

# 基于使用模型的嵌入式软件测试用例生成研究

熊利,周宽久

(大连理工大学 软件学院 嵌入式系统工程系,辽宁 大连 116622)

**摘要:**基于 Markov 链使用模型的软件统计测试是产生高效测试用例,实现软件可靠性定量评估的有效方法。介绍了基于使用模型的统计测试方法,论述了使用模型的概念和特点,以及从使用模型中可以计算出的静态参数和它们在统计测试和软件开发中所起的作用。提出 Markov 链使用模型用于嵌入式软件的测试,从理论上得到嵌入式软件的可靠性。具体阐述了嵌入式软件的 Markov 链使用模型自动生成测试模型的方法。同时使用改进的 Kullback 判别式探讨测试的可靠性问题,从理论上证明了测试链到使用链收敛的必然性。理论分析和初步的实例证明该方法是可行的和有前途的。

**关键词:**Markov 使用模型;嵌入式软件;测试用例生成

**中图分类号:**TP311.5

**文献标识码:**A

**文章编号:**1673-629X(2010)05-0092-04

## Research of Embedded Software Generation of Test Cases Based on Usage Model

XIONG Li, ZHOU Kuan-jiu

(Dept. of Embedded Systems Eng., Sch. of Software, Dalian Univ. of Techn., Dalian 116622 China)

**Abstract:** Statistical software testing based on the use of Markov chain model, is effective approach which can produce test case and implement the evaluation of software reliability quantitatively. Statistical testing based on a usage model as well as properties of usage model is introduced in this paper. The statistical theory is used on embedded software test to obtain the reliability of the embedded software theoretically. The paper also introduced how to automatically generated test case from the Markov chain usage mode of embedded software. The Kullback discriminant is used as the reliability judgment criteria of the test-chain to usage-chain, and the inevitability of the convergence is testified theoretically. The primary experiments and theorized analysis prove the method is approving and promising.

**Key words:** Markov chain usage model; embedded software; generation of test cases

### 0 引言

软件测试是软件开发的重要步骤和软件质量保证的重要环节,软件统计测试则是高可靠软件测试的重要内容。

近年来国内外学者利用 Markov 的统计理论生成测试用例<sup>[1~5]</sup>。其中颜炯<sup>[1]</sup>从 UML 角度构建 Markov 链使用模型,高海昌<sup>[2]</sup>等提出了基于路径使用的 Markov 链模型来分析模块内部代码结构的统计测试方法。Walton G H<sup>[3]</sup>提出了使用模型中转移概率的生成方法,Guén H L<sup>[4]</sup>对基于 Markov 使用模型生成测试用例的可靠性估计进行了讨论和研究。

对于嵌入式软件而言,其安全性的失效可能会产

生灾难性的后果。无论是“阿丽娜 5 型”火箭的爆炸还是国内某型飞机由于数字电传系统软件的错误而造成坠机的重大事件都可以说明这一点<sup>[6]</sup>。随着嵌入式技术的不断发展,嵌入式系统在航天航空、军事科学、工业控制、通讯及消费电子等领域的应用越来越广泛,性能要求越来越高,功能需求越来越复杂,规模也越来越大<sup>[7]</sup>。同时,嵌入式软件在嵌入式系统中所占得比例也越来越大,其质量对整个系统的安全与稳定,对产品的最终质量都起着决定性的作用。因此,如何提高嵌入式软件的质量,保障其可靠性成为人们关注的焦点,而嵌入式软件的测试无疑是提高软件质量的主要方法之一,其自动化测试技术成为当前研究的热点。然而,由于嵌入式系统的自身特点,如实时性,内存不丰富,输入、输出通道少,开发工具昂贵,并且与硬件紧密相关,CPU 种类繁多等原因,嵌入式软件的测试比一般商用软件的测试更为复杂。可以说嵌入式软件是最难测试的一种软件,传统的软件分析和测试手段已经不能满足嵌入式软件分析测试的基本要求<sup>[8]</sup>。因此,引

收稿日期:2009-09-10;修回日期:2009-12-23

基金项目:大连市信息产业 IT 专项基金(DL20080243)

作者简介:熊利(1983-),男,硕士研究生,研究方向为嵌入式系统;周宽久,博士,副教授,硕士生导师,CCF 高级会员,研究方向为可信嵌入式系统、嵌入式系统应用。

入新的测试方法对嵌入式软件的测试有着重要的现实意义。

可靠性测试作为可靠性工程的重要环节,是进行可靠性评价的主要手段之一。成功实现可靠性测试的关键在于设计出满足测试要求的一组用例,通过测试运行,获得能够比较真实的反映软件运行时可靠性、安全性等方面的评价数据,从而为软件的可靠性评估提供一个量化依据。目前对嵌入式软件可靠性测试、评估的探讨和研究<sup>[9]</sup>比较深入,实现了可靠性测试用例的设计。但设计可靠性测试用例的手段主要依靠手工分析,沿用传统的软件测试用例设计方法进行,不能够满足上述可靠性测试用例设计的基本要求,因此无法实现真正意义上的软件可靠性测试。

基于此,文中引进 Markov 理论,构造嵌入式软件的使用模型,然后设计算法自动生成相应的测试模型,并使用改进的 Kullback 对测试的可靠性问题进行了探讨和研究。文中测试用例生成算法与思想均在 VC 中得以实现。

## 1 Markov 链使用模型

### 1.1 Markov 链使用模型

用户对软件的使用过程用 Markov 过程进行描述,即任何下一个发生的事件只和当前的状态有关,不涉及历史信息。因此软件的使用模型可以表示为有穷状态、离散参数的 Markov 链,包括描述系统行为特性的有向图以及与有向图相对应的转移矩阵两个部分。有向图由一个状态集组成,状态之间由边(转移边)连接,状态表示软件使用过程中的内部环境,边表示状态间的转移关系,边上标记相应的激励事件以及事件发生的概率值,概率值确定了从给定状态发生特定激励事件的可能性大小,且特定状态所有出边的转移概率之和为 1。每一个使用模型都有惟一的初态和终态,软件的每一次使用都从初态开始经过若干个中间状态,最后到达终态。使用模型转移概率的确定,主要依据历史统计数据计算得到,或采用已有的类似系统的使用数据;在缺乏历史数据情况下也可由相应的专家结合应用背景制定,或采用均匀分布<sup>[10,11]</sup>。

图 1 简单描述了嵌入式多媒体类软件的使用模型,依概率从初始状态经过若干中间状态到终止状态。在这个使用模型中,状态系统启动是初态,状态系统关闭是终态。初始化、多媒体操作(包括播放、暂停、快进、后退、停止等)和参数设置是 3 个中间状态,a、b、c、e、f 是 5 种不同的激励输入,与之对应的括号中的数表

示该边的转移概率。

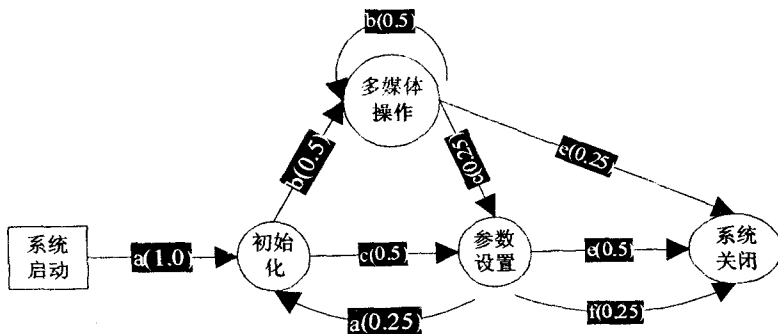


图 1 嵌入式多媒体类软件的 Markov 使用模型实例

### 1.2 Markov 链使用模型的静态参数

使用模型是对软件使用过程中软件行为特性的精确刻画,运用统计学理论对使用模型进行分析计算,可以获得软件使用特性相关的静态参数<sup>[12]</sup>。利用这些静态参数以及测试相关的资源和时间,可以制定合理高效的测试计划,从而更好地控制软件活动,实现软件设计、开发、测试等活动间的良性互动。同时,这些参数也是实现软件可靠性定量评估的基础。最主要的参数包括:转移矩阵;各状态在长时间运行的占有率;单个测试用例中各状态的发生概率;单个测试用例中各边的发生概率。

#### 1.2.1 转移矩阵

定义转移矩阵  $p = [p_{i,j}]$ ,其中  $p_{i,j}$  为从状态  $i$  的转移概率;定义  $(i,j)$  从状态  $i$  到状态  $j$  的转移边(弧)。因此从图 1 中可以得到一个  $5 \times 5$  的矩阵,矩阵的行和列分别是相应的状态,是模型分析与验证,软件可靠性评价的基本依据。

#### 1.2.2 各状态在长时间运行的占有率

状态在长时间运行中的占有率(Long Run Occupancy of Each State)描述了各状态的重要程度,可以用于软件开发中的资源分配。占有率大的状态使用率也比较高,开发过程中与之对应的模块可以分配较多的资源。很显然,状态在长时间运行中的占有率等于与之相邻的状态在长时间运行中的占有率和相邻状态转移到该状态的转移概率乘积之和,即

$$\pi_j = \sum_i \pi_i p_{ij} \quad (1)$$

各状态的长时间运行概率同时满足:

$$1 = \sum_i \pi_i \quad (2)$$

由公式(1)、(2)可以计算各个状态的长时间运行概率。

#### 1.2.3 单个测试用例中各状态的发生概率

单个测试用例中各状态的发生概率(Occurrence Probability of Each State In a Single Use)揭示了单个测

试用例中特定状态出现的频度。对于那些出现频率比较高的部分可以进行自动测试,而出现频率比较低的部分的测试可以手动进行。

假设状态  $j$  和终态被设置为可吸收的,  $y_{ij}$  代表从状态  $i$  开始最终到达状态  $j$  的概率,  $p_{ij}$  代表从状态  $i$  转移到状态  $j$  的概率, exit  $k$  代表使用模型的终止状态, 那么:

$$y_{ij} = p_{ij} + \sum_{k \in \{j, \text{exit}\}} p_{ik} y_{kj} \quad (3)$$

可以计算出矩阵  $Y = [y_{ij}]$ , 但实际上只需要它的第一行, 即从初始状态开始的序列经过状态  $j$  的概率, 也就是单个测试用例中状态  $j$  发生的概率。

#### 1.2.4 单个测试用例中各边的发生概率

单个测试用例中各边的发生概率 (Occurrence Probability of Each Arc In a Single Use) 揭示了单个测试用例中特定的边出现的频度, 在测试自动化中有很大的用处。对于那些出现频率比较高的边可以进行自动测试, 而出现频率比较低的部分的测试可以手动进行。

在原状态  $a$  和目标状态  $b$  之间添加一个中间状态  $\alpha$ , 根据状态在测试用例中的发生概率的计算方法:

$$y_{ia} = p_{ia} + \sum_{k \in \{a, \text{exit}\}} p_{ik} y_{ka} \quad (4)$$

人们就可以用状态在测试用例中的发生概率的方法来计算边在测试用例中的发生概率。

## 2 Markov 链测试模型

当使用模型完成后, 可以利用该模块的测试用例进行测试。提供测试用例需要同时提供当前测试用例可能经过的路径, 以便于实际执行时的路径进行比较计算。测试链是从使用链产生的, 它把使用链各边所对应的转移概率替换为一个初值为 0 的计数器, 随着测试的进行, 每当一个测试用例经过该边时计数器就加 1。当一组测试用例执行完毕后, 比较实际执行路径与给定预期的执行路径, 若二者不同则认为该模块发生错误, 标识为一个软件失效, 此时可进行跟踪测试以改正错误。当  $n$  组测试用例执行完毕, 进行测试 Markov 链的状态转移概率的计算。

概率计算方式为从测试 Markov 链的开始状态出发, 计算每一条过渡弧  $(i, j)$  的通过次数占从  $i$  出发的总测试用例个数的百分比即为该弧 (或从  $i$  到  $j$  的状态转移) 的测试概率。

对于图 1 的 Markov 使用模型实例, 假设测试用例序列为:  $T = \{a, b, b, e; a, c, e; a, b, b, b, e; a, c, a, b, b, c, f\}$ , 则产生的测试模型转移矩阵如下 (见公式 (5)):

初始计算时, 状态图上的使用 Markov 链的概率分布差别较大。经过  $m$  个  $n$  组测试用例执行, 测试模型的概率分布逐渐趋近于使用模型, 当二者的差值小于用户可接受的域值时, 即可停止测试。

$$P_t = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \frac{3}{5} & \frac{2}{5} & 0 \\ 0 & 0 & \frac{4}{7} & \frac{1}{7} & \frac{2}{7} \\ 0 & \frac{1}{3} & 0 & 0 & \frac{2}{3} \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (5)$$

## 3 测试的可靠性

软件统计测试的停止标准可以简单地描述为选择一个目标可靠性, 然后进行测试, 直到可靠性估计达到或超过目标可靠性时停止测试。测试链到使用链的逼近给出了直接基于使用链和测试链的统计属性的一个停止标准。使用链是模块的理想测试的模型, 每一条弧的概率都建立在实际使用的最佳估计, 并且没有失效情况出现。而测试链是特定测试历史的模型, 包括失效情况发生。因此, 使用链表示了没有失效情况下的统计测试情况, 而测试链代表了有失效情况下的统计测试。两个模型之间的差异成为测试过程的有用衡量。

测试链和使用链的比较可以通过 Kullback 判别式即一个称作 discriminant 的值<sup>[13]</sup>来进行, 它是两个随机过程似然度的期望值。但可以看出它并不总是可计算的, 在测试过程中只有在使用模型的所有边都覆盖以后它才有意义。因此产生了一种变体的计算方法<sup>[14]</sup>, 使得在任何情况下  $K(U, T)$  都是可计算的:

$$K(U, T) = \sum_{i=1}^n \pi_i \sum_j u_{ij} \lg \left( \frac{u_{ij}}{\epsilon - \epsilon(\text{sgn}(t_{ij})) + t_{ij}} \right) \quad (6)$$

其中  $u_{ij}$  和  $t_{ij}$  分别为使用链和测试链中从状态  $i$  到状态  $j$  的转移概率,  $\pi_i$  为状态  $i$  长时间运行中的占有率即长时间运行中各状态所占有的比例。 $\epsilon$  为一个很小的正数;  $\text{sgn}(x)$  为符号函数, 当  $x = 0$  时  $\text{sgn}(x) = 0$ , 当  $x < 0$  时  $\text{sgn}(x) = -1$ , 当  $x > 0$  时  $\text{sgn}(x) = 1$ 。显然, 在测试过程中当所有边都被覆盖以后  $K(U, T) = K(U, T)$ 。

可以看到测试链和使用链越接近,  $K(U, T)$  就越接近于 0。对于不同的软件根据参数选择可以接受的特定值。当  $K(U, T)$  小于设定的特定值的时候, 测试链到使用链的逼近达到阈值, 测试变得充分, 可以停止测试了。

## 4 测试用例生成算法

### 4.1 实现的算法步骤

实现的算法步骤如下:

(1) 在使用链的约束范围内随机地产生一有限的转移矩阵集合。

$$PP = \{P_1, P_2, \dots, P_m\} \quad (7)$$

(2) 按照输入测试用例个数  $k$  从  $PP$  中取出对应的转移矩阵。

$$p_k = (p_{kij}) (i, j) \in \Omega \times \Omega, 1 \leq k \leq M \quad (8)$$

(3) 依据公式(1)和(2)计算平稳分布。

$$\pi_k = \{\pi_{ki}, i \in \Omega\} \quad (9)$$

(4) 给定常数  $\epsilon$ , 依据  $p_k$  中的转移该路模拟的状态转移和给定的状态转移总数, 依据公式(6)计算对应的  $K$  值。

### 4.2 实例

表1(取  $\epsilon = 0.001$ ,  $k$  的阈值为 0.001)是图1使用模型在测试过程中的状态和边的覆盖率以及 discriminant 值。可以看到在测试用例达到一定的数量之后, 状态与边的覆盖率到达 100%, 所得的 discriminant 的值急剧减少, 当反复实验并不断增加测试用例得到的最优值小于 0.001 之后, 测试链到使用链的逼近已经达到可以接受的阈值, 可以终止测试了。

表1 不同的测试用例数的对应状态、边覆盖和 discriminant 值

测试用例个数	状态覆盖率	边覆盖率	$K'(U, T)$
1	0.8000	0.3333	0.9434
2	1.0000	0.5556	0.4806
3	1.0000	0.6667	0.2124
4	1.0000	0.8889	0.1059
5	1.0000	0.8889	0.0664
10	1.0000	1.0000	0.0458
100	1.0000	1.0000	0.0092
200	1.0000	1.0000	0.0079
400	1.0000	1.0000	0.0017
800	1.0000	1.0000	0.0008

## 5 结束语

基于 Markov 链使用模型的软件统计测试是产生高效测试用例、实现软件可靠性定量评估的有效方法。Markov 链使用模型能够较清晰地体现交互式系统的特点, 更加清晰地明确了软件使用过程中软件操作之间的执行顺序关系, 符合对象、消息机制的面向对象思想。在工程上由模型可产生大量的测试用例, 提高了软件的易测性。同时可以结合丰富的 Markov 统计分

析理论, 实现软件可靠性的定量评价<sup>[15,16]</sup>。

分析测试链在许多文献中一直作为可靠性模型的一个补充。文中通过对嵌入式软件使用模型的测试链构造与分析并利用改进的 Kullback 判别式对测试终止的判断, 提高了测试的可靠性和充分性。相信随着研究的不断深入和嵌入式软件的普及应用, 基于 Markov 使用模型的软件统计测试方法一定会在嵌入式软件中得到广泛的应用并发挥重要作用。

### 参考文献:

- [1] 颜炯, 王戟, 陈火旺. 基于 UML 的软件 Markov 链使用模型构造研究[J]. 软件学报, 2005, 16(8): 1386-1394.
- [2] 高海昌, 冯博琴, 曾明, 等. 基于 Markov 链路径使用模型的软件统计测试[J]. 计算机工程, 2006, 32(19): 20-22.
- [3] Walton G H, Poore J H. Generating transition probabilities to support model-based software testing[J]. Software: Practice and Experience, 2000, 30(10): 1095-1106.
- [4] Guen H L, Marie R, Thelin T. Reliability estimation for statistical usage testing using Markov chains[C]//Proceedings of the 15th International Symposium on Software Reliability Engineering (ISSRE'04). [s.l.]: [s.n.], 2004: 54-65.
- [5] Yang Shiping, Sang Nan, Xiong Guangze. A vector Markov model for structural coverage growth and number of failure occurrences[C]//Proceeding of 13th International Symposium Software Reliability Engineering (ISSRE'11). [s.l.]: [s.n.], 2002: 304-315.
- [6] Müllerburg M. Software intensive embedded systems[J]. Information and Software Technology, 1999, 41(14): 979-984.
- [7] Card D N, Glass R L. Measuring Software Design Quality [M]. [s.l.]: Prentice-Hall, 1990.
- [8] Fin A, Fummi F, Pravadelli G. AMLETO: A Multi-language Environment for Functional Test Generation[C]//Proceedings of the IEEE International Test Conference 2001. [s.l.]: [s.n.], 2001: 821-829.
- [9] Seon-Jae J, Hae-Geun K, Youn-Ky Ch. Manual specific testing and quality evaluation for embedded software[C]//Proceedings of the 7th IEEE/ACIS international conference on computer and information science. [s.l.]: [s.n.], 2008: 502-507.
- [10] Prowell S J, Trammell C J, Linger R, et al. Cleanroom software engineering technology and process[M]. [s.l.]: Addison-Wesley, 1998.
- [11] Prowell S J. JUMBL: A tool for model-based statistical testing[C]//Proceedings of the 36th Annual Hawaii International Conference on System Sciences (HICSS'03). [s.l.]: IEEE Computer Society Press, 2002.
- [12] Prowell S J. Computations for Markov Chain Usage Models

$$X' = \begin{bmatrix} 0.61465 & 0.49913 & 0.40532 & 0.23532 & 0.55199 & 0.74411 & 0.66366 & 0.82691 & 0.91589 \\ 0.06109 & 0.33276 & 0.11581 & 0.12943 & 0.01216 & 0.04556 & 0.14480 & 0.04771 & 0.02331 \\ 0.11192 & 0.41595 & 0.24609 & 0.09884 & 0.08794 & 0.07592 & 0.15687 & 0.07951 & 0.02997 \\ 0.68367 & 0.45754 & 0.61763 & 0.36945 & 0.80740 & 0.63781 & 0.61540 & 0.46911 & 0.37635 \\ 0.30546 & 0.29116 & 0.49217 & 0.85657 & 0.09169 & 0.13667 & 0.24133 & 0.15107 & 0.08992 \end{bmatrix} \quad (7)$$

然后利用式(2)进行相似度计算,其中设定各个故障特征的权重为  $\omega_k = (0.25, 0.025, 0.025, 0.1, 0.2, 0.05, 0.05, 0.1, 0.2)$ , 取分辨系数  $\zeta = 0.5$ ; 再利用式(3)~式(5)进行欧氏距离和相似度之间的转换, 可得当前案例与以往案例集中各个案例之间的相似度。如表 2 所示。

表 2 当前故障与各案例之间的相似度

	$SIM(x_0, x_1)$	$SIM(x_0, x_2)$	$SIM(x_0, x_3)$	$SIM(x_0, x_4)$	$SIM(x_0, x_5)$
相似度	0.273	0.590	0.673	0.303	0.536

根据表 2 计算的结果, 可知案例  $x_3$  与案例  $x_0$  的相似度最大, 即当前案例  $x_0$  与案例  $x_3$  特征向量最为相似, 因此在解决此故障时可以借鉴案例  $x_3$  的解决方案。

同时, 利用最近邻检索算法和余弦函数算法<sup>[11]</sup>计算出的相似度如表 3 所示。

表 3 最近邻与余弦函数算法计算的相似度

	$SIM(x_0, x_1)$	$SIM(x_0, x_2)$	$SIM(x_0, x_3)$	$SIM(x_0, x_4)$	$SIM(x_0, x_5)$
最近邻算法	0.546	0.868	0.905	0.612	0.877
余弦函数算法	0.515	0.868	0.908	0.572	0.900

根据表 3 的结果, 同样可知案例  $x_3$  与案例  $x_0$  的相似度最大, 从而证明了文中算法是正确有效的; 但就某一算法而言, 表 2 数据比表 3 数据更加分散, 即文中算法比另外两种算法具有更好的分辨率, 在同等条件下可以更容易地得到正确的结果。

4 结束语

文中将灰色关联理论引入到 CBR 中, 并结合欧氏距离计算故障案例之间的相似度问题, 提高了检索效

率。同时, CBR 作为新兴的人工智能推理技术, 与其它人工智能推理和学习方法的有效结合和应用成为其近些年来一个发展方向。

参考文献:

[1] 许六一, 李建璜, 胡柏青, 等. 一种改进的 CBR 故障诊断方法研究[J]. 微计算机信息, 2007, 23(3): 216-218.

[2] LIU Jia li, YAN Xiang bin, QI Wei. A Case-based Reasoning System for Mechanical Design[C]//International Conference on Management Science & Engineering. California: IEEE, 2008.

[3] 艾芳菊. CBR 中的检索模型研究[J]. 计算机工程与应用, 2005(19): 77-79.

[4] 李小青. 基于案例推理的故障诊断方法[J]. 计算机测量与控制, 2007, 15(9): 1130-1131.

[5] 杨善林, 倪志伟. 机器学习与智能决策支持系统[M]. 北京: 科学出版社, 2004: 79-112.

[6] 倪志伟, 李锋刚, 毛雪岷. 智能管理技术与方法[M]. 北京: 科学出版社, 2007: 34-39.

[7] 邓聚龙. 灰理论基础[M]. 武汉: 华中科技大学出版社, 2002: 135-150.

[8] 郭茂祖, 苏晓红, 王亚东, 等. 基于 IBL 算法的 CBR 系统中索引与检索机制研究[J]. 计算机工程与应用, 2001(5): 67-69.

[9] Burkhard H D. Similarity and Distance in Case Based Reasoning[J]. Fundamenta Informaticae, 2001, 47: 201-215.

[10] 王 东, 刘怀亮, 徐国华. 案例推理在故障诊断系统中的应用[J]. 计算机工程, 2003(12): 10-12.

[11] 陈富民, 吴垌沅, 林志航. 用于故障诊断的案例匹配算法分析[J]. 计算机应用研究, 2008, 25(5): 1352-1354.

(上接第 95 页)

[M]. [s.l.]: Software Quality Research Laboratory, 2000: 49-53.

[13] Kullback S. Information theory and statistics[M]. New York: Wiley, 1958: 10-83.

[14] Sayre K, Poore J H. Stopping criteria for statistical testing[J]. Information and Software Technology, 2000, 42(12): 851-857.

[15] Dulz W, Zhen F, Ma Te Lo. Statistical usage testing by annotated sequence diagrams, Markov chains and TTCN-3[C]//Proceedings of the Third International Conference on Quality Software (QSIC'03). [s.l.]: [s.n.], 2003: 336-342.

[16] Guen H L, Thelin T. Practical experiences with statistical usage testing[C]//Proceedings of the 11th Annual International Workshop on Software Technology and Engineering Practice (STEP'04). [s.l.]: [s.n.], 2004: 87-93.