基于构件的软件可靠性分析*)

樊林波[1,2] 吴 智[1] 赵 明[1]

(贵州大学贵州省可靠性工程研究中心 贵阳 550001)[1](遵义师范学院计算机科学系 遵义 563002)[2]

摘 要 现代软件工程的一个重要目标是实现聚合性和重用性,构件技术就是以此为目标的。随着该技术的快速发展,以构件设计复杂软件系统的软件开发方法日趋成熟。但是基于该技术的软件可靠性分析却比较落后。目前,在这领域的研究中,多数是使用软件可靠性增长模型(SGRM),对于软件系统的可靠性预测研究较少。在已有的文献中,给出了基于组件的具有层次结构的系统可靠性预测分析,但没有考虑模块之间循环调用的可靠性问题。循环调用在软件中是经常发生的事件,对软件系统的可靠性具有举足轻重的作用。本文在对已有模型的不足进行分析后,增加了循环调用的可靠性预测分析,建立了一个较为全面的系统可靠性预测分析模型。

关键词 构件,软件可靠性,依赖关系,可靠性预测,循环调用

Software Reliability Analysis of Component Based Software

FAN Lin-Bo[1,2] WU Zhi[1] ZHAO Ming[1]

(Reliability Engineering Center of Guizhou Province, Guizhou University, Guiyang 550001)[1]

(Department of Computer Science, Zhunyi Normal College, Zhunyi 563002)[2]

Abstract One of the most important goals of modern software engineering is to realize its aggregation and reusability. Component-based technology is one way to achieve this. With the rapid development of this technology, the component-based technology becomes quite mature. However, its reliability analyzing technology drops far behind. At present, in this research field, the widely used approach is SGRM. Little attention is on the study of software reliability prediction. Literature presents one component-based with hierarchical structure method for system reliability predication analyzing. But it does not consider the reliability of cycle calls among modules. As we know, cycle calls are prevalent in software, and it plays a critical role for the reliability of software system. In this paper, we analyze the existent models, augment reliability prediction analyzing for cycle call and build up one full-scale systematic reliability prediction analyzing model.

Keywords Component-based technology, Software reliability, Reliability prediction, Dependent relation

1 引言

随着软件构件技术的快速发展,以构件设计大型复杂软件系统的软件开发方法日趋成熟。但是,目前的工作更多地集中在构件的开发和构件的复用技术等方面,对利用构件组成的软件系统的可靠性和质量方面则关注较少[3]。在当前,对软件系统的可靠性评估模型中,要么没有考虑预测,通过测试后进行可靠性增长评估;要么是对预测模型约束条件苛刻,与实际工程技术相差较大。本文在对已有模型的不足进行分析后,增加了循环调用的可靠性预测分析,建立了一个较为全面的系统可靠性预测分析模型。这模型更符合实际工程应用。

2 对已有模型的分析

2.1 评估模型概述

在已有的软件系统评估模型中,一类是在系统开发完成、经过测试和实际运行之后才进行可靠性评估的模型,称为软件可靠性增长模型(SGRM)[1];另一类是在软件开发阶段,或者说是在软件未测试前对软件可靠性的预测,称为软件可靠性预测模型[4]。

第一类,也是目前使用得较多的软件可靠性增长模型(SGRM),它是以软件运行和测试期间得到的失效数据为输入,应用统计和随机过程的理论和方法,对软件中剩余的缺陷数目进行评估,并且预测软件在未来一段时间内的可靠性增长情况;其中包括文[5]给出的根据测试覆盖度来评估软件可靠性增长;文[5,6]给出的通过程序中数据流的正则性进行测试,并使用该正则性准则来评估软件可靠性。

第二类,有文[4]给出的RBD模型以及其扩展的RBD和Criticality模型,该模型研究了层次型软件模块之间的依赖关系;文[7]给出的层次型软件系统的可靠性评估模型;文[8]给出的构件概率迁移图模型,该模型使用了概率转移状态机来对软件可靠性进行评估,主要是动态分析;文[3]在文[4,7]的基础上进行了改进,定义了一个依赖关系函数,用来对层次结构模型的可靠性进行评估。

2.2 RBD和Criticality模型[4]及其不足

RBD and Criticality模型定义了一个函数 UTILIZES (m_i, m_j, X_{mi-mj}),该函数定义为如果 m_i 调用 m_j 时,如果 m_j 失效,则 m_i,以 X_{mi-mj} 的概率失效。该函数描述了一个模块调用另一个模块后,其可靠性预测。该模型是对原有的RBD模型的改进,在原有的RBD模型中,定义了USES函数,形如:

USES(*Modules* L_1, *Modules* L_{01})

USES(*Modules* L_1, *Modules* L_{02})

…

USES(*Modules* L_1, *Modules* L_{0n})。

其中 L_1 为父模块,$L_{01}, L_{02}, \cdots, L_{0n}$ 为 L_1 的子模块。该模型是以如下假设为前提:

*)国家自然科学基金项目(60473054)。樊林波 博士生,讲师,研究方向:软件工程。吴 智 硕士,研究方向:软件可靠性。赵 明 教授,博士生导师,研究方向:软件工程、可靠性工程。

1)同层模块是 s-independent 的,任何模块的失效概率不受其他模块失效概率影响。即有{∀(i,j)∈M,P(i | j)=P(i)},其中,M 为系统所有模块的集合;i 和 j 为系统的 2 个不同的模块。

2)失效(failure)只限于 1 个模块内部,即当 1 个模块的子模块失效时,它不会把失效传给父模块。

3)如果 1 个父模块不存在 fault,那么它调用的所有子模块也都不存在 faul。

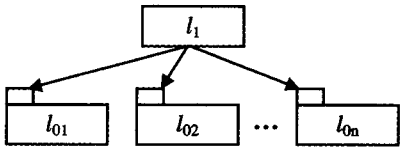

图 1 层次型软件结构

但是改进后的 RBD 和 Criticality 模型,仍然基于这一假设,即同层模块是 s-independent 的,任何模块的失效概率不受其他模块失效概率影响,表示为{∀(i,j)∈M,P(i | j)=P(i)},其中,M 为系统所有模块的集合,i 和 j 为系统的 2 个不同的模块,结构如图 1,这不符合实际的软件系统。

2.3 RBD 和 Criticality 模型的改进及其不足

文[2]在文[4]的基础上进行了改进,改进后不要求文[4]的假设。定义了 Depends 函数,用以约束层次内部模块之间的依赖关系。Depends(m_i,m_j,P_{mi-mj})表示模块 m_i 以概率 P_{mi-mj} 依赖于模块 m_j。Depends 函数定义在同一层次的模块之间,当 $P_{mi-mj}=0$ 时,m_i 和 m_j 相互独立,即 m_i 不依赖于 m_j;当 $P_{mi-mj}=1$ 时,m_i 完全依赖于 m_j,即当且仅当 m_i 正确时,m_j 正确;当 $0<P_{mi-mj}<1$ 时,P_{mi-mj} 给出了 m_i 和 m_j 之间的依赖程度。该模型描述了在同一层次的单向依赖,并在保证不存在循环依赖的情况下,给出了系统可靠性的计算公式。该模型如图 2,计算公式如下:

假设模块的依赖关系如下,

$Dep(L_{01})=\{L_{02},\cdots,L_{0i}\}$, $i\leqslant n$

$Dep(L_{0i})=\{L_{0i+1},\cdots,L_{0k}\}$, $k\leqslant n$

…

$Dep(L_{0n})=\Phi$

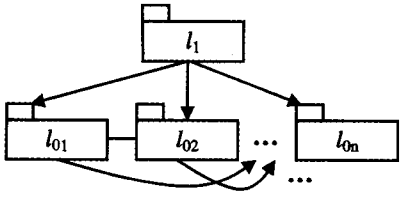

图 2 带有依赖的层次图

根据上述依赖关系,有:

$R(S)=P(L_1)[R(L_{01})+R(\sim L_{01})\cdot P(L_1|\sim L_{01})]\cdot\cdots\cdot[R(L_{0i})+R(\sim L_{0i})\cdot P(L_1|\sim L_{0n})]$

$R(L_{01})=P(L_{01})\cdot[R(L_{02})+P(L_{01}|\sim L_{02})]\cdot\cdots\cdot[R(L_{0i})+P(L_{01}|\sim L_{0i})]$

…

$R(L_{0i})=P(L_{0i})\cdot[R(L_{0i+1})+P(L_{0i}|\sim L_{0i+1})]\cdot\cdots\cdot[R(L_{0k})+P(L_{0i}|\sim L_{0k})]$

…

$R(L_{0n})=P(L_{0n})$

通过循环迭代解这个概率等式方程组,可计算出系统的可靠性。

本文中,作者排除了循环调用问题,但在实际软件工程中,循环调用问题经常出现,本文增加了循环调用的可靠性预测分析,建立了一个较为全面的系统可靠性预测分析模型。这样模型更符合实际工程应用。

3 基于构件软件的可靠性

构件是指封装了数据和功能、在运行时能够通过参数进行配置的模块。软件可靠性指的是在一段特定的自然单元或者时间间隔内,系统无失效运行的概率。

3.1 问题的说明

(1)本文是对基于构件的软件系统可靠性预测研究,不是对组件和模块组成的系统研究,原因有以下几点:1)构件技术发展较快,开发方法日趋成熟,符合现代软件工程思想。2)本文的目标是对系统开发完成或者开发中对系统的可靠性预测,预测的根据是基于该软件所用的基础部件,和部件的组织结构,要求部件的可靠性是已知的。3)构件通常是由第三方开发,除具有清晰的接口描述和常见的功能描述外,还应有表明该构件使用的场合和可靠性等性能指标。这符合 2)的部件选择。同时也是组件和模块一般不具有的。

(2)基于构件的软件系统,选择构件时一般要分析使用构件的中间件功能和构件使用的环境,本文对系统可靠性的研究,是基于中间件和使用环境都完全可靠的情况下进行的。在实际应用中应考虑它们的可靠度。

3.2 基本定义

定义 1(依赖关系) 设 $A=\{C_1,C_2,\cdots,C_n\}$ 是一个软件系统中所有构件的集合,$C_1,C_2,\cdots,C_n$ 是构件。$D\subseteq A\times A$ 是该软件系统中所有具有依赖关系的构件组成的集合,$A\times A$ 为 A 上的笛卡儿积。依赖是指处理事件一构件要完成它的功能或功能的一部分,必须依靠另一构件的功能或部分功能。例如,$D=\{\langle C_1,C_2\rangle\}$ 表示系统中只有构件 C_1 依赖于 C_2,注意,关系$\{\langle C_1,C_2\rangle\}$和$\{\langle C_2,C_1\rangle\}$表示不同的依赖。

定义 2(依赖程度) 如果构件 C_i 依赖于构件 C_j,假设 C_j 失效,就导致 C_i 以 P_{ij} 的概率失效,则我们称 P_{ij} 为构件 C_i 依赖构件 C_j 的依赖程度。

定义 3(循环依赖) 如果构件 C_i 完成其功能,需要借助构件 C_j 的功能,而构件 C_j 的该功能又要依赖于 C_i 的功能,则称这种依赖为循环依赖。注意:循环不是无限次的,因为构件功能的完成是基于有限次的循环,否则,系统进入死循环。

符号规定 $R(S)$表示系统可靠性,R_i 表示 C_i 构件的可靠性,R_{ij} 表示构件 C_i 调用构件 C_j 后的可靠性,$R_{[i,j,i]}$ 表示 C_i 调用 C_j,C_j 又调用 C_i 的一次循环调用后的可靠性。

3.3 可靠性计算公式

(1)"叶结点"调用计算,"叶结点"是指它不调用其他构件的构件或构件的组合。如 C_i 调用 C_j,而 C_j 不再调其他构件,或它对其他构件的调用结束。则调用后结点的可靠性计算如下:

$R_{ij}=R_i\cdot(R_j+(1-R_j)(1-P_{ij}))$。

(2)"带依赖的层次型构件"的调用计算,"带依赖的层次型构件"是指一个构件调用它"下层"的几个构件,"下层"构件中又存在调用关系,但不存在循环调用,如图 2 的情形。

假设 C_i 调用 C_j,C_i 又调用 C_k,而 C_j 又调用 C_k。

$R_{ijk}=R_i\cdot(R_{jk}+(1-R_{jk})(1-P_{ij}))\cdot(R_k+(1-R_k)(1-P_{ik}))$,

其中,$R_{jk}=R_j\cdot(R_k+(1-R_k)(1-P_{jk}))$

(3)带循环的型调用计算,带循环调用的可靠性计算是将一次循环的可靠性计算出来,n 次循环调用的可靠性是一次循环的可靠性的 n 次幂。在一个不可靠的系统里循环调用多

次，结论的可靠性将越来越低。

为了计算多次循环调用的可靠性，必须将一次循环调用的可靠性计算出来。一次循环调用的可靠性可将环撤为线性处理。如图3，可将图3化为图4处理。

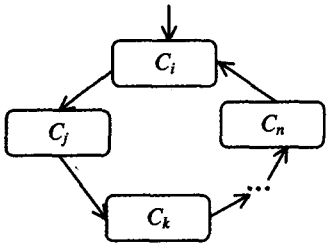

图3 循环调用图

C_i 调用 C_j，C_j 调用 C_k，…，C_n 又调用 C_i。一次调用后的可靠性为：

$R_{[i,j,\cdots,n]}=R_i\cdot(R_{[j,\cdots,n]}+(1-R_{[j,\cdots,n]})(1-P_{ij}))$

$R_{[j,\cdots,n]}=R_j\cdot(R_{[k,\cdots,n]}+(1-R_{[k,\cdots,n]})(1-P_{jk}))$

…

$R_{in}=R_n\cdot(R_i+(1-R_i)(1-P_{ni}))$。

如果是 n 次循环，可靠性是 $R_{[i,j,i]}$ 的 n 次幂。对于单个构件的自身调用和超过二个模块的循环调用，可是用该方法的推广。

(4)一个基于构件组成的软件系统，可把它分解为以上三种“部件”的组合，每种“部件”的可靠性是可计算的，则组合后系统的可靠性 $R(S)$ 是可以计算的，可根据系统中构件组成结构，运用以上三种运算的组合从“叶结点”一步一步向上计算。该方法适合静态分析。下面用一个简单例子来说明该方法的应用。

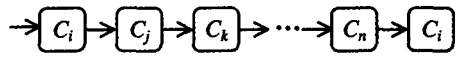

图4 将图3转化为线形结构

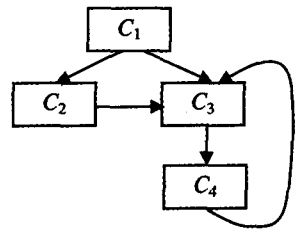

图5 综合调用图例

4 举例

假设图5是一个基于构件的软件结构图，该图中包含了循环和层次调用，我们假设调用 $C3$，则 $C3$ 必调用 $C4$，同时 $C4$ 必调用 $C3$。并且，每次调用都产生2次循环。这样我们可按如下顺序计算 $R(S)$。先计算 R_{43}，再计算 $R_{[3,4,3]}$，再计算 $R_{[3,4,3]}$ 的平方，这时就变成层次模型了。数据假设如表1。

表1 基本数据假设表

可靠性	$R_1=0.9980$	$R_2=0.9975$	$R_3=0.9985$	$R_4=0.9980$	循环次数2次
依赖程度	$P_{12}=0.2500$	$P_{13}=0.3000$	$P_{23}=0.1500$	$P_{34}=0.5000$	$P_{43}=0.3500$

$R_{43}=R_4\cdot(R_3+(1-R_3)(1-P_{43}))$

$=0.9980\cdot(0.9985+(1-0.9985)(1-0.3500))$

$=0.9979$

$R_{[3,4,3]}=R_3\cdot(R_{43}+(1-R_{43})(1-P_{34}))$

$=0.9985\cdot(0.9979+(1-0.9979)(1-0.5000))$

$=0.9974$

$(R_{[3,4,3]})^2=0.9974\cdot0.9974=0.9948$

$R_{23}=R_2\cdot((R[3,4,3])^2+(1-(R_{[3,4,3]})^2)(1-P_{23}))$

$=0.9975\cdot(0.9948+(1-0.9948)(1-0.1500))$

$=0.9967$

$R(S)=R_1\cdot(R_{23}+(1-R_{23})(1-P_{12}))\cdot((R_{[3,4,3]})^2+(1-(R_{[3,4,3]})^2)(1-P_{13}))$

$=0.9980\cdot(0.9967+(1-0.9967)(1-0.2500))\cdot(0.9948+(1-0.9948)(1-0.3000))$

$=0.9956$

由于每个构件的可靠性较，所以系统可靠性较高。

结束语 本文给出了基于构件的软件系统的可靠性预测方法，通过已知的构件的可靠性和构件之间的依赖关系，就可计算出该系统的可靠度，是一个可以在一个软件系统未经测试前，就可对其可靠性预测的模型。但本文是基于连接构件的中间件(或称为胶合逻辑和代码)，构件的使用环境完全可靠的基础上进行的，在实际软件系统中这是不能忽视的。从文章可看出系统的可靠性与系统的结构也有很大关系，对要更多地了解这些信息的读者可参考文[9～12]。

参 考 文 献

1 John D M. 软件可靠性工程[M]. 韩柯译. 北京：机械工业出版社，2003

2 柯尧，赵保华. 基于组件系统的可靠性分析[J]. 北京邮电大学学报，2005，28(6)：116～117

3 Goseva-Popstojanova K, Trivedi K, Mathur A P. How different architecture based software reliability models are related? In: Proc. of the Fast Abstracts 11th IEEE Int'l. Symp. on Software Reliability Engineering (ISSRE 2000). San Jose, California, 2000. http://www. chillarege. com/fastabstracts/issre2000/2000103. pdf

4 Leblanc S P, Roman P A. Reliability estimation of hier. archical software system [A]. In: Proceedings annual Relia. bility and Maintainability Symposium [C], 2002

5 Yashwant K M, Michael N L, JamesM B. Software reliability growth with test coverage [J]. IEEE Transactions on Reliability, 2002, 51(4): 420～426

6 孟洛明，王国伟. 面向 CHILL 数据流测试的研究与试验[J]. 北京邮电大学学报，1994，17(3)：38～42

7 马敏书，张仲义，吕永波. 层次型软件系统可靠性模型及预测[J]. 中国软科学，2003，12(6)：147～150

8 毛晓光，邓勇进. 基于构件软件的可靠性通用模型[J]. 软件学报，2004，15(1)：28～32

9 Goseva-Popstojanova K, Trivedi K. Architecture-Based approach to reliability assessment of software systems. Performance Evaluation, 2001, 45(2-3): 179～204

10 Gokhale S, Lyu M, Trivedi K. Reliability simulation of component based software systems. In: Proc. of the 9th Intl. Symp. on Software Reliability Engineering (ISSRE'98). Paderborn: IEEE Computer Society, 1998. 192～201

11 Krishnamurthy S, Mathur AP. On the estimation of reliability of a software system using reliabilities of its components. In: Proc. of the 8th Int'l. Symp. on Software Reliability Engineering (ISSRE'97). Albuquerque, NM: IEEE Computer Society, 1997. 146～155

12 Gokhale S, Wong W E, Trivedi K, Horgan JR. An analytical approach to architecture based software reliability prediction. In: Proc. of the 3rd Int'l. Computer Performance & Dependability Symp. (IPDS'98). Durham: IEEE Computer Society, 1998. 13～22