

Proactive Management of Software Systems

Kishor S. Trivedi

Center for Advanced Computing and Communication

Dept. of Electrical and Computer Engineering

Duke University

Durham, NC 27708-0294

Phone: (919) 401-0299 x306

Fax: (919) 401-5589

kst@ee.duke.edu

1 Introduction

With the explosive growth in Internet technology and the emergence of a number of new and advanced applications, assured availability of computer systems has become a critical issue. The challenge is to provide the desired availability and performance at a low cost. Computer system manufacturers like SUN, IBM, HP and Microsoft have recently announced major high-availability initiatives. We refer to the SunUP program [10] and the RASCAL laboratory of Sun Microsystems and Motorola's 5NINES initiative.

Outages in computer systems consists of both hardware and software failures. While hardware failures have been studied extensively and varied mechanisms have been presented to increase the system availability with regard to such failures, software failures and the corresponding reliability/availability analysis has not drawn much attention from researchers. The study of software failures has now become more important since it has been recognized that computer systems outages are more due to software faults than to hardware faults.

Recently, the phenomenon of *software aging*, one in which the state of the software system degrades with time, has been reported [2]. The primary causes of this degradation are the exhaustion of operating system resources, data corruption and numerical error accumulation. Eventually, software aging may lead to performance degradation or crash/hang failure or both.

The phenomenon of "software aging", far from being just anecdotal, has been reported while monitoring real applications [2]. It was observed that once the software was started, potential fault conditions gradually accumulated with time leading to either performance degradation or transient failures or both. Failures may be of crash/hang type or those resulting from data inconsistency because of aging. Typical causes of aging, i.e., slow degradation, are memory bloating or leaking, unreleased file-locks, data corruption, storage space fragmentation and accumulation of round off errors. Popular and widely used software like the web browser *Netscape* is known to suffer from serious memory leaks which lead to occasional crash/hang of the application. This problem is particularly pronounced in systems with low swap space. The newsreader software *xrn* also experiences problems due to memory leaks. Software aging has not only been observed in software used on a mass scale but numerous real life examples of aging in specialized software used in high availability and safety-critical applications also exist.

To counteract this phenomenon, Huang et. al [2] proposed a proactive approach of fault management, called *software rejuvenation*. It essentially involves occasionally terminating an application or a system, cleaning its internal state and restarting it. This process removes the accumulated errors and frees up operating system resources, thus preventing in a proactive manner the unplanned and potentially expensive system outages due to the software aging. Since the preventive action can be done at optimal times, for example when the load on the system is low, it reduces the cost of system downtime compared to reactive recovery from failure. Thus, software rejuvenation is a cost-effective technique for dealing with software faults that include protection not only against hard failures, but against performance degradation as well.

2 Research at Duke

The research work at Duke University was motivated by the need of pursuing preventive maintenance in operational software systems on scientific basis rather than on the then ad hoc practice. Thus, an important research issue is to determine the optimal times to perform the preventive maintenance. In this regard, two approaches have been taken - analytical modeling and measurement-based approach.

2.1 Analytical Modeling

The aim of the analytical modeling is to determine the optimal times to perform rejuvenation considering the tradeoffs of maximizing *availability* and minimizing the *probability of loss* or the *response time of a transaction*. This is particularly important for business-critical applications whose adequate response time can be as important as system uptime. The analysis is done for different kinds of software systems exhibiting varied failure/aging characteristics. The key paper was written by us [3] based on the preliminary work by Lucent Bell Labs researchers [2].

2.2 Measurement-based Approach

The measurement based approach deals with detection of the existence of software aging and predicting aging related failures, so that proactive methods can be applied to prevent unplanned outages. The basic idea is to periodically monitor and collect data on the attributes responsible for determining the health of the executing software. The data is then used to obtain clues about possible impending failures.

An SNMP-based distributed resource monitoring tool was developed as a part of Sachin Garg's Ph.D. thesis supervised by K. Trivedi [4]. This was to collect operating system resource usage and system activity data (real memory, swap space, CPU activity etc.) at regular intervals, from networked UNIX workstations (SunOS and Solaris) in the Dept. of ECE. Statistical trend detection techniques are applied to this data to detect/validate the existence of aging. For quantifying the effect of aging in these resources, the metric *Estimated time*

to exhaustion was proposed. which is calculated using the Sen's slope estimation technique. This was the first effort of its kind. While [4] does not take into account the system workload, as a part of K. Vaidyanathan's M.S. thesis supervised by K. Trivedi [6], we developed a measurement-based model to estimate the rate of exhaustion of operating system resources both as a function of time and the system workload state. A semi-Markov reward model is constructed based on workload and resource usage data collected from the UNIX operating system. We first identified different workload states using statistical cluster analysis and built a state-space model. Corresponding to each resource, a reward function was then defined for the model based on the rate of resource exhaustion in the different states. The model is then solved to obtain trends and the estimated exhaustion rates and time to exhaustion for the resources.

This method helps us to obtain a better estimation for time to exhaustion of the particular resource than the purely time-based approach. Further, a comparison between the estimated times to exhaustion can help ascertain the relative importance among different resources, thus pinpointing critical resources to be monitored and managed. The developed measurement-based models are the important steps towards predicting aging-related failures based on actual measurements, intended to help development of policies that automate the proactive handling of potential problems.

2.3 Rejuvenation for Cluster Systems (Collaboration with IBM)

Clustering techniques, a form of parallel and distributed computing, are becoming increasingly popular due to their ability in overcoming obstacles in processor scaling and availability. We have recently extended the use of rejuvenation to cluster systems. Our analyses [6] show that employing software rejuvenation in cluster systems results in a significant increase in system availability. Using the node failover mechanisms in a high availability cluster, one can maintain operation (though possibly at a degraded level) while rejuvenating one node at a time, assuming that a node rejuvenation takes less time to perform and is less disruptive than recovering from an unplanned node failure. Because the user's application has presumably been written to survive node failovers anyway, this environment has the added advantage of allowing rejuvenation to be transparent to the application. Simple time-based rejuvenation policies, in which the nodes are rejuvenated at regular intervals, can be implemented easily, using the existing cluster management infrastructure. The first commercial version of this kind of a software rejuvenation agent for the IBM Netfinity line of cluster servers running Windows NT/2000 has been implemented with our collaboration [1, 7]. System availability can be further enhanced by taking a proactive approach to detect and predict an impending outage of a specific server in order to initiate planned failover in a more orderly fashion. This is done by on-line collection and statistical analyses (curve-fitting and projection) of system data pertaining to system resources. This approach is currently being analyzed and implemented for IBM Netfinity cluster systems also with our collaboration [1, 8], and plans are in place to incorporate this into other operating systems/architectures also. IBM Software Rejuvenation is to be part of the recently announced multi-billion dollar Project eLiza for self-managing servers [9].

References

- [1] V. Castelli, R. E. Harper, P. Heidelberger, S. W. Hunter, K. S. Trivedi, K. Vaidyanathan and W. Zeggert. Proactive Management of Software Aging. *IBM Journal of Research & Development*, Vol. 45, No.2, March 2001.
- [2] Y. Huang, C. Kintala, N. Kolettis and N. D. Fulton. Software Rejuvenation: Analysis, Module and Applications. In *Proc. of 25th Symposium on Fault Tolerant Computer Systems*, pages 381-390, Pasadena, California, June 1995.
- [3] S. Garg, A. Puliafito, M. Telek and K. S. Trivedi. Analysis of Preventive Maintenance in Transactions Based Software Systems. *IEEE Trans. on Computers*, pages 96-107, Vol.47, No.1, January 1998.
- [4] S. Garg, A. van Moorsel, K. Vaidyanathan and K. S. Trivedi. A Methodology for Detection and Estimation of Software Aging. In *Proc. of the Ninth Intl. Symposium on Software Reliability Engineering*, pages 282-292, Paderborn, Germany, November 1998.
- [5] K. Vaidyanathan and K. S. Trivedi. A Measurement-Based Model for Estimation of Resource Exhaustion in Operational Software Systems. In *Proc. of the Tenth IEEE Intl. Symposium on Software Reliability Engineering*, pages 84-93, Boca Raton, Florida, November 1999.
- [6] K. Vaidyanathan, R. E. Harper, S. W. Hunter, K. S. Trivedi. Analysis of Software Rejuvenation in Cluster Systems. In *Proc. of the Joint Intl. Conference on Measurement and Modeling of Computer Systems, ACM SIGMETRICS 2001/PERFORMANCE 2001*, Cambridge, Massachusetts, June 2001.
- [7] IBM Netfinity Director Software Rejuvenation - White Paper. IBM Corporation, Research Triangle Park, NC.
- [8] <http://news.cnet.com/news/0-1003-200-4550757.html?tag=st.ne.1002.bgif.ni>
- [9] <http://news.cnet.com/news/0-1003-200-5744529.html>
- [10] <http://www.sun.com/availability/>