|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **序号** | **时间** | **作 者** | **文 章 题 目** | **发 现 方 法** | **应 对(预防) 方 法** | **关 键 词** | **不 足** |
| **0** | 2002 | M. Shaw, S. D. Gribble | Reverse Firewalls in Denali | 阻止本机向未知的或未授权的地址发起连接 |  | reverse firewall | 不能区分正常的应用（P2P应用会向未知的地址发起连接）和间谍软件的区别[] |
|  | 1994 | Yennun Huang, P. Jalote | Two techniques for transient software error recovery |  |  | 隐蔽性故障 |  |
|  | 1994 | David Lorge Parnas | Software Aging | 该文认为软件产品衰退不可阻止，是由需求变更引起的。软件产品的功能退化了，在不断的时间紧迫的升级、打补丁过程中，软件中遗留了一些errors。描述了软件衰退的起因、代价、预防软件退化的措施， | 预防措施：执行软件工程流程：良好的设计(灵活、预计可能的需求变化)、详尽的文档、软件评价、截肢(去掉频繁、草率的修改而陈旧的代码)、重构 |  | 主要针对软件产品功能的退化,草率的应对需求变更，引起代码质量下降；软件工程做的再好，仍不可避免会衰退 |
|  | 1995 | Yennun Huang, C. Kintala | [Software Rejuvenation：Analysis Module and Applications](../00-参考论文/(1995)%20Software%20Rejuvenation：Analysis%20Module%20and%20Applications（隐蔽性软件故障）.pdf) | 单个资源使用情况阈值，达到阈值，执行重启 |  | 不确定性因素，隐蔽性故障elusive bug，瞬时失效transient failure |  |
|  | 1995 | Garg. S | Analysis of software rejuvenation using Markov regenerative stochastic Petri net | 早期基于度量的方法。 | 基于建模技术的性能预测：建立性能分析模型，预测最优抗衰实施模型 | Markov，Petri net |  |
|  | 1996 | 1. Pfening, S. Garg, A. Puliafito, K.S. Trivedi | Optimal Software Rejuvenation for Tolerating Soft Failures |  |  |  | 现有的大部分方法都缺乏固定的/一致的控制基础[14] |
|  | 1997 | 1. Avritzer, E.J. Weyuker | Monitoring Smoothly Degrading Systems for Increased Dependability |  |  |  |  |
|  | 1998 | Garg. S | analysis of preventive maintenance in transactions based software systems | 最完善的基于时间的方法。 | 基于建模技术的性能预测： | time-based | 固定周期执行，不顾当前实际状态，代价大 |
|  | 1998 | Garg. S | A methodology for detection and estimation of software aging | 最早基于度量的方法。 | 基于趋势分析的性能预测方法。利用发生故障时所采集的数据进行分析，然而 | Measure-based | 不确定性大，方法不具备普适性。对于软件衰退评估和性能预测来说，仅利用故障数据是不够的[1] |
|  | 1999 | K Vaidyanathan, K.S. Trivedi | A measurement-based model for estimation of resource exhaustion in operational software systems | 基于度量的方法。 | 基于趋势分析的性能预测 | semi Markov |  |
|  | 2000 | T Dohi, K．Goseva-Popstojanova, K．S．Trivedi | Statistical non-parametric algorithms to estimate the optimal software rejuvenation schedule |  | 基于趋势分析的性能预测 |  |  |
|  | 2001 | V. Castelli, R.E. Harper, P. Heidelberger, K.S. Trivedi, K. Vaidyanathan | Proactive management of software aging |  |  |  |  |
|  | 2002 | Li L., K. Vaidyanathan, K.S. Trivedi | An Approach for Estimation of Software Aging in a Web Server |  | 基于趋势分析的性能预测 |  |  |
|  | 2003 | Y. Bao, X. Sun, K.S. Trivedi | Adaptive Software Rejuvenation: Degradation Model and Rejuvenation Scheme |  |  |  | 现有的大部分方法都缺乏固定的/一致的控制基础[14] |
|  | 2003 |  | An overview of software cybernetics（控制学） | 原来研究者认为软件与硬件不同，不会老化，但研究表明软件也会老化，本文提到软件老化不容忽视，尤其是网络环境下的软件，其老化会导致服务器挂起或崩溃；概述了潜在的导致软件老化的原因；作者提出现有的大部分方法都缺乏固定的/一致的控制基础，作者提出疑问：软件抗衰是一个软件控制学问题？如何适应性地执行抗衰？ | 定期或在软件服务挂起或崩溃前实施抗衰；研究内容包括：实时监测软件当前状态、收集信息、决定何时和软件的哪部分应该抗衰。服务的性能和质量对网络软件来说很重要，网络软件也应该调整它的运行和服务情况，以与网络流量的变化一致。 | Software rejuvenation adaptively | 综述性论文，提出软件抗衰是潜在的研究方向，未提出确切的方法 |
|  | 2004 | Wei Xie,Yignang Hong, K.S. Trivadi | Software Rejuvenation Policies for Cluster Systems under Varying Workload． |  | 基于建模技术的性能预测： |  |  |
|  | 2004 | M. Malek E Salfner, G.A. Hoffmann | Self Rejuvenation：An Effective Way to High Availability |  | 基于建模技术的性能预测： |  |  |
|  | 2006 | R. Matias Jr. , Paulo J. F. Filho | An Experimental Study on Software Aging and Rejuvenation in Web Servers | 衰退对web server的影响 |  |  |  |
|  | 2006 | 徐建 | 软件抗衰若干关键技术研究(PhD thesis) |  |  |  |  |
|  | 2007 | 徐建 | 软件抗衰研究综述 |  | 预测软件性能：认为时序分析方法从本质上讲都是线性模型，且统计学方法大都需要一些难以证明的显眼知识。故一些较为复杂的时间序列(如软件的性能预测数据)，这类模型的预测结果不能满足应用的需求。 | Heisenbug(瞬时性、隐蔽性bug) | 常见的清理内部状态的方法：清空缓存、内存垃圾收集，重新初始化内核表、清理文件系统、收集碎片等 |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

软件抗衰相比软件冗余等事后容错方法，是一种预防性的容错方法；它也被称为“软件再生”。

1. 软件衰退发现与应对要解决的本质问题
2. 现有方法已解决的本质问题
3. 未来研究尚需要解决的关键问题
4. 软件老化潜在的原因

软件老化潜在的原因有内存泄露、未释放的文件锁、未终止的线程的累积、数据越界错误的增长、文件系统碎片、共享内存池封锁（死锁？）等等[14]

1. 徐建，软件抗衰若干关键技术研究，2007