Availability in Telecommunication Management Distributed Systems

Patrícia A. Oliveira[a], José Marcos Nogueira[a] and Germán Goldszmidt[b]

[1 a]Computer Science, Federal University of Minas Gerais,
Belo Horizonte - MG - Brazil
(pati,jmarcos)@dcc.ufmg.br
[b]IBM Research - Hawthorne, NY, USA
gsg@us.ibm.com

Abstract. *This work presents the development of a Markov-based model for availability analysis of telecommunication management systems. The model was applied to analyze the Integrated System of Supervision (SIS), allowing to define the availability of parts of the system, to identify hardware and software components responsible for decreasing the availability of the system, and to define actions for availability improvement. The model is generic and can be applied in other management systems, such as management systems based on SNMP.*

Key-words: availability, Markov, distributed systems.

1. Introduction

Telecommunications management systems, when deployed in plants of medium to high complexity, can present complex structures with large numbers of hardware and software components [1]. In such cases, they are usually highly available, real time distributed systems with many interactions among their computational processes. Availability of these management systems is critical for the proper operation of the telecom plants. Both the systems and their components (hardware and software) require availability analysis, not only to monitor the system behavior, but also to evaluate the system design, configuration parameters and possible changes needed. Therefore, availability is one of the main parameters for service quality of a management system [2, 3, 4, 5].

This work analyzes the availability of distributed systems, considering both their hardware and software components, using the Markov model. The model is applied to answer questions like: how available is the entire system? How available is each one of the hardware and software components of the system? After answering these questions, it is possible to identify the components responsible for reducing the availability of the system, to evaluate their design and configuration, and to decide about changes to increase the system availability. Such a model can also help in identifying if a system crash was caused by an external factor or by one of their modules.

The motivation for this work comes from a telecommunication management system, the System for the Integration of Supervision (SIS), [6]. SIS is a distributed system responsible, for the management of a complex, huge, geographically distributed and heterogeneous telecommunication plant. SIS is used in fault, configuration, accounting, traffic and even in security management functional areas. The management levels include element, network, and services. Some outputs of SIS are used by the business personal, as well. The system has been in operation since 1993 covering an area of approximately 5 millions Km^2, with 20 million telephone terminals, and 80 million inhabitants. SIS is in operation at all times (7x24). It seldom stops operation, only in software version upgrades, for few hours, usually on Sundays dawn. The analysis of availability of SIS is an important source of information for their administrators, operators and users. The analysis objective is to increase the SIS availability, and to identify if a failure in the execution of a service was caused by an internal problem in SIS or by an external factor. For example, SIS has an interface with a external system named STC (Client Treatment System) for whom it executes operations in central switches. Every day, STC sends to SIS a list containing terminal telephone numbers to be enabled or disabled. If STC fails in a request, the operation fails, and the responsibility for the failure can not be assigned to SIS. With availability analysis it is possible to check the situation when the problem had happened.

The analysis introduced in this paper can be applied to others management systems, for example, in SNMP-based management systems. Even if a system is centralized, there are similarities with the SIS model in respect to availability of the modules: in the path where management information throws, there is a line of dependence among the modules. It is usual for a path to be composed of a network element, agent, proxy agent, manager and a data base.

This paper is organized as follows. Section 2. presents a formal definition of availability, the Markov model and related work. Section 3. presents architectures of the distributed systems and section 4. proposes a model for availability analysis of these systems architecture for both hardware and software components using the Markov model. Section 5. presents a case of study, where SIS availability is analyzed. Finally, the last section presents the conclusion and discusses future work.

2. Availability in a distributed systems

Availability is defined by [7] in the following way: "The ability of an item to be in a state to perform a required function at a given instant of time or at any instant of time within a given time interval, assuming that the external resources, if required, are provided".

In practice the *availability* term can be expressed in many different ways [8]. In this work, *availability* means a binary metric, which informs if a system and its hardware and software components are or are not available when requested. Based on this definition, the *limiting availability* measures the expected fraction of time that the system, its hardware and software components are operating, when time tends to infinite [9].

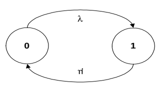
2.1. The Markov model

We can divided the main mathematical models of availability analysis in two groups: the models based on the states that the system may have, called state-space models, and the models that assume stochastic independence between system components, called non-state-space models [9, 10]. To analyze complex systems with complicated interactions between components, state-space models are used because they are more flexible, generic and capable of representing dependencies between the systems modules. The Markov model is a state-space model that has often been used due to its generality and its ability to represent complex iterations between the system components.

In order to measure system availability using the Markov model it is necessary to define three elements: a transition state diagram, a generator matrix and an initial probability vector [11]. The transition state diagram is a graph where its nodes represent the system states and the arrows are the possible state transitions. Based on this diagram it is possible to build the generator matrix. The element q_{ij} gives the rate of system interchange between the i and the j states, and the diagonal elements are calculated as: $q_{ii} = -\sum_{j \neq i} q_{ij}$.

The initial probability vector defines the probability of the system to be in some state at the initial time (t_0). If we assume the system is working at (t_0) its initial probability vector is $\pi_0(0) = 1$.

The following example illustrates the model: Consider a system that can be only in one of two states, **up** (state 1) and **down** (state 2). The failure rate is λ and the repair rate is μ. At the initial time the system is up ($\pi_0(0) = 1$). Figure 1.(a) shows the transition state diagram for the system. Based on this diagram the generator matrix is built (figure 1.(b)).



(a) State transition diagram

$$Q = \begin{pmatrix} -\lambda & \lambda \\ \mu & -\mu \end{pmatrix}$$

(b) Matrix generator

Figure 1: Markov model

An equation system is built using the probability vector and the generator matrix, and its solution gives the availability answer. In this work the equation system representing the SIS is built and solved using the software SHARPE. The SHARPE tool (Symbolic Hierarchical Automated Reliability and Performance Evaluator) allows to build and analyze performance, reliability and availability models. It includes both non-state-space and state-space models. SHARPE was developed by CACC *(The Center for Advanced Computing and Communication)*, a research center composed by enterprizers/university/government located in Raleigh and Duke Universities. (Additional information can be found in [11, 12].)

2.2. Related work

Lai et al. proposed a model for availability analysis of homogeneous distributed software/hardware systems [13]. The systems are composed by identical copies of distributed application software running on the same type of computers. A Markov-based model was developed and equations were derived to obtain the steady-state availability. Such a study is useful when studying optimal testing time or testing resource allocation. However, the work does not treat heterogeneous systems, as SIS. Goel and Soenjoto considered the performance of combined software and hardware system and developed stochastic models to analyze the system performance [4]. Laprie e Kanoun introduced the X-ware reliability and availability analysis, addressing the problem of modelling the system's reliability and availability with respect to the various classes of faults (physical and design, internal and external) which may affect the service delivered to its users [14]. This work showed that classical availability models can be generalized in order to cover both software and hardware faults. This idea was used in our work.

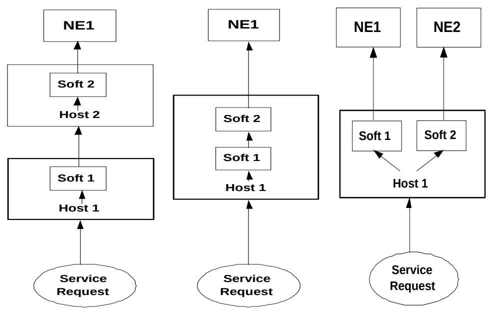
3. A availability analysis model

Telecommunication management systems are distributed systems composed by a large number of hardware and software components having different architectures. To simplify the very hard task of modelling such complex structures, three management systems exhibiting characteristics and properties common to real systems are defined, each one with a software/hardware architecture: (1) serial point-to-point system, (2) serial point-to-point system with shared resources, and (3) serial point-to-multi-point system with shared resources. In our study, the SIS availability analysis will be measured by decomposing the real system into the aforementioned classes.

A serial point-to-point system has only one network element, one data input, and one path between the network element and the data input. Therefore, the system is available only if all its hardware and software components are up. Figure 2.(a) represents a serial point-to-point system composed by two hardware devices (Host 1 and Host 2), and two software modules (Soft 1 and Soft 2). This system executes operations in the network element NE1 only if all its components are up.

The serial point-to-point system with shared resources has only one data input, one network element, only one path between the data input and the network elements and has shared hardware or/and software resources. Figure 2.(b) represents a serial point-to-point system with shared resources composed by one network elements (NE1), one host (Host 1), executing two software modules (Soft 1 and Soft 2). In order to execute operations in NE 1 it is necessary that all components are up.

The serial point-to-multi-point system with shared resources has only one data input, more than one network element, one path between the data input and each one of the network elements, and it has shared hardware or/and software resources. Figure 2.(c) represents serial point-to-multi-point system with shared resources composed by two network elements, NE 1 and NE2, and only one host, Host 1. In order to execute operations in NE 1 it is necessary that Host 1 and Soft 1 are up, while for operations in NE 2 it is necessary that Host 1 and Soft 2 are up. Therefore, Host 1 availability interferes with the operations of both network elements.



(a) Serial point-to-point system

(b) Serial point-to-point system with shared resources

(c) Serial point-to-multi-point system with shared resources

Figure 2: Distributed system architectures

To build a model of a system to measure its availability, several assumptions were made about the behavior of the components and their fault and repair models:

1. Both software and hardware components have one of two states: **up** and **down**;
2. The software and hardware components have a failure rate λ_s and λ_h, respectively, exhibiting an exponential distribution [15];
3. There is a maintenance personnel to repair the system upon software and hardware failure. The repair time follows an exponential distribution with parameter μ_s for software failure and μ_h for hardware failure [13];
4. All the software failures are independent from each other;
5. All the hardware failures are independent from each other;

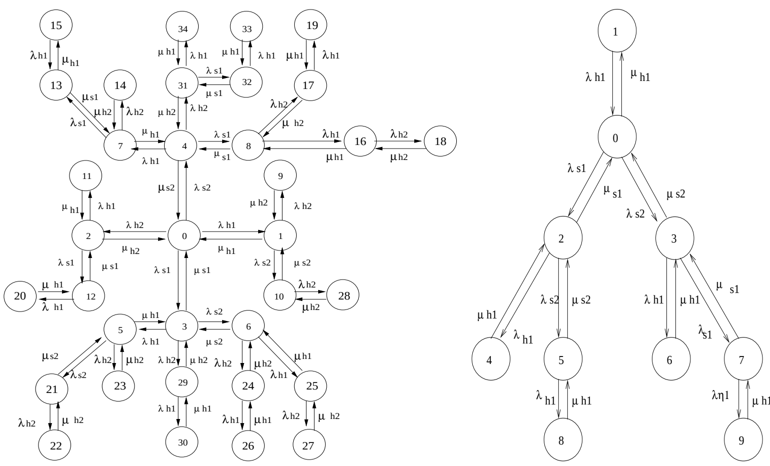
6. The software depend on their hosting hardware. If hardware fails, failure will be assigned to the hardware, and not to software. When the hardware is fixed, the state of the hosted software (up or down) will be the same as it was just before the hardware failure.
7. Failures cannot occur at the same time;
8. At the initial time t_0, all modules are working: the system is in the **up** state.

4. Availability analysis of distributed system

In this section, availability analysis of the three systems described in section 3. is developed using a Markov model. We first define the state transition diagram, the generator matrix and the initial probability vector.

4.1. Serial point-to-point system

Figure 3. shows the state transition diagram for a serial point-to-point system. It represents a Markov chain, with 35 states. The failure rates are λ_{h1}, λ_{h2}, λ_{s1} and λ_{s2} for Host 1, Host 2, Software 1 and Software 2, respectively. The repair rates are μ_{h1}, μ_{h2}, μ_{s1} and $.\mu_{s2}$ for Host 1, Host 2, Software 1 and Software 2, respectively. The state 0 is the initial state where all components are working (the system is up) and, therefore, $\pi_0(0) = 1$ is the initial probability vector.



(a) Serial point-to-point system.

(b) Serial point-to-point system with shared resources.

Figure 3: State transition diagram

Starting at state 0 the system can go to state 1 (Host 1 down), or to state 2 (Host 2 down), or to state 3 (Software 1 down), or to state 4 (Software 2 down). It is important to notice that the system is up only at the state 0. Once the system reaches the state 1, it can go to the state 9 (Host 2 fails before Host 1 is recovered), or to state 10 (Software 2 fails before Host 1 is recovered), or return to state 0 (Host 1 up), etc.

As discussed in the previous section, the generator matrix is built based on the state transition diagram, q_{ij} being the rate for system changing from the *i* state to the *j* state (the value of the ij arrow), and $q_{ii} = -\sum_{j \neq i} q_{ij}$:

$$Q = \begin{pmatrix} -\lambda_{h1} - \lambda_{h2} - \lambda_{s1} - \lambda_{s2} & \lambda_{h1} & \lambda_{h2} & \lambda_{s1} & \lambda_{s2} & 0 & 0 & 0 & 0 & \dots \\ \mu_{h1} & -\mu_{h1} - \lambda_{h2}\lambda_{s2} & 0 & 0 & 0 & 0 & \lambda_{h2} & \lambda_{s2} & 0 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \end{pmatrix}$$

4.2. Serial point-to-point system with shared resources

Figure 4. shows a state transition diagram for a point-to-point system with shared resources. The 0 is the initial state where the system is available. Based on this diagram, the generator matrix is built.

4.3. Serial point-to-multi-point system with shared resources

The availability analysis of a serial point-to-multi-point system with shared resources is similar to the serial point-to-point system with shared resources diagram once they have the same states and the same transitions. For both systems the number of components is the same and the host executes two software modules. However, in some states of the point-to-multi-point system the operations can be available only for NE 1 and not available for NE 2, and vice-versa. For the point-to-point system the operations are available only at the 0 state.

5. Obtaining SIS's data

A methodology for the SIS availability analysis is presented in 4 steps:

1) Defining the SIS configuration: The first step is to define the SIS configuration, identifying the networks elements, the hardware and software components, the paths between the network elements and the systems request point(the input data) and the interdependence of the components.

The SIS is spread over a huge geographical area, which includes 16 of the 27 states of Brazil and provides many different services. To exemplify, this work shows the availability analysis of the system in one region called BHE, the SIS functionalities restricted to the terminal enabling and disabling operation (configuration management). The BHE region is located in the Minas Gerais state, covering a large geographical area with more than 2 million terminals. The enabling/disabling terminal service has large economical importance for the operator (Telemar), the owner of SIS and the largest Brazilian telecommunication operator.

Figure 4 shows the terminal enabling/disabling structure in the BHE region. Seven switching systems are identified: SwS1, SwS2, SwS3, SwS4, SwS5, SwS6 and SwS7. A service for any switching system is executed only if all the path components between the input data and the switching system are up when the service is requested. To illustrate, if a service is requested to the SwS1 switch system, the Mesox software module and its host, the IBD, the manager 67 and their host, the agent 60604 and its host must be available for the service to succeed. Note that the enabling/disabling service availability depends on the availability of each one of the modules composing the path between the service request and the switching system.

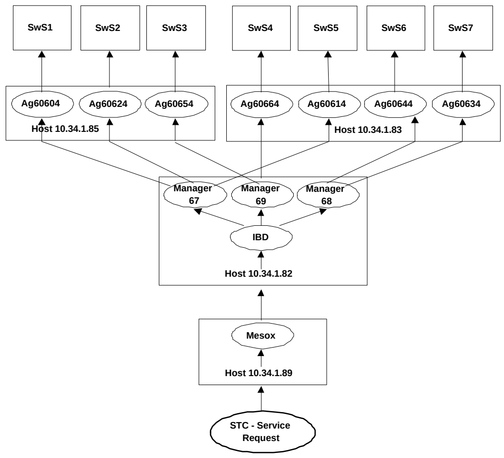


Figure 4: Access paths for terminal enabling/disabling service

2) Applying the Markov model in SIS: The Markov model is applied in the configuration described above to measure the switching systems operations availability using SIS. The architecture presented in section 3. is identified and the state transition diagram, the generator matrix and the initial probability vector are defined.

The path between the service request and one switching system is a serial point-to-point sub-system with shared resources. A script to automatically generate all the sub-system transitions was developed. Therefore, it was possible to define the state transition diagram and generator matrix. The SwS1 switching system generator matrix is showed below:

$$\begin{pmatrix} -\lambda_{89}-\lambda_m-\lambda_{82}-\lambda_{ibd}-\lambda_{g67}-\lambda_{85}-\lambda_{ag} & \lambda_{89} & \lambda_m & \lambda_{82} & \lambda_{ibd} & \lambda_{g67} & \lambda_{85} & \lambda_{ag} & \cdots \\ \mu_{89}-\mu_{89} & 0 & 0 & 0 & 0 & 0 & \mu_{82} & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

At the initial time the system is up, therefore the initial probability vector is $\pi_0(0) = 1$.

3) Defining the failure and the repair rates: Using supervising information collected by SIS itself it was possible to identify the number of failures per hour and the average repair time. This study was based on data stored in the SIS data base and on the development of scripts for the analysis of collected information and processed statistics.

The SIS has a self-supervision structure to monitor its software and hardware modules, generating alarms under failures. Modulo identifier, occurrence date, severity and fault description are some of the attributes of the failure data. Using these information, total faults and repair time of the components were defined. The data was collected from April 15 to June 24 of 2002 (71 days total). Based on these data, the SIS components failure rates (λ) and the repair rates (μ) were defined. Failure rate is the average number of faults per hour. To illustrate, a SIS component failed just once during the 71 days (1704 hours) and the operators took two hours to fix it. The failure rate is: $\lambda_{modulo} = 1/1704 = 0.00058$ faults/hour, and the repair rate is $\mu_{modulo} = 1/2 = 0.5$. The results show a low frequency for failure. The use of a larger sample would give a more reliable result, however it was not possible for the present work. In future, the same procedure will be used to analyze SIS using a larger mount of data. Table 1 shows the SIS components failure and repair rates.

At this point, the generator matrix, the initial probability vector, the failure and the repair rate have been defined. The next step is to measure the SIS availability using the SHARPE tool (described in section 2.).

4) Availability measure: Initially, the terminal enabling/disabling service limiting availability was measured for each one of the switching systems located in the BHE region. Further, the main modules responsible for damaging the system availability in switching systems SwS3 and SwS4 were identified. SwS3 and SwS4 are the most and the least available SIS switching systems, respectively. The repair and failure rates of the modules composing the path between the service requests and SwS3 and SwS4 was changed in order to analyze how they affect the system availability. Each component repair rate was increased and decreased by 20%, 40%, 60%, 80%, 100% of their initial value to analyze the changes in the system availability. A similar procedure was done for the failure rates, reducing it in 20%, 40%, 60%, 80% and 100% of their initial value to analyze the system availability. Finally, only the parameters of the main modules responsible for damaging the system availability were changed and the availability system was measured. The results are discussed in the next section.

Software modules	λ (faults/hour)	μ (/hour)
Agent-60604	0.02	2.79
Agent-60614	0.002	8.62
Agent-60624	0.002	0.14
Agent-60634	0.011	8.62
Agent-60644	0.0099	4.46
Agent-60654	0.002	25
Agent-60664	0.003	0.05
Manager-67	0.0046	1.21
Manager-68	0.002	0.67
Manager-69	0.003	0.82
IBD	0.0035	8.33
Mesox	0.0017	0.046
10.34.1.82 - Acerola host	0.00058	19.23
10.34.1.83 - Goiaba host	0.0035	18.52
10.34.1.85 - Abacate host	0.00058	71.42
10.34.1.89 - Mamão host	0.00058	71.42

Table 1: SIS components failure and repair rates

6. Results and availability analysis

6.1. Switching system operations availability

Table 2 shows the SIS switching system operation availability. The SwS3 has the better limiting availability, around 96%, while SwS4 has the worst, around 90%. The SwS3 access path is composed by the 10.34.1.89 host, mesox, 10.34.182, IBD, manager 69, 10.34.1.85 and Ag60654. The SwS4 access path is composed by the 10.34.1.89 host, mesox, 10.34.182 host, IBD, manager 69, 10.34.1.83 host and Ag60664. The only differences in the paths are the agents hosts and the agents. Therefore, the 10.34.1.83 and Ag60664 are the main modules responsible for damaging the SwS4 availability. Table 1 shows the host 10.34.1.83 had a big failure rate and a small repair rate. While agent Ag60664 had a medium failure rate, almost the same failure rate for agent Ag60654, Ag60664 had a much larger repair rate.

Switching system	SwS1	SwS2	SwS3	SwS4	SwS5	SwS6	SwS7
Limiting availability	0.95310	0.94606	0.96018	0.90368	0.95970	0.95853	0.95947

Table 2: Switching system limiting availability

6.2. Main modules responsible for damaging the system availability

Tables 5 and 6 show the main modules responsible for decreasing the SwS3 and SwS4 availability, respectively. In SwS3, the Mesox module fault state had the biggest probability to happen comparing with the other state faults. In SwS4, the agent fault state had the biggest probability to happen comparing with the other state faults. Therefore, the agent is the main module responsible for decreasing the SwS3 availability and mesox is the main responsible for SwS4 availability decrease.

State	Probability	Fault component
2	0.03548	Mesox
5	0.0035128	Manager
4	0.00040343	IBD
16	0.00012982	Mesox and Manager

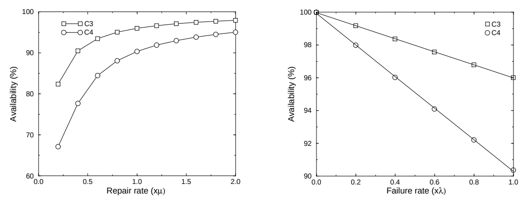
Figure 5: Main components responsible for SwS3 availability decrease

State	Probability	Fault components
7	0.054221	Agent
2	0.033397	Mesox
5	0.0033062	Manager
18	0.0020038e	Mesox and Agent
34	0.00019837e	Manager and Agent
43	0.00019837	Agent and IBD
6	0.00017078	10.34.1.83
16	0.00012218	Mesox and Manager

Figure 6: Main components responsible for SwS4 availability decrease

6.3. Changing the systems parameters

The SwS3 and SwS4 access path components had their failure and repair rates changed and the new system availability was measured. Initially, the failure rates were kept and each component repair rate was reduced (increased) in 20%, 40%, 60%, 80%, 100% of their initial value, followed by the limiting availability measure. Figure 6.(a) shows the SwS3 (squares) and SwS4 (circle) availability variation. The x axis goes from 0.2 to 2.0 corresponding, respectively, to 80% failure repair reduced and 100% repair rate increased $(y = 2.0 * \mu)$. The plot shows the SwS3 availability is always bigger than the SwS4 availability. At $(x = 0.2 * \mu)$, SwS3 availability is 82.39% and SwS4 availability is 67.126%. At $(x = 2.0 * \mu)$, SwS3 availability is 97.97%, and SwS4 availability is 95.06%, maximal values. As expected, the system availability increases when the repair rate increases.



(a) Availability × Repair rate

(b) Availability × Failure rate

Figure 7: Availability analysis

In the next step, the repair rates were kept and SwS3 and SwS4 access path components failure rates were reduces in 20%, 40%, 60%, 80% and 100% of their initial value and the availability was measured. Figure 6.(b) shows the SwS3 (square) and SwS4 (circle) availability variation. The X axis goes from 0.0 to 1.0. At x=0.8, for example, the fault rates were reduced in 20% $(x = 0.8 * \lambda)$. Consistently, the SwS3 availability is always bigger than SwS4 availability. The system availability has a linear relation with the system failure rate.

Finally, the main modules responsible for damaging the SwS3 and SwS4 repair rates were changed. As showed in section 6.2., these modules are Mesox and agent for SwS3 and SwS4, respectively. Their repair rates were increased in 20%, 40%, 60%, 80% and 100% of their initial values. Figure 8 shows (i) the comparison between changing all the SwS3 system components repair rates (empty squares) and only changing the Mesox repair rate (full squares), and (ii) the comparison between changing all SwS4 system components (empty circle) and only changing the Agent repair rate (full circle).

Note that changing only the SwS4 Agent does not increase the availability as much as changing all the components, while by changing the Mesox component in SwS3 there was almost the same availability as changing all the components. This result shows that for increasing the SwS3 availability it is necessary to invest only in the Mesox module improvement, and not in all the others components.

7. Conclusions and future work

This work has defined a methodology to analyze the availability of a distributed management telecommunication system (SIS) using the Markov mathematical model. The methodology makes it possible to calculate the availability of specific parts of SIS and to identify the main hardware and software modules responsible for

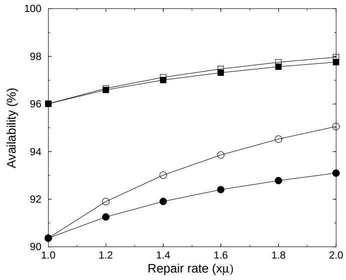


Figure 8: Availability × Changing the repair rate

decreasing the system's availability. It is also possible to simulate actions for availability changes, defining the best action for availability improvement. The methodology was applied to part of SIS (a particular geographical area) for the analysis of the entire system it would imply a much larger number of states, thus requiring a larger computational effort. For the complete analysis, it is necessary to study how to calculate the entire system availability based on the simulations of the many parts of the system.

References

[1] W.Meira, P.A. Oliveira, J.M. Nogueira, C.M. Machado. Performance monitoring of management systems, IEEE LANOMS, Rio de Janeiro, 1999.

[2] M. Satyanrayanan. *Dygest of Proceedings, 7th IEEE Workshop on Hot Topics in OS (HotOS-VII)*, Rio Rico, AZ, March, 1999.

[3] S.R. Welke, B.W. Johnson, J.H. Aylor. Reliability modeling of hardware/software systems, IEEE Trans on Reliability R-44(1995)413-418.

[4] A.L. Goel, J. Soenjoto. Models for hardware/software operational performance evaluation, IEEE Trans on Reliability R-30(1992)232-239.

[5] Brown, A. "Towards Availability and Maintainability Benchmarks: A Case Study of Software RAID Systems." *UC Berkeley Masters Report*, also available as *UCB CS Division TR UCB//CSD-01-1132*, Berkeley, CA, January 2001.

[6] Nogueira, Jose Marcos and Meira, Dilmar. The SIS project: A distributed platform for the integration of telecommunication management systems. IEEE/IFIP 1996 Network Operations and Management Symposium – NOMS'96, 1996.

[7] ITU-T home-page - http://www.itu.org.

[8] F. Piedad, M. Hawkins. High Availability - Design, Techniques and Processes, Prentice Hall PTR, 2001.

[9] K. S. Trivedi, S. Hunter, S. Garg, R. Fricks. Reliability Analysis Techniques Explored Through a Communication Network Example, CACC - The Center for Advanced Computing and Communication, Technical Report Series – TR 96/32.

[10] Jonh C. S. Lui and Richard R. Muntz. Computing Bounds on Steady State Availability of Reparaible Computer Systems, 1994 ACM 004-5411/94/0700-0676

[11] R.A. Sahner, K.S. Trivedi, A. Puliafito. Performance And Reliability Analysis of Computer System - An Example-Based Approach Using the SHARPE Software Package, Kluwer Academic Publishers, Massachussets, 1996.

[12] R.A. Sahner, K.S. Trivedi, SHARPE: Symbolic Hierarchical Automated Reliability and Performance Evaluator - Introduction and Guide for Users, 1992.

[13] C.D. Lai, M. Xie, K.L. Poh, Y.S. Dai, P. Yang. A model for availability analysis of distributed software/hardware systems, Information and Software Technology 44(2002)343-350.

[14] J.C. Laprie, K. Kanoun. X-Ware Reliability and Availability Modeling, IEEE Transaction on Software Engineering, vol. 18, No. 2, (1992) 651-664.

[15] T.T. Soong. Probabilistic Modeling and Analysis in Science and Engineering, Jonh Wiley and Sons, 1981.