第 34 卷 第 11 期 计 算 机 工 程 2008 年 6 月
Vol.34 No.11 Computer Engineering June 2008
·软件技术与数据库· 文章编号：1000—3428(2008)11—0052—03 文献标识码：A 中图分类号：TP311

软件可靠性建模研究

吴 超[1,2]，林家骏[1]，俞 岭[2]，邵玮炜[2]
(1. 华东理工大学自动化研究所，上海 200237；2. 总参通信部驻上海地区军事代表室，上海 200082)

摘 要：依据软件可靠性特征，提出以解决软件开发逻辑思维正确性为建模基本问题的可靠性建模思想。给出基于失效过程的软件可靠性定义，以及一种基于“任务-事件-功能”的软件体系结构准则，进行了形式化描述和理论证明。阐述从功能可靠性、事件可靠性到任务可靠性的分层可靠性预计方法，在建模中融合了软件可靠性设计以及软件测试和管理。

关键词：软件可靠性模型；软件工程；软件测试

Study on Software Reliability Modeling

WU Chao[1,2], LIN Jia-jun[1], YU Ling[2], SHAO Wei-wei[2]
(1. Research Institute of Automation, East China University of Science and Technology, Shanghai 200237;
2. Shanghai Military Deputy Office, Communication Department, General Staff, Shanghai 200082)

【Abstract】 According to software reliability characteristic, a software reliability modeling idea as its basic problem for solving the validity of logical thinking in the software development is put forward. A reliability define based on software invalidating process is brought forward, formalizing description and theoretic proving of a software systematic-frame principle based on “mission-event-function” are put forward. An estimating method of delaminating reliability in the reliability range from functions, events and missions is presented. Software reliability design, test and control are inosculated in modeling.

【Key words】 software reliability model; software engineering; software test

1 概述

如何反映软件在实现其功能时的可靠性组成结构变化对可靠性的影响，是软件可靠性建模亟待解决的问题。基于时域的模型只考虑了软件系统随时间变化的可靠性特性。基于输入域的“黑盒法”模型[1]只考虑软件输入而未考虑软件结构和执行路径。“白盒法”模型[2]则无法描述软件系统的动态过程对可靠性的影响。基于组件的可靠性建模给出了系统可靠性参数的估计方法，但由于军用软件几乎都是新开发的，组件可靠性未知，基于组件建模方法就失去了应用的前提。在经典可靠性理论中，任务阶段性(phased- mission)可靠性建模方法[3]考虑了系统的可靠性组成结构变化对可靠性的影响，但由于尚无实用的软件体系结构分解方法而未在软件可靠性建模中得到应用。

软件产品本质上是一个离散的数学模型，内部逻辑关系具有高度复杂性和抽象性。软件失效是其输入与其运行状态的函数，失效过程取决于软件系统结构和内部逻辑的复杂程度，而这种复杂程度又取决于软件开发人员的思维逻辑正确性。软件可靠性特征决定了用一种分解方法对一个软件进行系统结构分解是可能的，但要适用于所有的软件几乎是不可能的。

笔者认为软件可靠性建模的基本问题是如何保证软件开发逻辑思维正确性。问题的关键是涉及到人和对人的设计行为的管理。因此，应提出一种体系结构准则，以规范和保证软件开发思维逻辑的正确性，并由此作为基于软件可靠性组成结构建模的一个必要前提。一个良好的体系结构准则不仅可以成为可靠性设计的框架，同时也建立了可靠性测试与验证过程的框架，以融合可靠性设计、测试和管理。

2 软件可靠性建模

2.1 基于软件失效过程的可靠性定义

不失一般性，软件系统的输入和使用可设为特定程序的一个输入，它由m个量组成，这些量可以是字符型、数值型、逻辑型或结构型，用$i_1, i_2,\cdots, i_m$表示，这样的一个输入构成m维输入空间A。同样，假设程序的输出由n个量 $o_1,o_2,\cdots,o_n$ 组成，则它构成一个n维输出空间。当给定输入空间的一个量，经程序P计算或处理后，就能找到输出空间的一个量，因此，程序P实质上是输入到输出的一个映射。

由此产生以下 3 种情况：

(1)当唯一确定程序的一个输入空间 A 时，能唯一地映射到一个输出空间 B，如式(1)。

$$O=P(I),\ A\Rightarrow B \tag{1}$$

(2)当程序的输入空间不能唯一地确定时，或超出了预期的范围，程序就不可能正确地映射到所预期的输出空间，即

$$O+O_e=P(I+I_e),\quad A+A_e\Rightarrow B+B_e \tag{2}$$

(3)当程序中存在错误或缺陷时，虽然程序的输入空间能唯一确定，但由于输入空间的某些区域正好对应了程序 P 中隐含的错误代码或执行路径，因此程序也不能正确映射到所预期的输出空间，即

$$O+O_e=P(I),\quad A\Rightarrow B+B_e \tag{3}$$

军用软件面对的是特定的、有组织的用户群，需求清晰，

作者简介：吴 超(1957－)，男，高级工程师、博士研究生，主研方向：软件工程，软件可靠性，软件测试；林家骏，教授、博士生导师；俞 岭，工程师、硕士；邵玮炜，硕士

收稿日期：2007-06-20 **E-mail：**wuch818@163.com

使用要求明确。装备发展的继承性又形成了任务剖面和操作剖面一定的确定性。基于此，本文提出基于软件失效过程的可靠性定义如下：

设A为允许的预期用户操作，A_e为不允许的预期用户操作，用$f_1, f_2, \cdots, f_i$表示一个完成特定任务的装备软件P所具备的有限功能(点)的集合，对应功能点的允许与不允许的用户操作(或输入)是一个有限空间m和n，用$A_1, A_2, \cdots, A_m$和$A_{e1}, A_{e2}, \cdots, A_{em}$分别表示功能点上允许的预期用户操作与不允许的预期用户操作的集合，P及其功能规约B_f是(A, A_e)上的函数，即$B_f = P(A, A_e)$，P_f为(A, A_e)上的一个给定的概率分布。P在概率分布P_f下对应于B_f的可靠性$R(P, B_f)$定义为

$$R(P, B_f) = P_f(\{f \in A, A_e \mid P(A, A_e) = B_f(A, A_e)\}) \tag{4}$$

称其为 P 的可靠性。其失效率 λ_f 为

$$\lambda_f = \sum_{x=1}^{m} \frac{B_f}{A_x} + \alpha \sum_{y=1}^{n} \frac{B_f}{A_{ey}} \tag{5}$$

其中，$0<\alpha\leq 1$，视功能点对完成任务的重要度或危害度而定；

$$B_f = \begin{cases} 1 & B_f = P(A, A_e) \\ 0 & B_f \neq P(A, A_e) \end{cases}, \quad f \in A, A_e \tag{6}$$

显然，通过各功能点之间串联或并联 2 种联接形式，可得出 P 的总失效率 λ。则可靠性 R 为

$$R = 1 - \lambda \tag{7}$$

2.2 基于“任务-事件-功能”体系结构准则

基于失效过程可靠性定义，并根据军用软件特点，本文提出一种基于“任务-事件-功能”的软件体系结构准则，其抽象描述和理论证明为以下内容。

定义 1(有限任务定义) 设软件 S 被要求为完成某特定任务 M 的定制软件，对于 M 存在着若干事件 $x(x \in M)$，使任务 M 成为由 $x_i(i = 1, 2, \cdots, n)$ 组成的有限条件论域 $U, U = \{x \mid M(x)\}$，则称 U 为 M 的任务空间。

定义 2(任务条件定义) 事件 x 是任务 M 在论域 U 中的预期输入及环境条件 I $\{I = \{x \mid U(x)\}\}$，和预期目标 O $\{I \rightarrow O, O = \{x \mid U(x)\}\}$ 及其规约的集合，即 $I, O \in U(x)$。

定义 3(任务成功定义) 设 $I \in U(x)$，$I \in U(x) \Rightarrow O \in U(x)$。若存在 $I_e = \{x \mid x \notin I\}$，产生 $O_e = \{x \mid x \notin O\}$，则 $I \in U(x) \Rightarrow O \in U(x)$ 称为任务成功判据，记为 $I \times O$；而 $I_e \in U(x) \Rightarrow O_e \in U(x)$ 则称为任务失败判据，记为 $I_e \times O_e$。

定义 1 和定义 2 通过对有限任务空间的定义，把软件的输入空间与事件一一对应起来。定义 3 通过定义任务完成与软件输入的关系，把输入空间划分映射到软件系统结构，作为构建软件系统的前提条件。下文给出从任务空间至功能的分解准则。

定义 4(分解第 1 特性) 设 x 是论域 U 的事件，I_a，$I_b \in U(x)$，$I_a \in U(x) \Rightarrow O_a \in U(x)$，$I_b \in U(x) \Rightarrow O_b \in U(x)$。若对于任意 x，都有 $I_a = \{x \mid x \notin I_b\}$，且 $O_a = \{x \mid x \notin O_b\}$ 为分解第一特性。满足此特性即可称为论域 U 中的顶事件，记为 $X(I \times O)$。按此特性构成的软件系统，其结构称为第 1 分解范式，记为 1FD。

定义 5(分解第 2 特性) 设 x 是论域 U 的顶事件，$I_a, I_b \in U(x)$，$I_a \in U(x) \Rightarrow O_a \in U(x)$，$I_b \in U(x) \Rightarrow O_b \in U(x)$。若 $I_a - I_b = \{x \mid x \in I_a \wedge x \notin I_b\}$，且 $O_a - O_b = \{x \mid x \in O_a \wedge x \notin O_b\}$，为分解第 2 特性。满足此特性的 $I \times O$ 称为归集于 x 的功能，记为 $F(I \times O)$。按此特性构成的软件系统，其结构称为第 2 分解范式，记为 2FD。

定义 6(分解第 3 特性) 设 $F(I_a \times O_a) \in U(x)$，$F(I_b \times O_b) \in U(x)$ 分别属于论域 U 中的顶事件 x 的功能要求，有 $I_a - I_b = \{x \mid x \in I_a \wedge x \notin I_b\}$。若 $\exists O_a = \{x_a \mid x_a \in O_a\} \wedge \exists I_b = \{x_b \mid x_b \in I_b\}$，使得 $(x_a \Rightarrow x_b) \wedge O_b = \{x_b \mid x_b \in O_b\}$，则称 $F(I_a \times O_a)$ 与 $F(I_b \times O_b)$ 具有分解第 3 特性，即 $F(I_b \times O_b)$ 的实现依赖于 $F(I_a \times O_a)$，或称 $F(I_b \times O_b)$ 是 $F(I_a \times O_a)$ 的子功能。按此特性构成的软件系统，其结构称为第 3 分解范式，记为 3FD。

定义 7(功能模块定义) 设 $F(I \times O) \in U(x)$，程序 P 是 $F(I \times O)$ 的软件实现，即 $O = P(I)$，则 F 的实现程序 P 及其集合称为功能模块(或功能组件)。

定义 8(软件错误定义) 设 $I \in U(x)$，$I_e = \{x \mid x \notin I\}$，$I_e \in U(x) \Rightarrow O_e \in U(x)$，若 $(O, O_e) = P(I, I_e)$，则对应于 $I_e \times O_e$ 的实现程序 P 称为软件错误。

定义 4~定义 6 给出了从顶事件至功能剖面的分解准则。定义 8 在定义 7 的基础上给出了软件错误与功能模块和输入的关系，并逐一上溯到功能剖面和任务剖面，形成从功能独立、程序独立到软件错误独立的可靠性聚集和故障因素分离的结构特征。

软件系统结构是一种设计，它包括：由任务剖面确定软件预期功能；由任务独立来划分软件输入空间；由预期功能和输入空间构建功能模块和设计程序。下文给出软件体系结构准则的递归定理。

定理 1(任务空间划分定理) 一个完成特定任务 M 的软件 S，一定存在着有限个数的符合 1FD 的顶事件，构成有限条件论域 U 的任务空间，诸顶事件在任务空间中时序和其概率，构成软件 S 的任务剖面。

定理 2(功能空间划分定理) 顶事件一定包含了有限个数的符合 2FD 的功能和符合 3FD 的子功能，诸功能的有序集合和其执行概率，构成软件 S 的功能剖面。

定理 3(软件系统结构定理) 基于任务剖面的软件 S 的体系结构，是顶事件、功能和子功能的有序集合，且符合软件体系结构的 1FD, 2FD 和 3FD，而具有以上特征的软件系统结构，可称为具有结构的正则性。

定理 4(可靠性特征定理) 符合结构正则性的软件系统结构，一定具有故障因素分解和可靠性聚合的特征。

定义 9(任务可靠性定义) 设软件 S 被要求完成某特定任务 M，并可通过任务空间划分得出任务剖面 m，软件 S 的体系结构具备正则性。软件 S 的任务可靠性 $R(m)$是指：软件系统在规定的任务剖面 m 中完成规定功能的能力(概率)。

与通常情况不同，定义 9 是以规范软件开发的思维逻辑正确性为基础，以具有可靠性内聚和故障因素分离的系统结构为前提。

2.3 可靠性结构

图 1 给出了基于体系结构准则的软件系统结构框架。这个框架的构建始于软件需求，包括任务剖面、使用环境与要求等。

在基于体系结构准则中，每个事件是若干个功能的有序集合，由功能到功能模块是一种设计。当程序执行在功能模块中的驻留时间远远小于系统的失效间隔时间时，可按并行

系统结构进行分析[4]。通常基本配置结构有 3 种：串联实现配置，并联实现配置和*K*/*n*实现配置。基于任务剖面的软件系统结构框架如图 2 所示。并联实现配置与*K*/*n*实现配置是一种冗余设计方法，可通过某种方法转换为串联实现配置，如图 2(c)所示。

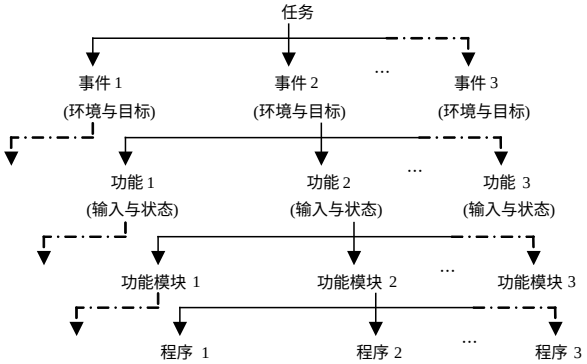

图 1　基于任务剖面的软件系统结构框架

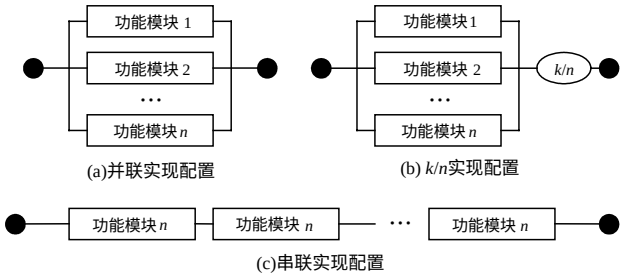

图 2　可靠性结构

在串联实现配置中，功能实现由*n*个相互独立的功能模块完成，其中任何一个功能模块失效都将导致该功能失效。设R_i为功能模块*i*的可靠度，则可靠度为

$$R_f = \prod_{i=1}^{n} R_i \tag{8}$$

符合体系结构准则的软件其输入域独立，因此，功能模块失效可视为常数。设 λ_i 为功能模块 *i* 的失效率，则失效率为

$$\lambda_f = \sum_{i=1}^{n} \lambda_i \tag{9}$$

2.4　可靠性预计

符合体系结构准则的软件具备以下特征：(1)软件功能与其预期使用条件或环境相对应；(2)软件是在预期使用条件下或环境中运行；(3)对程序中错误的检测是相互独立的；(4)所有检测到的错误就能从系统中排除；(5)软件失效率正比于程序中残留的错误数。

基于体系结构准则，功能*f*的输入域*E*划分为有限个独立子域 $E_i\{i=1,2,\cdots,M\}$，$E=\bigcup_{i=1}^{M} E_i$；同时求得 E_i 的邻域 $E_j\{j=1,2,\cdots,N\}$，使 $E_i \cap E_j = \phi, i \neq j$ 作为可能有的软件错误输入空间。根据E_i上的概率分布P_i，在E_i和E_j中随机各选取m_i与n_j个测试用例运行程序，若分别发生f_i与f_j个失效，其功能失效率为

$$\lambda_f = \sum_{i=1}^{M} \left(\frac{f_i}{m_i}\right) P_i + \sum_{j=1}^{N} \left(\frac{f_j}{n_j}\right) P_i \tag{10}$$

则功能 *f* 的可靠性估计值为

$$\hat{R}_f = 1 - \lambda_f \tag{11}$$

功能剖面已知，事件 *x* 的可靠性估计值为

$$\hat{R}_x = \prod_{i=1}^{n} \omega_{fi} \hat{R}_{fi} \tag{12}$$

其中，*n* 为实现事件 *x* 所需的功能数；$\hat{R}_{fi}$ 为第 *i* 功能模块的可靠性估计值；ω_{fi} 为第 *i* 功能在功能剖面中的执行概率。

任务剖面已知，软件系统的任务可靠性估计值为

$$\hat{R}_s = \prod_{i=1}^{m} \omega_{xi} \hat{R}_{xi} \tag{13}$$

其中，*m*为任务剖面中的事件数；$\hat{R}_{xi}$ 为第 *i* 事件的可靠性估计值；ω_{xi} 为第 *i* 事件在任务剖面中的执行概率。

2.5　分析和初步评价

本文提出的软件可靠性建模方法的特点如下：

(1)从功能、事件到任务的可靠性分层预计，工程概念清晰，公式简单明了，便于实施；反映了软件系统的可靠性动态结构对可靠性的影响，克服了软件输入“组合爆炸”带来测试和预计的困难。

(2)融合了可靠性设计与可靠性测试，分层测试和预计过程的同时也是对可靠性设计正确性、完备性和测试充分性的一种度量。

(3)把软件需求作为系统结构的组成部分和设计输入，结合分层测试与预计方法，建立了使用方和开发方在军用软件立项论证、可靠性设计、测试和验证等过程中各负其责、共同管理的基础。

3　结束语

可靠性问题的关键是工程问题[5]，本文与偏重于数学建模的不同点如下：

(1)依据软件可靠性特征，提出“以解决软件开发逻辑思维正确性为建模基本问题”的可靠性建模思想；根据军用软件特点，提出了基于软件失效过程的可靠性定义。

(2)基于可靠性定义，提出“任务-事件-功能”体系结构准则，规范并保证软件开发思维逻辑的正确性，以可靠性内聚和故障因素分离的软件系统结构，作为基于软件可靠性组成结构建模的必要前提。

(3)基于体系结构准则，给出了从软件功能、事件到任务的分层可靠性预计方法，建立了对可靠性设计正确性和测试充分性的一种度量。

本文提出的遵循“可靠性是设计出来的”工程思想，结合专业领域，从工程出发，综合考虑可靠性设计、测试和管理，是软件可靠性建模研究与应用的一种有益尝试。

参考文献

[1] Basitani F B. Input-domain-based Models for Estimating the Correctness of Process Control Programs[C]//Proc. of the International School of Physics. Amsterdam, Holand: [s. n.], 1986.

[2] Shooman M L. Statistical Computer Performance Evaluation[M]. New York, USA: Academic Press, 1972

[3] Dugan J B. Automated Analysis of Phased-mission Reliability[J]. IEEE Trans. on Reliability, 1991, 40(1): 45-53.

[4] 黄锡滋. 软件可靠性、安全性与质量保证[M]. 北京: 电子工业出版社, 2002.

[5] Musa J D. 软件可靠性工程[M]. 韩　柯, 译. 北京: 机械工业出版社, 2003.