1997 Pacific Rim International Symposium on Fault-Tolerant Systems, Taipei, Taiwan, Dec. 15-16, 1997, pp. 85-90.

Engineering Oriented Dependability Evaluation: MEADEP and Its Applications

Dong Tang, Myron Hecht, Jeffrey Agron, Jeffrey Miller and Herbert Hecht
SoHaR Incorporated, Beverly Hills, California, USA

Abstract

MEADEP is a user-friendly dependability evaluation tool for measurement-based analysis of computing systems. Features of MEADEP include: a data processor for converting data in various formats to the MEADEP format, a statistical analysis module for graphical data presentation and parameter estimation, a graphical modeling interface for building reliability block diagrams and Markov chains, a library of dependability models for constructing customer models, and a model solution module for availability/ reliability calculations with graphical parametric analysis. Use of the tool on failure data from measurements can provide objective evaluations of dependability for critical systems, while greatly reducing requirements for specialized skills in data processing, analysis, and modeling from the user. MEADEP has been applied to evaluate availability for two air traffic control systems based on operational failure data and results produced by MEADEP have provided valuable feedback to the project management of these critical systems. MEADEP has also been used to analyze a nuclear power plant safety model, based on the Eagle 21 architecture and its early field failure data, and results of sensitivity analysis on the model are discussed.

1. Introduction

Automation of dependability evaluation has been realized by the computer engineering community for over 15 years during which many dependability modeling tools were developed [1, 6]. Some of the representative tools are SAVE [2], SHARPE [8], and UltraSAN [10]. The emergence of these tools has given impetus to the applications of advanced modeling and evaluation techniques. These tools were developed for building models based on parameters, with emphasis on model solution techniques. Although many theoretical modeling and solution issues such as non-exponential failure arrival/recovery times and numerical stiffness[1] have been addressed, practical issues such as data analysis, parameter estimation and graphical user interface (GUI) have rarely been considered in these tools. An exception in the GUI aspect is UltraSAN.

Given a tool in the above category and a set of field failure data (typically in a database format), reliability engineers often hesitate to make use of the tool and data to generate a quantitative evaluation of dependability. This is due to the difficulties involved in data processing, parameter estimation (especially for cases where failures are rare), model specification for the tool, and appropriate mapping from data to models. The need to develop tools that can reduce these difficulties has thus been apparent.

From the research viewpoint, a capable tool should be able to handle any probability distribution. However, this capability is rarely used in practical engineering projects that require field measurements for the following reasons: First, it is very difficult to measure and identify failure arrival distributions for each component. When failures are rare and distributed over multiple replicas, this identification is impossible. Second, errors caused by using the exponential approximation of actual distributions may not be as significant as errors introduced in other evaluation steps such as measurement, parameter estimation and model construction. In our experience, a dependability assessment that is accurate to the right order of magnitude is good enough in practice, taking account of various possible errors. It is thus questionable, if not impossible, to make costly efforts to identify all actual distributions to correct minor errors. Third, in order to have a conservative evaluation, upper bounds based approaches (failure rate upper bounds, instead of means) are usually used in engineering. Official standards (e.g., Military Handbooks) for estimating failure rate confidence intervals typically assume the exponential distribution, and these standards have been followed by reliability engineers for many years.

Thus, from the engineering viewpoint, it is desirable to have software tools which integrate data processing, statistical analysis, reasonable dependability modeling and evaluation,[2] and a user-friendly interface to provide non-expert users with an easy-to-operate environment for producing quantitative dependability evaluations for real systems. This paper introduces just such a newly developed tool — MEADEP (MEAsure DEPendability). The purpose of this development

[1]In a Markov dependability model, failure rates tend to be very small and recovery rates tend to be much larger. Stiffness means the technical difficulty in model solution caused by the difference between the largest and the smallest parameters in the model.

[2]The exponential distribution is typically assumed in the model evaluation.

was to facilitate the use of measurement-based dependability analysis methods [4, 11] and to reduce the cost of such analyses so that they can become an integral part of engineering projects where dependability is an important consideration.

2. Overview of MEADEP

Figure 1 is a layout of MEADEP. The *Data PreProcessor* (DPP) module, interacts with the user to convert source data to MEADEP internal data. The source data can be manually generated structured trouble reports or computer generated event logs. The *Data Editor and Analyzer* (DEA) module is used to edit internal data and to perform statistical analysis on the data. Parameter values estimated from the data by this module can be inserted into the text modeling file generated by another module, the *Model Generator* (MG). The MG module provides a graphical user interface for the user to draw model diagrams and then to generate, from the diagrams, a text modeling file that contains model specifications suitable for solution. Model diagrams can be imported from library files containing predefined models to save development time. The *Model Evaluator* (ME) module produces results based on the model specifications and parameters in the text modeling file. All modules are integrated with the *Graphical User Interface* (GUI).

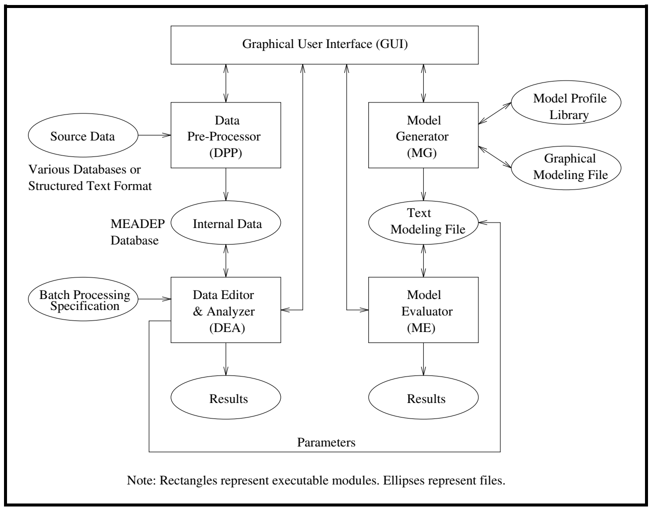

Graphical User Interface (GUI)
Source Data
Various Databases or Structured Text Format
Data Pre-Processor (DPP)
Model Generator (MG)
Model Profile Library
Graphical Modeling File
MEADEP Database
Internal Data
Text Modeling File
Batch Processing Specification
Data Editor & Analyzer (DEA)
Model Evaluator (ME)
Results
Results
Parameters
Note: Rectangles represent executable modules. Ellipses represent files.

Figure 1 Layout of MEADEP

MEADEP has the following features:

Support for data conversion: Structured data in a variety of formats (ASCII Delimited Text, Access, dBASE, Paradox, etc.) can be converted to the MEADEP data format.

Estimation of parameters from data: Typically used parameters (failure rate, coverage, etc.) and their upper and lower bounds at a certain level of confidence are estimated by statistical routines taken from mature numerical libraries.

Graphical presentation of data: A number of graphical formats are provided to display dependability characteristics for data (e.g., pie charts and histograms).

Graphical Input of models: A graphical "drag and drop" interface allows the user to create models hierarchically, out of reliability block diagrams (including the k-out-of-n block) and Markov reward models [3].

Parametric analysis in solution: The model solution portion of MEADEP allows a model to be run with a range of user-specified values for a selected parameter, and the results can be displayed graphically.

A library of dependability models: A library of dependability models, including primitive models for typical fault-tolerant architectures and complex models for real critical systems, are included in MEADEP for reuse by users.

Graphical user interface: For all of its functions, MEADEP provides a user-friendly GUI featuring menus, dialogs, pictures, printing previews, and extensive on-line help information.

MEADEP is developed based on the following software environments and tools: Microsoft Windows 95, Visual C++, Open Database Connectivity (ODBC) interface, IMSL C Numerical Libraries and the Olectra Chart graphical package. Because of the incorporation of these techniques, MEADEP provides many user-friendly features and can interface with various database formats, thereby reducing difficulties in model development and data conversion. Further details on the MEADEP structure are discussed in [14].

3. Numerical Methods Used in MEADEP

This section discusses parameter estimation and model evaluation methods used in MEADEP.

3.1 Parameter Estimation Methods

Three essentially different types of parameters can be estimated from data by MEADEP: failure rate or mean time between failures (MTBF), recovery rate or mean time to recovery/repair (MTTR), and coverage, including cases where failures are rare. For each type, the sample mean, a lower bound and an upper bound at a certain level of confidence are provided, whenever applicable.

For the failure rate estimation, the exponential MTBF distribution is assumed. If n failure events are included in the data for the measurement period T, the failure rate upper bound, λ_U, and lower bound, λ_L, at the $100(1-\alpha)\%$ confidence level (α is the significance level) are given by [5]

$$\lambda_L = \frac{\chi^2_{\alpha;2n}}{2T}, \quad \lambda_U = \frac{\chi^2_{1-\alpha;2n+2}}{2T} \tag{1}$$

where χ^2 represents the Chi-square distribution. The above λ_U

formula is also applicable to the case in which n is zero, i.e., no failure occurred in T. When n is zero, this formula is equivalent to the following estimator given in [12]:

$$\lambda_U = \frac{-\ln(\alpha)}{T} \tag{2}$$

For MTTR, either the exponential or normal distribution can be assumed. In the first case, estimators similar to Equation (1) are used. In the second case, the student's t distribution is used. Compared to the normal distribution, the student's t distribution provides better estimates when the variance is unknown and the sample size is less than 25 [5]. The estimators for the lower and upper bounds are the following:

$$MTTR_L = M - t_{\alpha;n-1}\sqrt{\frac{S}{n}}\ ,\ MTTR_U = M + t_{\alpha;n-1}\sqrt{\frac{S}{n}} \tag{3}$$

where t represents the student's t distribution, α is the significance level, and n, M and S are the sample size, sample mean and sample variance, respectively.

For the coverage C, the binomial distribution is used. If the number of successes, s, in n trials is greater than zero and less than n, the lower bound (C_L) and upper bound (C_U) of C can be approximated by [5]

$$C_L = \frac{1}{1 + \frac{n-s+1}{s} F_{1-\alpha;2(n-s)+2;2s}}\ ,\ C_U = \frac{1}{1 + \frac{n-s}{s+1} F_{\alpha;2(n-s);2s+2}} \tag{4}$$

where F represents the F distribution and α is the significance level. If s equals n (a 100% coverage), a conservative lower bound is given by [12]

$$C_L = \alpha^{\frac{1}{n}} \tag{5}$$

3.2 Model Evaluation Methods

Models are developed hierarchically from the top level to the bottom level, but the evaluation has to be performed from the bottom level to the top level. In the hierarchy tree, each node is a model representing a system or subsystem. For each node, four measures — failure rate (λ), recovery rate (μ), availability (A), and reliability (R) — are evaluated by MEADEP. At the bottom level, failure rates and recovery rates for all modeled components (elemental blocks) are given by the user or obtained from data. For a Markov model, all transition rates, the initial state, and the failure state are also specified by the user. Based on these parameters, the four measures are evaluated from bottom to top using methods discussed below.

Assume a reliability block diagram consists of n blocks connected either in series or in parallel. The four measures for block i are denoted by λ_i, μ_i, A_i and R_i. If block i is an elemental block, A_i and R_i are calculated by

$$A_i = \frac{\mu_i}{\lambda_i + \mu_i}\ ,\quad R_i = e^{-\lambda_i T} \tag{6}$$

where T is a time interval (an integer representing hours) specified by the user during evaluation. If block i is decomposed into a lower level diagram, λ_i, μ_i, A_i and R_i are calculated using the formulas for block diagrams described below.

Let λ, μ, A and R denote the four measures for the block diagram. If the n blocks in the diagram are connected in series, A and R are calculated by

$$A = \prod_{i=1}^{n} A_i\ ,\quad R = \prod_{i=1}^{n} R_i \tag{7}$$

If the n blocks are connected in parallel, A and R are calculated by

$$A = 1 - \prod_{i=1}^{n}(1 - A_i)\ ,\quad R = 1 - \prod_{i=1}^{n}(1 - R_i) \tag{8}$$

No matter whether the n blocks are connected in series or in parallel, λ and μ are calculated by

$$\mu = \sum_{i=1}^{n} \frac{\lambda_i \mu_i}{\sum_{j=1}^{n} \lambda_j}\ ,\quad \lambda = \frac{\mu(1-A)}{A} \tag{9}$$

For a Markov chain with n states, the availability A is calculated from state reward rates (r_i) and occupancy probabilities (p_i):

$$A = \sum_{i=1}^{n} r_i\, p_i \tag{10}$$

where r_i is defined by the user and p_i is obtained by solving the Q matrix (infinitesimal generator) based equations (Eq. 8.13 and 8.14 in [15]). Since reward rate is used, partially available states are allowed in the model. The reliability R at time T (an integer representing hours) is calculated based on the uniformization technique [7] and the Chapman-Kolmogorov equation:

$$R = \sum_{i=1}^{n} r_i\, p_i(T)\ ,\quad P(T) = P(0)U^T \tag{11}$$

where $P(T)=(p_0(T), p_1(T), ..., p_n(T))$ is the state probability vector at time T, $P(0)$ is the initial state probability vector, and U is the unit time transition probability matrix converted from the Q matrix by using the uniformization technique and by setting the failure state to the absorbing state:

$$U = (Q/2^s + I)^{2^s} \quad \textbf{(12)}$$

where s is the smallest integer such that 2^s is greater than the largest element, q_{max}, in Q. Since T is an integer, it can be expressed as the following sum:

$$T = \sum_{i=0}^{m} C_i 2^i \quad \textbf{(13)}$$

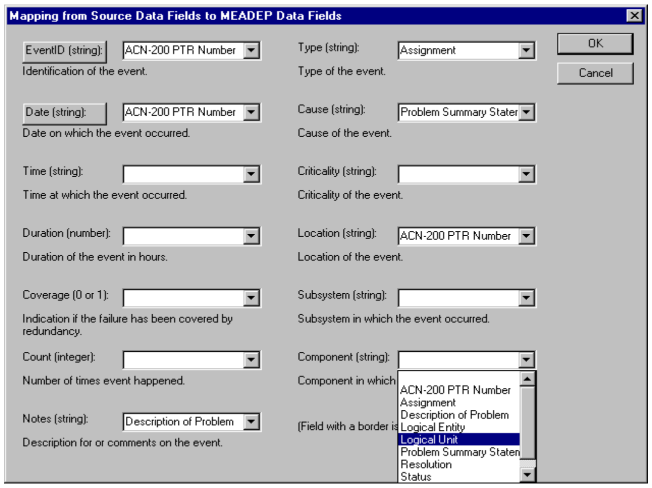
where C_i (coefficient) is either 1 or 0 and m is the maximum integer such that $2^m < T$. Thus, the computation of U and U^T can be done by a matrix squaring iteration [7] for $s+m$ times. The computation complexity for this algorithm is $O(n^3(s+m))$, or $O(n^3 \lg(q_{max}T))$, where n is the number of states in the Markov chain and is restricted to a maximum of 100 in the current MEADEP version. Normally we have $s<10$. T is restricted to a maximum of 10^9 hours (10^5 years) in MEADEP and m is thus bounded by 30. For the two remaining measures of the Markov chain, the recovery rate μ is simply the transition rate out of the failure state, and the failure rate λ is calculated from A and μ by Eq. (9).

For several test cases, including the complex VSCS availability model to be discussed in the next section and a Markov model that shows a certain degree of stiffness (having a failure rate of 10^{-8} and a recovery rate of 10^2), the steady-state and transient results produced by MEADEP matched those produced by SHARPE [9].

4. Applications of MEADEP

MEADEP has been used to evaluate availability for two major air traffic control (ATC) systems: the Voice Switching and Control System (VSCS) and the Air Route Traffic Control Center (ARTCC). The VSCS is a digital communication system responsible for voice switching between pilots and air traffic controllers. The modeled ARTCC components include a set of radars, radar data processing and display subsystems. Both evaluations were based on measurements from field systems and provided valuable feedback to the project management. Details on these evaluations can be found in [14]. Here we present two figures in this work for illustrating the data analysis functions of MEADEP.

Figure 2 shows a data conversion screen which maps the VSCS Program Trouble Reports (PTRs) data fields that are useful for parameter estimation to the MEADEP data fields. Figure 3 shows the distribution histogram of time between outage events for a particular radar site, generated by MEADEP. MEADEP allows the user to super-plot, over a histogram, five different analytical probability distribution functions (pdf) determined by the sample mean and sample variance: exponential, gamma, Weibull, normal and lognormal. Meanwhile, the estimated parameters for these functions as well as the results of the Chi-Square and Kolmogorov-Smirnov goodness-of-fit tests [4] are displayed. The curves plotted on this figure are the exponential, gamma, and Weibull functions. All three functions passed both Chi-Square and Kolmogorov-Smirnov goodness-of-fit tests at the 0.1 significance level.


Mapping from Source Data Fields to MEADEP Data Fields

EventID (string): ACN-200 PTR Number
Identification of the event.

Date (string): ACN-200 PTR Number
Date on which the event occurred.

Time (string):
Time at which the event occurred.

Duration (number):
Duration of the event in hours.

Coverage (0 or 1):
Indication if the failure has been covered by redundancy.

Count (integer):
Number of times event happened.

Notes (string): Description of Problem
Description for or comments on the event.

Type (string): Assignment
Type of the event.

Cause (string): Problem Summary Stater
Cause of the event.

Criticality (string):
Criticality of the event.

Location (string): ACN-200 PTR Number
Location of the event.

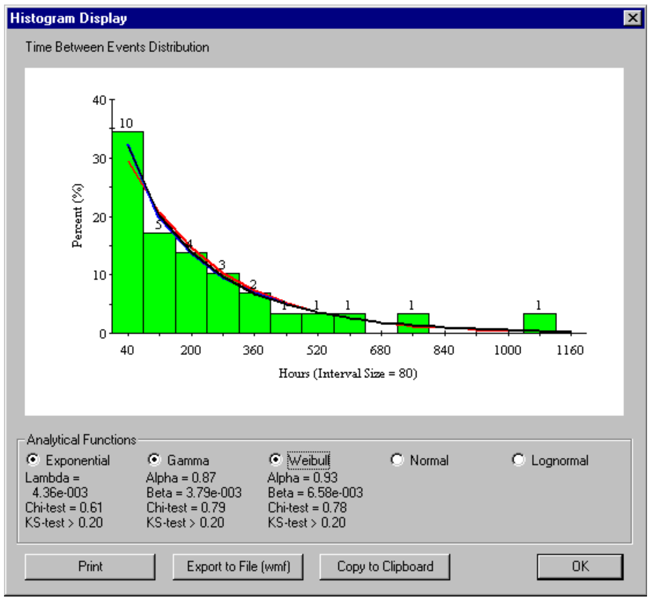
Subsystem (string):
Subsystem in which the event occurred.

Component (string):
Component in which

(Field with a border is

ACN-200 PTR Number
Assignment
Description of Problem
Logical Entity
Logical Unit
Problem Summary Staten
Resolution
Status

Figure 2 Data Conversion from VSCS PTRs to MEADEP Format


Histogram Display

Time Between Events Distribution

Analytical Functions

[✓ Exponential]	[✓ Gamma]	[✓ Weibull]	Normal	Lognormal
Lambda = 4.36e-003	Alpha = 0.87	Alpha = 0.93		
	Beta = 3.79e-003	Beta = 6.58e-003		
Chi-test = 0.61	Chi-test = 0.79	Chi-test = 0.78		
KS-test > 0.20	KS-test > 0.20	KS-test > 0.20		

Figure 3 Distribution of Time Between Outages for a Radar Site

Now we demonstrate how MEADEP can be used on safety systems. The modeled configuration has two major components: a nuclear plant and a digital safety system which protects the plant by responding to and processing challenges from the plant instrumentation. A 3-level hierarchical model was developed for this configuration, where levels 2 and 3 were based on the architecture of a real digital protection system — Eagle 21 [16].

Figure 4 shows the top level plant model which reflects the intermittent operating profile of safety systems (the heavy frame in this diagram means parameters λ_{ss} and μ_{ss} are evaluated from the lower level model SafSys, similar to the next diagram). Figure 5 shows the middle level model, a safety system which consists of four channels working on a basis of 2-out-of-4 votes for a reactor trip (shutdown). In this model, channel failures are assumed to be of the Byzantine type[3], because this type is the worst case failure mode and is hazardous to the protection function. The bottom level channel model is a diagram of reliability blocks representing the major components of Eagle 21. Failure rates for these components were estimated from field data.

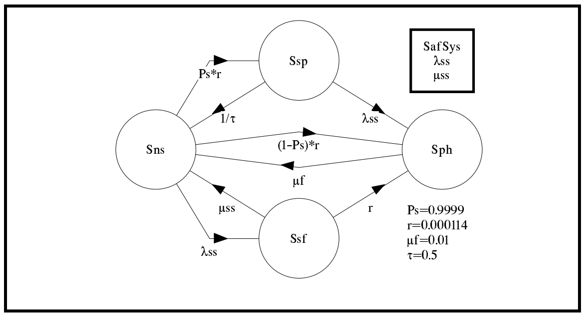


Figure 4 The Nuclear Plant Model

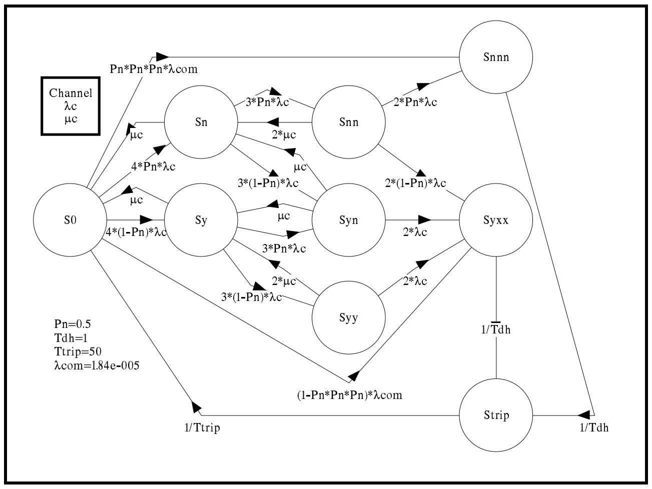


Figure 5 The Safety System Model (SafSys)

The notation used in the figures is as follows:

S_{ns}	Normal/safe state in which either both plant and safety system are functioning within technical specifications or the plant is in a safe trip (reactor is shut down safely)
S_{sp}	Safety processing state in which the safety system is processing a challenge
S_{sf}	Safety failure state in which the safety system is not able to respond to a challenge properly while the plant is functioning within technical specifications
S_{ph}	Plant hazard state which is the result of a failure of the safety system to process a challenge successfully in terms of initiating a necessary reactor trip
P_S	Probability of success upon demand, i.e., the safety system will be successful in responding a challenge (initially set to 0.9999)
r	Arrival rate of challenges from the plant requiring a response of the safety system (assumed to be once a year, a typical value)
τ	Challenge processing time (assumed to be a half hour, a conservative assumption)
λ_{ss}	Failure rate of the safety system (evaluated from the middle level model)
μ_{ss}	Rate for detection and handling of a safety system failure (evaluated from the middle level model)
μ_f	Recovery rate of the plant after a hazardous event
S_0	Normal state in which all four channels are functioning properly
S_n	State in which one channel has failed and the output of the failed channel votes for "no trip"
S_y	State in which one channel has failed and the output of the failed channel votes for "trip"
S_{nn}	State in which two channels have failed and both failed channels vote for "no trip"
S_{yn}	State in which two channels have failed and one failed channel votes for "trip" and another failed channel votes for "no trip"
S_{yy}	State in which two channels have failed and both failed channels vote for "trip"
S_{nnn}	State in which at least three channels have failed and at least three failed channels vote for "no trip"; This state is equivalent to state S_{sf} in Figure 4, because the safety system would generate a "no trip" signal should a challenge arrive.
S_{yxx}	State in which three channels have failed and at least one of the failed channels votes for "trip"
S_{trip}	Plant trip state (reactor is shut down)
P_n	Probability that the channel output votes for "no trip", given a channel failure (assumed to be 0.5)
λ_c	Failure rate of a channel (evaluated from the bottom level model)
μ_c	Recovery rate of a channel (evaluated from the bottom level model)
λ_{com}	Common mode failure rate for the safety system (80% confidence upper bound based on "no common mode failures for 10 years")
T_{dh}	Failure detection and handling time, given that at least three channels have failed (assumed to be one hour)
T_{trip}	Plant trip duration (assumed to be 50 hours)

The data source for this analysis is the failure reports generated for the early use of Eagle 21 in Unit 1 and Unit 2 at the Sequoyah nuclear power plant during a 2-year period [13]. The available data only allowed estimation of upper bounds for failure rates in the bottom level model. Other parameters were assumed to take typical or conservative values, as shown in the figure, for demonstration purposes.

The dependability measure to evaluate in this analysis is the plant Mean Time Between Hazards (MTBH), i.e., the mean time to state S_{ph} (Figure 4), which represents a failure of the safety system to initiate a necessary reactor trip in response to a challenge due to its computer hardware or software (design or random) faults. The MEADEP parametric

[3]The faulty channel continues execution and lies when asked for information [11].

analysis functionality was used on the above model to investigate the impact of the following three parameters upon the plant MTBH: (1) the safety system common mode failure rate λ_{com}, (2) the safety system failure detection and handling time T_{dh}, and (3) the probability of success upon demand P_S. As each of these parameters was selected for sensitivity study, it was varied within a reasonable range, and all other parameters were left unchanged.

The results showed that the plant MTBH is not very sensitive to λ_{com} and T_{dh}. For the selected parameter ranges, the variance of MTBH is about 8% and 2%. However, the plant MTBH is extremely sensitive to P_S: when P_S increases from 0.999 to 1, MTBH increases from 1000 years to 291,000 years, i.e., an increase of 290 times (Figure 6). The largest increment segment is between 0.99999 and 0.999999 (from 74,500 years to 225,600 years), and achieving a value in this range is the most rewarding. It is clear that the most important parameter is P_S, the probability of success upon demand, and achieving a high value and estimating the achieved value for this parameter should be a key effort in the system development.

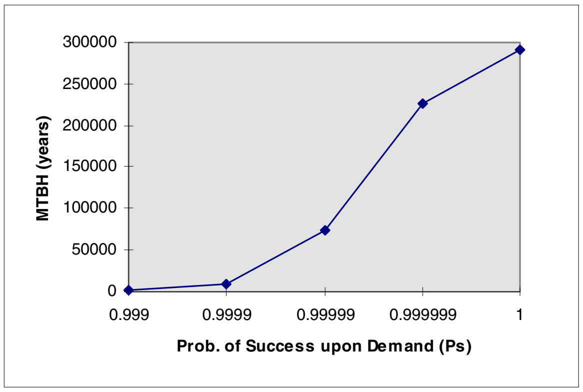


Figure 6 Sensitivity of Plant MTBH to P_S

5. Conclusion

In this paper, we presented a user-friendly measurement-based dependability evaluation tool — MEADEP. The development of MEADEP integrated techniques in GUI programming, database engineering, dependability modeling, and statistical/numerical analysis. Features of MEADEP include: converting data in various formats to the MEADEP format, graphical data presentation and parameter estimation, graphical construction of reliability block diagrams and Markov chains, a library of dependability models, and availability/reliability calculations with graphical parametric analysis. Use of the tool on failure data from measurements can provide objective evaluations of dependability for critical systems, while greatly reducing requirements for specialized skills in data processing, statistical analysis, and dependability modeling from the user.

Acknowledgments

This work was supported partially by the U.S. Nuclear Regulatory Commission under Contract NRC-04-95-081 and partially by the Federal Aviation Administration under Contract DTFA01-93-Y-0169, Subcontract 96-107.

References

[1] R. Geist and K. S. Trivedi, "Reliability Estimation of Fault-Tolerant Systems: Tools and Techniques," *IEEE Computer*, July 1990, pp. 52-61.

[2] A. Goyal, et al., "The System Availability Estimator," *Proceedings of the 16th International Symposium on Fault-Tolerant Computing*, June 1986, pp. 84-89.

[3] A. Goyal, S. S. Lavenberg and K.S. Trivedi, "Probabilistic Modeling of Computer System Availability," *Annals of Operations Research*, No. 8, March 1987, pp. 285-306.

[4] R. K. Iyer and D. Tang, "Experimental Analysis of Computer System Dependability," *Fault-Tolerant Computer System Design*, D. K. Pradhan (Ed.), Prentice Hall PTR, Upper Saddle River, NJ, 1996, pp. 282-392.

[5] D. Kececioglu, *Reliability and Life Testing Handbook*, Vol. 1 & 2, PTR Prentice Hall, Englewood Cliffs, NJ, 1993.

[6] J. F. Meyer, "Performability: A Retrospective and Some Points to the Future," *Performance Evaluation*, Vol. 14, Feb. 1992, pp. 139-156.

[7] A. Reibman and K. S. Trivedi, "Numerical Transient Analysis of Markov Models," *Computational Operations Research*, Vol. 15, No. 1, 1988, pp. 19-36.

[8] R. A. Sahner and K. S. Trivedi, "Reliability Modeling Using SHARPE", *IEEE Transactions on Reliability,* vol 36, February 1987, pp. 186-193.

[9] R. A. Sahner, K. S. Trivedi and A. Puliafito, *Performance and Reliability Analysis of Computer Systems: An Experimental-Based Approach Using the SHARPE Software Package*, Kluwer Academic Publishers, 1996.

[10] W. H. Sanders, W. D. Obal II, M. A. Qureshi, and F. K. Widjanarko, "The UltraSAN Modeling Environment," *Performance Evaluation*, Vol. 24, No. 1, Oct/Nov 1995, pp. 89-115.

[11] D. P. Siewiorek and R. W. Swarz, *Reliable Computer Systems: Design and Evaluation, Digital Press*, Bedford, Mass., 1992.

[12] D. Tang and M. Hecht, "Evaluation of Software Dependability Based on Stability Test Data," *Proceedings of the 25th International Symposium on Fault-Tolerant Computing*, Pasadena, CA, June 1995, pp. 434-443.

[13] D. Tang, M. Hecht, H. Hecht, and R. Brill, "Measurement-Based Dependability Evaluation for Safety-Grade Digital Systems," *Proceedings of the 1996 American Nuclear Society International Topical Meeting on Nuclear Plant Instrumentation, Control and Human-Machine Interface Technologies*, May 1996, pp. 535-542.

[14] D. Tang, M. Hecht, J. Handal and L. Czekalski, "MEADEP and Its Applications in Evaluating Dependability for Air Traffic Control Systems," *Proceedings of the 1998 Annual Reliability and Maintainability Symposium*, Anaheim, California, Jan. 1998.

[15] K. S. Trivedi, *Probability & Statistics with Reliability, Queuing, and Computer Science Applications*, Prentice-Hall, Englewood Cliffs, NJ, 1982.

[16] *EAGLE 21 Technical Description*, Westinghouse Electric Corporation, Process Control Division.