第41卷 第4期 西 安 交 通 大 学 学 报 Vol.41 №4
2007年4月 JOURNAL OF XI′AN JIAOTONG UNIVERSITY Apr. 2007

基于UML的软件可靠性测试用例生成的混合模型

王昕，覃征，韩峰岩
（西安交通大学电子与信息工程学院，710049，西安）

摘要：为了解决实时控制系统软件可靠性测试用例生成的问题，在分析操作剖面模型和Markov链模型的基础上，提出了一种基于UML的混合模型.该模型用操作剖面模型来定义使用用例，并将状态图嵌入其中以表述该用例的动态特性.通过平展状态图获得使用图，使用图按一定的概率迁移，从而获得用Markov链表示的使用模型，而操作剖面模型定义的使用用例集与Markov链表述的状态迁移模型可结合生成可靠性测试用例.通过雷达波束调度软件可靠性测试表明，按所提模型在各测试周期生成的测试用例集的框架稳定性比较好，测试用例极少出现重复现象，它综合了操作剖面模型和Markov链模型的优点，可用于开发实时控制系统的软件可靠性测试用例.

关键词：软件可靠性；测试用例；操作剖面；混合模型
中图分类号：TP311 **文献标识码**：A **文章编号**：0253-987X(2007)04-0421-05

UML Based Hybrid Model for Generation of Software Reliability Test Cases

Wang Xin, Qin Zheng, Han Fengyan
(School of Electronics and Information Engineering, Xi′an Jiaotong University, Xi′an 710049, China)

Abstract: In order to produce reliability test cases for real-time control software, a hybrid model based on UML is presented by analyzing the operational profile model and Markov chain model. The use-cases are defined by using operational profile model with imbedded state-graph to describe its dynamic characteristics. The usage graphs can be obtained by flattened state diagrams, and the usage model denoted by Markov chains is obtained according to certain transfer probability. The use-cases defined by operational profile model combine with Markov chain state transfer model to generate software reliability test cases. The reliability test of radar beam control software shows that the frame of test case set generated by the proposed model in each test period is stabler and few repeated case occurs. This model has the advantages that the operational profile model and Markov chain model possess, and is applicable for developing reliability test cases for real-time control software.

Keywords: software reliability; test case; operational profile; hybrid model

软件可靠性测试是提高软件可靠性、定量评定可靠性水平的关键技术，但其难点是测试用例的生成.目前，软件可靠性测试用例生成模型主要有2种，即基于操作剖面的测试用例生成模型和基于Markov链的测试用例生成模型.操作剖面的概念由Musa提出[1]，之后人们进行了大量的研究，包括软件模块操作剖面的表达、测试用例生成和基于操作剖面测试的可靠性评估[2]，以及构造具有并发特性、反应特性的软件操作剖面的方法[3]，适用于不同用户模式下操作剖面的细化描述方法[4]等.基于Markov链的测试用例生成方法包括状态分层Markov模型、基于概率状态图的Markov链建模方

收稿日期：2006-09-01. 作者简介：王昕(1978～)，女，博士生；覃征(联系人)，男，教授，博士生导师. 基金项目：国家重点基础研究发展规划资助项目(2004CB719401).

法[5-8]，这2种方法为软件的建模和可靠性测试用例生成奠定了基础，但仍然存在一些难以克服的缺点.

本文在分析操作剖面、Markov链模型利弊的基础上，提出了一种基于UML的综合运用操作剖面和Markov链的软件可靠性测试用例生成的混合模型，它可以简化测试用例生成过程，提高测试效率，生成实时控制系统的软件可靠性测试用例.

1 操作剖面和Markov链模型分析

1.1 操作剖面模型的特点

操作剖面是软件系统操作及其发生概率的集合，操作剖面模型的主要特点如下.

(1)具有树状分解结构，可提供层次化、渐进化的建模方法，剖面生成过程清晰，适用于大型复杂软件建模.

(2)没有明确考虑软件系统不同输入之间以及同一输入不同时刻之间的依赖关系. 无法明确地表示连续操作之间的关系，也不知道这些操作间的协议，难以为具有反应性、并发性的实时控制系统软件建模.

(3)操作剖面构造的用例集，对操作剖面扩展出来的运行集合有较强的依赖性，而运行集合基本上是定制的，因此每次构造的用例集都相对稳定，不便于表现测试输入的随机性.

1.2 Markov链模型分析

Markov链模型是一种迁移具有概率特征的有限状态机，其主要特点如下.

(1)通过对状态图的遍历所生成的测试用例直观、形象，状态转移充分考虑了不同输入之间以及同一输入不同时刻之间的依赖关系.

(2)测试用例生成的随机性很强.

(3)随着软件复杂程度的增加，状态空间呈指数增长，所以难以处理复杂的软件系统.

由以上分析可见，操作剖面的建模过程类似于功能分解，而Markov链模型更近似于有限状态机. 从测试的角度看，单独使用任何一种方法都难以构造出完备、有效的测试用例集，但二者具有很强的互补性，综合运用它们可以取得较好的效果.

2 模型表述

UML用使用用例(UC)来描述系统的功能，用状态图(SC)来模拟用例的动态执行情况，用类图表示类元素之间的关系，用顺序图表示对象之间相互交换消息的序列等，所以UML比较适合于描述软件测试过程. 目前，UML已用于生成软件测试用例[9-12]，但在实时、反应、并发特性的实时控制系统的软件可靠性测试方面，仍有许多问题需要解决.

本文提出的基于UML的软件可靠性测试用例生成模型的思路是：通过加入统计约束的UML的UC和SC来生成可靠性测试用例，如既考虑UC的发生概率，也关注SC中的迁移概率. 具体做法如下.

(1)用操作剖面建模方法生成软件的操作集合，用加入统计约束的UC来描述软件操作及其出现的概率.

(2)将SC附加到UC之中，以描述用例的场景，表现用例的动态特性，刻画软件对象之间以及软件对象与环境之间的交互过程，并通过为SC中的迁移添加发生概率，来表现不同场景下用户使用的分布.

(3)利用用例视图生成测试用例集的框架，用SC生成每个测试用例中的测试序列.

2.1 使用用例和状态图的构造

2.1.1 *使用用例的构造* 为了生成UC，需面向使用来生成软件系统的动态行为模型，特别是需要描述系统与环境的交互. 因此，需要对基本的UC进行扩充和规格化.

定义1 UC可定义为5元组

$$C_{\mathrm{U}}=\langle I_{\mathrm{UcID}},C_{\mathrm{PreCond}},S_{\mathrm{C}},R_{\mathrm{OtherReq}},S_{\mathrm{PostCondSet}},p_{\mathrm{UC}}\rangle \tag{1}$$

式中：I_{UcID}是用例标识；C_{PreCond}是用例执行的前置条件；S_{C}是与UC相对应的状态图；R_{OtherReq}是针对用例的其他需求，如用例执行的时间要求等；$S_{\mathrm{PostCondSet}}$是后置条件的集合，表示UC在成功执行之后系统进入的状态；p_{UC}是用例执行概率.

建立UC需要软件操作剖面的相关信息，可以通过以下步骤实现.

(1)确定操作发起者. 这相当于构造用户剖面，但只需要得到操作发起者列表，不需要完整的用户剖面.

(2)定义操作模式. 这相当于构造系统模式剖面，而系统模式是系统不同的使用模式或测试所需的不同环境的集合.

(3)创建操作列表. 操作列表是操作剖面中的操作集合，为了该列表，需要先为每一个操作发起者建立一个操作列表，然后合并其为总操作列表. 对操作列表中的每一个操作，用UC的5元组定义进行规范.

(4)确定出现频率. 出现频率即在一段时间内某一操作发生的次数,可依据实际数据,或从相似系统的使用中得到.

(5)确定发生概率. 为了确定操作的发生概率,需将每个操作的出现频率除以总的出现频率. 如果操作的发生概率很低,而其又是关键的操作,则应使用调节系数对该操作的发生概率进行调整,并将最终确定的发生概率替换到相应的 UC 之中.

2.1.2　*状态图的构造*　UC 中的 SC 是描述软件系统动态特性的关键. 为了表述软件操作的实时、并发特性,描述 UC 间的相互调用关系,可以层次状态图为基础,通过改进和扩充使 SC 更适合于软件测试.

定义 2　UC 中的 SC 可定义为 9 元组

$$S_C = \langle S_{StateSet}, S_{TranSet}, \rho, S_{LabSet}, S_0, S_{FinishSet}, S_{EventSet}, S_{CondSet}, S_{ActSet} \rangle \quad (2)$$

式中:$S_{StateSet}$ 是状态集,非空且有限;ρ: $S_{StateSet} \to 2^{StateSet}$ 描述了状态之间的树状层次求精关系;$S_{TranSet} \subseteq 2^{StateSet} \times S_{LabSet} \times 2^{StateSet}$ 是迁移关系;S_{LabSet} 是迁移标记集合;S_0 是初始状态集合;$S_{FinishSet}$ 是终止状态集合;$S_{EventSet}$ 是激励事件集合;$S_{CondSet}$ 是监视条件集合;S_{ActSet} 是动作集合.

$\rho(s)$ 定义了状态 s 的子状态集,$\rho^*(s)$ 代表包含自身状态的子状态集,$\rho^+(s)$ 代表不包含自身状态的子状态集,即 $\rho^+(s)=\rho^*(s)-\{s\}$. 对于每一个状态图,存在一个根状态 $r \in S_{StateSet}$,对 $\forall s \in S_{StateSet}$,$r \notin \rho^+(s)$,即层次结构中不存在环.

设 X 表示迁移的源状态集,Y 表示迁移的目标状态集,L 表示迁移的标记,则对 $\forall t=(X,L,Y) \in S_{TranSet}$,存在 $X, Y \subseteq S_{StateSet}$,$X \neq \varnothing$,$Y \neq \varnothing$,$L \in S_{LabSet}$.

对于迁移 $L \in S_{LabSet}$,有 $L=\langle E[c]/A \rangle$,其中 E 是激励事件,c 是监视条件,A 是动作. c 为真时,E 才能产生动作 A,否则 E 不产生动作. 迁移 L 可以是简单迁移、并发迁移、进入组合状态、从组合状态中出来.

2.2　使用模型和测试模型的导出

2.2.1　*从状态图导出使用图*　将 UML 状态图转换为使用图的关键在于,平展状态图,消除 UML 状态图的层次. SC 的层次关系主要由 OR 状态和 AND 状态描述.

定义 3　对于一个状态 s,其类型可以是基本状态、OR 状态、AND 状态.

当 s 为基本状态时,$\rho(s)=\varnothing$.

当 s 为 OR 状态,且 $\rho(s) \neq \varnothing$ 时,$\rho(s)$ 是状态 s 的 OR 分解,而当对象处于状态 s 时,实际上 $\rho(s)$ 处于 s 的某个子状态.

当 s 为 AND 状态,且 $\rho(s) \neq \varnothing$ 时,$\rho(s)$ 是状态 s 的 AND 分解,而当对象处于状态 s 时,实际上 $\rho(s)$ 处于 s 的所有子状态.

定义 4　对于一个状态 s,进入该状态后将进入的第一个子状态 s' 称为 s 的缺省子状态.

定义 5　如果 $s_1 \in \rho^*(s)$,则称 s_1 是 s 的子孙,s 是 s_1 的祖先. 如果 $s_1 \in \rho^*(s)$ 并且 $s_1 \neq s$,则称 s_1 是 s 的严格子孙,s 是 s_1 的严格祖先.

遵循以下原则可实现由 SC 到使用图的转换.

(1)递归地引入目前已经用替代符号替代的从属状态图,以对 SC 进行平展. 平展从第一个子状态 s' 开始,在更高层图之中,从属状态的初始转换被指向各自状态的弧线所替代,状态标号的应用状态图的名字作为其前缀经迭代后命名,以避免重名. 这样,将得到一个无层次结构的状态框图.

(2)对激励进行标准化.

(3)用带有 ε 的迁移替代自动迁移.

(4)使用图中不应存在不可达或死循环的状态,如果有,需要修正.

2.2.2　*使用图转换为使用模型*　在使用模型中,状态迁移具有概率特征. 为了获得使用模型,需要确定软件预期使用的概率分布,即激励事件 E 发生的概率 $p(E)$ 和监视条件 c 为真的概率 $p(c)$. 对于迁移标记 $L \in S_{LabSet}$,被扩充为 $L=\langle E(p(E))[c(p(c))]/A \rangle$.

为获得迁移概率可采用以下方法.

(1)基于假设的方法. 假定在一个状态下的全部 n 个激励事件具有相同的发生概率 $1/n$,迁移上监视条件为真或假的概率均为 1/2.

(2)基于历史数据的方法. 通过对旧版本软件或相同应用的类似软件的使用情况进行记录和分析,获得软件的历史使用数据,用该数据推算迁移概率.

(3)基于评估的方法. 通过经验丰富的专家或用户对软件未来使用进行分析、预测和评估,推断软件的使用情况,得到迁移概率.

2.2.3　*使用模型的 Markov 链的表示*

定义 6　软件 Markov 链使用模型可定义为 5 元组

$$C_M = \langle S, \Gamma, \delta, q_0, F \rangle \quad (3)$$

式中:S 是软件执行的状态集合;Γ 是迁移标记集合,对于 $\forall t, t \in \Gamma$,则 $t=\langle \text{Name}, p \rangle$,其中 Name 是迁移名称,$p$ 是迁移概率;δ 是一个函数,$\delta: S \times \Gamma \to S$,它可规范执行过程中的软件状态间的迁移关系;

q_0 是初始状态，即软件执行前所处的状态；F 是软件终止执行时所进入的终止状态集合.

将 UC 的使用模型用定义 5 规范，即可生成该用例的 Markov 链使用模型.

2.3 测试用例集生成算法

定义 7 测试用例(TC)集可定义为 5 元组

$$C_T = \langle S_{TcIdSet}, S_{PreCondSet}, S_{SequSet}, S_{PostCondSet}, S_{RuleSet} \rangle \tag{4}$$

式中：$S_{TcIdSet}$ 是 TC 的有序标识集合；$S_{PreCondSet}$ 是 TC 执行的预置条件集合；$S_{SequSet}$ 是 TC 的执行步骤集合；$S_{PostCondSet}$ 是 TC 的预期执行结果集合；$S_{RuleSet}$ 是 TC 的成功与失败的判定准则集合.

有了软件系统扩展的 UC 集、对应的 SC 和 Markov 链使用模型集，利用以下算法可生成可靠性测试用例集.

输入：扩展的 UC 集、对应的 SC 和 Markov 链使用模型集.

步骤 1：初始化，包括：①设定生成总的 TC 数 N；②TC 标识计数器 Count=1；③$C_T=\varnothing$.

步骤 2：按发生概率随机抽取一个 C_U.

步骤 3：提取 UC 信息并按照下列顺序放入 TC 之中，即①按照 TC 的定义，建立一个 t，并初始化；②t. TcId=Count；③t. PreCond=C_U. PreCond；④ t. PostCond = C_U. PostCond；⑤ t. Rule 根据 C_U. OtherReq和 C_U. PostCondSet 提取判定准则.

步骤 4：按下列顺序生成 TC 执行序列，即①取 C_U 对应的 Markov 链的 C_M；②用 C_M 随机生成一个 TC；③取 C_U 对应的状态图 s，对照使用序列中的状态和迁移，从 s 中获取每个使用步骤的精确信息；④将使用序列和使用步骤的精确信息合成，形成测试序列；⑤让 $S_{SequSet}$ 等于测试序列.

步骤 5：将 TC 放入测试用例集

$$C_T = C_T \bigcup \{t\}$$

步骤 6：判断 TC 数是否小于 N，若小于 N，赋 Count=Count+1，上转步骤 2，否则结束用例生成.

3 实例分析

为了验证所提模型的实际效果，在某火控雷达波束调度软件可靠性测试中进行了应用. 该雷达波束调度软件的主要功能是，将雷达的时间资源分配给校正、搜索和特殊模式的波束. 波束调度软件根据操作员的多重要求，在充分利用系统容量、资源配备的条件下，完成相应模式的波束参数和波束指向等计算准备工作，并根据最新一帧发射波束的实际统计数字，对下一帧扫描的时间资源重新分配，将准备好的波束供波束形成单元子系统.

首先，按照模型提供的方法生成 TC，针对波束调度软件生成了 19 个 UC. 为了获得 UC 出现的概率，通过对雷达工作任务记录软件收集到的实际数据进行分析，确定任务出现的频率和用例发生的概率，其中部分用例数据如表 1 所示. 详细的 UC 表述如表 2 所示.

表 1 部分用例数据

用例标识	用例名称	发生概率
BD-001	波束调度初始化	0.01
BD-002	雷达状态描述及转换	0.03
BD-003	波束调度状态管理	0.03
BD-004	惯导数据处理	0.02
BD-005	雷达波束选择	0.12
BD-006	波束产生	0.15
BD-007	波束发送处理	0.12
BD-008	波束发送	0.12
BD-009	填写波束参数	0.12
BD-010	计算生成波束参数	0.03
BD-011	响应通信链路维护信息	0.01
BD-012	更新已有的扇区	0.02
BD-013	插入搜索扇区	0.02

根据不变式、主要成功想定、变异、延伸、包含的 UC 等栏目的内容，生成 SC，再通过对 SC 进行平展处理，获得各 UC 的 Markov 链使用模型.

利用 2.3 节的算法生成 TC，并对软件进行了 3 个周期的测试，每个系统测试周期随机生成了 300 个 TC，共得到 37 个失效数据.

实际应用表明，本模型生成的各测试周期的 TC 集框架基本稳定，具体的 TC 极少重复，所以用它可提高实时控制软件的可靠性测试效率和有效性.

4 结束语

本文提出的基于 UML 模型的可靠性测试用例生成模型，利用了 UML 的 UC 和 SC，将操作剖面模型和 Markov 链模型相结合，并综合了这 2 种模型的优点，克服了操作剖面难以描述软件实时特性的弱点，其 UML 描述过程清晰，比较适合于实时控

制软件可靠性测试用例的开发.

表 2　波束发送处理的 UC

用例标识	BD-007.
用例名称	波束发送处理.
目的	向下一位分系统发送一个可对扫描空域进行对空常规扫描的波束.
参与者	操作员.
前置条件	波束调度处于工作状态，已定义的扇区个数少于最大可定义的扇区数.
退出条件	新定义的对空常规扇区在雷达的可扫描范围内，并且不与其他非对空常规扇区重叠.
不变式	无.
主要成功想定	(1)操作员打开扇区操作对话框. (2)操作员点击“扇区插入”按钮. (3)操作员在“插入扇区”对话框中输入不与其他非对空常规扇区重叠的扇区扫描范围，并确定该扫描范围在雷达可扫描的范围内. (4)操作员点击“确定”按钮. (5)波束调度在下一个雷达工作周期中且当到达新定义扇区的起始扫描位置时，形成波束发射命令并向下一位分系统发送该命令.
其他需求	无.
变异	无.
延伸	(1)波束调度无法发出该波束请求，即波束调度处于非工作状态下接收扇区插入请求，波束调度处于工作态下而输入的扇区扫描范围是非雷达可扫描范围，输入的扇区被其他非对空常规扇区所覆盖，输入的扇区类型是非对空常规扇区. (2)操作员点击“取消”按钮.
所含的 UC	填写波束参数. 发送波束.

参考文献：

[1] Musa J D. Operational profiles in software-reliability engineering [J]. IEEE Trans Software, 1993, 10(2): 14-32.

[2] Woit D. Operational profile specification, test case generation, and reliability estimation for modules [D]. Kingston, Canada: Telecommunications Research Institute of Ontario (TRIO), McMaster University, 1994.

[3] Ouabdesselam F, Parissis I. Constructing operational profiles for synchronous critical software [C]// Proceedings of 6th International Symposium on Software Reliability Engineering. Los Alamitos, USA: IEEE Computer Society, 1995: 286-293.

[4] Elbaum S, Narla L S. A methodology for operational profile refinement [EB/OL]. [2006-01-21]. http://www.cse.unl.edu/elbaum/papers/workshops/rm01.pdf.

[5] Whittaker J A, Poore J H. Markov analysis of software specifications [J]. ACM Transactions on Software Engineering and Methodology, 1993, 2(1): 93-106.

[6] Whittaker J A, Thomason M G. A Markov chain model for statistical software testing [J]. IEEE Transactions on Software Engineering, 1994, 20(10): 812-824.

[7] Wohlin C, Runeson P. Certification of software components [J]. IEEE Transactions on Software Engineering, 1994, 20(6): 494-499.

[8] Shukla R, Carrington D, Strooper P. Systematic operational profiles development for software components [C]// Proceedings of the 11th Asia-Pacific Software Engineering Conference. Los Alamitos, USA: IEEE Computer Society, 2004: 528-537.

[9] Kim Y G, Hong H S, Cho S M, et al. Test cases generation from UML state diagrams [J]. IEEE Proceedings-Software, 1999, 146(4): 187-192.

[10] Hubner M, Philippow I. Statistical usage testing based on UML [EB/OL]. [2006-01-15]. http://www.theoinf.tu-ilmenau.de/~riebisch/publ/SCI2003-paper.pdf.

[11] Riebisch M, Philippow I. UML-based statistical test case generation [C]// International Conference on Objects, Components, Architectures, Services, and Applications for a Networked World. Berlin: Springer-Verlag, 2003: 394-411.

[12] 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.
Yan Jiong, Wang Ji, Chen Huowang. Deriving software Markov chain usage model from UML models [J]. Journal of Software, 2005, 16(8): 1386-1394.

（编辑　苗凌）