

基于测试用例和时间域软件可靠性模型

张玲¹,袁娜²,马永刚³,黄鹏⁴

(1. 西安美术科技学院 机电工程学院,陕西 西安 710077;

2. 西北工业大学 电子信息学院,陕西 西安 710072;

3. 西北大学 信息科学与技术学院,陕西 西安 710127;

4. 华为技术有限公司,陕西 西安 710075)

摘要:可靠性作为衡量软件质量的重要特性,其定量评估和预测已成为人们关注和研究的焦点。软件可靠性模型既是软件可靠性定量分析的基础,也是可靠性预测的核心和关键。在软件比重日益增加的今天,研究系统的软件可靠性对整个产品质量的提升具有很大的现实意义。现有的软件可靠性模型大都是基于概率统计建立的,考虑的因素比较单一,与工程实际有一定差别。文中对典型的软件可靠性模型进行比较研究,在综合考虑了输入域、缺陷等级、时间域等因素的基础上,经过严密的数学推导,建立了基于测试用例和时间域的软件可靠性混合模型,并对该模型的实际应用进行了介绍。

关键词:测试用例;软件可靠性;输入域;时间域

中图分类号:TP311

文献标识码:A

文章编号:1673-629X(2009)11-0167-04

Mixed Software Reliability Module Based on Testing Cases and Time Domain

ZHANG Ling¹, YUAN Na², MA Yong-gang³, HUANG Peng⁴

(1. Mechanical & Electrical Engineering School, Xi'an College of Fine Arts and Technology, Xi'an 710077, China;

2. School of Electronics and Information, Northwestern Polytechnical University, Xi'an 710072, China;

3. School of Electronics Engineering and Computer Science, Northwest University, Xi'an 710127, China;

4. Huawei Technologies Co., Ltd, Xi'an 710075, China)

Abstract: The reliability as the software quality's important characteristic, its quantitative assessment and the forecast has become the focal point. The software reliable model is not only the software reliable quantitative analysis foundation, but also the reliable forecast core and the key. Today in the software proportion, the research system's software reliability has the very big practical significance to the entire product quality's promotion. Most of the software reliability modules are based on probability and statistics. They consider less elements. There are relatively big gap between these modules and practical project. After analyzing and comparing typical software reliability modules, proposed a new software reliability module, using mathematical knowledge. This module is based on inputting domain, error level and time domain. Also introduce the application of the module briefly.

Key words: testing cases; software reliability; inputting domain; time domain

0 引言

一个优良的可靠性模型可以准确评估和预测软件可靠性行为,这对于软件的市场决策具有重要意义。软件可靠性模型这个领域的研究在20世纪70年代获得了较大发展后,很多可靠性模型已经投入使用。到

目前为止,发表了百余种软件可靠性模型,大多数模型是基于概率统计建立的,有指数失效时间模型、Weibull和Gamma分布失效时间模型、无限失效时间模型、Bayes模型等。文中介绍的软件可靠性模型综合考虑了输入域、缺陷等级、时间域等因素,使模型的建立更切合工程实际。

收稿日期:2009-03-30;修回日期:2009-06-24

基金项目:西安-美国应用材料创新基金资助项目(ZX05097-XA-AM-200514)

作者简介:张玲(1982-),女,硕士,助教,研究方向为嵌入式系统开发与设计应用。

1 典型的软件可靠性模型比较研究

Jelinski-Moranda模型^[1]是最具代表性的早期软件可靠性马科夫过程的数学模型,该模型是1972年提出来的。该模型的优点在于其简单,假设比较直观,数

据要求比较简单;缺点在于假设不完善,缺乏广泛适用性^[2]。

Goel-Okumoto 模型^[3]是 1979 年提出的,此模型是把累积故障数 $N(t)$ 作为时间的变化量,用一个非齐次泊松过程模型加以描述。该模型的优点为数学模型简单,不区分缺陷与失效。缺陷表现在:由于 $N(t)$ 被当作连续时间函数,只有当软件失效次数较大时才能保证精度,模型的广泛适用性尚未进一步确认。

上述两个模型都是基于测试故障数据进行参数估计而建立起来的可靠性评价模型。它们都是从输入域来考虑软件缺陷发生的概率或缺陷总数,却没有从缺陷自身的因素以及输入域数据的输入概率讨论。其实,造成软件缺陷的原因有很多种,比如软件系统设计缺陷、测试用例设计过程引入的人为错误等,而且不同缺陷对软件的可靠性影响不同。另外,每个输入数据出现的概率是不一样的,这些因素都会影响到对软件可靠性的评价。

2 基于测试用例的软件可靠性模型

2.1 基本概念及定义

所谓测试用例是指按照一定顺序执行的与测试目标相关的一系列测试。也就是说,对软件运行过程中所有可能存在的目标、执行、条件和结果的描述,这个特定目标可以是验证一个特定的程序路径或核实是否符合特定需求。因此,测试活动必须在一定的环境条件下,提供测试用例输入,并观察输出,将这些输入与输出进行比较,以确定测试是否满足软件要求。

测试用例的设计既要包含正常情况下的输入测试,也要包含异常情况下的测试。总之,只有面面俱到,保证测试用例的充分性,才能得到正确的测试结果。然而,可靠性测试以黑盒测试为主,黑盒测试用例设计方法包括等价类划分法、边界值分析法、因果图法、判定表驱动法、正交试验设计法、猜错法等^[4]。测试用例的设计不可能穷举所有的输入,只能选取输入域中有代表的集合,例如边界值等。因此,软件可靠性模型与输入的测试用例关系很大。完善的测试用例是可靠性模型的保证^[5]。

输入域主要包括需求规格说明、程序观察和额外的属性规约。其中,额外的属性规约主要指规格说明中没有但满足负面测试或可能用到的那部分数据。从可行性的角度考虑,软件测试时从输入域中选取输入点一般采取以下三步:

首先,按照功能或者模块结构将输入域分成若干个互不相交的子域。一个软件系统由多种功能组成,不同功能或模块的输入数据属性不同,正确划分输入

域是测试的第一步。

其次,在各子域中提取边界值点,构成集合 A_1 。边界值是考察软件正确与否的关键方法。一个软件在正常情况下正确不一定边界值正确,但边界值正确一般正常输入正确的概率较大,但绝非必然。

第三,在相邻边界点之间选取 N 个测试点,如果有 L 个子域,该步选取的测试点个数为 $L * N$ 。测试点选择方法可随机选取,也可按照一定的概率分布选取。具体选取的个数视实际情况而定。该步选取的所有测试点构成集合 A_2 。

为推导可靠性模型,现作如下定义:

(1) 定义测试用例的输入域为 D ,那么 $D = A_1 \cup A_2$ 。将测试用例的输入域 D 划分成 L 个互不相交的子域 $D_1, D_2, D_3, \dots, D_L$,即 $D = D_1 \cup D_2 \cup \dots \cup D_L$,且 $D_i \cap D_j = \emptyset (i \neq j \text{ 且 } i, j = 1, 2, \dots, L)$ 。将属于一个子域的元素排成一列,可得到一个 M 行 L 列的矩阵。定义这个矩阵中的行向量为测试向量,那么输入域就包含 M 个 L 维的测试向量 F_1, F_2, \dots, F_M 。

(2) $I(I \in D)$ 表示输入数据, O 表示输出数据。如果 O 符合设计要求,则认为输入数据不产生缺陷。

(3) 测试可靠性因子:引入测试可靠性因子 C 来表示测试结果,即输入与输出的关系。 C 值取 0 表示输入不引起相应缺陷,取 1 表示引起缺陷。将 $C(I)$ 表示为输入 I 的函数,它满足 $C(I) = 0$ 或者 $C(I) = 1$ 。

(4) 缺陷影响因子:不同的缺陷对软件可靠性的影响程度不同,缺陷影响因子用 α 表示。文献[6]中对软件错误进行深入分析,并将其分为 6 类,分别为:较小错误、一般错误、较严重错误、严重错误、非常严重错误和最严重错误。因此,可将缺陷影响因子分为六级。 $\alpha(I)$ 表示输入 I 对应的缺陷影响因子, $\alpha(I)$ 的值域可以设定为 $(0.5, 0.3, 0.1, 0.05, 0.03, 0.02)$,这些值可以根据实际情况略作调整。 α 值越大,表明这个缺陷对可靠性的影响越大^[6]。

(5) 缺陷产生概率:输入 $I(I \in D)$ 产生缺陷的概率为 $P(I)$ 。

根据上述定义可以得到一个比较通用的衡量软件可靠性的公式:

$$R = \int_{I \in D} P(I) C(I) \alpha(I) dD \quad (1)$$

2.2 实现方法

测试向量表示方式为: $F_i = (I_{i1}, I_{i2}, \dots, I_{iL})$ (i 为整数且 $i \in [1, M]$),其中的 L 个元素分别属于 L 个输入域。

由定义(3)可知,一个输入数据可能产生缺陷也可能不产生缺陷,那么测试向量 F_i 得到的测试可靠性

因子为:

$$C = [c_1, c_2, \dots, c_L] \quad (2)$$

$c_n = 0$ 或 1 (n 为整数且 $n \in [1, L]$), 0 表示没有缺陷, 1 表示有缺陷。

由定义(4)可知,产生的缺陷又分为六个等级,因此可将 c_n 扩展为 6 维列向量,

$$c_n = [c_{1n}, c_{2n}, c_{3n}, c_{4n}, c_{5n}, c_{6n}]^T \quad (3)$$

$c_{mn} = 0$ 或 1 (m 为整数且 $m \in [1, 6]$), 且最多有一个元素为 1。 $c_{mn} = 1$ 表示输入向量 F_i 中的第 n ($n \in [1, L]$) 个元素产生了缺陷, 缺陷等级为 m ($m \in [1, 6]$)。

因此对于由多个输入数据所构成的输入向量 F_i 产生的缺陷等级的描述将是一个 $6 * L$ 二维矩阵。用 C 表示这个 6 行 L 列的二维矩阵, 行数表示缺陷等级, 列数表示输入域:

$$C = \begin{bmatrix} c_{11} & \dots & c_{1L} \\ \dots & \dots & \dots \\ c_{61} & \dots & c_{6L} \end{bmatrix} \quad (4)$$

矩阵中元素 c_{mn} 取值为 0 或者 1, 0 表示没有缺陷, 1 表示输入向量 F_i 中的第 n ($n \in [1, L]$) 个元素产生了缺陷等级为 m ($m \in [1, 6]$) 的缺陷。

使用任一测试向量进行测试, 可得到上述矩阵。对矩阵的行向量采用均值法处理, 并考虑缺陷影响因子, 可靠度可表示为:

$$R_{F_i} = \sum_{m=1}^6 \alpha_m \frac{\sum_{n=1}^L c_{mn}}{L} \quad (5)$$

其中, α_m 表示各个缺陷等级的影响因子, L 表示输入子域的个数。 R_{F_i} 值越大, 可靠性越低。可靠度也可表示为:

$$R_{F_i} = 1 - \sum_{m=1}^6 \alpha_m \frac{\sum_{n=1}^L c_{mn}}{L} \quad (6)$$

这样 R_{F_i} 就与可靠度正相关。

因为测试向量 F_1, F_2, \dots, F_M 相互独立, 可采用均值法来处理 M 个测试向量的测试结果。测试用例的可靠度可表示为:

$$R = \frac{1}{M} \sum_{i=1}^M R_{F_i} \quad (7)$$

3 基于测试用例和时间域的软件可靠性混合模型

基于测试用例的软件可靠性模型是从输入域的角度对软件可靠性进行评估, 这样考虑的因素比较单一, 而现实情况是比较复杂的, 影响可靠度的因素多种多

样, 为了使结果更符合工程实际, 可从时间域的角度对该模型进行修正, 得到混合模型^[7]。从时间域来看, 测试用例在系统中的输入概率虽然是随机的, 但也有规律可循, 如数据的输入概率与使用者的个人习惯有关, 同样的机器每个人由于使用习惯不同, 机器的寿命也不一定完全相同。这个混合模型是在基于测试用例的软件可靠性模型的基础上进一步考虑数据的输入概率, 从而得到一个新的模型^[8]。

用 h 表示输入数据 I 在整个系统操作过程中一定时间段内的操作的概率, 对于测试向量 $F_i = (I_{i1}, I_{i2}, \dots, I_{iL})$, 其对应的输入概率向量为 $H_i = (h_{i1}, h_{i2}, \dots, h_{iL})$, 由公式(2)得:

$$C = [c_1 h_{i1}, c_2 h_{i2}, \dots, c_L h_{iL}] \quad (8)$$

考虑了输入概率之后, C 中的元素取值范围变为 0 到 1 之间的实数。

由公式(3)得:

$$c_n h_{in} = [c_{1n} h_{in}, c_{2n} h_{in}, c_{3n} h_{in}, c_{4n} h_{in}, c_{5n} h_{in}, c_{6n} h_{in}]^T \quad (9)$$

由公式(4)得:

$$C = \begin{bmatrix} c_{11} h_{i1} & \dots & c_{1L} h_{iL} \\ \dots & \dots & \dots \\ c_{61} h_{i1} & \dots & c_{6L} h_{iL} \end{bmatrix} \quad (10)$$

由公式(5)得:

$$R_{F_i} = \sum_{m=1}^6 \alpha_m \frac{\sum_{n=1}^L c_{mn} h_{in}}{L} \quad (11)$$

根据公式(11)可得到修正后的软件可靠度。

4 工程应用

上述章节已对软件可靠性的评价模型做了较为详细的论述, 并提出了易于工程化的可靠性评价方法。“基于 WinCE 的电子清纱器软件系统”进一步使用并验证了上述模型, 该项目软件系统结构如图 1 所示。

该系统的可靠性是在真实的工业现场进行测试的。该系统包含 12 个子模块, 因此将输入域划分为 12 个子域, 然后在各子域中提取边界值, 再从相邻边界值选取一系列的点集, 得到测试用例的输入域 D 。在功能及可靠性测试中, 运用黑盒测试方法, 对系统的各种功能进行严格的详细测试。通过三个月的实际测试, 得到了大量的测试数据。对这些故障数据采用文中所讲的混合模型进行处理, 得到了一系列的可靠性评价结果, 该系统的可靠性不断提高, 最终使软件可靠率达到 0.9836。

该产品自进入实际工厂生产测试三个月之后, 连续工作半年未出现任何软件失效现象, 从而使得相关

可靠性的研究在实际系统中进一步得到验证。

该模型的适用范围以及模型的一致性等问题有待于进一步的解决。

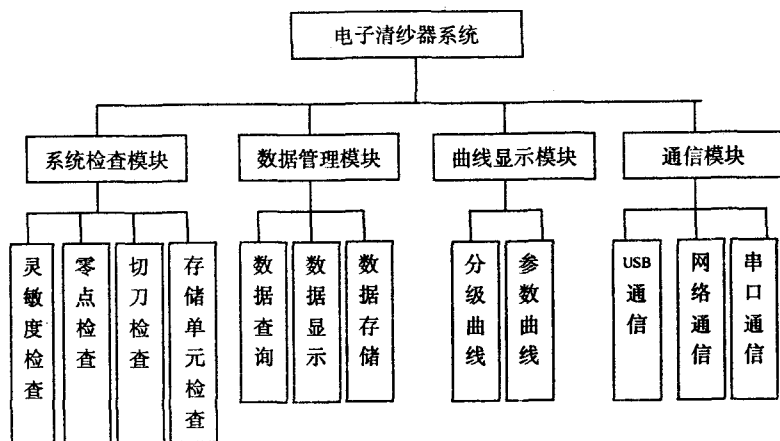


图 1 应用软件系统框图

5 结束语

在软件可靠性模型构建过程中,充分考虑了测试的各个环节对测试结果的影响,该模型不仅从输入域的试验结果来推导模型,更从各种缺陷对软件可靠性产生的效果不一样来分析,使得基于测试用例的模型与实际更接近。并在此模型的基础上,提出了一个基于输入域和时间域的混合模型,将使用者的输入概率综合考虑到模型的建立中,使得模型更具有合理性。该模型具有简单、实用、精确和易于工程化等特点,但

参考文献:

- [1] 宫云战. 软件测试[M]. 北京:国防工业出版社,2006:5-6.
- [2] 孙 勇. 软件可靠性模型应用研究[D]. 南京:东南大学,2004:25-30.
- [3] Lyu M R. Handbook of Software Reliability Engineering[M]. New York: McGraw-Hill and IEEE Computer Society Press, 1996: 18-24.
- [4] 吴艳征,宋志强. 浅谈黑盒测试用例设计方法[J]. 科技信息(学术研究), 2008(16): 178-179.
- [5] 蒋天乐,徐国治. 软件缺陷及软件可靠性技术[J]. 计算机仿真,2004,21(2):141-144.
- [6] 周卫东. 组合导航系统应用软件可靠性研究[D]. 哈尔滨:哈尔滨工程大学,2006: 95-96.
- [7] Jeske D R, Zhang X, Pham L. Adjust software failure rates that are estimated from test data[J]. IEEE Trans. on Reliability, 2005, 54(1): 107-114.
- [8] Gokhal S S, Trivedi K S. A Time/Struct Based Software Reliability Model[J]. Annals of Software Engineering, 1999, 8(1): 85-121.

(上接第 110 页)

4 结束语

.NET 环境下的 GDI+ 功能非常强大,文中的例程只是用到了其中的很小一部分,很多方法应用还在等待慢慢去探索。开发软件过程中,熟练使用 GDI+ 技术不仅可以摆脱编程环境固定组件种类的束缚,界面的风格化和独创性更是多数人评价一款软件的重要

因素。

参考文献:

- [1] 石云辉,黄 隽. 基于.NET 的网络故障监测报警系统的设计[J]. 微计算机信息,2008,24(3):199-200.
- [2] Nagel C, Evjen B, Gly J, et al. Professional C# 2005 (C# 高级编程)[M]. 第 4 版. 北京:清华大学出版社,2006.
- [3] 尹 勇,金一丞. 航海模拟器与分布交互仿真技术[J]. 计算机仿真,2007,17(6):68-70.
- [4] Chand M. Graphics Programming with GDI+ [M]. 北京:电子工业出版社,2004.
- [5] White E. GDI+ 程序设计[M]. 北京:清华大学出版社,2002.
- [6] Faison T. Visual C# 基于组件的开发[M]. 北京:清华大学出版社,2003.
- [7] 李顺亮,张均东. GDI+ 技术在综合船舶监控系统中的应用[D]. 大连:大连海事大学,2005: 42-44.
- [8] 邓海生,李军怀,刘红英. 基于 ASP.NET 的 Web 服务性能优化[J]. 计算机技术与发展,2007,17(10):94-98.

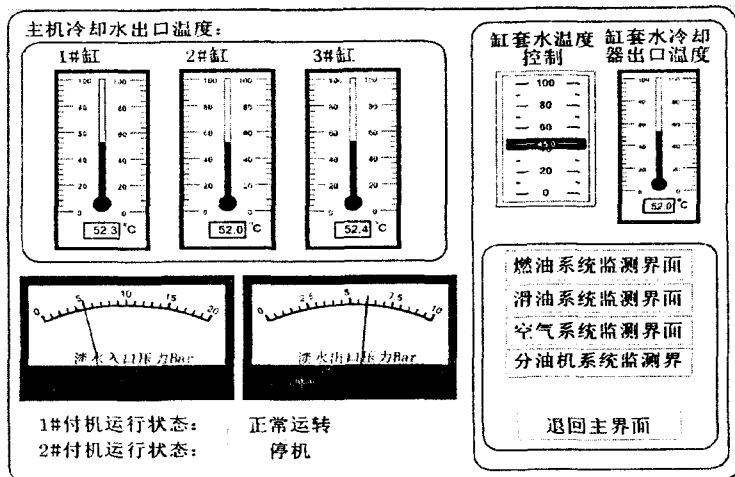


图 3 组件技术在工控模拟器监控系统中的应用