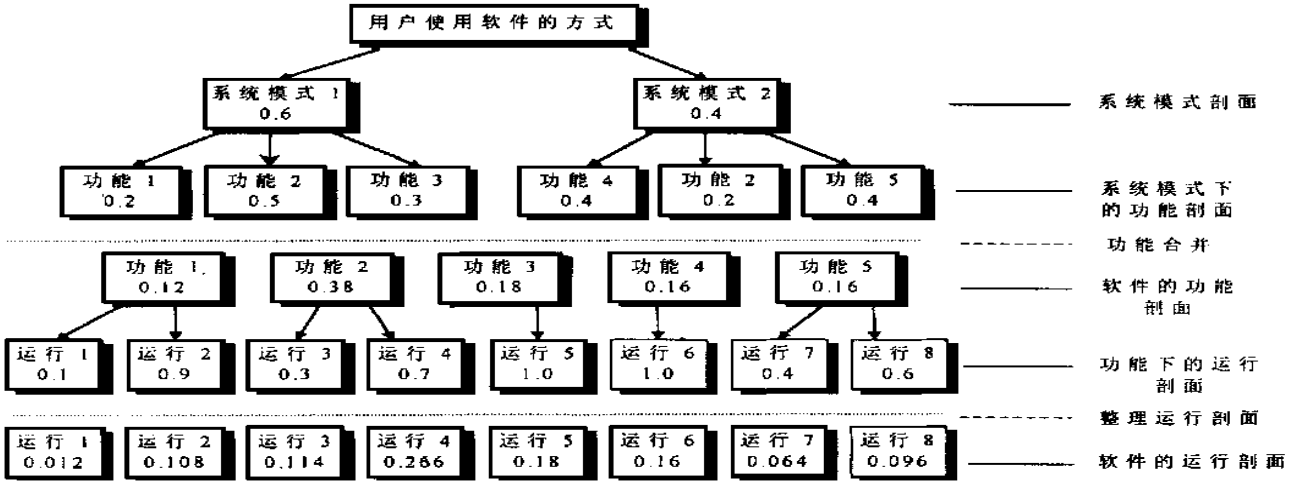

图 2 系统模式剖面、功能剖面和运行剖面的关系

测试用例是根据运行剖面随机生成的。在运行剖面中规定了每个输入变量的取值区间,并且认为变量在取值区间内均匀分布或分段均匀分布(由于很难确定变量的具体分布,这里假设为均匀分布)。软件可靠性测试是一种随机测试,测试用例的选取方式是随机选取。因此,根据随机测试的原则,在运行剖面给定的输入变量的取值区间内任意抽取一个变量的实际取值。将各个变量按顺序组合起来便生成了测试用例。用于软件可靠性测试的测试用例可以定义为:根据运行剖面生成的、完成对某一功能进行测试的、按顺序输入到被测软件的一系列输入变量的有序组合。

根据运行剖面生成测试用例的过程为:运行剖面由一系列变量的取值区间和该运行发生的概率组成。

首先,要随机抽取一个运行来实现对某一功能的一次测试。抽取运行的过程如下:

①将运行剖面$\{OP_i \mid OP_i = < O_i , P_i > , i = 1,2, \cdots, N\}$中所有运行发生的概率 P_i 求前 j 项和,形成一个数列$\{S_j\}$,$S_j = \sum_{i=1}^{j} P_i$,其中,$j = 1,2, \cdots, N$;N 为软件运行剖面中运行总数,规定 $S_0 = 0$,并有 $S_1 = P_1$,$S_n = 1.0$,$S_j - S_{j-1} = P_j$。这里运行相互独立。

②任给一个随机数 $\eta \in (0, 1.0)$,观察 η 落在哪个区间,若 η 满足 $S_{j-1} < \eta \leq S_j$,则该随机数 η 与 P_j 这个概率值对应,那么这次随机抽到的运行为 O_j。

其次,要进行第二次抽样来确定运行中每个取值区间将取到的实体(即具体取值)。实体的确定将按照输入变量的属性分两种情况进行:

①对于连续型输入变量,运行剖面给出的是该变量的取值区间的上下限$[R_{ij}.down, R_{ij}.up]$。抽样时将根据输入变量的数据类型,在区间$[R_{ij}.down, R_{ij}.up]$内任意抽取一个满足输入变量数据类型的具体值,作为该输入变量的实体。

②对于离散型输入变量,运行剖面给出的是一组离散点 R_{ij},$j = 1,2, \cdots, m_i$;m_i 为离散点的个数。抽样时将在$[1, m_i]$内任意抽取一个整数 j,以确定选哪一个离散点作为该输入变量的实体,并将该实体转化为该输入变量的数据类型。

通过对运行和各个实体两个步骤的抽样,完成了一个测试用例的生成。不断重复上述步骤,直到生成所需数量的测试用例为止。

对一个已通过验收鉴定并投入使用的软件,据其实际使用的情况生成了相应的运行剖面与测试用例,然后对该软件进行可靠性测试。用生成的 400 个测试用例,共测试出 60 个软件故障。对这些故障进行分析,剔除了相同故障后,剩下 9 个故障。测试结果可参见表 1。

表1 软件可靠性测试记录 测试用例号:1～400

测试用例序号	测试日期	开始测试时间	失效发生时刻	测试运行时间/s	累计运行时间/s	失效序号	失效现象	失效等级
15	1996-11-09	15:18:13	15:19:12	59.8	897.8	1	解释图形少一个门,用户程序出错	1*
27	1996-11-10	13:02:26	13:04:11	105.3	1711.1	2	拖动滚动条画图时死机	1
44	1996-11-10	15:37:58	15:40:38	160.3	2708.2	3	非最大化画图时死机	1
78	1996-11-11	14:20:10	14:22:14	124.0	4471.8	4	输入故障树信息存不住	1
108	1996-11-12	10:50:29	10:51:31	61.3	6444.8	5	诊断解释说明字符串无结尾	3
160	1996-11-13	10:16:15	10:16:47	32.3	9134.9	6	程序执行非法操作,未死机	3
236	1996-11-14	15:23:07	15:24:09	62.5	12951.9	7	开始画图时死机	1
324	1996-11-17	22:02:05	22:03:11	66.2	17820.9	8	画图位置偏,移滚动条死机	1
386	1996-11-18	14:48:02	14:48:57	54.3	21116.0	9	禁门条件错解释不画图	1

*注:失效等级是失效造成的危害程度的级别。一级是最为严重的后果。

3 软件可靠性仿真测试环境

一般的软件测试环境较为简单,本文将着重阐述嵌入式软件的可靠性仿真测试环境。所谓嵌入式软件是指嵌入式计算机系统用的软件,绝大多数军用和民用产品中使用的都是这类软件。

对于嵌入式软件,在系统集成完成到投入实际的使用两个阶段之间,缺乏一种有效的手段对其进行测试,特别是进行可靠性测试。在进行系统集成时进行的测试由于无法加入其交联系统,从而许多诸如时序错误、接口错误很难暴露出来,而一旦系统投入到实际环境中进行测试,则引发两个问题:一是由于软件测试,特别是可靠性测试需要施加大规模的测试用例进行长时间的测试,许多机械设备的寿命如陀螺仪等不允许进行如此长的试验;二是如果嵌入式软件一旦在实际环境下出错,有可能会影响其他交联系统,甚至对其他系统及整个系统造成无法弥补的损失,有时也可能会危及人的安全。因此构建一种仿真测试环境就成为迫切的需要。

嵌入式软件仿真测试环境是一个自动的、实时的、非侵入性的(non-intrusively)的闭环测试环境。它能够逼真地模拟被测软件运行所需的真实物理环境,并且能够组织被测软件的输入,来驱动被测软件运行,同时接收被测软件的输出结果,从而完成对嵌入式软件的测试。

3.1 测试环境的体系结构

测试环境的体系结构定义系统的组成和各个节点的定义以及它们的物理连接和数据通信协议。它同时决定了其功能是如何组织和整个测试环境的载荷是如何分布的。以下对最基本的结构进行说明。

仿真测试环境的基本结构如图3所示。

(1) 主机(Host)

Host通常是一台UNIX工作站或Windows NT PC机。它的主要任务是:用户命令接口;系统配置;测试用例及测试方案生成;测试脚本编写;测试过程监控;

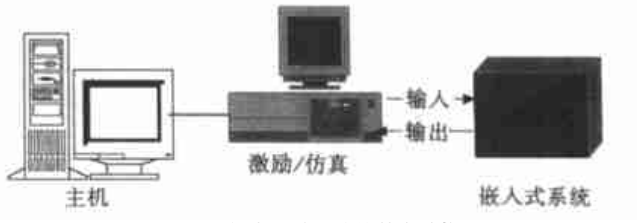

图3 仿真测试环境的基本结构

测试回放;测试结果分析和处理;可靠性评估;测试文档辅助生成;数据库管理。有时,数据库采用专门的数据库服务器。

通常,Host在测试开始后,同激励/仿真(S&S)的通信仅限于用户的命令和一些监控信息。如果实时约束许可,并且在可以使用大缓冲技术的条件下,或者采用诸如FastLink和ScramNet等网络延迟确定性较好的网络时,可以考虑在Host和S&S之间进行实时数据的传输。

(2) 激励/仿真(Stimulator/Simulator,S&S)

S&S一般采用具备实时处理能力的工作站或微机,它们可以是运行实时操作系统的工作站和微机,也可以是本文后面提及的对普通操作系统进行实时扩展的微机。S&S还包括一系列的I/O设备(如MIL-STD-1553、ARINC429、ARINC629、RS-232/422、A/D、D/A等)和它们的驱动程序。

S&S的任务主要包括:解释测试脚本,对数据进行仿真处理;生成激励信号,驱动被测软件运行;接收测试数据,进行实时比较;实时显示。

(3) Host和S&S的通信

Host和S&S通信通常采用以下方式进行通信:TCP/IP;Shared Memory;FastLink;ScramNet。

通常在使用TCP/IP的情况下,由于其固有的随机性,使得网络延迟确定性较差,因此在测试开始后,Host同激励/仿真(S&S)的通信仅限于用户的命令和一些监控信息。如果实时约束许可,并且可以使用大缓冲技术的条件下,或者当采用诸如FastLink和ScramNet等网络延迟确定性较好的网络时,可以考虑在Host和S&S之间进行实时数据的传输。

3.2 测试环境的基本功能

为了完成软件可靠性测试,一个完备的仿真测试环境应具备以下的基本功能:初始化;测试准备;测试;测试结果分析处理;测试文档管理;日常维护管理。

(1) 初始化是指测试环境由启动到可以进行一次测试之前要完成的一系列工作,这些工作包括:自检、系统配置、确立测试方案等。

(2) 测试准备功能起一个过渡性的作用,当系统配置和测试方案确定后,系统并不能马上进行测试,必须有一个准备阶段,测试准备并不是一个单一功能,而是由以下功能构成的:

①数据准备:它主要指测试时需要的数据从磁盘填充到数据缓冲区。

②预运行:它的主要作用是使得系统能进入稳定的运行。因为诸如定时器启动、I/O 服务启动(这通常表现为多线程的创建)需要这样的一个过程。

(3) 测试

测试自然是整个仿真测试环境的重点,它在一个实时调度器的调度下进行以下工作:

①测试数据仿真:主要是解释测试脚本(测试用例),判断它们分别应由哪一个仿真模型处理和操作。每个仿真模型代表被测系统的一个或几个交联系统。仿真模型产生针对一个或多个 I/O 的命令和数据,交给实时调度程序处理。

②产生激励信号:实时调度程序依据逻辑关系和定时关系,通过一组 I/O 服务程序生成被测软件的物理输入,激励被测系统运行。一般一个 I/O 服务程序负责一个接口硬件设备(如 MBI、并口、ADC、DAC 等)。当然不排除通过一个 I/O 服务程序控制多个硬件设备,这样做的收益是减少了系统的并发要求,但是会降低系统的通用性。

③测试结果接收:被测软件的输出结果也是通过 I/O 服务程序来执行的。需要指出的是,有的 I/O 服务程序处理的硬件设备是全双工的。被测软件的激励和输出可能通过一个或多个既产生激励又负责接收数据的 I/O 服务程序完成或部分完成。测试结果被放入数据缓冲区,其理由在上面的数据准备中已经提及。

④测试过程监控:它的内容是被测系统在测试中,仿真测试环境各个部件的状态、测试的进展情况、各类缓冲区的状态等。

⑤测试结果处理:分为实时处理和测试后处理,这里所指的是实时处理,它包括数据比较、数据过滤和记录,以及实时显示。

(4) 测试结果分析处理

测试仅仅是得到了一组原始的数据,虽然在测试中对测试结果数据作了简单的处理,但这远远不够。在测试结束后,仿真测试环境还应对测试结果数据作更进一步的分析处理,这包括:

①数据回放:用于测试结束后,回放显示测试数据和测试结果数据。数据回放应包括确定显示内容、显示方式、回显时标的缩放比例。显示方式有仪表显示、文本显示、图表显示、三维实体可视化显示等。

②测试结果数据比较、分析。

(5) 测试文档管理

负责对测试的各类文档进行管理,生成测试报告等工作。

(6) 系统维护管理

系统日常维护的主要工作有:记录平台测试的配置和测试方案,生成带时间标签的配置文件;工作日志;系统故障记录;系统故障修复;系统软件完好性检查。

软件可靠性仿真测试的工作流程参见图 4、图 5 和图 6。

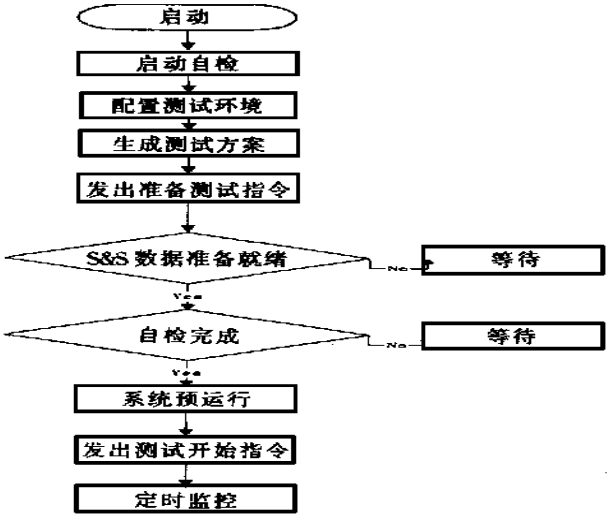

图 4 Host 工作流程(测试前)

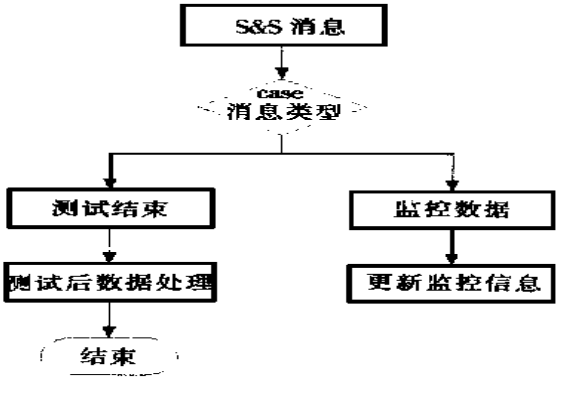

图 5 Host 工作流程(测试和测试后)

(下转第 16 页)

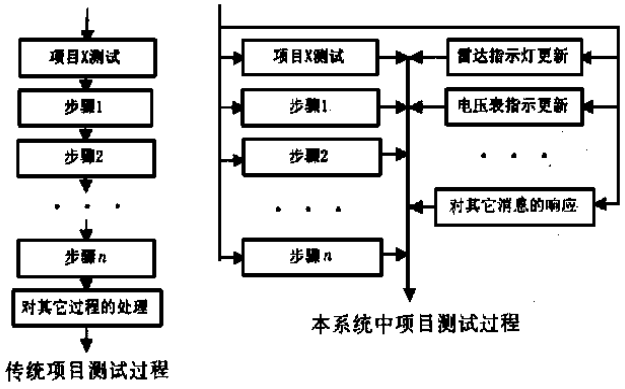

图 3　该软件系统与以往测试系统项目测试过程的区别

焦(focus)一个事件的办法来缩短对一个子步骤的处理与响应周期(0.01 s),这样既兼顾了时间测量精度,也使 CPU 在其他情况下,有足够的时间响应 Windows 及用户的其他消息。

由于该软件系统采用的是多任务机制,系统在运行"前台"一些子过程的同时,也运行了大量的"后台"子过程。在某些情况下会大量占用计算机的运行时间,阻碍其对其他消息的响应,降低系统的测量精度和灵活性。因此,当"后台"子过程与当前面板及当前测试过程无关时,就应及时关闭。实践证明,这是提高整个系统软件灵活性和高效的有效方法。

以往的自动测试系统所提供的用户对自动测试过程的干预功能很少,一个完整的测试流程所需时间长达二十几分钟,实际使用时很不方便。根据部队对雷达自动测试的实际需求,该系统的测试软件彻底改变了过去"全过程全项目自动测试"的软件设计方法,用户在自动测试前,不但可以选择自动测试项目的数目和测试流程,定义参数超差的处理办法,还可以暂停和终止测试过程。因而,提高了系统自动测试过程的灵活性,方便了用户对意外事件的处理。

3　结束语

该测控软件是"末制导雷达通用检测设备"中的一个重要组成部分,其开发工作量已经大大超过系统其他部分的工作量。在其中一种型号的末制导雷达测控程序中,虚拟面板总数超过 40 个,控件总数超过 1 000 个。该软件工程中包含工程文件 64 个,与软件系统运行有关的其他辅助文件 18 个,源文件大小超过 1MB。

通过对该软件的开发,我们认识到,虚拟仪器技术在武器装备测控领域有广阔的应用前景。采用虚拟仪器的软件设计技术,可以提高装备测控软件的灵活性,节约系统的硬件成本,缩短系统开发周期,从而大大提高武器装备的军事经济效益。

参考文献

1　张公学等. 一种可扩展的末制导雷达通用检测系统. 宇航计测技术, 1999, (1)
2　LabWindows/CVI User Manual. National Instruments Corporation, 1996
3　庄梓新等. 测量与控制核心系统手册. 北京:航空工业出版社, 1989

(收稿日期:1999 - 05) □

(上接第 12 页)

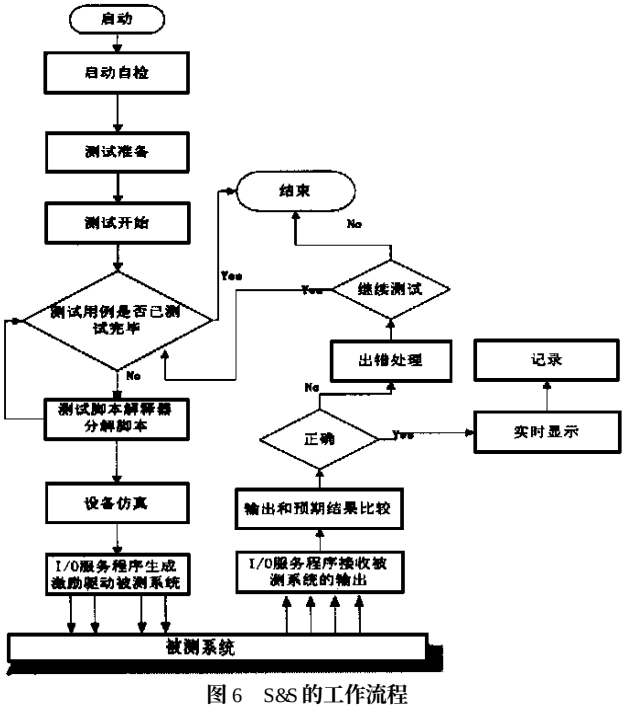

图 6　S&S 的工作流程

4　结束语

我们已成功开发了一个嵌入式软件的可靠性仿真测试平台,并对某个航空电子系统的嵌入式软件进行了可靠性测试,发现了该软件的缺陷和错误,从而确认了上述软件可靠性仿真测试技术与测试环境概念的正确性与可行性。

参考文献

1　Musa J D. Software-Reliability-Engineered Testing. AT&T BELL Lab, 1998
2　Chen Xuesong, Lu Minyan, Ruan Lian. A Study of Constructing Software Operational Profile And Generating Test Cases. ICRMS '99

作者简介:阮镰,男,1938 年生。北京航空航天大学工程系统工程系教授,博士研究生导师。现为国家"九五"重点预先研究项目"嵌入式软件可靠性仿真测试与验证技术"的项目负责人。现为 IEEE member。

(收稿日期:2000 - 01) □