The Dependability Problem: Introduction and Verification of Fault Tolerance for a Very Complex System

Algirdas Avižienis

Dependable Computing & Fault-Tolerant Systems Laboratory
UCLA Computer Science Department
Los Angeles, California 90024

Abstract

The Advanced Automation System must meet availability requirements that pose new and severe challenges to designers and analysts of fault-tolerant systems. The problems of introducing sufficient fault tolerance and verifying its effectiveness are discussed in the context of AAS requirements.

1. Introduction

The Advanced Automation System (AAS) is probably the most complex real-time system being designed today that has to meet extreme requirements for near-perfect availability and safe operation [1]. Three modes of AAS operation have been defined, with availability requirements of $1 - 5 \times 10^{-6}$ for the Full Service mode, $1 - 10^{-6}$ for the Reduced Capability mode, and $1 - 10^{-7}$ for the Emergency mode. The availability requirement for one operational (controller's) position is also $1 - 5 \times 10^{-6}$, which corresponds to the average of 2.6 minutes down time per year. An extensive discussion of the requirements and other dependability aspects of the AAS has been presented in [2].

The complexity of AAS functional requirements and the severity of the availability requirements lead to the conclusion that the solution requires a systematic and hierarchically structured introduction of fault tolerance techniques throughout the entire AAS design process. The probability of success can be much enhanced by the use of rigorous design and evaluation guidelines, or *paradigms,* that are described in this paper.

2. The Design Paradigm For Fault-Tolerant Systems

It is interesting to note that the first effort to define design guidelines for fault-tolerant systems was presented at another FJCC exactly twenty years ago. They appeared in the 1967 Fall Joint Computer Conference paper "Design of Fault-Tolerant Computers" [3]. This paper introduced the concept of a "fault-tolerant system", presented a classification of faults, and outlined the alternate forms of masking, detection, diagnosis, and recovery techniques along with some criteria for choices between "massive" (i.e., masking) and "selective" application of redundancy. The design of the experimental fault-tolerant JPL-STAR computer was used to illustrate the application of these criteria in choosing the fault tolerance techniques for a spacecraft computer that had long life and autonomy requirements with strict weight and power constraints.

The 1967 FJCC paper was the start of a long-term effort to formulate an evolving view of dependable computing as the consequence of a judicious introduction of fault tolerance and fault avoidance during system design. Two different kinds of faults - those due to physical causes, and those due to human mistakes and oversights are considered. This evolving view has appeared in a series of papers on the techniques, scope, and aims of fault tolerance, the milestones of which are references [4], [5], [6]. The unifying theme over the past twenty years has been the evolution of a design paradigm for fault-tolerant systems that guides the designer to consider fault tolerance as a fundamental issue throughout the design process. The series of papers shows a progressive refinement of concepts and an expansion of the scope to include the tolerance of "human-made" design and interaction faults. Other recently introduced themes are: (1) the balancing of performance and fault tolerance objectives during system partitioning, and (2) the integration of subsystem recovery procedures into a multi-level recovery hierarchy. Strong emphasis is also placed on the application of design diversity in all parts of a multichannel system in order to attain tolerance of design faults [7].

Because of its great functional complexity, the AAS is the most severe challenge yet posed in the design of fault-tolerant systems. The introduction of fault tolerance into the AAS is most likely to succeed if a methodical approach is employed that begins with the initial design concepts and requires the collaboration of performance and fault tolerance architects during the critical tasks of system partitioning, function allocation, and definition of inter-subsystem communications and control. Such a design approach has been formulated as the *design paradigm* for fault-tolerant systems discussed above. The major steps of the paradigm are as follows:

1. Identify the classes of expected faults over lifetime of the system;

2. Specify goals for system performance and dependability (service modes, recovery time bounds, reliability, availability, and safety goals);

3. Partition the system into subsystems (hardware and software) for implementation of *both* performance and fault tolerance;

4. Select error detection and fault diagnosis algorithms for every subsystem;

5. Devise recovery algorithms for every subsystem;

6. Evaluate the effectiveness of fault tolerance and its impact on performance;

7. Integrate subsystem fault tolerance attributes on the systemwide (global) scale;

8. Refine the design by iteration of steps 3-7.

The above sequence is an abstraction of the actual specification and design process. In practice, the system level specification (defining steps 1 and 2) together with past experience of the designers leads to the derivation of initial choices for the steps 3 - 5 of the design paradigm. During design, iterations between any pair of steps 3 to 7 may take place as inadequacies are discovered and the design is refined. The objective of the design paradigm is to minimize the probability of oversights, mistakes, and inconsistencies in the process of meeting the specified dependability goals (step 2) with respect to defined classes of faults (step 1) by means of the given implementation of fault tolerance (steps 3 - 7).

The design paradigm presented above summarizes many insights gained during successful design efforts in the past. The AAS System Level Specification and the Statement of Work have not explicitly required that such a paradigm should be employed, but both have encouraged a systematic approach to the incorporation and demonstration of fault tolerance as the means of attaining the required availability. As it might be expected, both contractors of the current AAS design competition phase have carried out many of the analysis and design activities prescribed by the design paradigm. The major question that comes next is: "Will the proposed designs meet the exceptionally severe availability requirements when the AAS is built and deployed?" The remaining part of this paper addresses the question of evaluating the design of a very complex, highly fault-tolerant system.

3. Evaluation Of System Dependability

Successful completion of the design of a fault-tolerant AAS requires a convincing verification of two system properties: the *completeness* of the design and *its potential to meet the stated goals* of availability and safety. The verification therefore must consist of the sequential application of two distinct evaluations: first *qualitative*, then *quantitative*, or *numerical*.

The *qualitative evaluation* of fault tolerance attributes verifies that all the necessary defenses against the expected classes of faults have been properly incorporated into the design. A study of the design paradigm leads to the guidelines for a structured qualitative evaluation. It is a "pass/fail" type of examination that employs an "inverse" of the design paradigm, called the *qualitative evaluation paradigm,* to apply a series of searching questions about the completeness and appropriateness of fault tolerance techniques.

It is essential that the qualitative examination and evaluation should be satisfied *prior* to generating numerical predictions of system reliability and availability. Otherwise, unreasonably optimistic predictions can be obtained because of unjustified simplifications that remain unnoticed. Examples are: (1) Insufficient fault assumptions, such as omissions of design faults and man/machine interaction faults, disregard of nearly concurrent manifestations of two or more faults, disregard of latent errors and dormant faults, etc.; (2) Overestimation of the effectiveness and speed of error detection, fault diagnosis, system state recovery, and reconfiguration methods; (3) Implicit assumptions that all fault tolerance functions do not contain design faults themselves, that they are fully protected against physical faults, and that they always interact perfectly on a systemwide basis.

3.1 The Qualitative Evaluation Paradigm

The goal of the qualitative evaluation (QE) paradigm is to attain the discovery and elimination of design weaknesses and of the above described unjustified simplifications. Once that has been accomplished, the numerical evaluation can proceed with confidence. A typical sequence of steps of the *QE* paradigm are as follows:

1. *Validate fault assumptions:*

   1a. Are the assumptions about the classes of expected physical faults and about the frequency of their occurrence clearly stated and systematically justified?

   1b. Are design faults and man/machine interaction faults appropriately identified and treated?

   1c. Are faults in the implementation of fault tolerance functions (error detection, fault diagnosis, state recovery, reconfiguration) considered?

   1d. Are the effects of latent errors and dormant faults taken into account?

2. *Verify completeness of fault tolerance attributes:*

   2a. Are (i) system partitioning, (ii) subsystem interfacing, and (iii) fault isolation mechanisms at the boundaries, clearly described and adequate for the given fault assumptions?

   2b. Does every subsystem contain sufficient error detection, fault diagnosis, state recovery, and autonomous repair provisions?

2c. Are the fault tolerance provisions for the elements of (2b) complete and are those elements themselves adequately protected?

2d. Are the time requirements for the elements of (2b) completely stated and justified?

2e. Are the following design decisions documented and explained:

i. Elimination of alternatives in the choice of fault tolerance elements (i.e., detection, diagnosis, recovery);

ii. Reconciliation of fault tolerance costs and performance requirements under both normal and fault conditions?

3. *Check whether recovery procedures of subsystems are properly integrated and protected to provide complete and consistent defenses against the anticipated faults:*

3a. Are subsystem recovery and repair procedures consistent, synchronized and coordinated in a fault-tolerant manner?

3b. Can the system execute concurrent recoveries in two or more subsystems?

3c. Has the occurrence of unexpected faults (not included in (1) above) been assessed and treated as an unlikely, but possibly catastrophic event?

4. *Verify quality and applicability of design and numerical evaluation tools used:*

4a. Is the system reliability model clearly presented and fully justified with respect to the description of the design?

4b. Are the numerical methods adequate?

4c. Is the trustworthiness of the tools assessed?

4d. Are independent validations of the numerical results feasible?

The QE paradigm, as stated above, presents a generic list of questions that are relevant to every fault-tolerant system. For a given design, such as the AAS, these questions need to be tailored to the properties of every hardware and software subsystem and to their interconnections.

3.2 The Numerical Evaluation

The *numerical evaluation* verifies that the quantitative requirements of availability, reliability, and safety can be met by the design that has passed the qualitative evaluation. A design first must be described in terms of a system reliability model. This model is characterized by sets of physical, structural, repair, fault tolerance, and performance parameters for every subsystem, including subsystem communication links [8].

The most critical and difficult task is the determination of the most likely ranges of *coverage* and *execution time* parameters for all fault tolerance functions of every subsystem. These functions are: error detection, fault location and removal, state restoration, reconfiguration, recovery validation. Their execution must be coordinated on a systemwide basis. For this reason, the coverage and execution time parameters have to be predicted for subsystem interactions, assuming that one or more than one overlapping fault manifestation can occur, or that latent errors may be encountered in the course of an attempted recovery.

In order to be able to predict the impact of faults on system performance, the availability of the AAS must be estimated in terms of two distinct measures:

(1) The *Mean Time between Mode Reductions* that indicates the expected *frequency* of transitions into modes of operation below Full Service.

(2) The *Duration of Mode Reduction* that indicates the expected length of stay in the less desirable modes of operation, expressed in various measures (mean value, 99th percentile, worst case, etc.).

These measures are derived using the system reliability model that has been validated by means of the qualitative evaluation paradigm. They are very sensitive to the values of the *coverage* and *execution time* parameters of fault tolerance functions. These parameters need to be obtained by means of a sequence of progressively more specific and precise methods: estimates, analyses, simulations, and incremental experimentation with prototypes of critical elements under fault conditions.

4. The Challenges Ahead

The size and complexity of the AAS present a most severe challenge to designers, evaluators and modelers. Even when all state-of-the-art techniques are properly applied, there still remain some major concerns. The problem areas that demand ongoing attention in the AAS are:

1. The need to tolerate *new fault modes* unique to distributed systems that lead to errors in synchronization and in state consistency of concurrent processes and distributed data bases, making system state recovery very difficult.

2. The need to cope with unanticipated, failure-inducing *improper interactions* of otherwise well-designed fault-tolerant subsystems that have not been perfectly integrated into a distributed, fault-tolerant system. Especially difficult to integrate are the various hierarchical fault diagnosis and recovery sequences that

support the localized fault tolerance of subsystems and those that support the fault tolerance of functional services ("threads") that are provided by two or more subsystems.

3. The need to handle two or more *nearly concurrent fault manifestations*. The large size and distributed nature of new systems lead to the possibility of two or more independent fault manifestations occurring close in time, most likely because of the previously undetected existence of latent errors and dormant faults. This, in turn, will require two or more recovery algorithms to be concurrently active, with the resulting risks of mutual interference, deadlocks, and unpredictable behavior.

4. The need to tolerate during operation the *residual design faults* that have remained undiscovered by the testing and validation of software or VLSI logic, and later may cause concurrent, identical errors in multiple redundant computing channels.

5. The need to detect *potential operator and repairman errors* caused by interaction faults in their use of man/machine interfaces during system operation, maintenance, and modification.

6. The need to *extend and reverify* the protection of the AAS that is offered by fault tolerance techniques after every improvement, modification, and addition of new functions to AAS.

7. The need for error detection, containment, and system state recovery/reconfiguration features in the system and application software to provide fast-reacting defenses against the consequences of generally anticipated, but not precisely definable forms of abnormal behavior that may be caused by unrecognized fault modes or by any of the other problems discussed in (1)-(6) above.

The existence of the above listed problem areas was recognized by the FAA Advanced Automation Program Office at an early stage of the AAS Design Competition Phase (DCP). As a consequence, both DCP contractors have been encouraged to address the issues discussed above during the current term of DCP work.

Special emphasis has been placed on *incremental demonstrations* of the capabilities, protection, completeness, and consistency of all AAS fault tolerance functions and their specific implementation not only under single, but also under *multiple overlapping* fault conditions. The ultimate objective has been suggested as the evolution of a distributed, system-wide test of the fault-tolerant design under complex simultaneous fault conditions.

The second area of special emphasis is the carrying out of *supporting analyses* for the above discussed incremental demonstrations. Key issues in the analysis effort are: (1) verification of fault handling protocols by both formal and experimental techniques; (2) in-depth analysis of system-level mode change and recovery capabilities as well as of the fault tolerance attributes of top-level software, database, and interface designs; (3) consistency analysis and documentation of the layering of fault-handling in application software, databases and system architecture; and (4) identification and analysis of all error detection, fault handling, and recovery capabilities provided by the commercially available software that is being proposed for use in the AAS.

All the demonstration and analysis activities that have been discussed in this section represent a specific, in-depth application of the qualitative evaluation paradigm to the evolving AAS designs as they existed at a certain point of the DCP. These activities should significantly enhance the probability of a timely delivery of an AAS that can meet its extreme availability requirements.

5. Concluding Remarks

The AAS system is unique among current very large-scale systems because of its exceptional availability requirements during a long service life and because of the early and rigorous emphasis on fault tolerance as the main means to assure such availability.

It is evident that a successful delivery of the AAS as a replacement of the current ATC system will not only provide a new level of dependability and safety for air traffic control in the U.S., but will also serve as the major milestone in the evolution of a new generation of fault-tolerant systems. The timely transfer of the AAS experience to benefit other large-scale system projects remains an important challenge to the FAA, the industrial contractors, and the research community at large.

Acknowledgement

The author thanks Mr. Danforth E. Ball of the MITRE Corporation, Mr. Phil DeCara of the FAA Advanced Automation Program Office, and Mr. Gregory V. Kloster of Knowlex Technology Corporation for many valuable discussions. Suggestions received from Mr. Robert Wiseman and Dr. Philip Yoh of the DOT Transportation Systems Center are also sincerely appreciated.

References

[1] Hunt, V. R., Zellweger, A., "The FAA's Advanced Automation System: Strategies for Future Air Traffic Control Systems," *Computer*, Vol. 20, No. 2, February 1987, pp. 19-32.

[2] Avižienis, A., Ball, D. E., "On the Achievement of a Highly Dependable and Fault-Tolerant Air Traffic Control System," *Computer,* Vol. 20, No. 2, February 1987, pp. 84-90.

[3] Avižienis, A., "Design of Fault-Tolerant Computers," *AFIPS Conference Proceedings, 1967 Fall Joint Computer Conference,* Vol. 31, Washington, D. C.: Thompson, 1967, pp. 733-743.

[4] Avižienis, A., "The Methodology of Fault-Tolerant Computing," *Proceedings of the 1st USA-Japan Computer Conference,* Tokyo, October 1972, pp. 405-413.

[5] Avižienis, A., "Fault-Tolerance: The Survival Attribute of Digital Systems," *Proceedings of the IEEE,* Vol. 66, No. 10, October 1978, pp. 1109-1125.

[6] Avižienis, A., Laprie, J. C., "Dependable Computing: From Concepts to Design Diversity," *Proceedings of the IEEE,* Vol. 74, No. 5, May 1986, pp. 629-638.

[7] Avižienis, A., "The N-Version Approach to Fault-Tolerant Software," *IEEE Transactions on Software Engineering,* Vol. SE-11, No. 12, December 1985, pp. 1491-1501.

[8] Ng, Y. W., Avižienis, A., "A Unified Reliability Model for Fault-Tolerant Computers," *IEEE Transactions on Computers,* Vol. C-29, No. 11, November 1980, pp. 1002-1011.