Prediction of Information System Availability in Mission Critical and Business Critical Applications

Myron Hecht • SoHaR Incorporated • Beverly Hills, CA

Twelfth Annual International Symposium of the
International Council On Systems Engineering (INCOSE)
28 July – 1 August 2002

Abstract. One of the most important attributes of on-line computer systems that are performing mission or business critical applications is availability. System Engineers are often called upon to predict the reliability of such systems as part of proposal preparation, architecture definition, design reviews, and operation. However, traditional modeling techniques are incapable of handling integrated hardware and software systems with multiple states and redundancy. This paper describes how Markov Modeling and Reliability Block Diagrams can be used together to model a large on-line information system and develop the answers to strategic questions on the configuration and operation of high availability computing systems. The analyses are performed using MEADEP, a reliability analysis tool capable of hierarchical modeling and integrating Markov and block diagram techniques. In the example 3-tier architecture e-commerce site described in this paper, it is shown that (a) the most frequently failing subsystem is not necessarily the availability bottleneck, and (b) that restoration time is often a more important parameter than availability when attempting to maximize system throughput.

1. INTRODUCTION

Availability is one of the most important attributes of online information systems – whether used for air traffic control, e-commerce, medical information, and financial transactions. This paper describes how reliability and availability modeling can be used to answer such questions as:

- How much redundancy and what service level agreements are necessary to support a system availability of 99.99%?
- I may know which networks, backbones, or servers are failing most often, but which are the biggest contributors to the downtime of my mission critical information services?
- How can my budget be spent most effectively to increase availability and performance?
- What are the highest impact items to negotiate in my service level agreement?
- Is the benefit of replacing a single server with a cluster and/or a RAID subsystem worth the acquisition and ongoing support cost?

This paper describes the use of modeling -- and in particular, the MEADEP (MEAsurement-based DEPendability tool)[1] (Hecht, 1997) -- for predicting the reliability and availability of a large e-commerce web site. Our approach uses hierarchical modeling incorporating both reliability block diagrams and Markov Models.

In this paper, we first describe the system to be modeled, a large three-tiered e-commerce site. We then demonstrate how to use a modeling hierarchy to create simple, easily understood component models that are integrated to represent this complex system. We also demonstrate the power of combined Markov and reliability block diagram modeling. Finally, we discuss the results produced by the MEADEP model and how they can be interpreted from the perspective of the enterprise.

The enterprise (economic) perspective is of particular importance because it supports business decisions to address the issues listed above.

To demonstrate the capability of MEADEP to represent a large system, we will use a simplified representation of the eBay e-commerce web site shown in Figure 1 (based on a description in the Los Angeles Times (Menn, 1999) reporting on their 22 hour outage and the resulting consequences). Buyers and sellers enter information (through browsers) that is routed into either of two Internet Service Providers (ISP's), designated as ISP_ A or ISP_ B, whose backbones are connected directly to the eBay site. Dual redundant

[1] MEADEP is listed on the INCOSE web site

routers are connected to a set of PC front-end web servers running on Windows NT™.

The web servers in turn pass requests on auctions and/or bids to redundant Database Management System (DBMS) servers hosted on Sun Solaris™ servers. These DBMSs in turn retrieve and update tables representing each of the auctions.

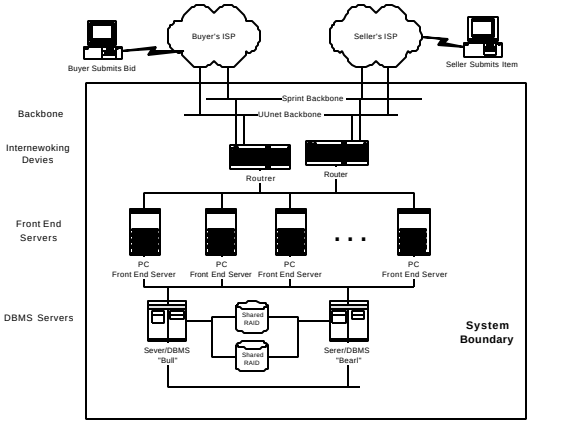

Figure 1 System Configuration

2. MODEL DESCRIPTION

MEADEP allows models to be created hierarchically and thus a complex system can be modeled as an ensemble of simpler, easily understood models which can be either system specific or adaptations of predefined models. Figure 2 shows the model hierarchy.

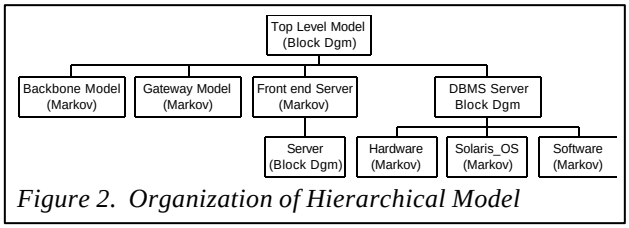

Figure 2. Organization of Hierarchical Model

The top level model is a reliability block diagram that partitions the system into 4 lower level models that represent the backbone communications and tiers of the system. Three of these lower level models are Markov chains while the fourth is a block diagram. Table 1 lists the symbols used in the models. Figure 3 shows the top level block diagram, the 4 top elements shown in Figure 1: the Internet backbones, the Internetworking devices (routers), the front end PC Web servers, and the bull and bear DBMS servers.

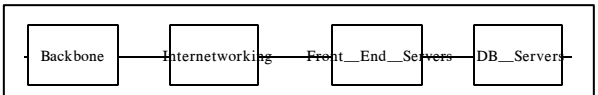

Figure 3. Top Level Block Diagram

Table 1 Nomenclature and Notation

Parameter	Meaning
μWinNT	Reboot/restoration rate of Windows NT running on PC front end servers (corresponding to 12 minute restoration time)
λWinNT	Failure rate of Windows NT running on PC front end servers (corresponding to 200 hour average time between crashes)
μWebserver	Reboot/restoration rate of web server software running on PC front end servers (corresponding to 12 minute restoration time)
λWebserver	Failure rate of web server running on PC front end servers (corresponding to 200 hour time between crashes)
μPCHW	Repair rate of PC front end server hardware (corresponding to 1 hour mean repair time)
λPCHW	Failure rate of web server running on PC front end servers (corresponding to 200 hour MTTF)
Nrs	Number of web servers
Csun	Failure detection and successful switchover probability of Sun Starfire server system from hardware failure
COS	Failure detection and successful switchover probability of Sun Starfire server operating systemfrom hardware failure
λOS	Failure rate of Solaris operating system on the Starfire server (corresponding to 200 hour MTTF)
μOS	Restoration time of Solaris operating system on the Starfire server (corresponding to 20 minute restoration time)
Cdbsw	Failure detection and successful switchover probability from DBMS failure
λdbsw	Failure rate of Oracle DBMS on the Starfire server (corresponding to 200 hour time between crashes)
μdbsw	Restoration time of Oracle DBMS on the Starfire server (corresponding to 20 minture restoration time)
μhw	Repair time of the Starfire hardware
λhw	Failure rate of the Cisco Starfire hardware
Chw	Failure detection and successful switchover probability from Starfire hardware failure
Csw	Failure detection and successful switchover probability from Cisco Switch failure
λsw	Failure rate of the Cisco Switch
μsw	Restoration time of the Cisco Switch
μISP A	Restoration time of ISP A backbone (corresponding to 2 hours)
μISP B	Restoration time of ISP B backbone (corresponding to 2 hours)
λISP B	Outage rate of ISP B backbone (corresponding to 1000 hour time between backbone outages)
C	Failure detection and successful switchover probability from

For the purposes of simplicity and clarity, we make the following assumptions on the constituent lower level models:

1. Electrical power is 100% available
2. The inter-arrival time of software failures is Poisson distributed (largely true of mature, infrastructure software such as operating systems [2])
3. System components function or fail completely (degradation is treated on a subsystem but not component level)
4. Failure rates and restoration times do not consider human induced failures, such as cable cuts or network configuration errors.
5. Components including network hubs, redirectors, and DNS servers are not included in this model.

Figure 4 shows the lower level reliability model for the two backbone networks. While reliability block diagrams use blocks to represent structural elements, Markov models represent the system in terms of states. Movement from left to right indicates failure and movement from right to left indicates recovery. In the model shown in Figure 4, there are 4 possible states: (1) both networks are up, (2) the ISP_ B network is up but the ISP_ A connection is down, (3) the converse, and (4) both networks are down. The second and third states are degraded in that only half the capacity is available. The fourth state is a completely failed state.

These capacities are represented by the reward, the relatively value of each state, depicted by a number below the state name (always in the range of 0 to 1). A reward of 1 to the fully functional state of both networks up, and a reward of 0.5 to the states of one ISP up. Specifically, if either ISP_ A or ISP_B is down, the system capacity has been reduced by half (assuming that both ISP_ A and ISP_ B have equal capacity). The expressions next to the arrows reflect the transition rates among the states. The failure rate of the ISP_ A network is defined by the parameter λISP_ A, and the recovery rate is given by μISP_ A. The meanings of parameters such as λISP_ B and μISP_ B are similar.

This model also considers the case when one backbone network fails and the system does not successfully make the switch to the second network. The parameter, C, or coverage, is the probability that there is a successful switch given that a failure in one of the networks occurs. In such a case, the quantity 1-C represents the probability that the transition is unsuccessful, that is, the Internetworking devices (routers) do not handle the transition from two networks to one. As a result, the web site moves directly from the first functional state to the failure state.

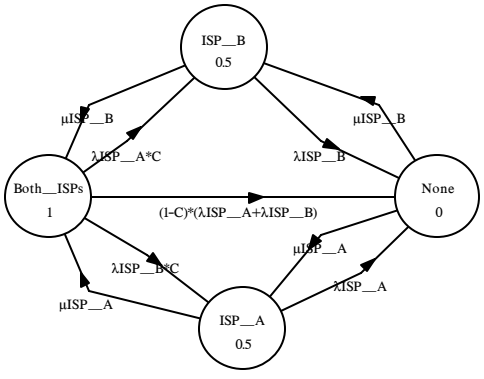

Figure 4. Markov Model for Backbone Network

We now move to the next block in Figure 2, the Internetworking devices (routers). Figure 5 shows the corresponding lower level Markov model. This model shows a classic 1 out of 2 hardware system, and has been taken directly from the MEADEP library of reusable models. The first state on the left, designated S0, represents both switches up, the second, S1, represents one switch up, and the final state (on the right), S2, represents both switches down. As was the case in the previous model, we have a finite probability, C, that the system will successfully transition from state S0 to state S1, and the complementary probability, 1-C, that the system will transition from the functional state S0 to the failed state Sf. In this model, we assign a reward of 1 to both the states S0 (both switches up) and S1 (one switch up) because either switch can handle the traffic of the entire network.

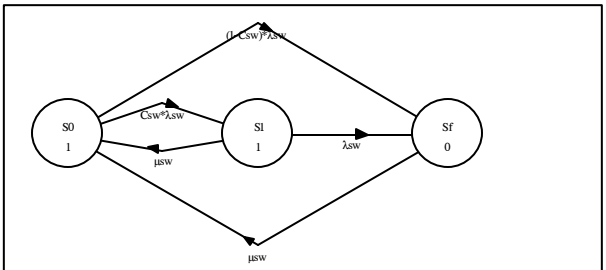

Figure 5. Internetworking (Router) Markov Model

The third block of Figure 2 represents the front end servers. The lower level model is shown in Figure 6, and consists of 11 PC-based servers. There are 8 states: S0 represents all servers functional, S1 represents 1 server down, S2 represents 2 servers down, and so on. Load balancing (within the routers or in a separate device, such as a level 4 switch, see assumption 5) assigns each incoming user to the most lightly loaded server. As servers fail, there is a reduction in overall capacity. We have assumed that once the system falls below 70% capacity (6 failures), a complete shutdown will occur. The rewards are reduced linearly under the assumption of a 5% loss in capacity for each server failure. Thus, we have assigned each with the appropriate reward function. For 11 servers, the reward is 1 (100% capacity), for 10 servers, it is 0.95, and so on. However, more precise estimates can be made on the basis of performance simulation modeling.

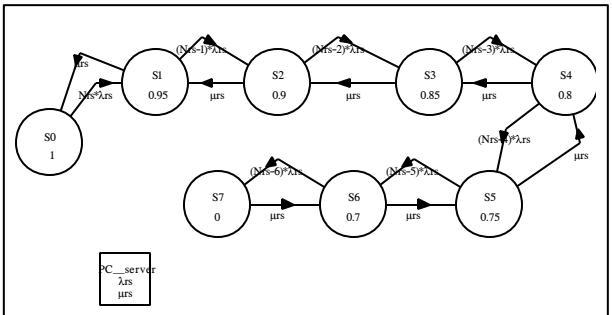

Figure 6. Portion of Front End Processor Model

Figure 7 shows a lower level model of a single PC front-end server. The model, which is used to calculate

λserver, as shown in Figure 6, consists of 3 elements: hardware, operating system (including the network infrastructure), and server software. This breakdown allows the uptime statistics from the Windows log files.

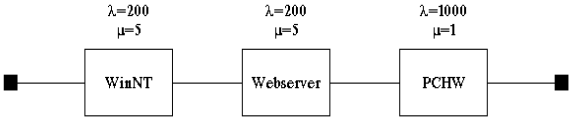

Figure 7. Server Model

The models for the database servers are shown in Figure 8 through 10. Figure 8 is a block diagram showing the dual redundant database server depicted in three segments: hardware, operating system, and DBMS software. The RAID device consists of redundant SCSI ports into redundant disks. For the purposes of this discussion, one channel is allocated to each of the servers.

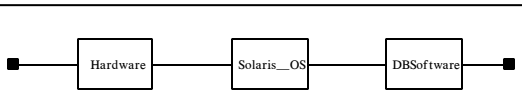

Figure 8. Database server top level diagram

Figure 9 shows the model of the dual redundant hardware platform, which is similar to Figure 5. There are 3 states: both processors up (S0, i.e., no failures), one processor up (S1, one processor down), and no processors up (Sf, failure state). State S0 has 2 transitions: the first being to state S1, and the second being to the failure state. The sum of these two transition rates is 2λhw, corresponding to the combined failure rate of the two functioning systems. State S1 also has two transitions out: restoration, i.e., μhw, and the condition of a second processor failure while the first is being restored. This occurs at the rate of failure

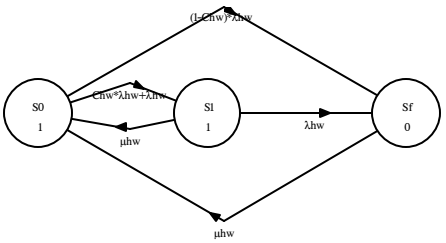

Figure 9 Database Hardware Model

of a single processor, or λhw.

Figure 10 shows the Markov model for the operating system portion of the DBMS servers, the block entitled Solaris_OS, and Figure 11 shows the corresponding diagram for the DBMS application running on the server. Both these models are similar to the hardware model shown in Figure 9. The primary difference is in the meaning of the transition parameters, which correspond to the failure rates of the DBMS and the operating system.

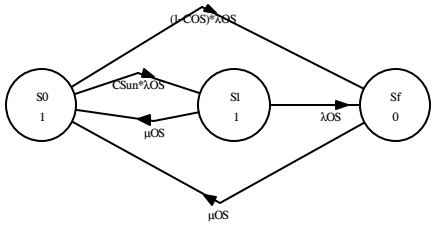

Figure 10. Solaris_OS diagram

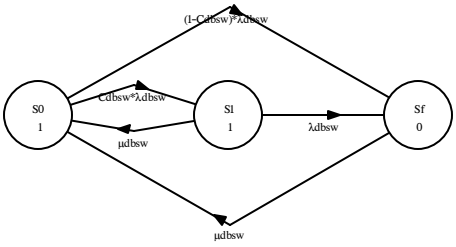

Figure 11. DBMS Block Markov Diagram

3. RESULTS

Table 2 shows the values used in the model described in the next section (see Table 1 for symbols and nomenclature).

Table 2. Baseline Values of Model Parameters

Parameter	Value	Parameter	Value
μWinNT	5 hr^{-1}	μdbsw	3 hr^{-1}
λWinNT	.005 hr^{-1}	μhw	1 hr^{-1}
μWebserver	5 hr^{-1}	λhw	.001 hr^{-1}
λWebserver	.005 hr^{-1}	Chw	0.9
μPCHW	1 hr^{-1}	Csw	0.9
λPCHW	.001 hr^{-1}	λsw	.005 hr^{-1}
Nrs	11	μsw	3
Csun	0.95	μISP_A	0.5
COS	0.95	μISP_B	0.5
λOS	0.005 hr^{-1}	λISP_B	.001 hr^{-1}
μOS	3 hr^{-1}	C	0.95
Cdbsw	0.9	λISP_A	.001 hr^{-1}
λdbsw	.005 hr^{-1}		

Table 3 shows the results in numeric form for all 9 models in the hierarchy. The overall system availability is 0.9957, corresponding to an average downtime of 38 hours per year.

Table 3. Model Results

Model-Name	Failure-Rate (per hour)	MTTF (hours)	Repair-Rate (per hour)	MTTR (hours)	Availability	Unavailability
Ebay_ Main	0.016189	61.8	3.74	0.268	0.995688	0.004312
Backbone	0.0021	476.2	1.00	1.000	0.997905	0.002095
Cisco_ Switches	0.000507	1973.4	3.00	0.333	0.999831	0.000169
Front_ End_ Servers	0.007882	126.9	4.64	0.216	0.998303	0.001697
PC_server	0.013923	71.8	4.64	0.216	0.997006	0.002994
DB_ Servers	0.000986	1014.5	2.77	0.362	0.999644	0.000356
DB Software	0.000507	1973.4	3.00	0.333	0.999831	0.000169
Hardware	0.000102	9832.4	1.00	1.000	0.999898	0.000102
Solaris_OS	0.000257	3883.5	3.00	0.333	0.999914	0.000086

Figure 12 graphically shows the results of the model for the top hierarchical divisions shown in Figure 2. The subsystem with the highest failure rate is the front-end server PC subsystem, followed by the WAN backbone.

These results indicate a site with a relatively high availability and reliability. However, for high volume web sites, there is always an incentive to achieve even higher levels of reliability and availability. Parametric analysis can support the quest for more uptime. We first consider the case of increasing the system reliability, i.e., increasing its Mean Time Between Failures (MTBF). Figure 12 showed that the PC front end server subsystem has the highest failure rate. It would appear that increasing the reliability of the PC servers would therefore provide the highest benefit. However, parametric analyses show that such an effort will yield less benefit than other measures.

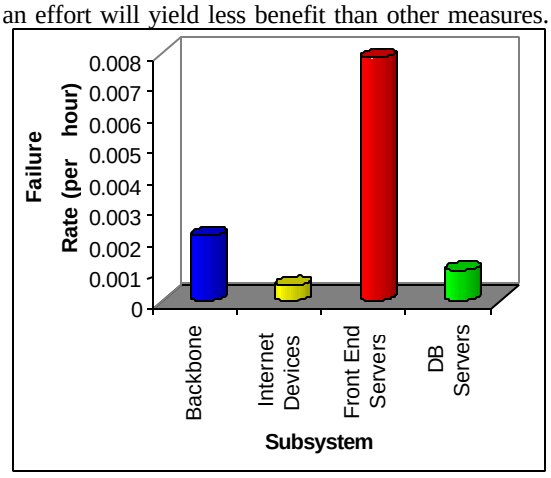

Figure 12. Failure Rates of Major Subsystems

The results from the model are not always obvious. For example, as the PC server operating system MTBF changes from 200 hours to 2000 hours, factor of 10 increase MTBF in each of the PC servers. The front server subsystem MTBF improves from 127 hours to 195 hours, corresponding to a decrease in the failure rate from 0.007882 per hour to 0.005128 per hour, a 65% decrease.

3.1 BUSINESS ANALYSIS OF A SERVICE LEVEL AGREEMENT

As was shown above, improvement of the WAN Service provider backbone restoration time provides the greatest benefit (for reliability). This section discusses the business case for two aspects of service level agreements: restoration time and reliability.

Figure 13 shows the impact of service provider restoration time on monthly revenue under the following assumptions: (a) a transaction rate of 650 per minute, (b) failure rates of 500 hours for both backbone networks, (c) a restoration time of 1 hour for the service provider of the second network, (d) the net value of each transaction is $0.10.

With a restoration time of 30 minutes, the expected number of lost transactions per year is approximately 1.2 million. However, as the restoration time approaches 5 hours, the number of lost transactions increases to 2.8 million. Under the assumptions listed above, the value of a service level agreement that guarantees a restoration time of 0.5 hours or less is more than $160,000 per year.

Figure 14 shows the impact of the failure rate of one of the two Internet service provider backbones on the number of lost transactions (the MTBO of the second Internet backbone is held constant). As the backbone MTBO is increased from 100 to 1000 hours, the number of lost transactions decreases from approximately 4. 2 million to just over 1 million. Under the assumption that each transaction is worth $0.10, the value of increasing reliability over this range is approximately $320,000 annually.

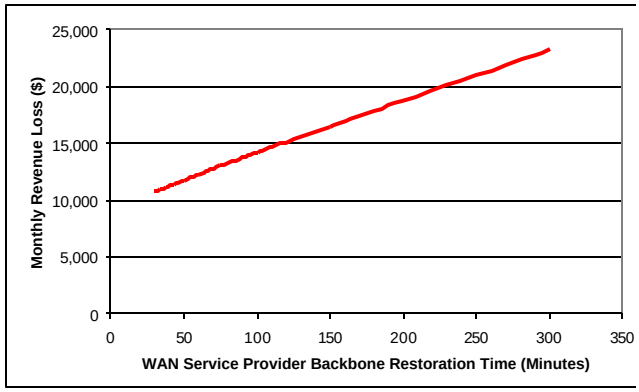

Figure 13 Impact of Service Provider Restoration Time on Monthly Revenue due to Lost Transactions

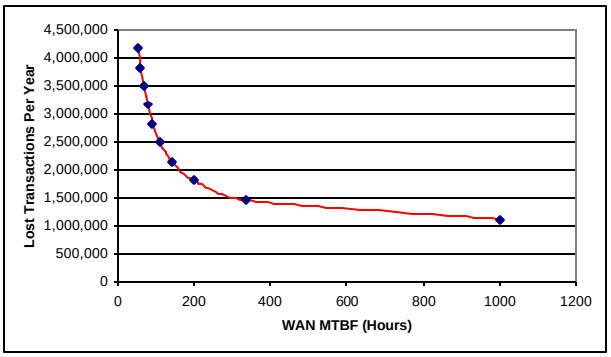

Figure 14 Impact of Backbone Outage Rate on Web Site Downtime

However, it may be impossible – or very expensive -- to obtain a service agreement with a guarantee of an average time between outages time of 1000 hours (approximately 6 weeks). It will probably be much more feasible to get an agreement of 200 to 300 hours. As is evident in the figure, most of the benefit can be achieved if the reliability is increased to 330 hours (a little less than two weeks). Thus, it may be more cost effective to set the reliability of the service level agreement at that level. The resulting reduction of transaction loss, approximatley 2.7 million annually, would be worth $270,000 annually.

It must be emphasized that these results are with the presence of *two independent and parallel* service providers. If a single backbone is used, the relative importance of MTBO and outage restoration time will differ.

3.2 VALUE OF ADDITIONAL CAPACITY

The information system model has two independent backbones, but neither has the capacity to carry the full traffic load (the model assumes that each has half the capacity). Given the MTBOs and restoration times of the networks, MEADEP can be used to evaluate the value the additional capacity through its *reward* function. Figure 15 compares the monthly revenue loss for two alternatives:

- *Half capacity backbones*: 2 WAN backbones each with half the capacity required for the average load, and
- *Full capacity backbones:* 2 WAN backbones each with the capacity for the entire load

The results show that increasing the capacity of both backbone WANs so that either could handle the full load could decrease the monthly revenue loss by as much as $30,000 or as little as $9,000 depending on the reliability (as measured by mean time between outages) of the service providers. The lower the reliability (MTBO), the greater the benefit of increasing the capacity.

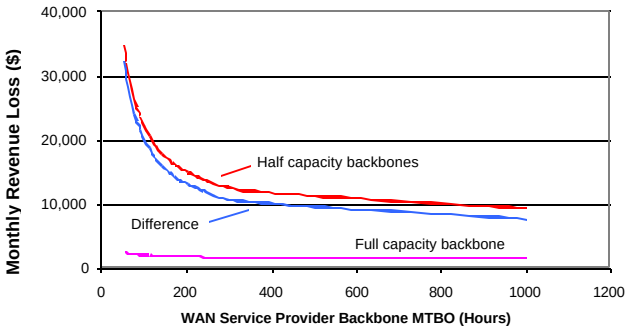

Figure 15. Revenue losses due to outages of WAN backbones

With these results, it is also possible to assess the value of a service level agreement with the additional dimension of capacity. For example, it may be that provisioning each of the backbones with double the capacity is a lower cost option than securing service level agreements with both WAN backbone service providers. With the reward function contained in the MEADEP tool, it is also possible to consider other options such as providing full capacity in the first backbone and half capacity in the second one, or providing 75% capacity in each. The optimum configuration depends on the traffic profile and the MTBO and restoration times of the WAN backbones, the load profile of the server, the value of the traffic, and other system-specific factors.

4. CONCLUSIONS

This paper has demonstrated how modeling can be used not only for verifying meeting requirements but also for performing tradeoff analyses and answering significant cost questions. However, the correct modeling approach and procedures must be used. Traditional reliability block diagrams do not account for the multiple state transitions that can occur in modern computing systems with multiple levels of redundancy and performance. However, Markov modeling, which has been used for nearly three decades for modeling computer system reliability, can be used for this purpose.

For a continuously operating information system, maximizing uptime (i.e., availability) is the primary objective. For a transaction-driven e-commerce or similar site, uptime translates directly into transactions, which in turn is the source of revenue for the enterprise. However, it is not always the most important attribute. MEADEP can handle these alternate figures of merit as well.

For example, if a system is taken down periodically in accordance with a scheduled maintenance plan, then the reliability, i.e., the probability that the system will

be up during the maintenance interval is the primary figure of merit. Another example arose in a recent election when a major candidate held a fund raising event on the web site. For this limited time purpose, reliability (i.e., minimizing the probability of an outage over the fixed time that the fund raising was occurring) was also more important than continuous availability.

It is important to note that representing software failures using stochastic processes is not valid for immature systems. However, for the COTS-based hardware and software processors used in modern systems, it is valid. Previous research on mature fault tolerant and other high integrity systems has shown that residual software faults lead to a failure behavior which can be characterized by an MTBF (Nagle, 1982; Adams 1984; Hsueh, 1988) and that a majority of such failures could be recovered from by the use of physical redundancy (Gray, 1990; Lee, 1995, Tang, 1995). These results provide a basis for modeling software failures as a stochastic process in systems where (1) the requirements definition (including specification of off-normal states and events in the operating environment) is substantially correct and complete, (2) a disciplined development, has been effective in eliminating defects with a high degree of determinism and repeatability, and (3) testing and past operational history are largely representative of future performance. Under these conditions, The failure rate of mature software can be determined through measure based methodologies (Hecht, 1997, Tang, 1995).

REFERENCES

Adams, Edward N., "Optimizing Preventive Service of Software Products", *IBM Journal of Research & Development*, Jan. 1984, pp 2-14.

Gray, J., "A Census of Tandem System Availability Between 1985 and 1990", *IEEE Transactions on Reliability*, May 1990, pp. 409-418.

Hecht, M., D. Tang, and H. Hecht "Quantitative Reliability and Availability Assessment for Critical Systems Including Software", *Proceedings of the 12th Annual Conference on Computer Assurance*, , Gaitherburg, Maryland, USA, June 16-20, 1997

Hsueh, M.C. and R Iyer, "Performability modeling based on real data: A case study", *IEEE Transactions on Computers*, vol. 37 no. 4, April 1988, pp. 478-484.

Lee, Inwhan and Ravi K. Iyer, "Software Dependability in the Tandem GUARDIAN System," in IEEE Transactions on Software Engineering, May 1995, pp. 455-467.

Menn, Joseph, "Prevention of Online Crashes Is No Easy Fix", *Los Angeles Times*, Section C, Page 1, October 16, 1999.

Tang D., and M. Hecht, "Evaluation of Software Dependability Based on Stability Test Data," *Proc. 25th Int. Symp. Fault-Tolerant Computing*, Pasadena, California, pp. 434-443, June 1995

Tang, D. , M. Hecht, and H. Hecht, "A Methodology and Tool for Measurement-Based Dependability Evaluation of Digital Systems in Critical Applications", *IEEE Transactions on Nuclear Science*, August, 1996.

Author Biography

Myron Hecht is co-founder and President of SoHaR Incorporated, a firm specializing in the reliability of computer systems, with offices in the Southern California and the Washington, DC metropolitan areas. He has worked in the areas of software and systems reliability. His activities in basic research and development at SoHaR have resulted in new architectures for real time distributed systems, methodologies for the development and verification of fault tolerant software, and he has developed designs for highly reliable distributed systems for both process control and C^3I. He has also worked on large projects supporting either the prime contractor or the Government customer in air traffic control and air defense systems. Mr. Hecht holds both B.S. and M.S. degrees in Engineering from UCLA, and a M.B.A in Information Systems from UCLA.