-- --

IEEE P1003.0 Draft 13 – September 1991

5.4 Fault Management

Responsibility: Rich Bergman

The trade-offs in this subclause involve:

— Testability and Verifiability vs. simplicity and time

— Confidence and reliability vs. cost vs. risk

— Downtime vs. availability

— Maintainability vs. higher system cost

These services allow the system to react to the loss or incorrect operation of system components at various levels (hardware, logical, services, etc.). The classical model of fault tolerance has a three-step approach. The three steps are fault detection, fault isolation, and fault recovery. Typically implementations divide these steps into multiple steps or integrate them into one or two steps. Additionally, fault diagnosis services support the other steps in the treatment of a fault.

Various fault tolerance strategies, such as checkpointing and voting, are implemented as a collection of services comprising one or more of the steps in the fault tolerance classical model. For example, services involved in implementing a three-node voting scheme will include a vote comparator service (fault detection), vote analyzer service (fault isolation/fault diagnosis), a service to pass the majority ‘‘answer’’ through (fault recovery) as well as a service to disable the faulty resource and reconfigure the voters (fault recovery/reconfiguration).

Fault Detection

Fault detection services are concerned with determining when a fault has occurred in the system. Fault detection services are both passive and active. Active services are those that attempt to determine the status of various system components by testing those components. Passive services, on the other hand, try to ascertain system components by passively gathering information and watching the behavior of the system.

Fault Isolation;

Fault isolation services attempt to determine the component at fault and segregate the faulty component from the rest of the system. Services may be shared between the fault detection and isolation service library in that they perform both functions.

Fault Recovery

Fault recovery services attempt to bring the system into a consistent state. These services may be very interrelated to the scheduling services, network services, and data base services, depending on the recovery scheme used.

Redundancy of resources is many times needed to support fault recovery. Resources may include data, process, processor, disk drive, etc.

As parts of the system fail, it may no longer be possible to satisfy all the requirements of the application. Services to support graceful degradation may be used to ensure that critical activities do not fail.

Fault Diagnosis

These services deal with the system's ability to analyze the attributes of a system fault and determine its cause. These services tend to be very interrelated with fault detection and fault isolation services.

Fault Avoidance

These services involve the avoidance of faults before a failure in the system component occurs. If a system can detect that the operation of a component is approaching the edge of its operational range, a standby or backup component could be phased in to replace it. Another form of fault avoidance is logging of shocks, temperature extremes, etc., so that it can be predicted that a component will not meet its original expected service life.

Software Safety

These services involve the system's ability to keep application software from causing harm to the system's software, hardware, or user. For instance, a process may attempt to write into another process's memory space without permission.

A good example of a reliability method that may provide software safety is a bounds checker. The checker compares an answer supplied against the bounds. If it is not within the bounds, the bounds checker will not allow the answer to propagate, possibly causing damage to the system's integrity. Additionally, it may send a fault message (or security violation information, depending on the type of answers expected) to the proper service.

To enhance software safety, other services and processes should be only given the resources necessary to complete their job.

Status of System Components

These services involve the obtrusive and nonobtrusive diagnosis of the state of system components. For further explanation of these services, see Fault Detection and Fault Diagnosis services. These services may additionally need to record and/or display information concerning performance, configuration, and general system information.

Reconfiguration

These services allow the system to reconfigure its view of the world. This services allow the system to substitute different resources to perform system functions such as substituting a new physical I/O channel to support a logical channel. These services are part of the API but their use may be restricted to specially authorized programs such as those used by the target system operator.

Maintainability

Maintainability services provide support for the maintenance of a system. A major component of that support is the collection and logging of information about the operation of the system. Typical information to be logged is:

— Software and hardware errors during operation

— Processes that failed or almost failed to meet scheduled deadlines

— Performance metrics for system tuning

— Times when the system operated in extreme environmental conditions

— Errors reported during startup self-testing

— Attempts to violate rules of the system's security policy.

Section 6: Profiles

Responsibility: Fritz Schulz

This section targets those who want to know more about what profiles are and those who are in the process of developing their own profiles. The latter group consists of those developing formal ''Standardized Profiles'' and those developing less formal profiles for their industry group (e.g., a banking trade association) or their own company or enterprise for procurement or strategic planning purposes.

Those not involved in the development of profiles should read 6.2. Parts of 6.3 also may be useful, especially the earlier subclauses that give definitions of terms and explain concepts more precisely.

Developers of profiles that are not formal POSIX Standardized Profiles (POSIX SPs) should read all of Section 6.

Developers of profiles that are formal POSIX SPs should read all of Section 6 and Annex A.

6.1 Scope

The information presented here about profiles is limited in scope to assist those needing to understand profile concepts as they apply to the POSIX Open System Environment. Covered are profiles constructed from standards (and profiles) listed within this guide (that, by design, are consistent with POSIX.1).

The goal is to create a common approach and documentation scope and style for POSIX-oriented profiles. Annex A goes further by giving specific guidance to developers of formal POSIX SPs.

6.2 Profile Concepts

Responsibility: Bob Gambrel

Introduction

This guide is designed to assist in the selection of standards in the procurement process or as a target application environment. Profiles also assist in the selection of standards. A profile is a suite of base standards with specified options. Profiles can be created by software developers to describe the environment they