面向服务软件中异常处理模块重要性的仿真分析方法

吴 青[1] 应 时[2] 贾向阳[2] 朱小刚[2,3]
(武汉大学教育科学学院 武汉 400072)[1] (武汉大学软件工程国家重点实验室 武汉 400072)[2]
(南昌大学科学技术学院网络与信息中心 南昌 330047)[3]

摘 要 基于蒙特卡洛方法,提出异常处理模块重要性的影响因子,其中包括异常处理模块所对应的保护区的重要性以及异常处理模块自身的稳态故障率。基于贝叶斯理论,计算保护区的重要性;运用马尔可夫链的遍历性计算异常处理模块的稳态故障率。试图提出一种面向服务软件中异常处理模块重要性的定量分析方法,设计人员可以根据分析结果对重要度高的异常处理模块给予更多的关注。

关键词 面向服务软件,软件可靠性,异常处理,蒙特卡洛方法
中图法分类号 TP311.5 **文献标识码** A

Simulation Analysis Method for Importance of Exception Handling Module in Service-oriented Software System

WU Qing[1] YING Shi[2] JIA Xiang-yang[2] ZHU Xiao-gang[2,3]
(College of Education Science, Wuhan University, Wuhan 430072, China)[1]
(The State Key Lab of Software Engineering, Wuhan University, Wuhan 430072, China)[2]
(College of Science and Technology, Nanchang University, Nanchang 330047, China)[3]

Abstract Based on Monte Carlo method, the impact factors of the importance of exception handling module were proposed, which include the importance of its protected areas and its steady-state failure rate. The former is calculateed, based on bayesian theory, while the latter using the ergodicity in Markov chain. The quantitative analysis tries to offer an effective solution. According to the result, designer could pay more attention to the specific module.

Keywords SOA, Software reliability, Exception handling, Monte Carlo method

1 引言

面向服务软件运行环境的动态性和不确定性及服务资源的自治性和松耦合性,使得面向服务软件系统中可能出现的异常情况繁多,且异常处理逻辑十分复杂。由于各异常处理模块对软件可靠性的影响不尽相同,若要提高软件可靠性,需对软件中各个异常处理模块一视同仁,这可能耗费开发人员过多的时间和精力,影响开发进度,因此,有必要对异常处理模块的重要性进行分析,以辅助开发人员对异常处理设计方案进行权衡和决策。然而,在面向服务软件的异常处理领域,现有的研究工作主要集中在异常处理方法[1,2]以及故障推理[3]等方面,对异常处理模块的重要性鲜有涉及。在软件可靠性评估领域,虽有部分工作关注组件重要性,但传统的可靠性模型无法恰当地描述面向服务软件的并发特性,且异常处理模块并非一个普通独立模块,传统的组件重要性评估方法并不适用。

针对上述问题,本文基于蒙特卡洛方法,试图提出一种面向服务软件中异常处理模块重要性的仿真分析方法。从保护区故障而引发系统异常的概率以及异常处理块自身的故障概率两方面,分析异常处理模型的实验数据,定量度量异常处理块的重要性。

本文第2节提出面向服务软件中影响异常处理模块重要性的因子及其求解方法;第3节结合"汽车装配流水线管理系统"展示了仿真分析方法的主要步骤和试验数据;第4节介绍可靠性评估的相关工作,并阐述传统方法的不足;最后对本文工作进行了总结和展望。

2 异常处理模块的重要性分析

2.1 方法概述

目前,面向服务软件中异常处理模块重要性方面的工作并不多见,但部分文献考虑了软件中各组件的重要性分析方法。Gokhale通过方差计算组件的重要性[4],该方法仅考虑组件可靠性大小对系统可靠性的影响。而Siegrist认为每个组件可靠性依赖于组件的平均运行次数[5],提出用偏导数来表示组件的重要性。文献[6]在前两者的研究基础上,提出组件的重要性是指系统可靠性相对于该组件可靠性的变化率,认为组件的可靠性同时依赖于组件运行次数以及组件自身的可靠性。上述方法均需要考虑软件系统的复杂结构,在评估

到稿日期:2012-03-20 返修日期:2012-06-20 本文受国家自然科学基金(61070012)资助。
吴 青(1982—),女,博士,讲师,主要研究方向为面向服务软件集成、Petri网应用,E-mail:wuqingwhu@gmail.com;应 时(1965—),男,博士,教授,博士生导师,主要研究方向为面向对象软件工程方法、基于组件的软件工程方法、软件体系结构和模式、软件的可重用性与互操作性等;朱小刚(1978—),硕士,讲师,主要研究方向为软件工程。

大规模复杂软件系统时，其可操作性有待进一步验证；并且异常处理模块的运行次数与其保护的正常业务模块的可靠性具有直接联系，但上述评估方法并未考虑它们之间的联系。

蒙特卡洛方法即概率模拟法，是一种在传统工业界评估硬件可靠性时广泛采用的仿真分析方法。使用该方法无需过多考虑复杂的系统结构以及状态爆炸的现象。本文基于蒙特卡洛方法的思想，首先利用 Petri 网善于描述系统并发行为的特性，基于着色随机 Petri 网建立面向服务软件中异常处理概率模型，接着利用仿真分析工具 Design/CPN 模拟模型运行，最后分析实验数据得到所求参数的统计特征。本文的研究重点是如何根据实验数据度量异常处理模块的重要性，对异常处理模型不做单独介绍。

本文基于 WSDL 协议描述服务，基于 BPEL 语言定义流程，研究面向服务软件的异常处理逻辑。当服务抛出异常时，本文将服务看作黑盒，捕获并处理服务接口层的异常。当流程抛出异常时，将流程看作白盒，捕获并处理流程内部产生的异常。同时，采用悲观估计的方式来评判系统是否成功执行。为便于后文阐述，给出几个相关的定义。

定义 1 保护区是流程中的某个作用域(Scope)或流程调用的服务，用符号 $pro()$ 表示。

定义 2 异常处理块是某保护区抛出异常后，捕获并处理异常的模块，用符号 $ehb()$ 表示。

异常处理块与保护区之间的关系由符号 Θ^{-1} 表示。例如，$pro(wheels)=\Theta^{-1}ehb(ehwh)$ 表示保护区 $wheels$ 中的异常均由异常处理块 $ehwh$ 捕获和处理。

学术界通常基于正常业务逻辑和异常处理逻辑分离的思想研究异常处理方法，故将面向服务软件中的模块分为保护区和异常处理块。一方面，若系统大部分的异常均来自某保护区，则该保护区是系统中比较脆弱的区域，加强此保护区对应的异常处理块的设计，可以保障该保护区的可靠性，同时较大地提高软件整体的可靠性。另一方面，Cristian 的实验研究表明[7]，异常处理代码比程序中的其他部分更易包含缺陷。所以异常处理块的自身故障率也极大地影响了软件可靠性。

2.2 影响因子的求解方法

2.2.1 保护区的重要性

美国电子设备可靠性顾问团在可靠性指标分配 AGREE 方法中提出，模块的重要度是指某一单元发生故障时对系统可靠性的影响程度。文献[8]提出运用贝叶斯公式计算模块的故障概率。借鉴上述思想，给出保护区重要性的定义。

定义 3 保护区的重要性指在异常处理块不参与修复操作的情况下，保护区 b 发生故障而引发系统异常的概率，用符号 $\psi(b)$ 表示。

$\psi(b)$ 值越大，保护区 b 越可能是系统的故障源，该保护区对软件可靠性影响越大。

本文采用悲观估计，保护区 b 对系统的影响可以分解成保护区对其父模块可靠性的影响与父模块对系统可靠性的影响。这种递归调用的方法消除了其他保护区出错对系统可靠性的影响且便于仿真实现。该方法虽然在理论上其复杂度为 $O(n^n)$，但从实际出发，面向服务软件的层次数量有限，故具有一定可行性。

引理 1 保护区 b 的父模块 $f[b]$ 出现故障，且出错由保护区 b 引发的概率用符号 $\psi(b,f[b])$ 表示。$f[b]$ 若为系统本身，则表示为 $\psi(b)$，即 b 相对系统的重要性。

设 f^x 代表保护区第 x 级父模块，且 f^x 的取值在 f^0 到 f^n 之间，f^0 代表系统本身，$f^n=\Theta^{-1}ehb(b)$，$g(f_i^x)$ 代表 f^{x+1} 作为 f^x 第 i 个模块，相对 f^{x+1} 的重要性。参考一个参数的原始递归式的写法，保护区重要性 $\psi[\Theta^{-1}ehb(b)]$ 的计算公式为：

$$\begin{cases}\psi(f^{x+1},f^x)=\psi(f^x,f^{x-1})*g(f_i^{x+1})\\ g(f_i^x)=P(f_i^{x+1}\mid f^x)=\dfrac{P(f^x\mid f_i^{x+1})*P(f_i^{x+1})}{\sum\limits_{i\in x}P(f^x\mid f_i^{x+1})P(f_i^{x+1})}\\ \psi(f^0)=g(f^0)=\psi(f^0,f^{-1})\end{cases}\tag{1}$$

下面分别讨论式(1)在面向服务软件几种典型的结构中各参数的取值。

(1)顺序结构

在顺序结构的子系统中，各个组件依次执行。因本文采用悲观估计的方式，故顺序结构中任意一个组件的故障均会引发系统故障。故式(1)中 $P(f^x\mid f_i^{x+1})$ 等于 f^x 中 f_i^{x+1} 的发生概率，$P(f_i^{x+1})$ 等于 f_i^{x+1} 在 f^x 中的时间占有率。

(2)并行结构

并行结构指系统中存在并发组件，所有组件均成功运行，整个并行结构才成功执行完毕。所以 $P(f^x\mid f_i^{x+1})$ 取值与顺序结构中相同，$P(f_i^{x+1})$ 等于 f_i^{x+1} 在 f^x 中的时间占有率乘以一个时间权值，该权值代表 f_i^{x+1} 所在分支的运行时间占并行结构运行总时间的比率。

(3)冗余备份结构

冗余备份结构目的在于提高系统的可靠性，只要有一个组件能成功运行就代表系统运行成功。设组件的冗余数为 n，则将冗余组件看作一个整体，整体重要性的 $1/n$ 为冗余备份结构中某一组件的重要性。

(4)其他结构

分支结构指每次运行时，会以一定的概率选择其中一个分支来运行，而循环结构指组件被循环调用。这两种结构可以看作特殊的顺序结构，参数取值与顺序结构相同。

2.2.2 异常处理块的稳态故障率

本节在给出异常处理块的稳态故障率的定义后，对它的存在性给予证明，接着给出求解公式。

定义 4 异常处理块的稳态故障率指异常处理块的执行次数 n 趋于无穷时，异常处理块出现故障的平稳状态概率，用符号 R 表示。

证明：保护区的运行存在正常和异常两种状态，该保护区相应的异常处理块对异常的操作同样存在正常和异常两种状态。这些状态之间的转移仅仅取决于当前时刻的状态，即状态之间无后效性。

由于文献[9]提出基于组件间的状态转移无后效性，因此可以将组件间的状态转移看作一个离散时间的马尔可夫链。故保护区和异常处理块之间的状态转移同样可看作两态的马尔可夫链。

设正常状态为 0，异常状态为 1，该马氏链的转移概率矩阵如下，

$$\begin{pmatrix}P(\Theta^{-1}ehb(b))_{00} & P(\Theta^{-1}ehb(b))_{01}\\ P(ehb(b))_{10} & P(ehb(b))_{11}\end{pmatrix}\tag{2}$$

根据马尔可夫可遍历性公式 $\lim_{n\to\infty} p_{ij}^{(n)} = p_j$ 可知，马尔可夫链经过相当长的时间或相当多的步数后，到达 j 状态的概率接近一个与初始状态无关的常数，即平稳状态概率。因此，将保护区与其异常处理块作为整体模块，该整体模块存在一组稳态成功率和稳态故障率，记作(y_0, y_1)。

又因为，整体模块的稳态故障率指保护区执行时抛出异常，异常处理块并未成功处理该异常的稳态概率。故存在一个异常处理块的稳态故障率，且异常处理块的稳态故障率等同于整体模块的稳态故障率，即 $P(ehb(b))=y_1$。证毕。

公式求解：

由文献[10]的推导可知，平稳状态概率行向量为 $P(n)$，P 为一步转移概率矩阵，且有 $P(n)*P=P(n)$，故有：

$$(y_0, y_1) * \begin{pmatrix} P(\Theta^{-1}ehb(b))_{00} & P(\Theta^{-1}ehb(b))_{01} \\ P(ehb(b))_{10} & P(ehb(b))_{11} \end{pmatrix} = (y_0, y_1) \tag{2}$$

联合公式：

$$y_0 + y_1 = 1 \tag{3}$$

可以求解出 y_0 与 y_1 的值，根据证明可知 $R(ehb(b))=1$。

3 实验与仿真

下面给出汽车装配流水线管理系统(简称 R2E 系统)中的部分功能。假设汽车车架到达流水线中安装车轮的工位，R2E 系统需要获取库存物料数目，即获取车轮的库存量。检查流水线中用于执行安装的设备状态。当安装工作完成后，R2E 系统首先检查采集条码设备状态，选择工作状态良好且当前状态为空闲的采集设备，同时采集车架号和已安装的车轮条码号，若两者型号匹配，则将其信息记录到数据库，车轮安装成功。表 1 列举了上述步骤在实际使用中可能出现的多种异常情况。由此可见，此例中异常处理逻辑远比正常处理逻辑复杂。

表 1 R2E 系统中可能出现的异常情况

操作步骤	异常情况	异常处理
获取物料数量	①数量为零	拒绝后续安装
检查安装设备	②安装设备无应答	检查使用的 web 服务
检查采集设备	③采集设备忙碌	选择另一台设备
采集车轮条码	④未获取到车轮条码	重试 3 次不成功则实例终止
检查型号匹配	⑤型号不匹配	流程实例终止
数据入库	⑥数据库写操作出错	检查数据库连接后重试

运用着色 Petri 网工具 Design/CPN 构建 R2E 系统的异常处理模型，如图 1 所示。该流程存在 sco1 和 sco2 两大保护区。保护区 sco1 中包含获取物料数量变迁 cNum 和检查安装设备变迁 sta。保护区 sco2 是一个替代变迁，其子网如图 2 所示，包括具有冗余备份结构的检查采集设备状态操作变迁 sC1 和 sC2，冗余度为 2；并发执行采集车轮条码变迁 car 和采集车架型号变迁 wh。

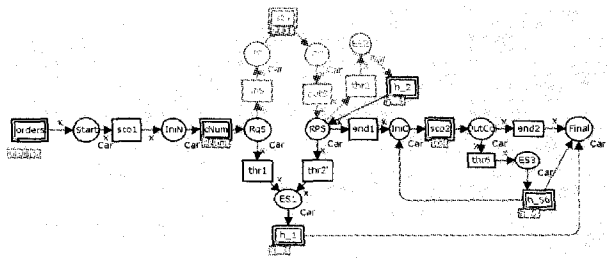

图 1 异常处理模型

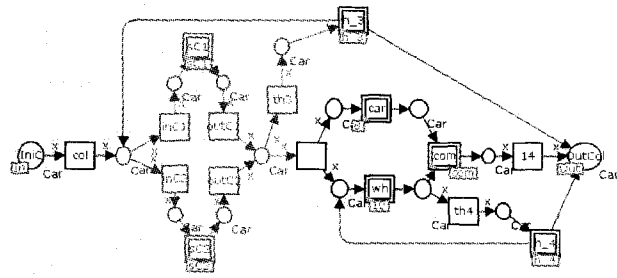

图 2 保护区 sco2

系统模拟运行 2000 次后得到的监控数据如表 2 所列。

表 2 模拟数据

保护区	操作次数	故障次数	修复次数	运行时间
sta	1962	58	45	11375
sc	1958	92	74	7339
wh	1976	86	11	14654
sco1	2000	73	31	18462
sco2	1958	173	36	26568

根据本文提出的方法所得的计算结果如表 3 所列。

表 3 分析结果

异常处理块	保护区重要性	稳态故障率
h_1	22%	15%
h_2	8%	4%
h_3	9%	6%
h_4	32%	24%
h_56	78%	30%

由此可见，系统中的异常 78%来自保护区 sco2，其中采集车架型号操作抛出异常的可能占 32%，上述保护区的重要性远高于其他模块，且其异常处理块的稳态故障率分别为 24%和 30%。故异常处理块 h_4 和 h_56 的重要性相对较高，设计人员应该给予更多的关注。

4 软件可靠性评估的相关工作

目前，软件组件重要性的研究尚不多见，研究主要集中在软件可靠性的评估。但软件可靠性的评估方法对研究异常处理模块重要性分析具有一定借鉴意义，故将软件可靠性评估方法方面的研究工作总结如下。

软件可靠性评估方法主要分为：基于状态的评估方法、基于路径的方法以及基于累加的评估方法。

基于状态的评估通常使用控制流图表示软件架构，把软件的执行过程看成模块间的状态转移过程，然后利用随机过程的理论，如马尔可夫过程，进行分析。其代表方法包括 Littlewood 提出用不可归约的半马尔可夫过程描述软件体系结构[11]，以及 Cheung 运用离散时间马尔可夫链描述由 n 个状态构成的软件体系结构[9]。但这些方法很难描述和分析软件的并行结构，然而异步并发是面向服务软件的基本特征之一，因此传统基于状态的评估方法并不完全适用于面向服务软件的可靠性评估；并且基于状态的评估方法容易发生状态爆炸，大规模复杂系统的可靠性评估并不适用。

基于路径的评估方法通过模拟运行所有可能执行路径，计算程序的可靠性。执行路径是通过测试实验或算法获得，例如 Sherif 提出基于场景的可靠性分析方法[12]。此类方法虽然可以描述和分析软件的并发行为，但它们通常在软件实

(下转第 147 页)

工具，支持对 FSP 进程的分析、图形化显示和组装，并可以对进程进行安全性、活性等性质的检测。但 LTSA 对系统状态采用显式(explicit)表示，验证中占用的存储空间过大。FSP 不支持对时间系统的建模。TCBV 使用并改进了 LTSA 的一些模块如协议编辑、图形化显示等部分来支持对 TBP 的编辑、相容性(安全、活性)的验证，增加了时间行为协议约简、时间属性的描述和可替换性验证等功能。与 LTSA 相比，TCBV 在时间行为协议的状态约简、实时性质验证等方面功能更强大。

TCBV 的设计注重功能完善、简单易用，系统的用户界面简明清晰，窗口功能以及菜单、按钮选项意义明确，力图通过简单的文本输入、编辑和单击操作，就可在工具视图中进行时间行为协议描述和验证；并注重健壮可靠，支持时间行为协议的语法分析，对语法错误在原文本上给出提示，对于用户所有可能的错误输入，能够给出出错信息提示。

结束语 各种复杂 CPS 构件系统在航空航天、医疗、交通等各种高可靠需求领域的应用越来越广泛，对复杂实时构件系统行为进行形式化描述与验证可以提高系统的正确性、可靠性。本文介绍了基于时间行为协议的时序行为形式化建模与组合相容性验证方法，给出了建模与验证工具 TCBV 的框架与功能模块，结合验证工具给出了应用实例。结果表明，基于时间行为协议的建模和验证方法可以对复杂实时构件系统时序行为进行准确的建模，并可以方便地检验系统的各种时序行为错误。在今后，我们将进一步开展更复杂的现实应用系统时序行为的建模、验证工作。

参 考 文 献

[1] 李建中. 信息物理融合系统(CPS)的概念、特点、挑战和研究进展[R]. 2009 年中国计算机科学技术发展报告，2010：1-17

[2] 贾仰理，李舟军，邢建英，等. 基于模型检验的构件验证技术研究进展[J]. 计算机研究与发展，2011，48(6)：913-922

[3] 贾仰理，张振领，李舟军. 构件行为协议实时性扩展及相容性验证[J]. 计算机科学，2010，37(10)：143-1147

[4] Plasil F, Visnovsky S. Behavior Protocols for Software Components[J]. IEEE Transactions on Software Engineering, 2002, 28(11): 1056-1076

[5] He Ji-feng. A Classical Mind: Essays in Honour of C. A. R. Hoare[M]// Roscoe A W, ed. International Series in Computer Science. Prentice Hall, 1994: 171-189

[6] 郭亮，唐稚松. 基于 XYZ/E 描述和验证容错系统[J]. 软件学报，2002，13(5)：913-920

[7] Lynch N, Segala R, Vaandrager F. Hybrid I/O automata[J]. Information and Computation, 2003, 185(1): 105-157

(上接第 138 页)

现后对软件系统的可靠性进行评估，并不适用于软件设计早期对设计方案进行预测和评估。

基于累加的评估方法通过系统测试获得各组件的执行时间和相对使用率，计算单个组件的失效数和失效率函数，该方法并不直接考虑软件系统的构架，而是间接通过组件的可靠性来估计系统架构，如 M.Xie 提出的附加可靠性模型[13]。此类方法同样是在软件实现后对软件系统的可靠性进行评估。

运用蒙特卡洛方法时无需考虑复杂的系统结构，因此它特别适合大规范复杂的面向服务软件系统的建模与分析。本文在运用该方法时，提出使用着色 Petri 网的形式化描述方法取代可靠性方框图。一方面，着色 Petri 网可以清晰地描述面向服务软件中包括异步并发在内的各种复杂结构。另一方面，着色 Petri 网有成熟的工具支撑，可以对各模块的故障情况进行模拟仿真。

结束语 本文利用着色 Petri 网作为面向服务软件异常处理逻辑的形式化描述工具，并借鉴传统的组件重要性评估方法，提出一种面向服务软件中异常处理模块重要性的仿真分析方法，该方法可以为软件优化提供定量分析依据；接着对传统软件可靠性评估方法的特点及其不足进行了分析。

蒙特卡洛方法的采样次数与系统规模无关，它容易处理各种实际运行控制策略，故可以用于大型面向服务软件系统的可靠评估。但蒙特卡洛方法存在计算时间与计算精度的矛盾，即为了获得精度较高的可靠性指标，需要进行长时间的模拟计算。如何利用方差加快蒙特卡洛模拟的收敛速度，提高仿真效率，可以作为今后的研究方向。

参 考 文 献

[1] Carbone M. Session-based Choreography with Exceptions[J]. Electronic Notes in Theoretical Computer Science, 2009, 241(C): 35-55

[2] Diez F J, Maurtua I. Dynamic Exception Handling Based on Web Services and OPC XML-DA[C]// Proceeding of IEEE International Conference on Web Services (ICWS08). IEEE Computer Society, 2008: 593-599

[3] Men Peng, Duan Zhen-hua, Yu Bin. Utilizing Fuzzy Petri Net for Choreography Based Semantic Web Services Discovery[C]// ICATPN 2007. 2007: 362-380

[4] Gokhale S. Quantifying the variance in application reliability[C]// Proceedings of the Pacific Rim Dependability Conference. Papeete, Polynesia, 2004: 113-121

[5] Siegrist K. Reliability of systems with Markov transfer of control[J]. IEEE Transactions on Reliability, 1988, 14(7): 1049-1053

[6] 陆文，徐锋，吕建. 一种开放环境下的软件可靠性评估方法[J]. 计算机学报，2010，33(3)：452-462

[7] Cristian F. Exception handling and tolerance of software faults [J]. IEEE Transactions on Computers, 1982, 31(6): 531-540

[8] 柳永坡，吴际，金茂忠. 基于贝叶斯统计推理的故障定位实验研究[J]. 计算机研究与发展，2010(4)：707-715

[9] Siegrist K. Reliability of systems with Markov transfer of control[J]. IEEE Transactions on Reliability, 1988, 14(7): 1049-1053

[10] 刘延夫. 基于马尔可夫分析方法的软件系统可靠性研究[D]. 长春：长春理工大学，2006

[11] Littlewood B. Software reliability model for modular program structure [J]. IEEE Transactions Reliability, 1979, 28(3): 241-246

[12] Sherif M. Scenario-based reliability analysis of component-based software[C]// 10th International Symposium on Software Reliability. Boca Raton, Florida, 1999: 22-31

[13] Xie M, Wohlin C. An additive reliability model for the analysis of modular software failure data[C]// Proceedings of the Sixth International Symposium on Software Reliability Engineering (ISSRE'95). 1995: 188-194