On Operational Availability of a Large Software-Based Telecommunications System

Randy Cramp and Mladen A. Vouk
North Carolina State University
Department of Computer Science, Box 8206
Raleigh, NC 27695-8206

Abstract
Modern telecommunications systems are dependent on software for their successful operation. In many cases, the software operates on an established hardware platform and subsequent releases of the system differ primarily in the software component. Because of the increased dependence of the society on advanced telecommunications systems, the reliability and availability of network switching elements are very important. Empirical information on the operational unavailability due to total system outages of a large software-based telecommunications system is presented and discussed. About 50% of these outages are reported as being caused by software. The data is used to show that the unavailability of this system can be described well using a classical two-state availability model.

1. Introduction
The importance of reliability and availability in the context of modern telecommunications systems cannot be overemphasized. These systems, to a very large extent, depend on software for their successful operation. In many cases, the software operates on an established hardware platform and subsequent upgrades of the system differ primarily in the software component.

Because of the society's increased dependence on advanced telecommunications, Bellcore has established generic requirements for the performance evaluation of telecommunications systems [Bellcore, 1990]. In order to diagnose the root causes of a failure (or outage) in network switching elements "it is important to gather outage data for all service failures, regardless of cause, and to classify the data by cause of failure" [Bellcore, 1989]. This includes hardware causes, software causes, human error causes, and environmental variables.

In this paper, we discuss availability of a large telecommunications switching system. A typical distribution of the reported causes of outages for the system is shown in Figure 1.1. We see that although software causes appear to account for almost 50% of the outages, system availability is significantly affected by a variety of the other types of causes. This illustrates the importance of addressing all causes of failures when modeling the availability of a practical software-based system. Many models have been proposed for evaluating availability of software-based systems (e.g., [Trivedi, 1975], [Costes et al., 1978], [Shooman, 1983], [Laprie et al., 1990], [Laprie et al., 1991], and [Laprie and Kanoun, 1992]). Some incorporate interaction with hardware and attempt to account for different types of failure causes (e.g., [Laprie and Kanoun, 1992]).

In the remainder of this section, we provide some background information concerning the system we studied and the availability measures we use throughout this paper. In Section 2, we discuss empirical unavailability. In Section 3, we examine system failure and repair rates, and in Section 4, using an example, we show how a very simple availability model can be used to describe and predict availability of the system studied. In order to protect proprietary information, we have removed numerical information from the graph ordinates.
Similarly, the units have been removed from the abscissae.

1.1 System

The system discussed in this paper has two principal switching products running similar software, but distinguished by their hardware compositions. We refer to these separate systems (or products) as P1 and P2. Since the software running on these systems generally changes more frequently than the hardware, we let a new software release represent the upgrade to a new system. A software release is normally installed at hundreds of sites. We distinguish among the releases by their release numbers (e.g., R7, R8, etc.). Software libraries for this system exceed 10 million lines of high level code. A typical executable software load consists of approximately 7 million lines of high level code of which 10% is new or modified code.

As part of the software process and product quality assurance activities a large amount of data is collected regularly on the development and performance of each product release. The data includes failure and fault information resulting from the product testing and its operational, or field, phases. For example, in addition to the exact product version and installation information for each site, the database stores calendar dates of the outages, duration of the outages, and cause classifications made by the field office (e.g., hardware, software, procedural, etc.).

1.2 "Time"

System usage is central to the idea of reliability and availability. Therefore, it is very important to choose an appropriate measure of usage. For example, system usage can be measured in inservice time or calendar time. We define inservice time as the cumulative time a given software release has been executed over all sites using that release. In contrast, calendar time is the cumulative time (i.e. weeks, months, etc.) that has elapsed since the software was released. At different calendar times different software releases will have been installed at a different number of sites. This means that the usage intensity of the system varies over calendar time and changes according to the amount of time the sites using the release have been in service. Therefore, from both hardware and software viewpoints exposure to usage inservice time is a more representative measure of usage than calendar time. However, calendar time is a measure that in all likelihood better reflects the perception of users (i.e., the telecommunications companies) since calendar time availability and degradation of services are very important from the customers' point of view.

The number of sites that operate a particular product release at any time varies with product type and release version. Figure 1.2 illustrates the number of sites in service for the two different product types. Figure 1.3 compares the inservice time with calendar time for different releases of one particular product type. We see that the "load" on the product software varies considerably among product platforms and releases. For example, in Figure 1.3, we see that at 50 calendar time units release R10 has had less inservice time than release R9 and R11.

![Figure 1.2 Calendar time variation in the number of sites operating release R10.](image)

This needs to be taken into account when detailed performance models of the system are considered. For instance, the larger the number of sites using a product release within a given calendar time period the larger is the rate of accumulation of inservice time during that period. This means that releases which are installed on more sites provide a larger exposure sample within a shorter calendar time period and achieve larger inservice time earlier in the calendar life of a release. This in turn means that some of the characteristics of the availability functions, such as its transient period, may end much sooner in the widely deployed releases than in the case of releases that are installed at only a few sites.

![Figure 1.3 Inservice time versus calendar time for various releases of P2.](image)
1.3 Availability

We use the general term of "failure" to represent a system outage. An outage is defined as a loss or degradation of service to a customer for a period of time (outage duration). In general, outages can be caused by hardware or software failures, human errors, and environmental variables (e.g., lightning, power failures, fire, etc.). A failure resulting in the loss of functionality of the entire system is called a "total system outage" [Bellcore, 1989]. A total (telephone) system outage implies a "loss of origination or termination capability in all terminations for a period in excess of 30 seconds, or loss of all stable calls for a period in excess of 30 seconds." Successful recovery from a total system outage occurs "when both of the following apply: i) origination and termination capability in all terminations is restored, and ii) the system's engineered call handling capacity has been restored" [Bellcore, 1989]. In this paper, we are only concerned with total system outages.

1.3.1 Instantaneous Availability

Instantaneous availability is the probability that the system will be available at any random time t during its life [Sandler, 1963]. We estimate "instantaneous" availability in a period i as follows:

\[ \hat{A}(i) = \frac{\text{uptime in period } i}{\text{total inservice time for period } i} \]  

where the inservice time is the total time in the period i during which all switches of a particular type (i.e., P1 or P2) operated a particular software release (whether fully operational, partly degraded, or under repair), while uptime is the total time during period i at which the switches were not in the "100% down" state (or total system outage state). Correspondingly, the instantaneous unavailability estimate is (1 - \( \hat{A}(i) \)). Associated with this measure are "instantaneous" system failure, \( \hat{\lambda}(i) \), and recovery rates, \( \mu(i) \), which are estimated as follows:

\[ \hat{\lambda}(i) = \frac{\text{number of outages in period } i}{\text{total uptime for period } i} \]  

\[ \hat{\mu}(i) = \frac{\text{number of outages in period } i}{\text{total downtime for period } i} \]  

where "inservice time" for period i is the sum of the downtime and uptime in that period.

1.3.2 Uptime Availability

Since the raw data are often "noisy", the data are usually presented after some form of smoothing has been applied. This gives rise to a family of "smoothed" availability metrics. In this study, we frequently used both one-sided and symmetrical moving average smoothing (e.g., 11-point symmetrical moving average). An extreme form of smoothing is provided by the uptime, or average, availability. Uptime availability is the proportion of time in a specified interval [0, T] that the system is available for use [Sandler, 1963].

We estimate uptime availability up to and including period i as follows:

\[ \hat{\lambda}_c(i) = \frac{\sum_{x=1}^{i} \text{uptime in period } x}{\sum_{x=1}^{i} \text{total inservice time for period } x} \]  

Total uptime and total inservice time are cumulative sums starting with the first observation related to a particular release. Uptime includes degraded service. Associated with uptime availability are average system failure, \( \hat{\lambda}_c(i) \), and recovery rates, \( \hat{\mu}_c(i) \), which are estimated as follows:

\[ \hat{\lambda}_c(i) = \frac{\sum_{x=1}^{i} \text{number of outages in period } x}{\sum_{x=1}^{i} \text{uptime in period } x} \]  

\[ \hat{\mu}_c(i) = \frac{\sum_{x=1}^{i} \text{number of outages in period } x}{\sum_{x=1}^{i} \text{downtime in period } x} \]  

2. Empirical Unavailability

Figure 2.1 illustrates a typical unavailability observed for the system. In addition to the "instantaneous" data, we show the influence of different smoothing approaches. Each "raw" data point on the graph corresponds to the data collected during one calendar period. Abrupt changes in instantaneous unavailability are smoothed by uptime averaging. Under the assumption that product unavailability becomes smaller with time, we would expect uptime unavailability estimates to be generally conservative. We found the 11-point symmetrical moving average smoothing useful for examining trends in the instantaneous unavailability.

Note: In order to draw the "raw" data on the logarithmic scale, the data points (periods) where no failures were observed (i.e. zero failure rate) were censored and only the adjacent non-zero failure rate points are shown.
Figure 2.2 shows uptime unavailability of P2 for releases R8 through R11 versus inservice time. In all releases, we see that there is a period immediately after the product is made available to customers in which there is considerable oscillation in the observed unavailability. We refer to the time period from the point where the product is made available to the customers (i.e., time zero) to the point where the instabilities abate as the "transient region." It corresponds to the transient part of the availability function. The duration of this region of instability depends on the product type and release. Once the instability has decayed, all releases exhibit fairly smooth unavailability decay curves which in this case can be approximated by almost straight lines (Note: the ordinate is drawn on logarithmic scale).

Figure 2.2 Variability of uptime unavailability for different P2 releases.

Figure 2.3 compares smoothed instantaneous (11-point symmetric moving average) and uptime unavailability of P1 and P2 for release R10 using linear ordinate. Notice the maximum in the P2 unavailability functions characteristic of reliability growth [Laprie and Kanoun, 1992].

3. Failure and Recovery Rates

Two measures which directly influence the availability of a system are its failure rate and its field repair rate (or switch recovery rate). Figure 3.1 shows P2 failure and recovery rates for release R11. Apart from the censored "raw" data two other representations are shown. In one the data are smoothed using an 11-point symmetrical moving average. In the other, we show uptime data, or the cumulative average of the data. In a system which improves with field usage we would expect a decreasing function for failure rate with inservice time (implying fault or problem reduction and reliability growth). This was behavior is observed (see Figures 3.1 and 3.2). Immediately after the product release date, there is considerable variation in the failure rate. Later the failure rate reduces and stabilizes.

Failure rate is connected to both the operational usage profile and the process of problem resolution and correction. Recovery rate depends on the operational usage profile, the type of problem encountered, and the field response to that problem (i.e., the duration of outages in this case). If the failures encountered during the operational phase of the release do not exhibit durations which would be preferentially longer or shorter at a point (or period) in the life-cycle, then we would expect the "instantaneous" recovery rate to be a level function with inservice time (with, perhaps, some oscillations in the early stages). This behavior was generally observed.

In Figure 3.1, we see that the smoothed recovery rate is approximately constant with age, although there are considerable oscillations in the "raw" data. It is interesting to note that the recovery rate is about 3 to 4 orders of magnitude larger than the failure rate, and since the recovery rate is approximately constant, it would be expected that the availability would be governed primarily

2 Zero valued data points are not shown in order to allow the use of logarithmic scale on the ordinate.
by the stochastic changes in the failure rate (e.g., [Laprie et al., 1991]).

Figures 3.2 and 3.3 depict the uptime failure and recovery rate curves for releases R7 through R11. Failure rate curves have similar shapes and suggest an exponentially or logarithmically decaying function once past the transient region. There is more variety among the recovery rate shapes. The transient period in the recovery rate occurs on a longer interval than the failure rate and suggests considerable variance in the duration of failures in the transient region.

4. Models

The time varying nature of both the failure rate and repair rate indicates that a full availability model which would describe the system in this study should be nonhomogeneous. In addition, the distribution of outage causes as well as the possibility of operation in degraded up states suggest that a detailed model should be many-state. In this section, we show that nonetheless a very simple two-state model may provide a reasonable description of the system availability beyond the transient region.

Figure 3.1 Field recovery and failure rates for P2 release R11 computed using cumulative outages, downtime and uptime.

Figure 3.2 Uptime failure rates for P2 systems for releases R7 - R11.

Figure 3.3 Uptime field recovery rates for P2 systems under releases R7 - R11.
4.1 Homogeneous Two State Markov Models

Consider a discrete-time stochastic process \( (X_n, n = 1, 2, \ldots) \) where \( X_n \) is a random variable denoting the state of the process at time \( n \). Let the stochastic process be a Markov chain. Let there be two states which we call state 0 and state 1. In state 0, the system is operational, while in state 1, the system is under repair. Let the conditional probability of failure in the period \( t, t + dt \) be \( \lambda dt \) (i.e., probability of transition to states 1 given the system is in state 0), and the conditional probability of completing repair in \( t, t + dt \) be \( \mu dt \) (i.e., probability of transition to state 0 given the system is in state 1). Then \( \lambda \) is failure rate and \( \mu \) is repair and/or recovery rate, and the one-step state transition diagram is

\[
\text{Running} \rightarrow (1-\lambda) \Delta t \rightarrow \text{Under Repair} \rightarrow (1-\mu) \Delta t
\]

The one-step transition probabilities are \( P_{00} = 1 - \lambda dt, \quad P_{01} = \lambda dt, \quad P_{10} = \mu dt \) and \( P_{11} = 1 - \mu dt \), where \( P_{ij} \) denotes transition probability between state \( i \) and state \( j \). The one-step transition matrix is then:

\[
P = \begin{bmatrix}
P_{00} & P_{01} \\
P_{10} & P_{11}
\end{bmatrix} = \begin{bmatrix}
1 - \lambda dt & \lambda dt \\
\mu dt & 1 - \mu dt
\end{bmatrix}
\quad (4.1)
\]

In general, \( n^{th} \) step transition probability, \( P^n \), (i.e., probability that the process starting in state \( i \) will be in state \( j \) after \( n \) transitions), is expressed by Chapman-Kolmogorov equations (e.g., [Trivedi, 1982], [Musa et al., 1987]). Given that the initial system probability vector is \( v(t=0) = [P_0(0), P_1(0)] \), where \( P_i(t) \) denotes the probability that the system is in state \( i \) at time \( t \), then after \( n \) transitions (and at time \( t \))

\[
v(n,t=t) = v(0)p^n
\]

The probability that the system that starts in state 0 (i.e., \( v(0) = [1, 0] \)) ends in state 0 after \( n \) transitions and at time \( t \) is the system availability \( A(t) = 1 \times P_{00}^n \), and \( \bar{A}(t) = 1 - A(t) \) is system unavailability.

Furthermore, assuming a Poisson arrival process for failures, an exponential distribution for repair times, and the same initial probability vector, we can set and solve differential equations describing the homogeneous process (e.g., [Trivedi, 1982], [Shooman, 1983]):

\[
A(t) = P_0(t) = \frac{\mu}{\lambda + \mu} + \frac{\lambda}{\lambda + \mu}e^{-(\lambda + \mu)t}
\]

\[
1 - A(t) = P_1(t) = \frac{\lambda}{\lambda + \mu} - \frac{\lambda}{\lambda + \mu}e^{-(\lambda + \mu)t}
\quad (4.3)
\]

The measure of uptime availability can be formally defined as the expectation

\[
A_c(T) = \frac{1}{T} \int_0^T A(t) dt
\]

From (4.3) we get

\[
A_c(T) = \frac{\mu}{\lambda + \mu} + \frac{\lambda}{(\lambda + \mu)^2} - \frac{\lambda}{(\lambda + \mu)^2}e^{-(\lambda + \mu)T}
\]

The system becomes independent of its starting state after operating for enough time for the transient part of (4.3) and (4.5) to decay away. This steady-state availability of the system is \( A(\infty) = \lim_{T \to \infty} A(t) = \frac{\mu}{\lambda + \mu} \), i.e.,

\[
A(\infty) = \frac{\mu}{\lambda + \mu}
\quad (4.6)
\]

4.2 Nonhomogeneous Two-State Markov Models

The two-state model discussed above represents a system which can be either fully operational or completely offline and under repair. However, not all realistic systems follow this simple model. In fact, \( P_1 \) and \( P_2 \) not only have failure rates and repair rates which vary with time, and can have different down states (e.g., FCC reportable or not [FCC, 1992]), but they can also function in more than one up state (i.e., the system may remain operational but with less than 100% functionality for some failures). Thus, a many-state nonhomogeneous Markov model may be more appropriate for describing the details of these systems (e.g., [Trivedi, 1975], [Ibe and Wein, 1992], [Ibe and Wein, 1992]).

Nevertheless, a classical two-state model for availability of recoverable systems based on constant failure rate \( \lambda \) and constant recovery rate \( \mu \) can be used to approximate behavior of more complex nonhomogeneous systems such as \( P_1 \) and \( P_2 \). We illustrate this through two approaches: one we call "steady-state" approximation, and another we call \( n \)-step approximation.

The "steady-state" approximation is based on observations made by Trivedi [Trivedi, 1975] and Shooman [Shooman,
They noted that once the system has been operational for some time, the steady-state equation (4.6) may be used to approximate the instantaneous availability by assuming a piecewise-constant variation of \( \lambda \) and \( \mu \) in time. Letting \( \tilde{\lambda}(t) \) and \( \tilde{\mu}(t) \) be estimates at time \( t \) for \( \lambda \) and \( \mu \), respectively, we can estimate instantaneous availability as

\[
\tilde{\lambda}(t) = \frac{\tilde{\mu}(t)}{\tilde{\lambda}(t) + \tilde{\mu}(t)}
\]  (4.7)

The \( \tilde{\lambda}(t) \) and \( \tilde{\mu}(t) \) approximations can be obtained from the empirical data: the former through application of a reliability model, the latter is often assumed to be a constant (e.g., [Laprie et al., 1991]).

The n-step transition approximation is a numerical approach. Chapman-Kolmogorov equations are replaced by a series of matrix multiplications describing a series of one-step transitions with time-varying values for \( \lambda \) and \( \mu \). The \( \lambda \) and \( \mu \) functions are obtained from separate models and are recomputed and substituted with each increment, \( \Delta t \), in time. If the time increment is small enough and the arithmetic is accurate enough the approximation provides an acceptable description of the actual system behavior even in the transient region. That is

\[
P_{n}(t+\Delta t) = P_{n-1}(t) \left[ \begin{array}{cc} 1 - \tilde{\lambda}(t)\Delta t & \tilde{\lambda}(t)\Delta t \\ \tilde{\mu}(t)\Delta t & 1 - \tilde{\mu}(t)\Delta t \end{array} \right] (4.8)
\]

Availability follows from (4.2).

### 4.3 Example

This example provides a simple illustration of the application of the above approximations. Consider the release R11 for product P2. Figure 3.1 shows the P2 failure and repair rates for the same release. From Figure 3.1, we see that the uptime recovery rate is approximately constant once sufficient in-service time has passed. There is more variation in the instantaneous rate and the 11-point smoothed rate. We make the simplifying assumption that the recovery rate is constant and choose it to be the average of the period being considered (i.e., it is the uptime recovery rate of the sample point with the largest in-service time).

Figure 4.1 shows the unavailability profile of P2 for R11. Figure 4.2 shows the corresponding instantaneous failure rate and a model fit. From earlier work by Jones (e.g., [Jones, 1991], [Jones, 1992]) we know that the Logarithmic Poisson execution time (LPET) model ([Musa and Okumoto, 1984], [Musa et al., 1987]) provides a good descriptive, as well as predictive, model for the failure rate of the P1 and P2 systems. Therefore, we used the LPET model to model the instantaneous failure rate. The least squares fit is shown in Figure 4.2.

![Figure 4.1 Unavailability data for P2 Release R11.](image1)

![Figure 4.2 LPET least squares fit to failure rate.](image2)

We see that both approximations practically coincide once past the transient period. However, both approximations lie below uptime unavailability, shown in the figure as the thick solid line, because they model instantaneous unavailability which tends to be less conservative than the uptime unavailability.
It is interesting to note that the n-step approximation shows a maximum typical of systems that experience reliability growth. In contrast, the "steady-state" approximation does not exhibit this mode.

In practice, models would be used to predict future unavailability of a system. Of course, only the data up to the point from which the prediction is being made would be available. We refer to the time point at which the prediction is made as the "cut-off point". The prediction
would differ from the true value depending on how well
the model describes the system. In Figure 4.4, we show
both instantaneous and uptime unavailability
approximations for release R11 of product P2 based on the
average recovery rate at the "cut-off point" and the LPET failure fit to points from the beginning of the release's
operational phase up to the "cut-off point". The uptime
unavailability approximation was calculated using the
discrete form of equation (4.4). We see that the
approximation for uptime unavailability follows the
empirical uptime unavailability quite well. Similar
approximation for uptime unavailability follows the
results have been obtained for other releases using various
"cut-off points".

5. Summary

We have presented empirical data on unavailability of a
large software-based telecommunications system due to
total system outages. The total system outages were
found to occur for a number of reasons with software as
the major cause. Different releases of the system exhibit
similar failure and recovery rate profiles. The system
recovery rate appears to be approximately constant beyond
the transient region, while the failure rate is a decreasing
function of the inservice time. A complete unavailability
model for this system needs to incorporate time-dependent
parameters, as well as more than one operational state, and
more than one failure state to account for software and
other types of causes and different classes of failure
durations. However, using an example, we demonstrated
that it is possible to reasonably approximate the empirical
behavior of the system unavailability due to total system
outages using a simple two-state Markov model.

We are currently investigating unavailability of the system
due to individual causes, such as software, procedural,
hardware, etc. We will use the results to evaluate a
many-state model of the system that accounts for degraded
system operation, different failure causes, and different
classes of failure durations.

6. References

[Bellcore, 1989]. Network Switching Element Outage
Performance Monitoring Procedures - SR-TSY-
000962, Issue 1, April 1989.
[Bellcore, 1990]. Reliability and Quality Measurements for
Telecommunications Systems (RQMS) - TR-TSY-
000929, Issue 1, June 1990.
[Costes et al., 1978]. A. Costes, C. Landrault, and J.C.
Laprie, "Reliability and Availability Models for
Maintained Systems Featuring Hardware Failures and
[FCC, 1992]. Federal Communications Commission,
"Notification by Common Carriers of Service
57 (44), March 5, 1992, pp 7883-7885.
[Ibe and Wein, 1992]. O.C. Ibe and A.S. Wein,
"Availability of Systems with Partially Observable
Failures," IEEE Transactions on Reliability, IEEE,
vol. 41, no. 1, March 1992, pp. 92-96.
[Jones, 1991]. W.J. Jones, "Reliability Models for Large
Software Systems in Industry," Proceedings First
International Symposium on Software Reliability
Engineering, pp. 35-42.
[Jones, 1992]. W.J. Jones, "Reliability of
Telecommunications Software: Assessing Sensitivity
of Least Squares Reliability Estimates," To appear in
[Laprie et al., 1990]. J.C. Laprie, C. Beoumes, M. Kaaniche,
and K. Kanoun, "The Transformation Approach to the
Modeling and Evaluation of the Reliability and
Availability Growth," Proc. 20th IEEE Int. Symp. on
Fault Tolerant Computing: (FTCS 20), IEEE, pp. 364-
371.
[Laprie et al., 1991]. J.C. Laprie, K. Kanoun, C. Beoumes,
and M. Kaaniche, "The KAT (Knowledge-Action-
Transformation) Approach to the Modeling and
Evaluation of Reliability and Availability Growth,"
IEEE Transactions on Software Engineering, IEEE,
vol. 18, no. 4, April 1991, pp. 701-714.
[Laprie and Kanoun, 1992]. J.C. Laprie, and K. Kanoun, "X-
Ware Reliability and Availability Modeling," IEEE
Transactions on Software Engineering, IEEE, vol. 18,
[Musa and Okumoto, 1984]. John D. Musa and Kazuhira
Okumoto, "A Logarithmic Poisson Execution Time
Model for Software Reliability Measurement,"
Proceedings Seventh International Conference on
[Musa et al., 1987]. John D. Musa, Anthony Iannino, and
Kazuhira Okumoto, Software Reliability:
Measurement, Prediction, Application, McGraw-Hill,
[Sandler, 1963]. Gerald H. Sandler, Systems Reliability
Engineering, Prentice-Hall, Englewood Cliffs, N.J.,
1963.
[Shooman, 1983]. M.L. Shooman, Software Engineering,
[Trivedi, 1975]. A.K. Trivedi, "Computer Software
Reliability: Many-State Markov Modeling
Techniques," Ph.D. Dissertation, Polytechnic Institute
of Brooklyn, June, 1975.
[Trivedi, 1982]. K. S. Trivedi, Probability & Statistics with
Reliability, Queuing, and Computer Science
Applications, Prentice-Hall, Englewood Cliffs, N.J.,
1982.