



***MEADep Application Note:***

---

# **Availability Modeling for a Networking Device using MEADep**

July, 2000

**DRAFT**

SoHaR Incorporated  
8421 Wilshire Blvd., Suite 201  
Beverly Hills, CA 90211  
**323-653-4717**

[www.sohar.com/meadep](http://www.sohar.com/meadep)  
[info@sohar.com](mailto:info@sohar.com)

## Table of Contents

Introduction .....	1
Model Description.....	1
CPU Subsystem.....	2
WAN Interface Subsystem .....	4
Power Subsystem .....	6
Results .....	8
Parametric Studies.....	8
Conclusions .....	10
MEADEP Text Modeling File.....	11

## List of Figures

Figure 1. Top Level Block Diagram .....	1
Figure 2. Model Hierarchy .....	2
Figure 3. Example Subsystem Top Level Diagram .....	2
Figure 4. CPU Subsystem Lower Level Model .....	2
Figure 5 . Markov Model for the Single Board Computer (SBC) Subsystem .....	3
Figure 6 . Markov Model for the WAN Interface subsystem .....	5
Figure 7 . Dual Redundant Power Supply Subsystem.....	7
Figure 8. Single Power Supply .....	7
Figure 9. Alternative Representation of Dual Redundant Power Supply Subsystem with Different Failure Rates when a single supply is supplying the entire load. ....	7
Figure 10. Impact of SBC Coverage on System Availability .....	9
Figure 11. Impact of Operating System MTBF on System Availability .....	9
Figure 12 . Impact of Hardware MTTR on System Availability .....	9

## List of Tables

Table 1. Variables in CPU Model .....	3
Table 2. Transitions in CPU Model .....	3
Table 3. SBC Model Variables .....	4
Table 4. SBC Model Transitions.....	4
Table 5 . WAN Interface Variables.....	6
Table 6. WAN Interface Model Transitions .....	6
Table 7. Baseline Results.....	8

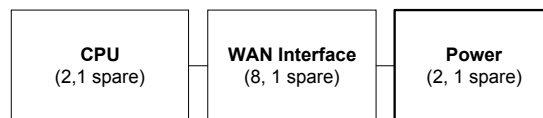
## Introduction

Availability and reliability are major concerns for networking devices, including routers, gateways, DSL modems, and ATM switches. This Application describes an example of such modeling with the MEADEP (Measurement-Based Dependability) Tool. The example model addresses not only device reliability but software failures and integration of capacity with availability in a single model.

The next section of this application note describes both the example networking device and the hierarchical reliability model which is used to represent it. The subsequent section discusses the results for both the baseline values and for several parametric analyses. The conclusions section, discusses how the model can be extended to include logistics constraints (spares level, travel time, spares replenishment time) and reliability growth. The MEADEP text modeling file is included in the appendix.

## Model Description

Figure 1 shows a block diagram for a generic data communication device that consists of 3 subsystems: Central Central Processing Unit (CPU), Wide Area Network (WAN) Interface, and Power. The CPU Subsystem consists of dual redundant Single Board Computers (SBCs), the WAN Interface consists of 8 line cards, including a redundant spare, and the Power Subsystem consists of dual redundant power supplies.

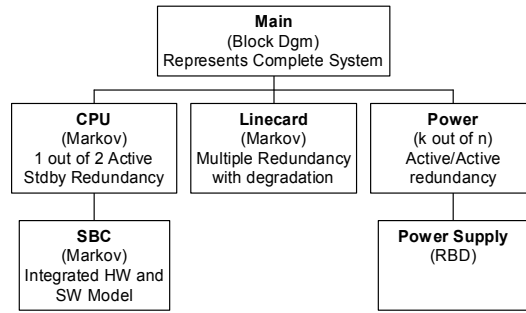


**Figure 1. Generic Data Communication Device Top Level Block Diagram**

The following additional assumptions will be used in the analysis:

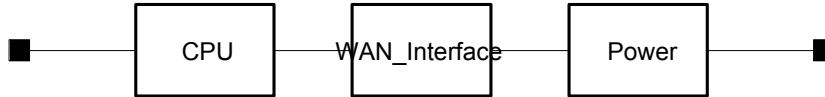
1. There is active/standby redundancy within the CPU subsystem with a recovery probability
2. There is active/active redundancy within the power subsystems and that the failure rate is constant if either the primary or backup power supply fails.
3. Within the WAN Interface Subsystem, we assume that loss of a single card has no impact, but loss of two or more cards causes a degradation based on the amount of capacity. That is, the loss of a second card causes a loss of  $1/N_{lc}$  where  $N_{lc}$  is the number of line cards.
4. SBCs are periodically polled either by their companions or by another connected monitor. When a software failure occurs, they are rebooted either from an internal command (e.g., the equivalent of a software watchdog timer) or from an external command
5. Boards are “hot swappable”, that is, the switch does not need to be brought down in order to replace a board.

With these assumptions, the system can be modeled hierarchically model as shown in Figure 2.



**Figure 2. Model Hierarchy**

Figure 3 shows the main or top level model of the communications device. The three blocks in the diagram directly respond to the system block diagram shown in Figure 1.

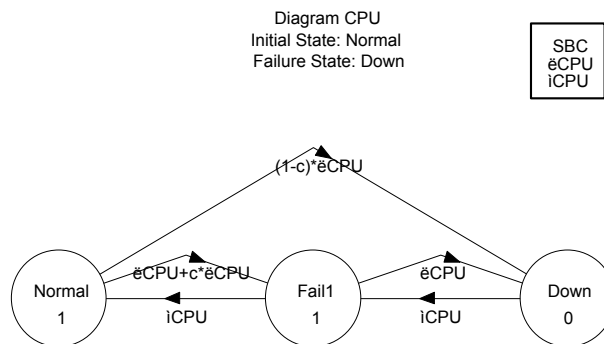


**Figure 3. MEADep Main (Top Level) Model**

Each of these blocks is a lower level model as will be described in the following section.

### **CPU Subsystem**

Figure 4 shows the first lower level diagram for the CPU subsystem. The Markov model represents an active/standby redundant system consisting of two Single Board Computers (SBCs). There are three states: Normal (both SBCs up), Fail1 (Single SBC up), and Down (both SBCs down). The numbers underneath the state names are rewards. States Normal and Fail1 have rewards of 1 because the device will continue to function in the presence of a single SBC failure. State Down has a reward of 0 because, with 2 SBCs failed, the device will no longer function. This model has a total of 4 input variables and 6 transitions that are described in Tables 1 and 2. The frame in the upper right hand corner of Figure 4 represents a reference to the lower level SBC model which will in turn define the hardware and software failure rates.



**Figure 4. CPU Subsystem Lower Level Model**

**Table 1. Variables in CPU Model**

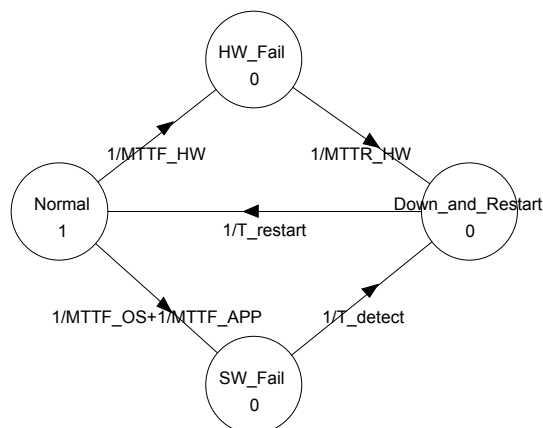
Variable	Baseline Value	Meaning
$\lambda_{CPU}$	TBD by Lower Level SBC model	Failure rate of the SBC board (combined hardware and software, see SBC model) that will be further defined in the lower level SBC model
c	0.9	Probability of successful detection and recovery of failed SBC
$\mu_{CPU}$	TBD by Lower Level SBC model	Restoration Rate of SBC (combined hardware and software, see SBC model) that will be further defined in the lower level SBC model

**Table 2. Transitions in CPU Model**

From	To	Transition	Comment
Normal	Failed1	$\lambda_{CPU} + c * \lambda_{CPU}$	$c * \lambda_{CPU}$ represents the probability of a successful switchover to the standby SBC after a failure of the active SBC. $\lambda_{CPU}$ represents the failure of a standby SBC
Normal	Down	$(1 - c) * \lambda_{CPU}$	This represents the failure to successfully switch to the standby SBC
Failed1	Down	$\lambda_{CPU}$	Represents the failure of an SBC running in a
Down	1	$\mu_{CPU}$	Repair/restoration rate of the SBC (see table 1)
Failed1	Normal	$\mu_{CPU}$	Repair/restoration rate of the SBC (see table 1)

Figure 5 shows the SBC model. The diagram has four states: Normal, Software failure (SW\_fail), Hardware failure (HW\_fail), and Down with restart. The software failure state is separated from the hardware failure state because of differences in recovery time. In the case of the software, recovery time the reciprocal of detection time. When a software failure is detected, the SBC will be restarted. However, in the case of a hardware failure, a technician must physically replace the SBC in order to restore service to that board. Hence, the transition rate is related to the hardware Mean Time to Repair, or MTTR\_HW. The Normal state has a reward of 1 because the SBC is functional. All of the other states have a reward of 0 because the device is failed. The Down\_and\_Restart state is designated as the failed state because it is common to all recovery paths

Table 3 lists the variables in the model and Table 4 describes the transitions.



**Figure 5 . Markov Model for the Single Board Computer (SBC) Subsystem**

**Table 3. SBC Model Variables**

Variable	Baseline Value	Meaning
MTTF_HW	25000 hours	SBC Hardware mean time to failure (MTTF)
MTTF_OS	10000 hours	SBC operating system mean time to failure
MTTF_APP	5000 hours	Device application software mean time to failure
T_detect	0.1 hours	Time needed to detect an SBC failure and initiate a recovery
MTTR_HW	4 hours	Hardware repair time
T_restart	0.05 hours	Time to restart the system

**Table 4. SBC Model Transitions**

From	To	Transition	Comment
Normal	HW_Fail	1/MTTF_HW	Transition rate is the reciprocal of the hardware MTTF (equivalent of $\lambda_{HW}$ )
Normal	SW_Fail	1/MTTF_OS+1/MTTF_APP	Transition rate is the sum of the reciprocal of the application software and operating system MTTFs. Because the detection and recovery from either the application or the operating system (OS) is the same, the recovery time for both the application and the OS is the same
SW_Fail	Down_and_Restart	1/T_detect	Related to the reciprocal of the periodic polling or other message detecting the health of the communication device
HW_Fail	Down_and_Restart	1/MTTR_HW	Transition is the reciprocal of the hardware repair time (MTTR)
Down_and_Restart	Normal	1/T_restart	Transition is the reciprocal of the restart time after a hardware or a software recovery action

The CPU and SBC models (Figures 4 and 5) together demonstrate how software failures can be analytically modeled. We consider the probability of correlated and common mode software failures in the parameter  $c$  (Table 1). For such an event, both the active and standby SBCs will fail. However, in a mature and well-tested system, nearly all residual failures are difficult to reproduce (otherwise they would have been found and removed). Such failures are often due to specific sequences of events, states of registers, or unusual timing. As has been shown in other studies<sup>1</sup>, most of these residual faults result in uncorrelated failures that behave like transient hardware failures. Thus, they can be modeled as random failures using the stochastic modeling techniques.

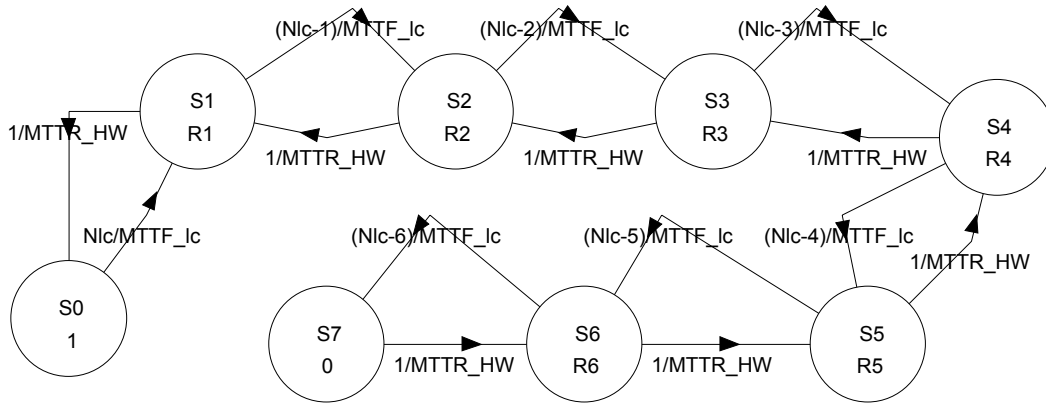
### ***WAN Interface Subsystem***

The WAN interface subsystem consists of eight<sup>2</sup> line cards. The failure of a single line card does not impact the system. However, failures of multiple line cards causes a degradation in capacity.

<sup>1</sup> See for example D. Tang 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; and in Inwhan Lee and Ravi K. Iyer, "Software Dependability in the Tandem GUARDIAN System," in *IEEE Transactions on Software Engineering*, May 1995, pp. 455-467.

<sup>2</sup> Or, in general,  $Nlc$ , where  $Nlc$  is an arbitrary number of line cards

Figure 6 shows a Markov model of the line card subsystem and demonstrates how these failure effects are represented in a MEADep model.



**Figure 6 . Markov Model for the WAN Interface subsystem**

The model has 8 states, S0 through S7, representing the failure of 0 through 7 line cards (there are actually 8 line cards in the modeled device; however, if only a single line card is up, the unit is not capable of switching). Unlike previous models, the rewards in the WAN Interface model are represented *using an expression*. The general form of this expression is

$$R_n = (Nlc - n + m) / Nlc \quad \text{Equation 1}$$

Where

- m is the number of redundant line cards
- n is the number of simultaneously failed line cards
- R<sub>n</sub> is the reward of the n<sup>th</sup> failure state where n > m (otherwise, the reward is 1 because of the presence of sparing)
- Nlc is the number of line cards.

The effect of Equation 1 is to account for a capacity reduction through a gradual lowering of the reward as more linecards fail. Other representations are also possible. For example, if the device is defined as being available so long as it has a capacity above a certain threshold that might be met if no more than *k* line cards have failed, then the reward for the states in which 0...*k* line cards failed would be 1 whereas all other states would have a reward of 0.

Tables 5 and 6 list the variables and the transitions used in the model. As is shown in Table 5, there are only three variables that are used in a repeating pattern to develop the 8 state model.

**Table 5 . WAN Interface Variables**

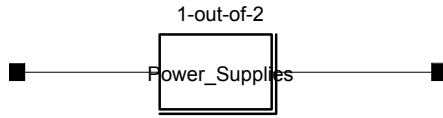
Variable	Baseline Value	Meaning
Nlc	8	Number of line cards in the device
MTTR_HW	4 hours	Hardware failure repair time (assumed to be the same for line cards as for CPUs, see earlier table)
MTTF_lc	5000 hours	Line card MTTF (assumed to be hardware only)

**Table 6. WAN Interface Model Transitions**

From	To	Transition Rate	Discussion
S0	S1	$Nlc/MTTF\_lc$	MTTF_lc is the time to failure of a single line card. The failure rate of the system in this state is Nlc representing the 8 initial line cards, any of which might fail
S1	S0	$1/MTTR\_HW$	Inverse of the hardware repair rate (to replace 1 card)
S1	S2	$(Nlc-1)/MTTF\_lc$	The failure rate of the system in this state is Nlc-1 representing the 7 remaining line cards, any of which might fail
S3	S2	$1/MTTR\_HW$	Inverse of the hardware repair rate (to replace 1 card)
S2	S1	$1/MTTR\_HW$	Inverse of the hardware repair rate (to replace 1 card)
S3	S4	$(Nlc-3)/MTTF\_lc$	The failure rate of the system in this state is Nlc-3 representing the 5 remaining line cards, any of which might fail
S2	S3	$(Nlc-2)/MTTF\_lc$	The failure rate of the system in this state is Nlc-2 representing the 6 remaining line cards, any of which might fail
S4	S3	$1/MTTR\_HW$	Inverse of the hardware repair rate (to replace 1 card)
S4	S5	$(Nlc-4)/MTTF\_lc$	The failure rate of the system in this state is Nlc-4 representing the 4 remaining line cards, any of which might fail
S5	S4	$1/MTTR\_HW$	Inverse of the hardware repair rate (to replace 1 card)
S5	S6	$(Nlc-5)/MTTF\_lc$	The failure rate of the system in this state is Nlc-5 representing the 3 remaining line cards, any of which might fail
S6	S5	$1/MTTR\_HW$	Inverse of the hardware repair rate (to replace 1 card)
S6	S7	$(Nlc-6)/MTTF\_lc$	The failure rate of the system in this state is Nlc-6 representing the 2 remaining line cards, any of which might fail
S7	S6	$1/MTTR\_HW$	Inverse of the hardware repair rate (to replace 1 card)

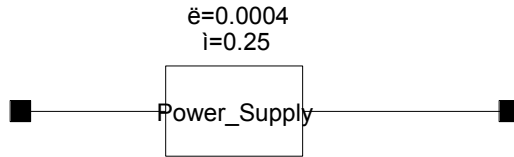
### **Power Subsystem**

The power subsystem consists of two power supplies running in an active-active configuration. The use of this redundant configuration allows us to model the subsystem using a block diagram. Figure 7 shows the model of the power supply with the k out of n construct.. In contrast to Markov models, block diagrams do not involve the concept of states. The only variables are the number of required (k) and the number available (n). In this case, there is 1 required and 2 available, hence, the 1 out of 2 legend at the top of the block.



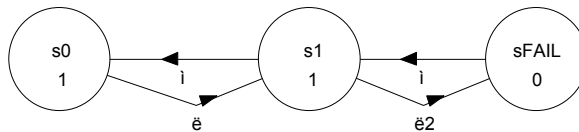
**Figure 7 . Dual Redundant Power Supply Subsystem**

The failure rate for each of the 2 power supplies is defined by a lower level reliability block diagram shown in Figure 8. MEADep displays the failure rate (inverse of MTTF) and recovery rate (inverse of MTTR) parameters corresponding to a 25,000 hour MTTF and a 4 hour MTTR. In this model, we have represented the power supply as a single block, but further detail could be included depending on the nature of the design and the needs of the model.



**Figure 8. Single Power Supply**

A model in which the failure rate of the power supplies varies by load could also be represented using Markov models as shown in Figure 9. In this case, we have three states, s0, both supplies working, s1, one supply failed, and sFail, both supplies failed. The transition between s0 and s1 is the failure rate  $\lambda$  whereas the transition between s1 and sFail is  $\lambda_2$ , the failure rate when only a single power supply is operational (and the load is higher with a resultant greater failure rate)



**Figure 9. Alternative representation of dual redundant power subsystem with different failure rates when two supplies are sharing the load and when a single power supply is supplying the entire load.**

## Results

Table 7, shows the reliability and availability results for the initial values for the overall system and each of the subsystems based on the baseline values shown in the tables for each of the subsystem variables..

**Table 7. Baseline Results**

Model-Name	Failure-Rate (per hour)	Recovery-Rate (per hour)	Availability	Unavailability
Main	0.000554	19.91663	0.999972	2.78E-05
CPU	0.000416	20	0.999979	2.08E-05
Power Supplies	6.38E-07	0.25	0.999997	2.55E-06
Linecards	1.12E-06	0.25	0.999996	4.49E-06
SBC	0.00414	20	0.999793	0.000207
PS	6.38E-07	0.25	0.999997	2.55E-06

The table shows that the largers contributors to unavailability are the power supplies, CPU, and line cards. The table also shows that the most frequently failing components are the single board computers (a reset occurring approximately once every 241 hours).

### ***Parametric Studies***

While the initial value results from the model are certainly of interest, the real power of MEADep becomes apparent when system tradeoffs and parametric modeling become important issues.

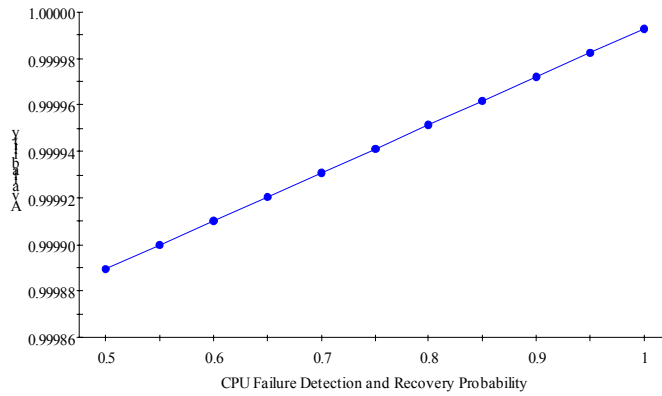
To demonstrate the power of such modeling, let us assume that it is desired to increase the current predicted availability from 0.99997 to 0.99999. After a system engineering study, it is determined that two options are available:

- a. Increase the coverage through additional failure/recovery testing
- b. Replace the operating system with an alternative with a higher demonstrated MTTF.

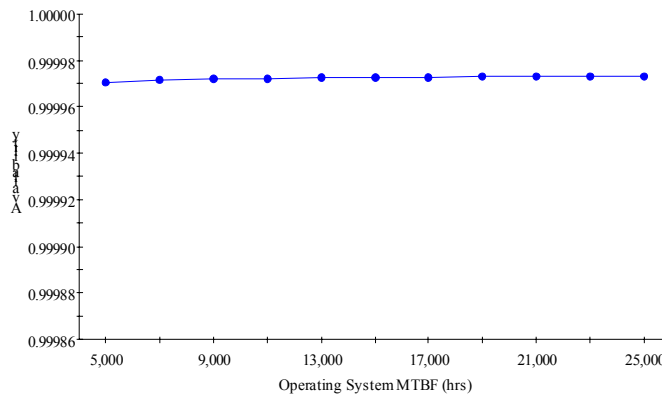
MEADep shows that with option a, it is feasible to reach the availability goal but that with option b, it is not. Figure 10 shows that if the coverage is increased from 0.90 to 0.96, then the availability will reach the goal of 0.99999. However, Figure 11, which shows impact of the operating system MTTF, shows that even if the MTTF is increased by a factor of 5 (from 5000 to 25000 hours). the availability increases from 0.999970 to 0.999974, which, on the same scale as which coverage is plotted, appears almost flat.

Logistics and support strategies can also impact availability and can be factored into the MEADep tradeoff analysis as well. For example, if the repair time is reduced from 4 hours to 30 minutes, the availability requirement can also be met. The impact of repair time on system availability is shown in Figure 12.

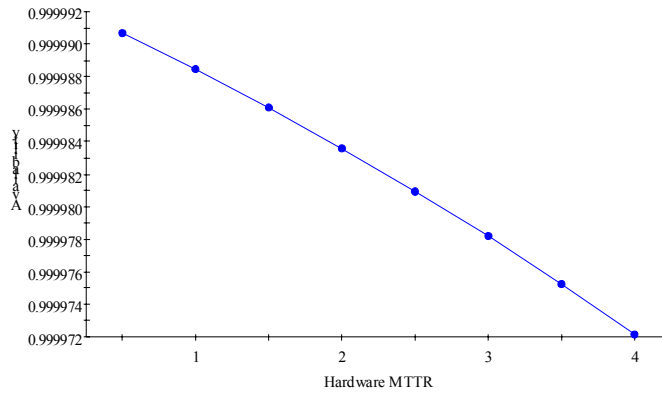
It is possible that the best option is a combination of several approaches. For example, lowering the maintenance time to 2 hours and increasing the coverage to 0.95 will achieve the availability objective as well.



**Figure 10. Impact of SBC Coverage on System Availability (Option a)**



**Figure 11. Impact of Operating System MTBF on System Availability (Option b)**



**Figure 12 . Impact of Hardware MTTR on System Availability**

There are many other such combinations that can be modeled and analyzed. The value of MEADEP is in the versatility it provides in enabling tradeoffs among a wide variety of parameters that affect reliability and availability.

## Conclusions

Modeling is an effective tool to analyzing on-line systems and developing the answers to strategic questions to achieve higher availability and performance. This white paper has demonstrated how MEADEP can be used for modeling a communications device. The paper has shown that

- hardware reliability and software reliability can be integrated into a single model,
- capacity can be incorporated into an availability model
- the importance of various contributors to system failures and unavailability can be easily identified,
- parametric analyses can be used to trade off a variety of parameters in hardware, software, and support.

The model presented here addresses some aspects of on-line system availability. However, the richness and completeness representation of the development environment and surrounding post-deployment support infrastructure is not limited to what is shown here. Other papers on the MEADEP web site ([www.sohar.com/meadep](http://www.sohar.com/meadep)) describe related analyses topics including:

- operational availability: modeling logistical constraints including
- trading off time to market against reliability goals: modeling software reliability growth

Finally, this paper has addressed the modeling capability of the MEADEP tool but not the data analysis. Papers addressing this topic are available from the publications subsystem on the SoHaR web site ([www.sohar.com](http://www.sohar.com)). Additional application notes are now in preparation.

## MEADep Text Modeling File

```
bind
Nlc 8
m 1
R6 0
Power_Supplies$k 1
Power_Supplies$n 2
μPower_Supply 0.25
λPower_Supply 0.0004
R1 1
MTTR_HW 4
MTTF_lc 5000
Nlc 8
c 0.9
T_restart 0.05
T_detect 0.1
MTTF_APP 5000
MTTF_OS 10000
MTTF_HW 25000
end

evaluate
R2 (Nlc-2+m)/Nlc
R3 (Nlc-3+m)/Nlc
R4 (Nlc-4+m)/Nlc
R5 (Nlc-5+m)/Nlc
R$Power_Supply Exponential(λPower_Supply)
λ$Power_Supply λPower_Supply
μ$Power_Supply μPower_Supply
A$Power_Supply μ$Power_Supply/(λ$Power_Supply+μ$Power_Supply)
R$Power_Supplies R$Power_Supply
A$Power_Supplies A$Power_Supply
μ$Power_Supplies μ$Power_Supply
λ$Power_Supplies μ$Power_Supplies*(1-A$Power_Supplies)/A$Power_Supplies
R$Temp R$Power_Supplies
A$Temp A$Power_Supplies
μ$Temp μ$Power_Supplies
R$Power_Supplies kofn(Power_Supplies$k,Power_Supplies$n,R$Temp)
A$Power_Supplies kofn(Power_Supplies$k,Power_Supplies$n,A$Temp)
μ$Power_Supplies μ$Temp
λ$Power_Supplies μ$Power_Supplies*(1-A$Power_Supplies)/A$Power_Supplies
R$PS R$Power_Supplies
A$PS A$Power_Supplies
μ$PS μ$Power_Supplies
λ$PS μ$PS*(1-A$PS)/A$PS
R$Linecards markovR(Linecards)
A$Linecards markovA(Linecards)
```

```

μ$Linecards    markovMu(Linecards)
λ$Linecards    μ$Linecards*(1-A$Linecards)/A$Linecards
R$CPU_Failures    markovR(CPU_Failures)
A$CPU_Failures    markovA(CPU_Failures)
μ$CPU_Failures    markovMu(CPU_Failures)
λ$CPU_Failures    μ$CPU_Failures*(1-A$CPU_Failures)/A$CPU_Failures
μCPU    μ$CPU_Failures
λCPU    λ$CPU_Failures
R$CPU    markovR(CPU)
A$CPU    markovA(CPU)
μ$CPU    markovMu(CPU)
λ$CPU    μ$CPU*(1-A$CPU)/A$CPU
R$Main    R$CPU*R$Linecards*R$PS
A$Main    A$CPU*A$Linecards*A$PS
μ$Main    (λ$CPU*μ$CPU+λ$Linecards*μ$Linecards+λ$PS*μ$PS)/(λ$CPU+λ$Linecards+λ$PS)
λ$Main    μ$Main*(1-A$Main)/A$Main
end

```

```

markov
Linecards 8 14
0 1 Nlc/MTTF_lc
1 0 1/MTTR_HW
1 2 (Nlc-1)/MTTF_lc
3 2 1/MTTR_HW
2 1 1/MTTR_HW
3 4 (Nlc-3)/MTTF_lc
2 3 (Nlc-2)/MTTF_lc
4 3 1/MTTR_HW
4 5 (Nlc-4)/MTTF_lc
5 4 1/MTTR_HW
5 6 (Nlc-5)/MTTF_lc
6 5 1/MTTR_HW
6 7 (Nlc-6)/MTTF_lc
7 6 1/MTTR_HW
reward
0 1
1 R1
2 R2
3 R3
4 R4
5 R5
6 R6
7 0
end

```

```

markov
CPU 3 5
0 1 λCPU+c*λCPU
0 2 (1-c)*λCPU
1 2 λCPU

```

```
2 1 μCPU
1 0 μCPU
reward
0 1
1 1
2 0
end
```

```
markov
CPU_Failures 4 5
0 1 1/MTTF_HW
0 2 1/MTTF_OS+1/MTTF_APP
2 3 1/T_detect
1 3 1/MTTR_HW
3 0 1/T_restart
reward
0 1
1 0
2 0
3 0
end
```

```
output
Main
CPU
Power_Supplies
Linecards
CPU_Failures
PS
end
```