文章编号:1000-582X(2010)02-0062-07

分布式实时系统的软件故障注入

徐光侠[1a,2],陈蜀宇[1b],常光辉[1a],刘宴兵[2],刘国良[1a]
(1. 重庆大学 a. 计算机学院;b. 软件学院,重庆 400044;
2. 重庆邮电大学 软件学院,重庆 400067)

摘 要:针对分布式实时系统的可信验证的难题,建立通用故障模型,将故障模型分为:内存故障、CPU 故障、通信故障和服务故障 4 种。提出一种建立在分布式实时系统环境中的软件故障注入系统结构,它分为 3 个层次:目标系统、通信网络、软件故障注入系统。软件故障注入系统分 3 个部分进行设计,软件故障注入器、数据收集模块和故障数据分析模块。对基于该结构的软件故障注入过程进行了说明,实现软件故障注入系统并做了相应的实验分析,实验检测到故障多数为通信故障、内存故障和 CPU 故障,其覆盖率分别为 37.68%、15.47%和 15.17%。实验证明这种体系结构很适合分布式实时环境的应用,同时也为进一步研究软件可信验证提供了理论基础和实例依据。

关键词:分布实时系统;系统可信验证平台;软件故障注入;故障模型
中图分类号:TP302 文献标志码:A

Software implemented fault injection for distributed real-time systems

XU Guang-xia[1a,2] *CHEN Shu-yu*[1b] *CHANG Guang-hui*[1a] *LIU Yan-bing*[2] *LIU Guo-liang*[1a]
(1 a. College of Computer Science; b. College of Software, Chongqing University, Chongqing 400044, P. R. China;
2. School of Software Engineering, Chongqing University of Posts and Telecommunications, Chongqing 400065, P. R. China)

Abstract: Aiming at the problem of dependability validation in the distributed real-time systems, the universal fault model is established, which is classified into four groups: memory fault, CPU fault, communication fault, and service fault. A software implemented fault injection architecture (SWIFIA) for the distributed real-time systems is proposed, which is classified into three levels: target system, communication network, and software implemented fault injection system (SWIFIS). The SWIFIS is designed with three parts: software implemented fault injector, data collection module, and fault data analysis module. The process of software implemented fault injection based on the architecture is illustrated. SWIFIS is implemented and analyzed with experiments. The major faults detected in the experiments are communication faults, memory faults, and CPU faults while the coverage rates were 37.68%, 15.47%, and 15.17%, respectively. The experimental results demonstrate that this architecture is suitable for the applications under distributed real-time environment. They offer theoretical base and evidence for further research of software dependability validation.

Key words: distributed real-time system; system dependability validation platform; software implemented fault injection; fault model

---

收稿日期:2009-10-12
基金项目:教育部新世纪优秀人才支持计划(NCET-04-0843);科技部国际科技合作项目(2007DFR10420);国家自然科学基金资助项目(60973160);重庆市自然科学基金资助项目(CSTC,2008BB2307)
作者简介:徐光侠(1974-),女,重庆邮电大学副教授,重庆大学博士研究生,主要从事可信计算、分布式计算方向研究,(Tel)13638325460;(E-mail)xugxia@163.com。

在分布式的网络环境中构筑起来的实时系统称为分布式实时系统[1]。实时系统的运行不仅要求逻辑上的正确性，同时要求满足时限[2]。系统的失效会带来较为严重的后果，分布式实时系统的设计者面临着一个主要问题就是对系统可信性的评价与度量。迄今，已经提出许多方法用以评价和度量系统可信性，如形式化方法、分析建模、仿真等[3]，Chillarege 也提出了“失效加速”的概念[4]。故障注入是“失效加速”的一种实现方法，它已成为验证系统可信机制的有效方法。目前，文献[5-10]对故障注入的研究很多，文献[11-12]硬件故障注入已有许多的成熟模型和实验系统，文献[13-15]在分布式实时系统的软件故障注入工具方面做了讨论和创新，文献[16-21]就软件故障注入的生存性、设置断点以及分布式可信计算的框架和编程模型等进行了探讨，但专门对分布式实时系统的软件故障注入结构模型进行讨论的论文并不多。笔者旨在设计通用的分布式实时系统的软件故障注入体系结构，通过软件故障注入技术以加速系统中节点的故障、差错或失效的发生，建立分布式实时系统的可信验证平台。

1　软件故障注入

计算机中的故障是按照时间特性和输出特性进行分类的。如果一个故障在物理上产生错误，称它是活跃的，否则，称它是良好的。按时间特性可以分为 3 种故障类型：永久性的、间歇性的和暂时性的。永久性故障不会随时间的推移而消除，直到故障部分地被修复或替换，故障才可能消除；间歇性故障介于故障活跃和故障良好的状态间反复转换；暂时性故障会在一段时间之后消失[22-23]。如上所述，有关故障分类的状态如表 1 所示，故障分类的状态转换框图如图 1 所示。在表 1 中，$a(t)$、$b(t)$、$c(t)$和 $d(t)$是故障转换状态的频率，t 是发生故障的时间。

表 1　故障分类的状态

故障类型	状态
永久性	$a(t)>0, b(t)=c(t)=d(t)=0$
暂时性	$a(t)>0, b(t)=0, c(t)>0, d(t)=0$
间歇性	$a(t)>0, b(t)>0, c(t)=0, d(t)>0$

当系统的一部分出现故障或者差错时，如果没有检查出来，就会迅速扩散，因此采用故障注入技术加速故障的发生，以达到对系统中采用的可信性机制进行验证的目的。故障注入技术包括硬件故障注入、软件故障注入和模拟故障注入[22]。基于硬件的方法在目标系统的硬件中注入物理故障；模拟的方法用目标系统的一个模型来注入模拟故障；而软件实现的方法用程序来模拟系统中硬件或软件的故障和错误，被称为软件实现的故障注入（software implemented fault injection，SWIFI）[23]。软件故障注入对目标系统造成一定的影响，但与其他故障注入技术相比有明显的优势，例如，它的开发成本较低廉、开发过程较简单、容易收集注入信息、可移植性较好、并且几乎不会对目标系统的硬件产生任何的损伤[24]。

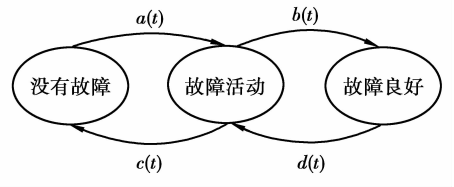

图 1　故障分类的状态转换

2　故障模型

2.1　软件可靠性模型

软件可靠性模型用于预测软件产生错误的速率，通过在每个调试阶段对软件可靠性提供一些指导，确定何时终止调试[22]。它可以分成 4 类，即失效间隔时间模型、缺陷计数模型、差错插入模型和基于输入域模型。其中缺陷计数模型关心的是在特定的时间间隔内，软件的失效数或差错数。该类模型由于不断被排除，在每单位时间内所发现的失效数不断减少，典型代表有 Shoman 模型、Goel-Okumoto 模型和 NHPP 模型等。而差错插入模型是将一组已知的差错，人为地插入到一个固有错误数尚未知的软件中，然后在软件的测试中观察并统计发现插入的错误数和软件的错误数，通过估计软件的总固有错误数来进行软件可靠性及其他有关指标的评价。这种模型的代表有 Niller 模型和 Basin 模型[25]。软件可靠性模型是软件可靠性定量分析的基础，可靠性是衡量软件可信的重要特征之一。对软件可靠性模型研究有助于为软件故障注入的失效加速时间特性的正确评价与设置提供参考，有关的典型模型如下。

Jelinski-Moranda 模型[22]：该模型认为错误产生率同软件中的故障数目成比例。即 $\lambda(i)=Ji$，其中 $\lambda(i)$表示有 i 个故障的软件故障率，J 是未知参

数。如果软件中存在 N_0 个故障，软件运行了 t 个单位时间，那么没有出现错误的概率为 $P_0=1-\exp(-\lambda(N_0)t)$。该模型明显的缺点就是假设故障率同系统中留存的故障数据量成比例。

Goel-Okumoto 模型：该模型假设错误产生是一个速率为 $\lambda(t)=abe^{-bt}$ 的非齐次泊松过程，其中 a 为最终观察到的失效期望值，b 为对每个错误的发现率。此模型假定在某随机时刻由于系统中的软件错误引起软件失效。它将软件的固有错误视为随机变量，其观察值与测试和其他环境因素有关，认为从第 $i-1$ 次错误到第 i 次错误发生的间隔时间依赖于第 $i-1$ 次错误发生的时间[25]。

据统计，目前软件的可靠性模型已有 100 余种。文献[26]提出一种具有实时特征的分布式软件可靠性评估方法。文献[27]给出了一种基于软件体系结构的高可信软件可靠性测评框架。文献[28]对分布式实时系统的可信性研究现状与存在的问题进行了综述。文献[28]作者还定义了分布式系统的可靠度，其表达式为

$$R_s=R_c\cdot R_p=\Pr\{\bigcap_{i=1}^{m}(\bigcup_{j=1}^{k_i}I_{i(j)})\}\cdot \Pr\{\bigcap_{i=1}^{m}(\prod_{j=1}^{a_i}M_{i(j)})\}。\quad(1)$$

目前，国内外学术界对分布式系统、实时多任务系统的可靠性模型已有深入研究，但对分布式实时系统的可靠性建模的研究还需深入。

2.2 分布式实时系统的故障模型

故障模型是在一定的系统层次上对目标系统真实故障的抽象。系统通过软件的方法，根据故障模型在故障注入时刻实现对特定故障的模拟。故障模型一般用故障的公共属性来表征[29]，故障实例是指故障模型中的每一个属性都取定值时所对应的故障，通常采用故障的 4 个属性来描述目标系统的故障模型，即故障位置、故障类型、故障持续时间和故障注入时刻[30]。

对故障模型的分类方法很多[3,31]，将分布式实时系统的故障模型初步分为以下几种：内存故障、CPU 故障、通信故障和服务故障。在故障注入时可以选择以上几种模型的任意组合作为注入条件。表 2 对每一种故障类型进行了说明。在注入故障时也能加强系统级差错注入的性能，如加快或放慢进程、终止或挂起进程、腐蚀时钟/定时器服务等[3]。

表 2 通用故障模型

模块名称	故障分类	故障模型	发生位置/故障描述
内存故障	一位	设置	堆栈
	两位	重置	全局变量
	单字节	触发	用户代码
	多字节	用户定义	操作系统核心部分
CPU故障	一位	设置	数据寄存器
	两位	重置	地址寄存器
	单字节	触发	栈指针
	字	用户自定义	程序计数器
	用户自定义		状态寄存器
通信故障	消息丢失	故障的链接选择	数据丢失
	消息重写	故障维度选择	内容改写，损坏
	消息更改	更改位置	数据发送错误
	消息延迟	更改运行	接收不及时
	用户自定义	延时控制	时间控制
服务故障	服务生存期	服务的建立、服务的撤消	服务的生命周期

1）内存故障

修改内存内容是一种基本的技术，它用于软件故障注入器。故障可能污染内存的某一部分，故内存故障不但能够表现 RAM 差错，也能仿真系统其它部分的故障发生。尽管内存故障模型比较复杂，但是一些故障仍然可以采用一种极小或非常规的方式来影响系统内存的内容。脱离内存故障注入而单独的仿真一个故障行为是很困难的，因此需要一个更复杂的故障模型。

内存故障注入可以注入一位、两位、单字节或多字节，也可以部分或全部设置、重置、触发被选中内存的内容。确定被污染内存的位置和选择故障类型很重要，用户可以明确指定故障注入位置，也可以随机选择物理内存空间。内存故障注入手段是必要的，如用户程序段、用户堆/栈或者系统软件区域的故障注入。

2）CPU 故障

CPU 故障发生在数据寄存器、地址寄存器、数据单元、控制寄存器、操作代码译码单元、ALU 等。在独立体系结构层中，选择仿真 CPU 故障的结果，例如，控制流可能被总线差错改变，指令译码逻辑差错、条件代码标志差错或者控制寄存器差错。

3)通信故障

在分布式实时系统中通信故障可能引起消息丢失、改变、复写或延迟。在系统中，消息可能间歇性丢失，或是交替丢失，即每一个消息在确定时间段内被交替地丢失。与内存故障相似，消息也会被改变，即一位、两位或者多字节的突发性差错。在故障注入时，用户可以指定差错是注入到消息体或是消息头。对于延迟信息，延迟时间可以是一定的，也可能呈现一定的或然分布。预先确定通信故障类型的集合后，用户也可以定义附加的通信故障。这些故障可以是预先定义故障类型的组合，也可以是基于个体消息的内容或者基于过去的消息历史记录等。通信失效的多样性，表现在将现在故障类型和定义新故障类型合并的能力方面，要考虑到失效语义的多样性注入，包括拜占庭失效[3]。

4)服务故障

服务故障主要由服务的建立与撤消引起的，还要涉及服务的生命周期。

3　软件故障注入方案

3.1　软件故障注入结构

笔者提出了一个分布式实时系统的软件故障注入体系结构(software implemented fault injection architecture of distributed real-time systems，SWIFIADRS)，如图 2 所示。其中，软件故障注入系统提供产生负载、注入控制、注入代理等功能，分布式实时系统的处理器结点与系统通信网络连接，通过互联网连接到故障注入系统结点。软件故障注入系统的服务器独立于目标系统，在目标系统中，故障注入代理(fault injection agent，FIA)采用子代理的形式在各结点上进行故障注入，可增加 SWIFIADRS 的轻便性，由于目标系统的每个结点是相对独立运行的，有利于减小目标系统整体运行效能的冲突。

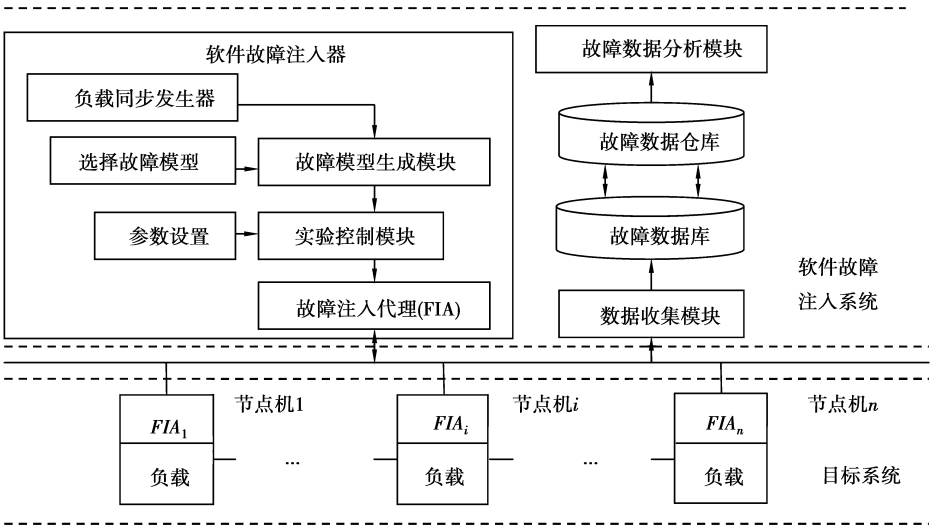

图 2　故障注入体系结构

SWIFIADRS 分 3 个层次，目标系统、通信网络、软件故障注入系统。软件故障注入系统分为 3 个部分，其中，软件故障注入器包含的模块有负载同步发生器、选择故障模型、故障模型生成模块、参数设置、实验控制模块和 FIA；数据收集模块在各结点运行时收集实验数据；故障数据分析模块分析实验完成后的离线数据。

在软件故障注入器中，负载同步发生器提供同步发生的不同实验负载。选择故障模型模块主要作用是故障模型的选择和组合。故障模型的特性包括类型、持续时间、位置、时刻等，故障空间是这些特性的笛卡尔积[30]。故障模型生成模块根据所选择的故障模型和负载同步发生器产生的同步信号生成故障模型负载。故实验控制模块经某些参数设置以管理和强化故障注入实验的执行。

3.2　注入过程

在 SWIFIADRS 注入过程中，首先对软件故障注入系统进行初始化，再对故障模型进行单独设置或组合设置，使负载同步发生器发生负载同步发生信号，故障模型生成模块生成将要注入的故障序列，其中故障序列选取有三种方法：一是穷尽故障空间的所有点；二是简单随机抽样法；三是方差减小技术，如分层抽样、分级抽样以及故障扩展等，通常采用方法三[32]。然后，经过参数设置，实验控制模块管理与强化控制故障序列，由故障注入代理将故障

序列分派至目标系统各结点的子代理，子代理将其注入到目标系统，最后将从数据收集模块收集到的数据送到故障数据分析模块进行分析。完整的软件故障注入过程如图3所示。

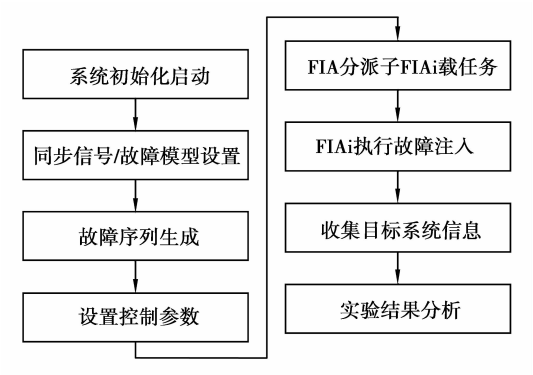

图3 软件故障注入过程

在故障注入过程中，故障注入实验与工作负载是完全透明的。每个有特殊工作负载的故障注入实验称为一次运行，判定故障数据分析结果质量的是故障注入实验中所收集到的运行数据，因此，自动多线程运行实验是非常有用的。实验控制中的关键涉及进程同步和重新初始化问题。重新依赖于完成每个负荷后目标系统的状态，在某些情况下，有必要重启动整个系统，而有时只需重启目标系统。

4 注入实验与结果分析

目前软件故障注入系统的主要模块与数据收集模块的功能已基本实现，该系统具有注入位置、故障模型和注入时间等可随意选择的特点。应用该系统对某能量管理系统进行了分布式实时故障注入实验。该系统包括能量支撑系统和电网分析系统两大部分，支撑系统由实时数据库管理系统、人机交互系统、进程管理系统和网络通信系统4部分构成，用Visual C++语言编程实现，硬、软件具有扩充性。实验环境：该系统使用TCP/IP协议进行网络通信，数据采集器将采集的数据传送到服务器。该系统运行的操作系统采用RTLinux硬实时操作系统，实验过程中所选用的时间片为20 ms。

为保证实验的有效性，将每两次故障注入的时间间隔设置足够长，以保证系统已恢复正常的运行状态。目标系统的节点机操作系统不仅负责正常运行能量管理系统任务，还要运行故障注入子代理(FIA_i)，进行故障注入与定时自检，一旦检测到故障，数据收集模块就进行相应的数据处理。

注入实验过程中，注入次数设定为8 000次，其中有效注入5 887次，占注入总数的73.59%，无效注入2 113次，占注入总数的26.41%，注入实验过程中故障触发条件采用的是定时触发，即利用预先设定时间的定时器来触发故障。它适用于瞬时故障和间歇性故障，缺点是不能重现故障注入，可能产生不可预料的故障影响和系统行为。故障注入实验中故障分布及检测机制检测到的各种故障类型的覆盖率分别如表3和表4所示，其中，故障诊断的覆盖率α指系统中可正确诊断的故障数N_1占可能发生的总故障数N的百分比[25]

$$\alpha = \frac{N_1}{N} \times 100\%。 \tag{2}$$

表3 故障分布

类型	故障数目	所占百分比/%
已检测有效故障	4 359	54.49
未检测有效故障	1 528	19.10
无效故障	2 113	26.41
合计	8 000	100

表4 故障注入实验统计结果

位置属性	电网分析系统	实时数据库管理系统	人机交互系统	进程管理系统	网络通信系统	故障总数	覆盖率/%
内存故障	154	108	143	382	124	911	15.47
CPU故障	267	29	95	399	103	893	15.17
通信故障	1 067	84	64	147	856	2 218	37.68
服务故障	74	31	89	50	93	337	5.72
故障总数	1 562	252	391	978	1 176	4 359	74.04

从实验的结果可以得出，除去无效故障，有效故障的检测覆盖率为74.04%。不过，有25.96%的故障没有覆盖，原因主要有以下几点：1)受RTLinux操作系统的影响与限制，部分检测已将个别故障进行屏蔽，如系统中异常处理模块。2)部分故障已被能量管理系统的纠错和校验方式自行纠正。3)系统运行过程中被注入的故障数据从未被使用。4)软件容错机制对于故障处理有待进一步完善。

实验中的大多数故障为通信故障、内存故障和CPU故障，其覆盖率分别为37.68%、15.47%和

15.17%。检测到通信故障最多，这与目标系统是有关系的，由于能量管理系统为分布式实时系统，且通信处理功能复杂、任务频繁，因此，可信验证平台监视通信任务的运行过程中，检测出较多的通信故障。本应该检测出更多的内存故障和 CPU 故障，实验中只有 911 次和 893 次，这主要是由于部分故障被 RTLinux 系统的纠错机制所纠正与屏蔽。服务故障次数为 337，次数较少主要与目标系统是有关。

5 结　论

给出了课题组研究提出的 SWIFIADRS，它弥补了已有故障注入模型中的不足，为软件可信验证建立了一种形式化的理论框架，并搭建了平台。实验证明，SWIFIADRS 能够准确描述软件可信验证的基本原理，在所选定的目标系统上进行了试验性运行，该验证平台的运行效果良好，其应用具有较强有效性和普遍适用性。同时也为下一步研究故障诊断专家系统课题和进一步研究软件可信验证提供了理论基础和实例依据。

参考文献：

[1] 库劳里斯，多利莫雷，欣德贝里著.（金蓓弘译）. 分布式系统概念与设计(第 4 版)[M]. 北京：机械工业出版社，2004.

[2] AIDEMARK J，FOLKESSON P，KARLSSON J. A framework for node-level fault tolerance in distributed real-time systems [C]// Proceedings of the International Conference on Dependable Systems and Networks. Yokohama，Japan：[s. n]，2005，6.

[3] SEUNGJAE H，KANG S G，HAROLD A. ROSENBERG. DOCTOR：an integrated software fault injection environment for distributed real-time systems[C]// In IEEE Int. 7 Computer Performance and Dependability Symposium，1995：204-213.

[4] BURNS A，WELLINGS A. Real-Time systems and programming languages：ada 95，real-time java and real-time POSIX [M]. Beijing：China Machine Press，2001.

[5] SIEWIOREK D P，杨孝宗，CHILLAREGE R，等. 可信计算的产业趋势和研究[J]. 计算机学报，2007，30(10)：1645-1661.
SIEWIOREK D P，YANG XIAO-ZONG，CHILLAREGE R，et al. Industry trends and research in dependable computing [J]. Chinese Journal of Computers，2007，30(10)：1645-1661.

[6] LÓPEZ-ONGIL C. A unified environment for fault injection at any design level based on emulation[J]. IEEE Transactions on Nuclear Science，2007，54(4)：946-950.

[7] 王建莹，孙峻朝，杨孝宗. 一种用于容错计算机系统整体验证的故障注入试验策略[J]. 计算机研究与发展，2001，38(1)：61-67.
WANG JIAN-YING，SUN JUN-ZHAO，YANG XIAO-ZONG. An experimental strategy of fault injection for the whole validation of fault tolerant computer systems [J]. Journal of Computer Research and Development，2001，38(1)：61-67.

[8] MAISTRI P，VANHAUWAER P，LEVEUGLE R. Evaluation of register-level protection techniques for the advanced encryption standard by multi-Level fault Injections [C]//Proceeding of the 22nd IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems. Rome：[s. n]，2007：499-507.

[9] BENSO A et al. A functional verification based fault injection environment [C]//Proceedings of the 22nd IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems. Rome：[s. n]，2007：114-122.

[10] GAWKOWSKI P，SOSNOWSKI J. Dependability evaluation with fault injection experiments[J]. IEICE Trans Inf Syst，2003 E86-D(12)：2642- 2649.

[11] TAEGHYOON K，SUNGMOON C，DOHOON L. Effective fault injection model for variant network traffic [C]//Proceedings of 2007 International Conference on Convergence Information Technology. Gyeongju：IEEE，2007：1189-1194.

[12] FIDALGO A V，ALVES G R，FERREIRA J M. Real time fault injection using a modified debugging Infrastructure [C]//Proceedings of the 12th IEEE International On-Line Testing Symposium (IOLTS06). Italy：[s. n] 2006：242-250.

[13] HOARAU W，TIXEUIL S. A Language-Driven Tool for Fault Injection in Distributed Systems [C]// Proceeding of the 6th IEEE/ACM International Workshop on Grid Computing. Seattle. Washington：[s. n]，2005：194-201.

[14] GALLA T M，HUMMEL K A，BURKHARD P. Exploiting mobile agents for structured distributed software-implemented fault injection [C]//HICSS'06：Proceedings of the Annual Hawaii International Conference on System Sciences. United states：[s. n]，2006(1)：4-7.

[15] 董剑，曲峰，刘宏伟，等. 嵌入式故障注入器 HFI-4 的研究与设计[J]. 小型微型计算机系统，2006，24(12)：2335-2337.
DONG JIAN，QU FENG，LIU HONG-WEI，et al.

Study and design of embedded fault injector[J]. Mini-micro Systems, 2006, 24(12):2335-2337.

[16] SMAILI I, ADEMAJ A. Setting break-points in distributed time-triggered architecture[C]//Proceedings of the Seventh IEEE International High-Level Design Validation and Test Workshop. Cannes, France: [s. n], 2002, 57-62.

[17] VOAS J M, GHOSH A K. Software fault injection for survivability[C]//Proceedings of DARPA Information Survivability Conference and Exposition. South Carolina:[s. n], 2000:338-346.

[18] GORENDER S, MACEDO R J D A, RAYNAL M. An adaptive programming model for fault-tolerant distributed computing [J]. IEEE Transactions on Dependable and Secure Computing, 2007, 4 (1): 18-31.

[19] KLONOWSKA K et al. Extended golomb rulers as the new recovery schemes in distributed dependable computing [C]//Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium. Denver, USA:[s. n] 2005:279.

[20] MOTET G, GEFFROY J C. Dependable computing: an overview[J]. Theoretical Computer Science, 2003, 290(2): 1115-1126.

[21] 孙峻朝,李运策,杨孝宗. 故障注入方法与工具的研究现状[J]. 宇航学报, 2001,22(1):99-104.
SUN JUN-ZHAO, LI YUN-CE, YANG XIAO-ZONG. A theory framework of fault injection study[J]. Mini-Micro Systems, 1999, 20(11):816- 819.

[22] KRISHNA C M, KANG G SHIN. 实时系统[M]. 戴琼海,译. 北京:清华大学出版社,2004.

[23] ROSENBERG H A, SHIN K G. Software fault injection and its application in distributed systems[C]//Proceedings of the 23rd International Symposium on Fault-Tolerant Computing. Toulouse, France:[s. n], 1993:208-217.

[24] 彭俊杰,黄庆成,洪炳熔,等. 一种用于星载系统可靠性评测的软件故障注入工具[J]. 宇航学报, 2005,26(6): 823-827.
PENG JUN-JIE, HUANG QING-CHENG, HONG BINGRONG, et al. A software fault injection tool for evaluation of the dependability of onboard system[J]. Journal of Astronautics, 2005, 26(6): 823-827.

[25] 李海泉,李刚. 系统可靠性分析与设计[M]. 北京:科学出版社,2003.

[26] 金海,谢夏,李运发,等. 一种具有时间约束的分布式软件可靠性评估方法[J]. 计算机研究与发展,2004, 41(2),311-316.
HAN ZONG-FEN, LI YUN-FA, XIE XIA, et al. A reliability Evaluation of time-constrained distributed software [J]. Journal of Computer Research and Development, 2004,41(2),311-316.

[27] 覃志东. 高可信软件可靠性和防危性测试与评价理论研究[D]. 四川:电子科技大学,2005.

[28] 胡华平,金士尧,王召福. 分布式系统的可信性研究[J]. 计算机工程与科学,1998,20(1):48-53.
HU HUA-PING, JIN SHI-YAO, WANG ZHAO-FU. Dependability study of distributed computer systems[J]. Computer Engineering & Science, 1998,20(1):48-53.

[29] WASZNIOWSKI L, KRÁKORA J, HANZÁLEK Z. Case study on distributed and fault tolerant system modeling based on timed automata [J]. Mini-micro Systems, 2009,10(82):1678-1694.

[30] 蒋支运,陈欣. 软件实现的无人机故障系统[J]. 哈尔滨工业大学学报, 2006,38(11):1993-1995.
JIANG ZHI-YUN, CHEN XIN. Fault injection system based on software for unmanned aerial vehicle [J]. Journal of Harbin Institute of Technology, 2006, 38(11):1993-1995.

[31] DREBES R J. A Kernel-Based communication fault injector for dependability testing of distributed systems [C]//Proceedings of the First International Haifa Verification Conference. Haifa, Israel:[s. n], 2005: 177-190.

[32] 王建莹,杨孝宗,徐海智. 用软件实现的故障注入工具评估错误检测机制[J]. 小型微型计算机系统, 2000, 21(5): 497-499.
WANG JIAN-YING, YANG XIAO ZONG, XU HAI-ZHI. Evaluation of error detection mechanisms using software implemented fault injector [J]. Mini-micro Systems, 2000, 21(5): 497-499.

(编辑 侯 湘)