Dave Penkler
Manfred Reitenspiess
Francis Tam (Eds.)

LNCS 4328

Service Availability

Third International Service Availability Symposium, ISAS 2006
Helsinki, Finland, May 2006
Revised Selected Papers

Springer

Dave Penkler Manfred Reitenspiess
Francis Tam (Eds.)

Service Availability

Third International Service Availability Symposium, ISAS 2006
Helsinki, Finland, May 15-16, 2006
Revised Selected Papers

Springer

Volume Editors

Dave Penkler
Hewlett-Packard
5, avenue Raymond Chanas - Eybens, 38053 Grenoble Cedex 9, France
E-mail: dave.penkler@hp.com

Manfred Reitenspiess
Fujitsu Siemens Computers GmbH
Domagkstr. 28, 80807 München, Germany
E-mail: Manfred.Reitenspiess@fujitsu-siemens.com

Francis Tam
Nokia Research Center
P.O. Box 407, 00045 Nokia Group, Finland
E-mail: francis.tam@nokia.com

Library of Congress Control Number: 2006938409

CR Subject Classification (1998): C.2, H.4, H.3, I.2.11, D.2, H.5, K.4.4, K.6

LNCS Sublibrary: SL 3 – Information Systems and Application, incl. Internet/Web and HCI

ISSN 0302-9743
ISBN-10 3-540-68724-6 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-68724-5 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11955498 06/3142 5 4 3 2 1 0

General and Program Chairs' Message

The 3rd International Service Availability Symposium (ISAS 2006) continued with the tradition of its predecessors by bringing together researchers and practitioners from both academia and industry to address the problems of service availability. The unique characteristic of a strong academic and industrial partnership was vividly reflected in this year's event, from the Organizing Committee to the contributions and the participants. Recognizing the value of broadening the scope of ISAS 2006, we included new topic areas that cover service-oriented architectures, dependability of information and communications technology services, and Java.

We received a total of 38 submissions, each of which was thoroughly reviewed by at least three members of the Program Committee. Due to the limited time allocated for the symposium, many worthwhile manuscripts unfortunately did not make it into the final program. Our sincere thanks go to the Program Committee for conducting a vigorous review process in a rather tight time schedule. The detailed review and their generous comments have shaped the contributions into an excellent program.

Under a Nokia Research Center scientific conference sponsorship, we introduced a one day pre-symposium tutorial. We are grateful to Kishor Trivedi, Veena Mendiratta, and Miroslaw Malek for stepping forward to deliver the presentations "Assurance for Continuous Availability" and "Predictive Algorithms and Technologies for Availability Enhancement". Another new feature in this year's program was a special session on European Union Sixth Framework Programme (FP6) projects and actions in the area of dependability and security, for which we thank Manfred Reitenspiess for its organization. The presentations, which were not formally reviewed, can still be accessed under http://www.saforum.org/events.

We are indebted to the University of Helsinki and Nokia Research Center for providing the support and resources needed for hosting ISAS 2006 in Finland. The local arrangements team, Minna Uimonen and Lauri Liuhto, did a tremendous job of assisting the planning, organizing, and coordinating all the local activities. Their dedication and precise execution deserve our special thanks. We would also like to acknowledge the involvement and support given by the Service Availability Forum and GI/ITG Technical Committee on "Dependability and Fault Tolerance".

We hope that you will find many contributions that are of interest to you in this volume. And of course, we look forward to seeing you at ISAS 2007, which will be held at the University of New Hampshire, Durham, NH in May 2007. The call for papers can be downloaded under http://www.saforum.org/events.

October 2006

Francis Tam
Kimmo Raatikainen
Dave Penkler
Kishor Trivedi

Organization

ISAS 2006 was organized by the Software and Application Technologies Laboratory, Nokia Research Center and the Department of Computer Science, University of Helsinki in cooperation with GI (German Computer Society) and Service Availability Forum (http://www.saforum.org).

ISAS 2006 Steering Committee

Miroslaw Malek (Humboldt Universität, Germany)
Dave Penkler (Hewlett-Packard, France)
Manfred Reitenspiess (Fujitsu Siemens Computers, Germany)
Francis Tam (Nokia Research Center, Finland)

Program Committee

General Co-chairs:	Francis Tam (Nokia Research Center, Finland)
	Kimmo Raatikainen (University of Helsinki, Finland)
Program Co-chairs:	Dave Penkler (Hewlett Packard, France)
	Kishor S. Trivedi (Duke University, USA)
Tutorials:	Francis Tam (Nokia Research Center, Finland)

Referees

A. Avritzer (Siemens, USA)
D. Bakken (University of Oslo/WSU)
S. Benlarbi (Alcatel)
A. Bobbio (University of Turin)
A. Bondavalli (University of Florence)
I. Chen (Virginia Tech)
T. Dohi (Hiroshima, Japan)
C. Fetzer (TU Dresden)
S. Garg (Avaya)
M. Garzia (Microsoft)
R. German (University of Erlangen)
H. Hermanns (University of Saarland)
S. Hunter (IBM)
M. Kaaniche (LAAS)
G. Le Lann (INRIA, France)
M. Lyu (CUHK, Hong Kong)
M. Malek (Humboldt University, Germany)
R. Mansharamani (TCS, India)
M. Marathe (Cisco)
V. Mendiratta (Lucent)
B. Murphy (Microsoft)
E. Nett (University of Magdeburg)
V. Nicola (University of Twente)
P. Portugal (University of Porto)
J. Posegga (TU Hamburg)
A. Rodriguez Vargas (Siemens)
H. Sun (Sun Microsystems)
N. Suri (Darmstadt, Germany)
H. Szczerbicka (University of Hannover)
A. Van Moorsel (University of Newcastle)

B. Vashaw (IBM, USA)
A. Wolski (Solid Tech.)
J. Xu (University of Leeds)
M. Yin (Cal State Poly University)

Sponsoring Institutions

Nokia Research Center, Helsinki, Finland

Table of Contents

International Service Availability Symposium 2006

Availability Modeling, Estimation and Analysis

Dependability Techniques and Their Applications

Performability: Measurements and Assessments

Service Availability Standards: Experience Reports and Futures

Model Based Approach for Autonomic Availability Management

Kesari Mishra and Kishor S. Trivedi

Dept. of Electrical and Computer Engineering, Duke University
Durham, NC 27708-0294, USA
{km, kst}@ee.duke.edu

Abstract. As increasingly complex computer systems have started playing a controlling role in all aspects of modern life, system availability and associated downtime of technical systems have acquired critical importance. Losses due to system downtime have risen manifold and become wide-ranging. Even though the component level availability of hardware and software has increased considerably, system wide availability still needs improvement as the heterogeneity of components and the complexity of interconnections has gone up considerably too. As systems become more interconnected and diverse, architects are less able to anticipate and design for every interaction among components, leaving such issues to be dealt with at runtime. Therefore, in this paper, we propose an approach for autonomic management of system availability, which provides real-time evaluation, monitoring and management of the availability of systems in critical applications. A hybrid approach is used where analytic models provide the behavioral abstraction of components/subsystems, their interconnections and dependencies and statistical inference is applied on the data from real time monitoring of those components and subsystems, to parameterize the system availability model. The model is solved online (that is, in real time) so that at any instant of time, both the point as well as the interval estimates of the overall system availability are obtained by propagating the point and the interval estimates of each of the input parameters, through the system model. The online monitoring and estimation of system availability can then lead to adaptive online control of system availability.

1 Introduction

Growing reliance upon computer systems in almost all the aspects of modern life has made things more manageable and controllable but at the same time has imposed stricter requirements on the dependability of these systems. System availability and associated downtime of technical systems have acquired critical importance as losses due to downtime have risen manifold. The nature of losses due to downtime vary, depending upon the field of deployment. In e-commerce applications like online brokerages, credit card authorizations, online sales etc., system downtime will directly translate into financial losses due to lost transactions in the short term to a loss of customer base in the long term.

D. Penkler, M. Reitenspiess, and F. Tam (Eds.): ISAS 2006, LNCS 4328, pp. 1–16, 2006.

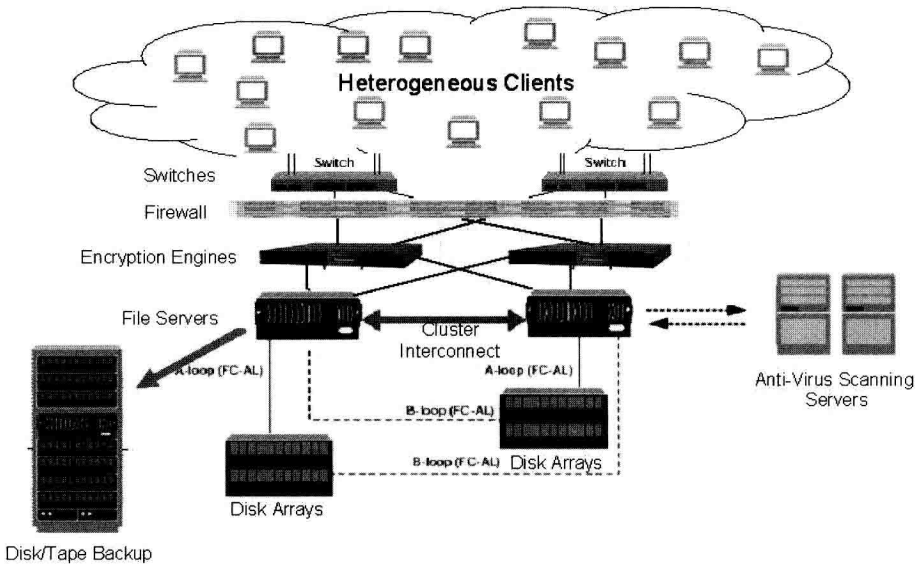

Fig. 1. A high availability data storage setup

In the infrastructure field, a downtime of the controlling systems may lead to disruption of essential services. In safety-critical and military applications, the dependability requirements are even higher as system unavailability would most often result in disastrous consequences. In such critical systems, there is a need to continuously monitor the availability of various components in the system and to take corrective/control actions in a timely manner, to maximize the system availability.

While the availability of hardware and software has significantly improved over time, the heterogeneity of systems, the complexity of interconnections and the dependencies between components has grown manifold. As a result, though the node level availability has improved, system level availability still needs improvement. Figure 1 shows a commonly deployed high availability data storage setup. In this common setup, the switches, the fire-wall (h/w or s/w), the encryption engines, file servers, disk arrays, backup system, anti-virus servers, fiber channel loops etc. are all interconnected with very complex dependencies. Most likely, all of these components will be from different manufacturers, following different best-practices guides and behaving differently, making it extremely difficult to plan for and anticipate every interaction between these components in the design phase of the setup.

The heterogeneity of system components(both hardware and software) and the complexity of their interconnections now makes the task of availability monitoring and management even more important as it will be more difficult to predict and design for all the interactions between these diverse components in the design phase and these interactions will need to be handled at runtime.

Therefore, in this paper we propose an approach for autonomic [11] management of system availability, which provides real-time evaluation, monitoring and management of the availability of systems in critical applications.

In this approach, an analytic model captures the interactions between system components and the input parameters of the model are inferred from data obtained by online (realtime) monitoring of the components of the system. The system availability model is solved online using these estimated values, thereby obtaining a realtime estimate of the system availability. The confidence interval for the overall system availability is obtained by propagating the confidence intervals of the individual parameter estimates through the system model, using a generic Monte-Carlo approach. Based on the parameter estimates and the overall system availability, control strategies such as alternate hardware and software configurations [17,10] or load balancing schemes can be decided. Optimal preventive maintenance schedules can also be computed [16,18]. This automates the whole process of availability management to ensure maximum system availability, making the system autonomic.

This paper is organized as follows: Section 2 discusses the background and related work in the field of availability evaluation, section 3 describes realtime availability estimation for a system viewed as a black box (when the whole system can be monitored from a single point). Section 4 discusses the steps in implementing autonomic availability management for a system (viewed as a grey box) with many hardware and software components, Section 5 illustrates this concept with the help of a simple prototype system and Section 6 finally concludes the paper.

2 Background and Related Work

Availability of a system can be defined as the fraction of time the system is providing service to its users. Limiting or steady state availability of a (non-redundant) system is computed as the ratio of mean time to failure (MTTF) of the system to the sum of mean time to failure and mean time to repair (MTTR). It is the steady state availability that gets translated into other metrics like uptime or downtime per year. In critical applications, there also needs to be a reasonable confidence in the estimated value of system availability. In other words, the quality or precision of the estimated values needs to be known too. Therefore, computing the interval estimates (confidence interval) of availability is also essential.

Availability evaluation has been widely researched and can be broadly classified into two basic approaches, measurement-based and model-based. In the model-based approach, availability is computed by constructing and solving an analytic or simulation model of the system. In the model-based approach, the system availability model is prepared (offline), based on the system architecture to capture the system behavior taking into account the interaction and dependencies between different components/subsystems, and their various modes of failures and repairs. Model-based approach is very convenient in the sense that it

can be used to evaluate several what-if scenarios (for example, alternate system configurations), without physically implementing those cases. However, for the results to be reasonably accurate, the model should capture the system behavior as closely as possible and the failure-repair data of the system components is needed as inputs. In measurement-based approach, availability can be estimated from measured failure-repair data (data comes from a real system or its prototype) using statistical inference techniques. Even though the results would be more accurate than using the availability modeling approach, elaborate measurements need to be made and the evaluations of what-if scenarios would be possible only after implementing each of them. Direct measurements may also provide lesser insight into the dependencies between various components in the system. Furthermore it is often more convenient to make measurements at the individual component/subsystem level rather than on the system as a whole [13]. The availability management approach advocated in this paper, combines both measurement based and model based availability evaluation approaches to complement each other's deficiencies and makes use of the advantages of both the approaches.

Traditionally, availability evaluation has been an offline task using either the model based or the measurement based approach. A few projects combined availability modeling with measurements, but they did not produce the results in an online manner [4,19,7]. In other words, these previous approaches were good only for a post-mortem type analysis. In today's complex and mission-critical applications, there is a need to continuously evaluate and monitor system availability in real-time, so that suitable control actions may be triggered. The growing heterogeneity of systems and complexity of their interconnections indicate the need for an automated way of managing system availability. In other words, the system should ultimately self-manage its availability to some degree (autonomic system). Kephart et. al. [11,31] suggest that an autonomic system needs to comprise of the following key components : a continuous monitoring system, automated statistical analysis of data gathered upon monitoring, a behavioral abstraction of the system (system model) and control policies to act upon the runtime conditions. This paper describes ways to implement each of these key components and hence outlines a way for autonomic availability management of technical systems.

3 Realtime Availability Estimation for a System Viewed as a Black Box

Availability evaluation addresses the failure and recovery aspects of the system. In cases where it is possible to ascertain the operational status of the whole system by monitoring at a single point (thus considering the system as a black-box), the availability of such systems can be computed by calculating the mean time to failure (MTTF) and mean time to repair (MTTR) from direct measurements of times to failure(TTF) and times to repair(TTR) of the system. To monitor the status of a system, two approaches can be followed. Either the

system under observation sends heartbeat messages to the monitoring station or the monitoring station polls the monitored system for its status. Each poll will either return the time of the last boot up or an indication of the system's failure. The heartbeat message will also contain similar information. In either case the information about the monitored system is stored as the tuple (*current time stamp, last boot time, current status*). The i^{th} time to failure is calculated as the difference between i^{th} failure time and the $(i-1)^{st}$ boot up time, (assuming the system came up for the first time at $i = 0$). The i^{th} repair time is calculated as the difference between i^{th} boot up time and the i^{th} failure time. At any observation i , the time to failure(ttf) and time to repair(ttr) are calculated as

$$ttf[i] = failure\ time[i] - \ bootup\ time[i-1]$$
$$ttr[i] = bootup\ time[i] - \ failure\ time[i]$$

The sample means of mean time to failure and mean time to repair are calculated as the averages of these measured times to failure and times to repair. If at the current time of observation, the system is in an up interval, the last time to failure will be an incomplete one and the point estimate of MTTF can be obtained as

$$\hat{MTTF} = \frac{\sum_{i=1}^{n} ttf[i] + x_{n+1}}{n}$$

where, x_{n+1} is the current(unfinished) time to failure. The point estimate of MTTR can be obtained as

$$\hat{MTTR} = \frac{\sum_{i=1}^{n} ttr[i]}{n}$$

Similarly, if the system is in a down state at the current time of observation, the last time to repair will be an incomplete one and in this case, the MTTR can be estimated as

$$\hat{MTTR} = \frac{\sum_{i=1}^{n-1} ttr[i] + y_n}{n-1}$$

where, y_n is the current(unfinished) time to repair. The point estimate of MTTF can be obtained as

$$\hat{MTTF} = \frac{\sum_{i=1}^{n} ttf[i]}{n}$$

The point estimate of the steady state availability of the system is then obtained as $\hat{A} = \frac{\hat{MTTF}}{\hat{MTTF}+\hat{MTTR}}$. It can be shown that the point estimate of steady state availability depends only on the mean time to failure and mean time to repair and not on the nature of distributions of the failure times and repair times [1]. However, the nature of distributions of the failure and repair times will govern the interval estimates of availability.

Assuming the failure and repair times to be exponentially distributed, both the two sided and the upper one-sided confidence interval of the system availability can be obtained with the help of *Fischer-Snedecor F distribution.* If at the current time of observation, the system is in an up state, the $100(1-\alpha)\%$ upper one-sided confidence interval for availability $(A_L, 1)$, can be obtained as

$$A_L = \frac{1}{1 + \frac{1/\hat{A}-1}{f_{2n,2n;1-\alpha}}}$$

where $f_{2n,2n;1-\alpha}$ is a critical value of the F distribution with $(2n, 2n)$ degrees of freedom. The $100(1-\alpha)\%$ two-sided confidence interval (A_L, A_U) for this case, can be computed as

$$A_L = \frac{1}{1 + \frac{1/\hat{A}-1}{f_{2n,2n;1-\alpha/2}}}$$
$$A_U = \frac{1}{1 + \frac{1/\hat{A}-1}{f_{2n,2n;\alpha/2}}}$$

where $f_{2n,2n;\alpha/2}$ and $f_{2n,2n;1-\alpha/2}$ are critical values of the F distribution with $(2n, 2n)$ degrees of freedom.

If the system is in the down state at the time of the current observation, the $100(1-\alpha)\%$ upper one-sided confidence for availability $(AL, 1)$ is computed as:

$$A_L = \frac{1}{1 + \frac{1/\hat{A}-1}{f_{2n,2n-2;1-\alpha}}}$$

The $100(1-\alpha)\%$ two-sided confidence interval (A_L, A_U) for this case, can be computed as

$$A_L = \frac{1}{1 + \frac{1/\hat{A}-1}{f_{2n,2n-2;1-\alpha/2}}}$$
$$A_U = \frac{1}{1 + \frac{1/\hat{A}-1}{f_{2n,2n-2;\alpha/2}}}$$

where, $f_{2n,2n-2;1-\alpha}$, $f_{2n,2n-2;\alpha/2}$ and $f_{2n,2n-2;1-\alpha/2}$ are critical values of the F distribution with $(2n, 2n-2)$ degrees of freedom [1,3].

Figure 2 shows a screenshot of the online point and interval estimates of availability of a system viewed as a black-box, with exponentially distributed time between failures and repairs. The monitored system was forced to crash and bootup at exponentially distributed times and the boot up and crash times were recorded at the monitoring station. As the number of samples increases, the confidence interval for the system availability, becomes tighter. In cases, where calculation of exact confidence intervals is time consuming (and hence, non-feasible for an online application), approximate confidence intervals are calculated using

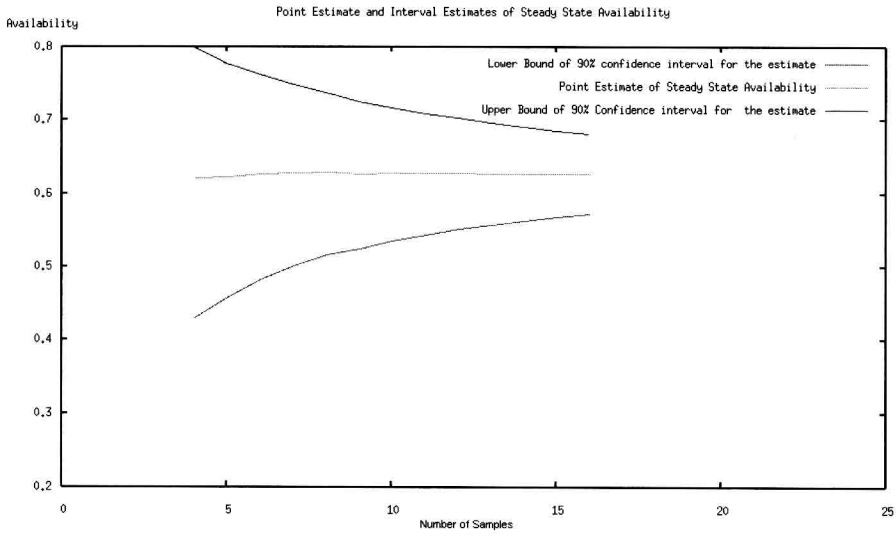

Fig. 2. Point and Interval Estimates of Steady State Availability of a Black box System

normal distribution assumptions [19]. Fricks and Ketcham [21] have proposed the method of *Cumulative Downtime Distribution*(CDD) that makes use of the sample means of the cumulative system outage time and provides both the point estimate as well as confidence intervals for the system availability without a need to make any assumption about the lifetime or time to repair distributions of the system being monitored.

4 Availability Management for a Complex System

High availability systems in critical applications have diverse and redundant hardware and software components configured together into a system with the help of complex interconnections. Availability/unavailability of these components affects the overall system availability. It might be difficult to ascertain the status of the whole system by monitoring a single point [13]. In this section, we outline the basic steps needed to implement autonomic availability management for such systems. The steps in its implementation can be summarized as:

1. Development of System Availability model (Behavioral Abstraction of the System)
   Based on the system architecture, the system availability model is constructed to capture the system behavior with respect to the interaction and dependencies between different components/subsystems, and their various modes of failures and repairs [22]. State-space, non-state space or hierarchical models are chosen based on the level of dependency in failure and

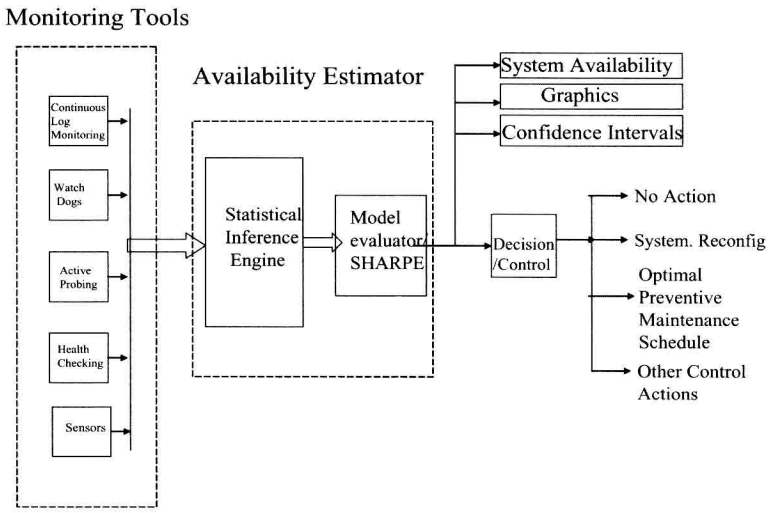

Fig. 3. The Availability Management System

repair between different components. In the case where all the components have statistically independent failures and repairs, non-state space models (e.g. Fault Trees, Reliability Block Diagrams) can be used. The non-state space models like fault trees and reliability block diagrams cannot easily handle dependencies in failure/repair of components or shared repair facilities. Therefore, in such complex cases, state space models are required. State space models like Markov chains [1,14], stochastic reward nets [5,8], semi-Markov process [18] and Markov regenerative process [20,23] have been widely used to model the availability of complex computer systems. Hierarchical availability models can be constructed for cases where failures and repairs across subsystems are independent of each other. They are used in cases when it is more intuitive to model the parts of the system(sub-systems) individually rather than the whole system at once and then have a model on top of these sub-system models to account for the interactions between these lower level models. Hierarchical models scale better with the number of subsystems and subsystem components than does a composite model and thus help avoid largeness of the state space of models [9].

2. Development of Monitoring Tools
For every component/subsystem in the system availability model, in order to gather its failure-repair data, there needs to be a mechanism to assess its operational status, at regular intervals. The monitoring mechanism needs to be designed to facilitate the monitoring of all the subsystems and components from a central location (the monitoring station). The system needs to be configured so that all the *relevant* (of the required severity) system log messages are directed to the monitoring station. Some of the steps needed in developing a monitoring mechanism have been summarized below.

- The system event logging mechanism in the monitored systems, should be configured so that the messages of required severity, from components being monitored are directed to the monitoring station. The monitoring station needs to be configured also as the log server for the log messages from all the monitored systems. Tools need to be developed to continuously inspect these log messages for error messages from I/O devices, hard disk, memory, CPU and related components like caches and buses, daemons and user processes, fans and power supplies. These tools can be either one of the available log monitoring tools like Epylog [25] or a light-weight custom developed one, running at the monitoring station. In Unix or Linux based systems, hardware failures like memory failure (uncorrectable ECC error), hard disk errors, I/O device errors, cache parity errors, bus errors etc. can be kept track of, by keeping a watch on emergency, alert, critical, error and warning level messages from the kernel [32].
- Polling agents need to be developed to poll the system regularly to determine the status of some components (e.g. sending periodic ICMP echo messages for network status, probing the sensor monitoring tools for status of fans and power supplies [28]).
- In cluster systems, watchdog processes need to be developed to listen for heartbeat messages from applications running on the cluster elements. Available redundancy (both process and hardware) in the cluster systems, provides additional mechanisms for monitoring by peer cluster elements [24,29,30]. Heart beat messages and Watch dog processes [29,30] and hardware [24] help detect processor and application failures.
- If SNMP (Simple Network Management Protocol) based monitoring is used, then periodic queries need to be sent from the monitoring station [26].

3. Development of the Statistical Inference Engine
   Once the monitoring tools have begun their task of data collection, parameters of the system availability model need to be estimated from the data by using methods of statistical inference. The tasks of the statistical inference engine are :
   - The statistical inference engine should first perform goodness of fit tests (Kolmogorov-Smirnov test and probability plot) upon the failure and repair data of each monitored component or subsystem. The parameters of the closely fitting distribution need to be calculated next.
   - The point estimate of limiting availability for any component or subsystem will be calculated as the ratio of mean time to failure and sum of mean time to failure and mean time to repair.
   - Depending upon the distribution of time to failure and time to repair, exact or approximate confidence intervals are calculated for the limiting availability of each component as discussed in the case of availability estimation for the non-redundant system.