

Novel Functional Availability Analysis Technique Using System Algebra For Safety Critical Systems

Submitted to International Institute of Information Technology, Bangalore
in Partial Fulfillment of
the Requirements for the Award of

Doctor of Philosophy

by

Manju Nanda
Roll No: PH2006503



International Institute of Information Technology, Bangalore
September, 2014

Certificate

This is to certify that the thesis titled **Novel Functional Availability Analysis Technique Using System Algebra For Safety Critical Systems** submitted to the International Institute of Information Technology, Bangalore, for the award of the degree of **Doctor of Philosophy** is a bona fide record of the research work done by **Manju Nanda, Roll No. PH2006503**, under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

SR

Professor Shrisha Rao
Thesis Supervisor

2014-09-29

Date



Novel Functional Availability Analysis Technique Using System Algebra For Safety Critical Systems

Abstract

Systems around us are often complex in design and functionality, and futuristic systems are going to be even more complex. There is a need to incorporate more formal method (FM)-based tools and techniques to fundamentally raise the bar with systems engineering processes, for lowering cost and time while increasing the safety assurance of complex systems. Such tools and techniques address problem spaces like generation of an integrated formal framework, a tool-based formal workbench, synthesis of critical functionalities, and real-life challenges. There already exist various FM-based tools and techniques being incorporated into various phases of the systems engineering process in order to address real-life system challenges. The research reported in this thesis adds a novel technique to the existing repertoire of systems engineering, to help increase the safety assurance level during the design of safety-critical systems.

Modern systems in critical applications, such as net-centric warfare and aerospace cyber-physical systems, are multi-functional. The availability requirements of these complex systems are very high. An ability to predict the future functional availability of a system given its design, and to achieve an assurance of a desired level of functional ability during design, are thus of much importance for safety-critical systems. The functional availability of a system, if known early on, can be used to better understand the pros and cons of the design and to detect design anomalies, which in turn leads to improvements in the design and life-cycle processes. There is a need for such a technique to detect design anomalies early on in the engineering process, to improve the design phase.

This thesis proposes a novel FM-based technique for analyzing functional availability of a system. The functional availability is determined using the

System Algebra (SA) FM. The SA technique uses the concept of a System State Machine (SSM) to compute system states and transitions between them over the life of a system. In the SA technique, a mathematical model is generated to represent the system under analysis. The relationships between different sub-systems of a system are modeled using this mathematical model by generating an SA expression. Applying operations on the SA expressions enables understanding of the system. As per SA, a system exists in one of the three states: the safe state, the hazardous state, and the unsafe state. The related SSM concept determines and analyzes the system state and its transition from one state to another over time, influenced by the failure of various sub-systems and components. The system state at any instant is determined by the health of its components and the states of its sub-systems. The health of a component is determined by its failure rate. Over time, as components start failing, the sub-systems degrade, thus degrading the overall system due to unavailable functionality. With SSM analysis, the design can be improved for better functional availability. This technique can help designers better understand system behavior and detect design anomalies. The system state and its transition determine the control flow between system states, and related changes in system functionality.

The proposed integrated design framework based on the SA technique has been automated to make the systems engineering process faster, more effective, and easier to implement. A design tool is being developed for easy plug-in into any engineering framework. The design framework has been validated by applying the technique for analyzing the design of existing safety-critical systems in different domains. In addition, the efficacy of the modified design framework has been validated by implementing it on completely new safety-critical aerospace systems. The SA technique introduced in the design phase of the integrated formal framework is seen to improve the performance of the process as compared to the conventional and model-based systems engineering framework.

Acknowledgements

First and foremost I would like to thank my advisor, Professor Shrisha Rao, without whose guidance, support, encouragement and timely advise, this thesis would not have been possible. His insightful discussions and constant motivation about the research encouraged me to work harder and start applying the work to my domain.

I take this opportunity to thank my Director, CSIR-NAL, Mr. Shyam Chetty, for giving me the opportunity to do my research.

I am grateful to Professor Sadagopan, Director, IIIT-B for giving me an opportunity to carry out research work at IIIT-B and for his support.

I am grateful to Professor Meenakshi D'Souza for her valuable guidance on the thesis chapters, scholarly inputs and consistent encouragement and suggestions. Without her my thesis would not have been in this shape. Her encouragement at a time when I was totally lost made it possible for me to submit my thesis.

I wish to thank Professor Dinesha for providing me scholarly inputs to be improvised in my research work. I also wish to thank faculty members of the institute who extended their help whenever I approached them. Special thanks to Mr. A.N. Ramachandra, Registrar IIIT-B, for his timely registration reminders and milestone reviews for smooth sailing of my research. I would like to thank all the supporting staff at IIIT-B for smooth processing of my research activities.

The thesis would not have come to a successful completion without the help I received from my colleagues at CSIR-NAL; their encouragement was vital in completing my research. I would like to thank Ms J Jayanthi, Ms Balamati and Mr. Shyamsunder Dhage for their scholarly suggestions and moral support. I would like to thank Mr. Madhan V, Mr. Nithin K. T, Mr. Arjun T.S and Ms. Chinamayi for helping me to apply my research work to the safety critical avionics domain. I would like to thank Head, ALD,

Mr.K.G.Venkatanarayana, for providing me resources, support and motivation to carry out my research work.

I would like to address special thanks to the unknown reviewers of my thesis, for accepting to read and review this thesis. I wish to thank the authors, developers and maintainers of the open source software used in this work. I would like to express my appreciation to all the researchers whose work I have referenced, to understand my field of research and keep it up to date.

Special thanks to Mr. Uday Nandiwad, who helped me in structuring and correcting my thesis. His experience helped in getting the thesis to this shape.

Special thanks to my lifeline, i.e my husband Alok and my son, Nikhil. Without their patience and understanding my work at IIIT-B and CSIR-NAL would not have been possible. Their unconditional support enabled me to get through the tough and rough at CSIR-NAL and IIIT-B. I am indebted to my family for their support, encouragement and love. Without my husband's support I would not have done well in my job and research. He is my motivation. My son who entered his teens during my research work also supported me. Without them I would not have achieved what I have achieved today.

Above all, I owe it all to Almighty God for granting me the wisdom, health and strength to undertake this research task and enabling me to its completion.

Thank you all.

Manju Nanda

Contents

List of Figures	x
List of Tables	xiii
1 Introduction	1
1.1 State of Art Research Work in Safety Critical Systems	2
1.2 An Overview of this Work	10
1.2.1 Formalization of System Analysis by Introducing System Functional Availability	11
1.2.2 Seamless Integration of SA with Reliability and the Safety Techniques	14
1.3 Thesis Layout	17
2 Existing FM Techniques, Methodologies And Systems Engineering	19
2.1 Systems Engineering Models	25
2.1.1 Waterfall Model	25
2.1.2 Prototype Model	26
2.1.3 V Model	26
2.1.4 W Model	27
2.1.5 Spiral Model	29
2.2 Evolution of the Systems Engineering Process, Integration of FMs with SE and Industry Standards	30
3 System Algebra	43
3.1 System Algebra Preliminaries	44
3.1.1 Mathematical Representation of a System	47
3.2 System Behavioral Analysis using the Functional Availability Property	49
3.2.1 Functional Availability using the System State and its Transition: Concept of System State Machine	49

CONTENTS

3.3	Verification and Validation of SA Approach for Analyzing Safety Critical Systems	58
3.4	Verifying and Validating the Critical System Parameters of Similar Systems using SA	62
3.4.1	FCS of Boeing B777	62
3.4.2	FCS of Airbus A380	63
3.4.3	Validating System Design based on System State Transition	65
3.4.3.1	Scenario A: <i>Failure at Input Stage</i> – 1	65
3.4.3.2	Scenario B: <i>Failure at Input Stage</i> – 2	67
3.4.3.3	Scenario C: <i>Failure at Computational Stage</i>	68
3.4.3.4	Scenario D: <i>Failure at Output Stage</i>	70
3.5	System Design Analysis using SA: Forward Engineering	70
3.5.1	Case Study 1: Functional Availability Analysis for SWS/AIC system design upgrade	71
3.5.2	Case Study 2: Functional Availability Analysis for eFM system design	75
3.6	Analyzing other existing Safety System Designs using SA: Reverse Engineering	80
3.6.1	Steer-by-wire Automobile: Automobile Domain	80
3.6.2	Railway Intelligent Transportation System Architecture (RITS): Transport Domain	83
3.6.3	A Distributed Fault-Tolerant Architecture for Nuclear Reactor Control and Safety Functions: Nuclear Domain	84
3.6.4	A Fault-Tolerant Architecture for Computer-based Railway Vehicle Brake Systems: Railway Domain	86
3.6.5	Distributed Architecture Block Diagram: Railway Domain	88
3.6.6	Typical Transport Power Source: Aerospace Domain	89
3.6.7	Integrated Elevator Electrical Power and Control using the Power-by-wire (PBW): Aerospace Domain	91
3.6.8	LCN – A Loosely Coupled Network System: Commercial Domain	92
3.6.9	System Complexity Analysis using SA: ATR72-600	93
3.7	Other System Analysis Methods and Performance Comparison of System Algebra with RBD and FTA	96
3.7.1	Reliability Block Diagram	99
3.7.2	Fault Tree Analysis	106
3.7.3	Comparison of System Algebra with other Popular System Analysis Methods	112

4 Proposed Modified Systems Engineering Life Cycle Process in Formal Design, Verification and Validation	117
4.1 Evolution of Systems Engineering Process to Integrated Formal Engineering Process	118
4.2 Proposed Novel Modified Systems Engineering	122
4.2.1 SA Analysis in the Design Phase of the Systems Engineering Lifecycle: Forward Engineering	127
4.2.1.1 Selection of Appropriate eFM Design Based on Functional Availability	127
4.2.1.2 Design of Crack Detection and Warning System (CDWS) Based on Functional Availability	133
4.2.2 SA Analysis in the Design Phase of Systems Engineering Lifecycle: Reverse Engineering	138
4.2.3 Measure of Performance and Effectiveness for the Modified Engineering Process	145
4.3 Tool Development	147
5 Conclusion And Future Work	159
5.1 Results & Outcome	161
5.2 Future Work	165
References	166
Index	176

List of Figures

1.1	Approach to Compute System Operational Availability	6
1.2	Addition of Functional Availability Property in System Design Analysis .	7
1.3	Paradigm Shift of Availability from Performance to Behavioral System Analysis	12
1.4	SA Approach	13
1.5	Relationship between Functional Availability and System States	14
1.6	Modified Systems Engineering Process	16
1.7	SA Process Automation using Tool	17
2.1	Engineering Process Flowchart	22
2.2	NASA Engineering Process Flowchart - 1 [1]	23
2.3	Systems Engineering Process Flowchart - 2 [1]	24
2.4	Systems Engineering Process Flowchart - 3 [2]	24
2.5	Waterfall Model	25
2.6	Prototype Model	26
2.7	V Model	28
2.8	W Model	28
2.9	Spiral Model	29
2.10	Relationship between the Engineering phases, Time-to-market, Cost and Errors	31
2.11	Graph Depicting the Cost and Error Relationship	32
2.12	Evolution of Engineering Process [3]	33
2.13	Modified Engineering Process for Certification [4]	38
2.14	V Model Adopted in Aerospace Process [5]	39
2.15	SCADE Engineering Process [5]	40
2.16	Formal Techniques and Tools to Realize an Effective Engineering Process by Miller <i>et al.</i> [6]	41
3.1	Safe State to Hazardous State to Unsafe State	52

LIST OF FIGURES

3.2	Safe State to Unsafe State	53
3.3	Hazardous State to Safe State to Unsafe State	54
3.4	Unsafe State	55
3.5	System State Transition	56
3.6	Jaguar FCS Block Diagram	59
3.7	B777 FCS Block Diagram	63
3.8	A380 FCS Block Diagram	64
3.9	Graphical Representation of Scenario A	66
3.10	Graphical Representation of Scenario B	68
3.11	Graphical Representation of Scenario C	69
3.12	Graphical Representation of Scenario D	70
3.13	Block Diagram of SWS/AIC	72
3.14	Block Diagram of Enhanced Fatigue Meter	76
3.15	Tree Diagram of eFM	79
3.16	Block Diagram of Steer-by-wire Automobile	81
3.17	Block Diagram of RITS	83
3.18	Block Diagram of Distributed Fault-tolerant Architecture for Nuclear Re- actor Control and Safety Functions	85
3.19	Block Diagram of Fault-tolerant Architecture for Computer-based Rail- way Vehicle Brake Systems	87
3.20	Block Diagram of Distributed Architecture	88
3.21	Block Diagram of Typical Power Source	90
3.22	Block Diagram of Integrated Elevator Electrical Power and Control	91
3.23	Block Diagram of LCN	93
3.24	Block Diagram of ATR72-600	95
3.25	Reliability Engineering	100
3.26	RBD Model [7]	101
3.27	Reliability Model of the Series System [7]	102
3.28	Reliability Model of Standby Parallel Redundancy [7]	102
3.29	Reliability Model of m-out-of-n Active Parallel Redundancy [7]	103
3.30	Reliability Model of m-out-of-n Systems [7]	103
3.31	Reliability Model of Simple Series and Parallel System [7]	104
3.32	Reliability Model of Non-series Parallel System [7]	104
3.33	Reliability Model of Complex Systems [7]	105
3.34	Reliability Model of Complex Systems [7]	105
3.35	RBD of B777 FCS	107
3.36	RBD of A380 FCS	108

LIST OF FIGURES

3.37	FTA Basic Gates [7]	109
3.38	FTA Symbol for Top or Intermediate Event [7]	110
3.39	FTA for AND Gate [7]	110
3.40	FTA for OR Gate [7]	111
3.41	FTA for Initiating Failure [7]	111
3.42	RBD, FTA Symbol for AND Gate [7]	112
3.43	RBD, FTA Symbol for AND Gate [7]	113
3.44	RBD, FTA Symbol for Inhibit Condition and XOR Gate [7]	114
3.45	RBD Representation for FTA System [7]	114
3.46	FTA of FCS System	115
4.1	Major Elements of Model-based Design	119
4.2	FM-based Availability Property Introduced in the System Design Analysis Attributes	121
4.3	Paradigm Shift of Availability Analysis Based on Data to Based on Simulation	122
4.4	Systems Engineering Lifecycle Process	123
4.5	Modified Systems Engineering Life Cycle Process	126
4.6	Block Diagram of Enhanced Fatigue Meter - Design I	128
4.7	Block Diagram of Enhanced Fatigue Meter - Design II	132
4.8	Block Diagram of Crack Detection and Warning System	134
4.9	Tree diagram of CDWS	137
4.10	Block Diagram of Conventional Clinical Laboratory System	141
4.11	Block Diagram of Autonomous De-centralized Clinical Laboratory System	144
4.12	Flowchart of the Tool	150
4.13	Tool : Opening Window	151
4.14	Tool : Parameter Window for System	152
4.15	Tool : Parameter Window for Sub-system	153
4.16	Tool : Feature Execution	154
4.17	Tool : Generation of System Block Diagram	155
4.18	Tool : Generation of SA Expression	156
4.19	Tool : System State Simulation	157
5.1	Top-Down Approach	163
5.2	Bottom-Up Approach	164

List of Tables

1.1	Relationship between Reliability, Maintainability and Availability	5
2.1	Systems Engineering Industry Standards	32
2.2	Scope of FM Use	36
3.1	Redundancy of the FCS Sub-systems and Components	60
3.2	Failure Rates of FCS Sub-systems and Components	61
3.3	Scenario A	66
3.4	Scenario B	67
3.5	Scenario C	69
3.6	Scenario D	70
3.7	Failure Rates of SWS/AIC Sub-systems and Components	73
3.8	Functional Availability Analysis of SWS/AIC	74
3.9	System State Transitions for eFM	78
3.10	Reliability Standards	106
4.1	Model-based Development Methodologies and Toolsets used in the Industry	120
4.2	Metrics for Model-based Development and Conventional Engineering Pro- cess	120
4.3	System State Analysis using the SA Expression	130
4.4	System State Transitions for CDWS	136
4.5	System State Analysis using the SA Expression	140
4.6	Attributes to Compute in MOP in Design Phase	146
4.7	System State Analysis using the SA Expression	147
5.1	Analysis of Systems Using SA	162

1

Introduction

Technology is user-driven. The never ending expectations of users make technology complex. The industry, which provides the new technology, needs a better, safer, faster, and cheaper system design methodology in order to be profitable. Some of the futuristic technologies being researched across the world are software-defined aircraft, situational awareness in the aircraft, intelligent transport systems, net-centric warfare, UAV flying on laser guidance, battery-powered human exoskeletons, synthetic blood, flying submarines and flying armored cars. The system software in these technologies is critical to provide the flexibility, configurability, and adaptability of these systems, e.g., the future electronic warfare system will have the flexibility of rapid online reprogramming to extend the basic capabilities of the system. The hardware and software of such systems are complex to achieve such functionality. In automotive applications, dozens of processors run 100 million lines of code to get a premium car out of the driveway. These complex systems are realized by incorporating an effective and well-defined process to validate system properties using formal techniques in the early phases of the engineering process. System properties aid in analyzing design during the initial phase of systems engineering, thus improving the final product. Effective analysis techniques incorporated in the engineering process reduce the time-to-market for the system from concept to implementation. System properties, such as reliability, safety, security, reach and maintainability, are analyzed against the project requirements to make the system design effective; this is discussed by Milutinovic and Lucanin [8], Wang *et al.* [9], and Reza *et al.* [10] in their work. There is always a need to improve the engineering process by integrating new techniques to design complex systems within resource constraints while ensuring the safety, security, and functionality of the system.

Systems engineering is a state-of-the-art methodology for managing high-integrity and complex system designs as well as production and maintenance activities of these

1. INTRODUCTION

systems. In response to increased demand for systems-of-systems, systems engineering practices have steadily become more formalized and specialized to reduce technical costs, improve schedules, and better products as discussed by Fieler [11], Honour [12], and Rushby [13]. This is achieved by modifying the systems engineering process to be more effective as discussed by Estefan [3]. The modification of the engineering process from document-centric to model-centric, is discussed by Erkinen [14]. This is an example of the process evolution for increased effectiveness. A typical systems engineering process can be either document-centric or model-centric consisting of various phases, such as concept, requirements, design, implementation, and maintenance as discussed in literature by Teresa [15], Ramos *et al.* [16], Koning *et al.* [17], and Estefan [3]. The systems engineering SysEngg [18] also discusses these phases in detail. Model-based safety analysis is a modification of the engineering process and is a key component in safety critical applications, especially avionics, where failure of the system will be catastrophic. Model based safety analysis integrates the advances in requirements engineering, test techniques, architecture, and engineering to deliver a safer and more reliable system. Formal methods and model-based approach are being used to improve these engineering processes with support from industry standards.

There is always a need to improve the engineering process by integrating new techniques to design complex systems within resource constraints while ensuring the safety, security, and functionality of the system. Improvements in the engineering process will raise the bar of system quality and lower project costs for complex systems.

1.1 State of Art Research Work in Safety Critical Systems

The complexity of the safety critical systems is also growing and there is always a need for effective tools and techniques to assure safety. Further, current and future safety critical systems consist of heterogeneous systems-of-systems and heterogeneous software to cater to these systems. The importance of software in driving these complex technologies is such that its failure may be catastrophic-loss of functionality, damage to environment and, in many cases, loss of life. Dulac and Leveson [19] discusses the need for software safety as the software controls the critical functionality of a system. The control of software over the system is so high that safety critical software, software system safety, and software fault tolerance are widely researched topics. Complex systems with functionality affecting safety are called safety critical systems. Safety critical systems are designed either for fail-safe, fail-off or fail-operational functionality. In a fail-safe system, failure

1.1 State of Art Research Work in Safety Critical Systems

makes the system non-operational without affecting the safety of the other interfacing systems. In the case of the fail-off system, failure switches- off the system, while a fail-operational system, design can tolerate multiple failures and continue to operate without affecting its functionality and that of the interfacing systems. These systems provide degraded functions till they stop functioning. Sababha *et al.* [20] discuss the concept of graceful degradation of safety critical embedded systems to make the systems highly dependable. The other advantage for designing systems with graceful degradation also helps in reducing the cost, weight and power requirements. Sababha *et al.* demonstrates this concept with the example of avionics system of a quadrotor unmanned aerial vehicle.

There exist various verification and validation techniques to ensure the safety of these critical systems. Formal Methods, FM, is considered to be an effective approach in ensuring the safety of a system design for its intended function. The analysis of a system by FM provides critical design feedback on performance and behavior. Rushby [13], and Woodcock *et al.* [21] discuss the effectiveness of FM in analyzing safety critical systems. FM's have the capability of describing a system either at the requirements, specification, architectural, design, implementation, testing, or maintenance phase of the process. FM's can be adapted in any or all of the engineering phases to provide higher quality, safety, and functionality assurance. The advantage of the FM in real-time applications is its capability of describing the system mathematically, proving its properties and performing system simulation to detect design faults early on, well before deploying the system.

The complexity of a system under development has its side effects, and so it is necessary to monitor its safety. Engineers were forced to develop effective techniques in the fields of system architecture, security, safety, model-based engineering, model-based testing, requirements engineering, integrated quality framework development, and formal methods for systems engineering in order to develop systems meeting all requirements.

An effective engineering process ensures that the system is built to meet project requirements within a stipulated time and budget. System requirements provide the functional and performance expectations from it in the field. Functional requirements describe the system behavior, which helps in designing the system for a particular application. Non-functional requirements describe the system's overall characteristics such as reliability, safety, availability, reachability, and maintainability. The non-functional requirements help in improving performance of the system. The functional and non-functional requirements in the requirement phase describe the complete system to be designed. The system design is validated for its functional and non-functional requirements. Safety, security, reachability, availability, and maintainability properties validate the system performance. The functionality analysis examines the system design for the intended function. Safety analysis examines the intended safety features in the system. Security analysis

1. INTRODUCTION

validates the security features in the system. Similarly reach, availability, and maintainability analyses look into the intended reach, availability, and maintainability features in the system-; there exist techniques to validate these properties. Of the system properties being analyzed during the functional and performance analysis, system availability analysis can be explored to demonstrate and validate the system functional requirements. This will integrate a new technique to analyze the system functionality.

At present, the availability property is used only to analyze the system to validate the non-functional requirements. In our work, we propose to use the availability property to validate the functional requirements of a system. This is required in today's high assurance critical systems where the system is expected to be available for longer periods of time and be fully functional. Introduction of this technique in the design phase of the engineering process not only modifies and enhances the process, but also makes it effective as designers can detect design faults early, thus improving system design and increasing system safety assurance.

The availability property has so far been used to determine the time taken by the system to transit from the operational phase to the maintenance phase. In other words, system availability property refers to the operational availability, the ability of the system to deliver the functionality when desired. In other words, operational availability is defined as the probability of the system operating properly in field or that a system has not failed is not undergoing a repair action when it is expected to be fully functional.

Research is underway to analyze and maximize this operational availability of distributed systems, operating systems, and other complex systems using availability algorithms or Markov models that describe the system's state switching processes to compute. Myrefelt [22] defines operational availability as a technical capability of the system to keep operating while under maintenance and defines functional availability as a quantification of the system's capability to be operational while maintaining intended functionality. Richards *et al.* [23] determine the probability of mean time between failure, MTBF, and mean time to repair, MTTR. These figures provide the probability that the system will be operational. Myrefelt [22] determines the availability based on MTTR and mean time to fail, MTTF. Trivedi *et al.* [24] discuss operational availability based on models instead of steady state calculation. The availability models analyze a system's availability based on its state change from good to repair state which is the operational availability. The Markov model, composite Markov model and semi-Markov models are used to determine integrity, performance, and confidentiality availability of the system used for dependable computing. These models analyze a system based on its failure and repair times. Work on high availability is undertaken to make systems available for a longer period of time in a net-centric application. System state analysis is performed based on the functionality,

1.1 State of Art Research Work in Safety Critical Systems

but does not provide the information about input scenario that cause changes in system state. The availability of the system is analyzed using system state chart, a part of the system model approach, e.g., SysML (System modeling from Rational). The system state chart provides qualitative visual information of system states as defined in the system design. The availability of the system is computed from the availability of components. The operational availability of the component is computed with the MTBF and the MTTR.

$$A = \frac{MTBF}{MTBF + MTTR}$$

These operational availability analyses examine systems for the transition of state, from operational to non-operational. The analysis is required for maintenance of the system and to determine its performance. There is a relationship between reliability and operational availability of a system. At first glance, it might seem like if a system has high availability, then it should also have high reliability- however, this is not necessarily the case. Reliability represents the probability that components, parts, and systems perform their required functions for a desired period of time without failure in specified environments. Reliability, in itself, does not account for any repair actions that may take place. Reliability defines the time it will take for a component, part, or system to fail while it is operating. It does not define the time taken to get the unit back to working condition.

As stated earlier, operational availability represents the probability that a system is capable of producing the functionality required of it, given that it has not failed or is not undergoing a repair action. Therefore, availability is a function of not only reliability, but also maintainability. Table 1.1 below shows the relationship between reliability, availability, and maintainability for determining system performance.

Reliability	Maintainability	Availability
Constant	Decreases	Decreases
Constant	Increases	Increases
Increases	Constant	Increases
Decreases	Constant	Decreases

Table 1.1: Relationship between Reliability, Maintainability and Availability

As mentioned in the Table 1.1, even if the reliability of the system is held constant at a high value, the system may not have high availability. This means that the availability decreases as the time to repair increases. Even a system with low reliability may have high availability if the time to repair is short. The approach to compute the operational availability is shown in Figure 1.1.

1. INTRODUCTION

This figure shows an approach to determine operational availability using the mean up time, mean time between failures, mean down time, and mean time to repair. These failure rates are numbers for the components, sub-systems, and system respectively.

The functionality property of the system is validated by means of modeling and simulation. Tools and techniques exist to validate a design for its functionality. For better behavioral analysis of a system, effective techniques are required to determine the availability of a system's functionality at any given instant. This technique will help in analyzing a system's functional availability and in detecting the design anomalies to further improve its design. This is important with increasing complexity, functionality, and desired longevity of systems. This thesis proposes a novel technique to determine the system's functional availability in the design phase of engineering process.

For an effective engineering process, FM techniques are integrated into the process. Akerlund *et al.* [25], Whalen *et al.* [26], and Chen *et al.* [27] discuss the integration of FM into system safety and reliability analysis. Whalen *et al.* [26] discusses the integration of FM techniques into model-based systems engineering using Simulink and SCADE approaches. Chen *et al.* [27] discusses a formal framework for validating Simulink models for safety critical systems with the TIC formal verification techniques. Metrics derived mathematically logically determine the effectiveness of techniques used in the engineering process for high reliability and safety of the product. For example, reliability is de-

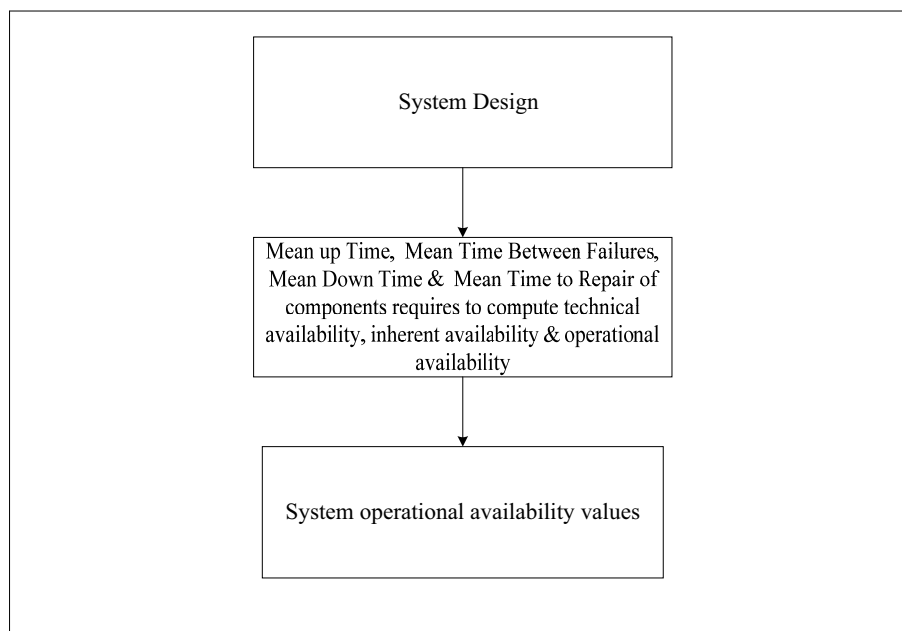


Figure 1.1: Approach to Compute System Operational Availability

1.1 State of Art Research Work in Safety Critical Systems

terminated by means of reliability block diagram (RBD), and safety by means of fault tree analysis (FTA), failure hazard analysis (FHA), mutation analysis and fault tree analysis.

System functionality and its availability during the life of a system play a critical role in determining the behavior of a system. Availability of a system is defined as the availability of its functionality at a given time in its life span. Functional availability of a system can be used to detect design anomalies improving the design and life-cycle process. Hence a FM-based technique to determine the availability of a system's functionality is proposed. The integration of this technique into the engineering process will make the process more effective. The integrated process-FM and the existing techniques-is visualized in Figure 1.2. This integrated process detects design anomalies in the system, aiding designers to come up with a better product. The FM-based technique provides additional system design metric leading to a better, safer, more reliable, and overall an effective system design.

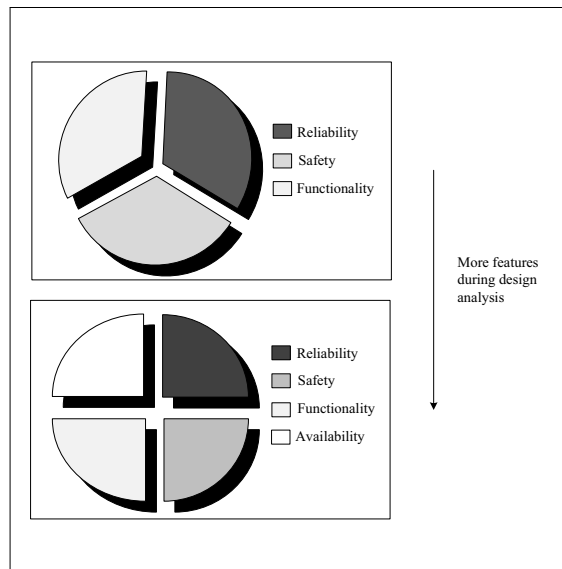


Figure 1.2: Addition of Functional Availability Property in System Design Analysis

There is a need of FM-based technique to determine system's functional availability in the design phase. There is a need for such a technique to detect the design anomalies early in the process to improve the design for increased confidence in design and system behavior. Further, induction of the FM-based technique will build a high-assurance approach in designing the systems.

The functional availability we have proposed is different from operational availability discussed in reliability engineering. SAE 4754A [28] provides guidelines for development of civil aircraft and systems. *Figure 1* [28] of this document provide guidelines

1. INTRODUCTION

for development and in-service operational phases. The development phase life cycle consists of system safety assessment, system development process, hardware, and software development. The operational phase consists of analyzing the system operations for commercial service. We propose the FM-based technique to analyze the system design against the project requirements in the development phase. On the one hand, the proposed approach validates the requirements by means of functional availability of the system, as shown in **Figure 6** [28]. On the other hand, system reliability is a quantitative approach used for in-service operational phases. The reliability of the system considers the operational availability as a part of system reliability analysis. Operational availability is part of the reliability engineering whereas functional availability is part of the design engineering. Operational availability is computed based on data about the components and sub-systems like MTTR and MTBF. Functional availability is computed based on system state transition and analysis after modeling the system mathematically using SA approach. Operational availability of a system is used to verify the reliability requirements of the system for in-service/operational phases. Functional availability of a system is used to verify the functional requirements of the system in the design phases.

This thesis presents a semantics-based integration of a novel FM, a technique based on System Algebra, SA, to determine the functional availability of a system. The proposed technique analyzes ‘functional availability’ to determine a system’s behavior using the concept of ‘System State Machine’, SSM. The technique helps designers to understand system behavior using functional availability property, to detect design anomalies and to modify design all in order to provide the desired functionality. The system state and its transition determine the control flow of the states and related functionality. This analysis improves system design, allowing designers to analyze various design options and to compare their behavior against the specifications of the project. This formalization of the design process and its seamless integration with the existing processes makes the systems engineering more effective.

The concept of SA, proposed by Rao [29] visualizes a system algebraically based on its composition. As per SA, a system exists in one of the three states: the *safe* state, the *hazardous* state, and the *unsafe* state. A system in safe state, is fully functional, behaves as specified and has no impact on safety. For a system in the hazardous state, functionality is limited, leading to degraded performance, with partial effect on safety. For a system in the unsafe state, system functionality is unavailable and safety is not guaranteed. SA technique models (abstracts) a system by decomposing it into independent sub-systems that cannot be further decomposed. This decomposition to the lowest level (atomic) is such that the sub-systems can be combined in series and parallel to form the system again. The inter-relationship amongst the sub-systems helps in deriving the SA expression. The

1.1 State of Art Research Work in Safety Critical Systems

syntax and semantics of an SA expression determines the complexity and fault tolerance of a system as well as its best and worst-case fault tolerance. The failure rates with the semantics of the components/sub-systems in the SA expression determine the system state at any given instant. The SSM, concept proposed, determines and analyzes the system state and its allowed transition from one state to another over time, influenced by failure of various sub-systems and components. The system state at any instant is determined by the health of its components and the state of its sub-systems. The health of a component is determined by its failure rate. Over time, as components start failing, the sub-systems degrade, thus degrading the system due to unavailable functionality. This is the functional availability analysis of the system. With this analysis, the design can be improved for better functional availability. The designer can decide the trade-off for selecting the best design is based on the time, budget, complexity, and functionality.

SSM is similar to *Finite State Machine* (FSM), which has entry, exit, input and transition actions. The entry action is the change in system functionality when it enters a state. The exit action is the change in the functionality of the system when it leaves a state. The input action is the system response, depending on the current state and the trigger input that prompts the system to change state. The transition action is the system response during certain transitions, dependent on system design and application. Applying SA-based design analysis during the design phase of an engineering process modifies the process to make it formal and more effective.

SA is implemented using both the top-down and bottom-up approach. The top-down approach performs a series of decompositions of a system into independent sub-systems and components in the system hierarchy. This decomposition helps in analyzing a system of any complexity. The relationship between between the sub-systems and between the components is expressed by means of simple algebra where '+' is referred to as 'direct sum' and '×' referred as 'direct product'. The bottom-up approach combines the sub-systems using SA expressions and the SSM concept determines the functional availability of the system. The composition of the sub-systems to form the system uses the series or parallel composition. Mathematically, the series composition is represented by '+' and parallel composition is represented by '×'. The SSM concept uses the relationship between the independent sub-systems and the operational rules, proposed by Rao [29], to compute the system state.

The proposed SA approach in the design phase is compared with work of Brosch *et al.* [30] and Melhart and White [31]. Brosch *et al.* proposes a novel approach to validate the reliability of different software architectures or system configuration. The system is modeled using tool which models the system using Palladio component model. The tool then converts the model into Markov chain to determine the system reliability based

1. INTRODUCTION

on software and hardware reliabilities. The proposed approach is validated by taking up industrial case studies of web-based media stores and industrial control systems. The observation of this approach provides feedback on different software architectures to provide a robust fault-tolerant architecture for software product line. Our approach models the system using SA and generates SSM flowchart to determine the system state and its transition. Unlike the O.k and failed state of Brosch *et al.*, our approach considers the system to be either in safe, hazardous and unsafe state. With each state, functionality the system can deliver is analyzed. This simulation outcome is verified against the system requirements. Similar to Brosch *et al.* work we have validated this approach by taking up proven industrial case studies. We have applied this approach on new technology development and modified the existing technology for in-house projects as discussed in sections 3.5.1, 3.5.2, and 4.2.1.2 of Chapters 3 and 4 respectively. Melhart and White [31] discuss the need for providing dependability requirements during the requirement phase of a project. The product and process requirements to achieve the desired system dependability are discussed. Air Traffic Controller example is used to illustrate the concepts of dependability. Our work contributes to provide greater detailing to the availability measure of dependability of the system. As the definition says "Availability addresses the ability of the system to provide service at any given time". SA can be used to provide details on the functionalities provided by the system at a given time based on the system state.

1.2 An Overview of this Work

The research work proposes an integrated formal framework for effective design of safety critical systems. The research was carried out in the following sequence: validation of the applicability of SA to system analysis, formalization of the design phase and the seamless integration of SA-based functional analysis technique with performance analysis techniques such as RBD and FTA.

The applicability and effectiveness of SA for system design analysis are validated by analyzing the safety-critical systems. To prove the effectiveness of a new concept, the best approach is reverse-engineering. Reverse-engineering is used to understand the system requirements. With reverse-engineering, system design can be modified for improved performance while meeting the same requirements. Reverse-engineering is used in upgrading system design. This approach is better than the other system design approaches such as forward engineering and the middle-out approach. Forward engineering approach is used for new system design and development. This approach has lot of unknowns, and involves research and development. In the middle-out approach, the system design may

have to be changed even for slight changes in requirements. In this thesis, SA technique is applied to an already proven system to demonstrate improvement to the design quality even with changes to the requirements. The advantage of reverse-engineering is that new ideas can be applied to an already proven system to demonstrate their capabilities through improved results.

To generate experimental data for the validation of functional availability property, it is applied to already proven systems in aerospace, medicine, railways and automotive sectors through reverse-engineering analysis. In these case studies, each system is modeled and represented by an SA expression. The system design is mathematically modeled and represented by the SA expression; while the operational semantics, proposed by Rao [29], are used to analyze the system state and its transitions. The availability of the system functionality is determined by system states and their transition flow under various failure scenarios. This analysis provides a visual state transition of the system and its available functionality with time. The detailed functional analysis is compared to the functional requirements of the system. This comparison helps in detecting design anomalies, i.e., non-availability of functionality under certain failure scenario. Identifying these design anomalies will enable the design improvements for a better and safer system.

The case studies are steer-by-wire automobile, clinical laboratory medical system, and the flight control systems (FCS) of the Jaguar, Airbus A380, and Boeing B777 aircraft. The simulation observations validated the capability of the SA in analyzing the functional availability of the system. The proposed integrated engineering process is implemented for the flight control systems of Jaguar, Airbus A380 and Boeing B777. This process integrates SA with RBD and the FTA. The comparison of SA with RBD and FTA proves the effectiveness of this integrated process in the behavioral and functional analysis of safety critical systems.

1.2.1 Formalization of System Analysis by Introducing System Functional Availability

This research work proposes a novel FM-based technique to determine the functional behavior of the system during the design phase of the process. The functional behavior is analyzed by the functional availability of the system. The capability to simulate the design using the SSM concept shifts the analysis from data driven domain to simulation based domain. The paradigm shift of availability is shown in Figure 1.3.

1. INTRODUCTION

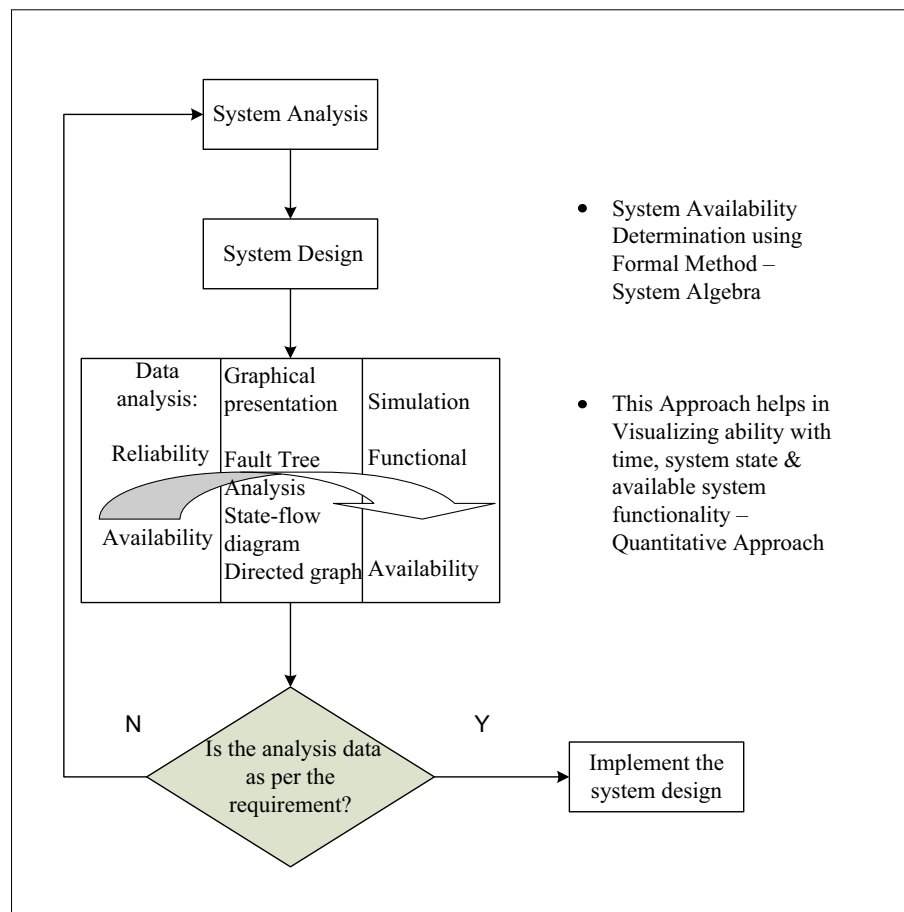


Figure 1.3: Paradigm Shift of Availability from Performance to Behavioral System Analysis

In Figure 1.3, system reliability and availability are analyzed using the component failure data and related calculations. Graphical analysis techniques such as fault tree analysis, state flow, and directed graph analysis, are used to analyze the system safety. The system functional analysis is done using system design simulation. The results of these analyses are compared with the project requirements and if they match, then the design is implemented. The availability property is shifted from data analysis approach to simulation approach, resulting in the behavioral analysis of the system. Figure 1.4 shows the proposed approach to determine the system functional availability, both visually and numerically. The proposed approach models the system using the SA technique. The derived SA expression with component failure rate computes the system's state and its transition over time; determining the system functional availability. The automation of this process is a step towards making the approach user-friendly, especially for complex systems. The SA-based tool will be used to develop the tool-based FM workbench.

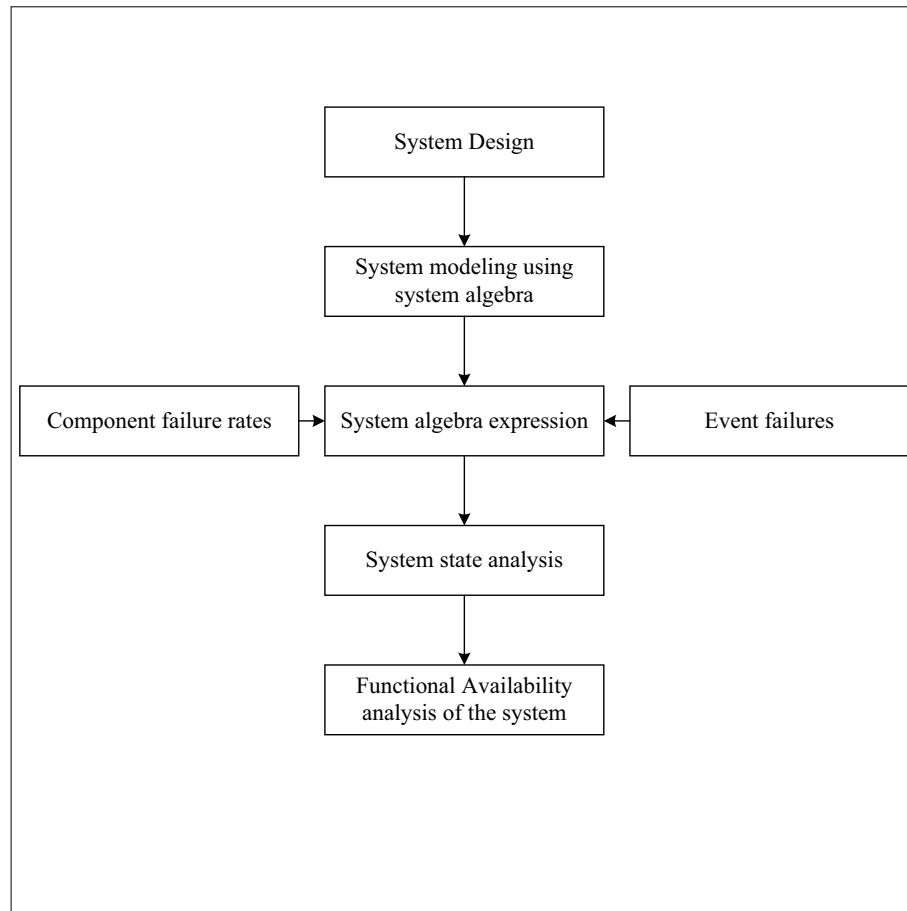


Figure 1.4: SA Approach

In Figure 1.4, the approach to determine system functional availability using SA is shown. The system is modeled and SA expression derived. The component failure rates and the asynchronous external events trigger state transitions from safe to hazardous to unsafe states. The designers can visualize these state transitions over time and determine the system's functional availability accordingly. Figure 1.5 depicts the relationship between system state transitions and functional availability with a simple safety critical system like clinical chemistry analyser in medical domain. Similar relationship exists for aerospace systems like the flight control system, and stall warning system to name a few.

1. INTRODUCTION

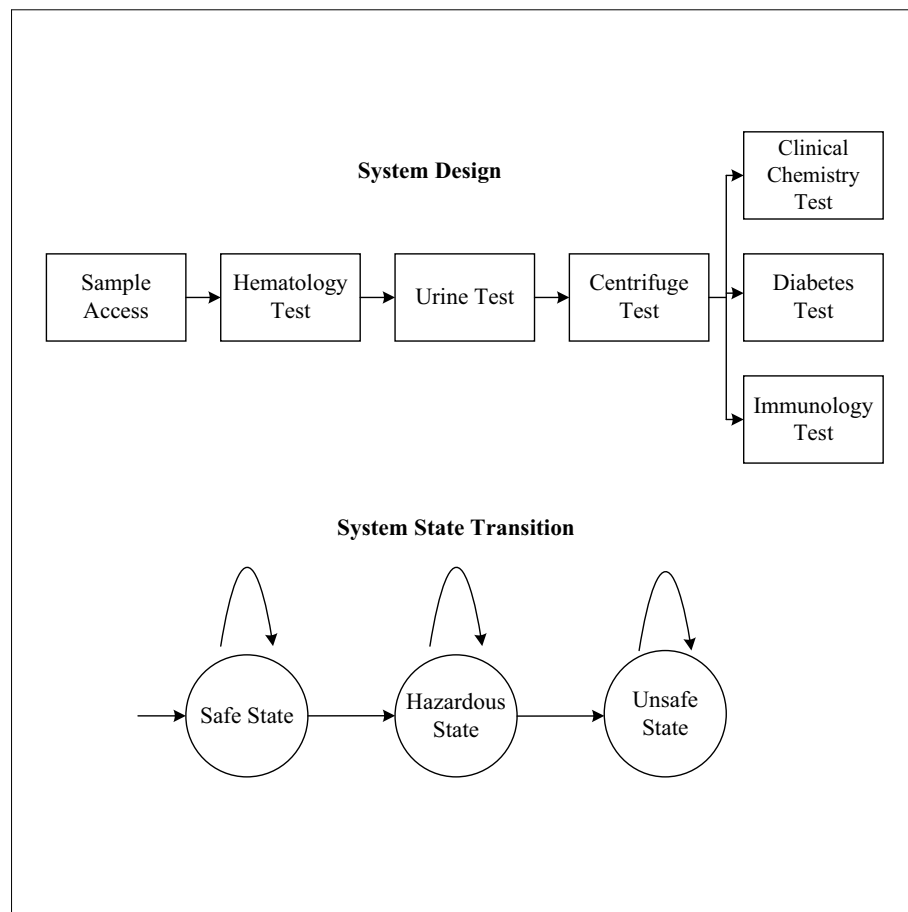


Figure 1.5: Relationship between Functional Availability and System States

1.2.2 Seamless Integration of SA with Reliability and the Safety Techniques

The integration of the SA with existing reliability and safety techniques creates a seamless formalized approach for the engineering process. The migration from an informal to a more formal approach enables a better and more effective engineering process. This integration of the SA with RBD and FTA was performed on the flight control system as a case study. The system analysis for the system functional availability, reliability, and safety helped in improving the system analysis. The introduction of this approach modifies the current engineering process, raises the bar of system quality, and lowers the project cost for futuristic complex systems.

Figure 1.6 shows the modified system's engineering process, where the SA is introduced in the design phase. The introduction of SA approach adds a new dimension to an-

1.2 An Overview of this Work

alyze the system design by means of functional availability property. The transition from the conventional engineering process to a modified process is shown in the Figure 1.6, where the modified process brings in functional availability in addition to the safety, reliability, and maintainability techniques. Various tools are used in the process. Examples of tools/techniques that are used for systems engineering are: DOORS for systems requirement capture; Clearquest for project management; SysML or Matlab/Simulink or Rhapsody toolsets for design simulation to verify functionality at the system and/or software level. FTA for safety analysis and RBD for reliability analysis are examples of techniques employed for such analyses. Documents generated as artifacts for certification in each of the engineering phases as per industry standard guidelines are, shown by dotted arrows in the figure.

1. INTRODUCTION

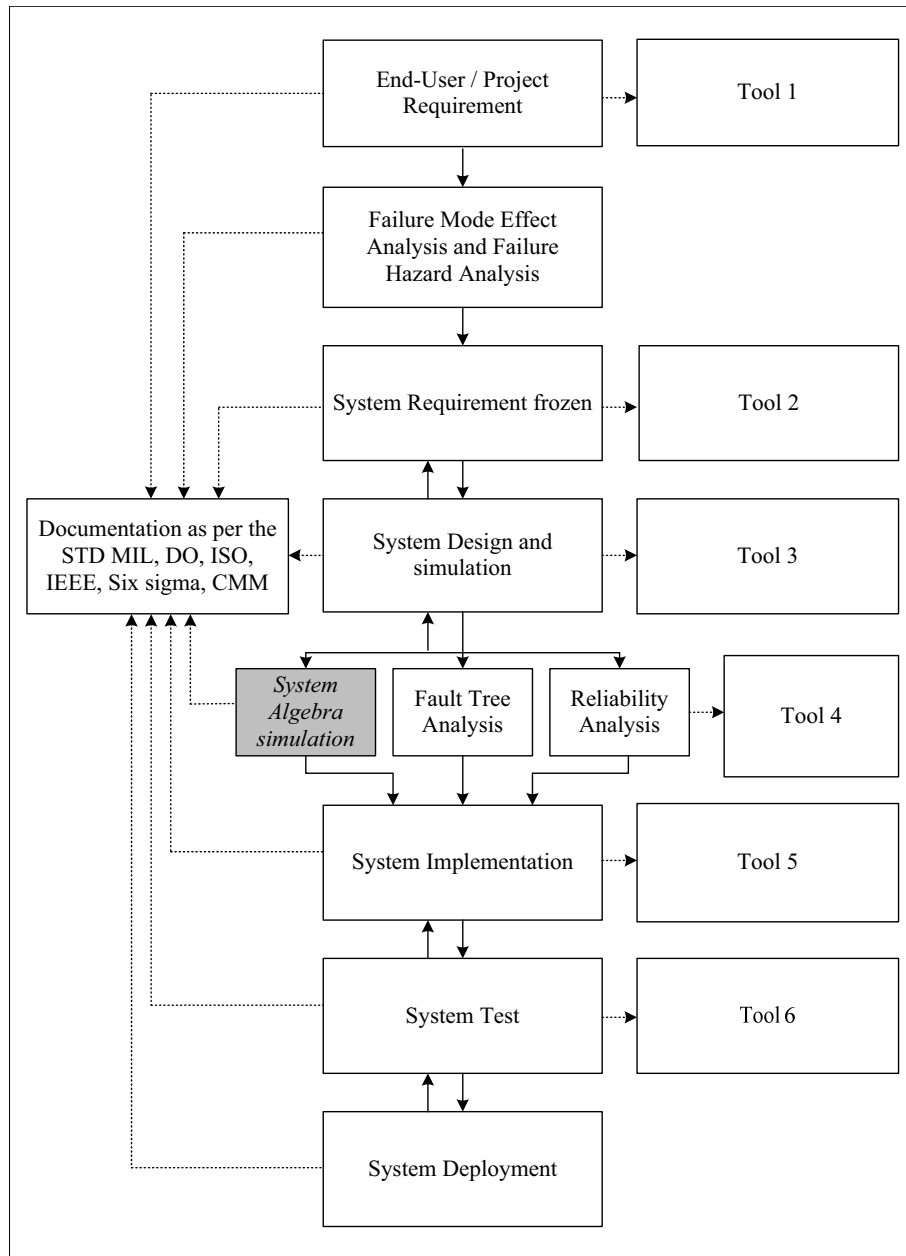


Figure 1.6: Modified Systems Engineering Process

Process automation is done by means of a tool to analyze the system functional availability. The laboratory level in-house tool is developed and its features and capabilities are shown in Figure 1.7.

The tool is capable of:

- Generating the SA expression based on system inputs such as the sub-systems, components, dependency, redundancy, and failure rates.
- Generating a directed graph based on this information and providing qualitative information about system coupling.
- Depicting the system state transitions from the safe to unsafe based on failure rates, redundancy, and dependency of the components.

The tool developed in-house not only automates the technique, but also adds to the existing list of formal method-based tools to provide a formal workbench for design, development, and verification of safety critical applications.

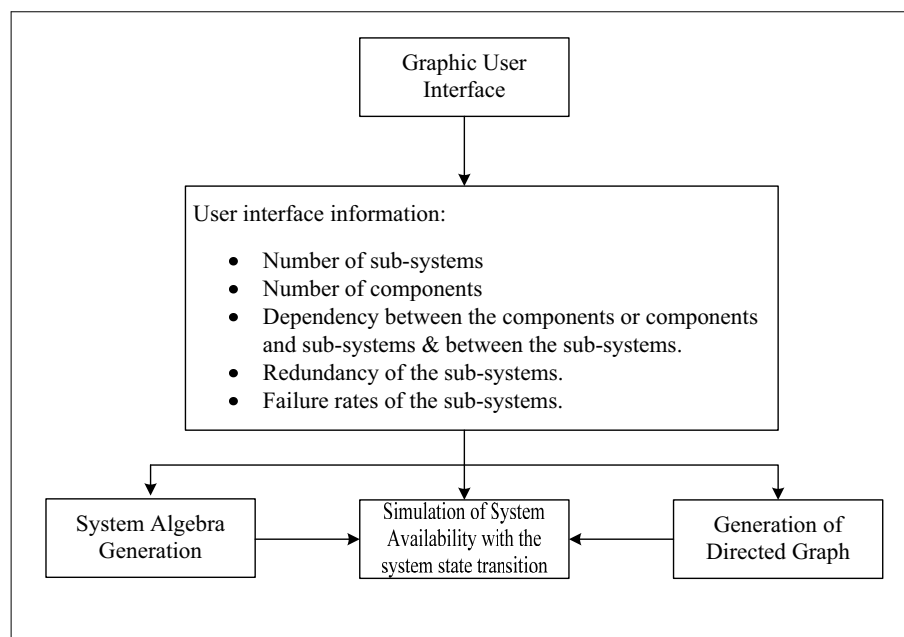


Figure 1.7: SA Process Automation using Tool

1.3 Thesis Layout

The thesis document is structured as follows:

1. INTRODUCTION

- **Chapter 2** discusses the realization, implementation, and improvement in the process by introducing formal methods into real-time systems. It discusses the approach, techniques, technologies, and algorithms implemented in real-time systems. The case studies of the existing formal methods, tools, and techniques to prove the properties of the systems in various phases of the engineering process have also been discussed here, highlighting, the need, importance, and impact of FM on systems engineering. Also discussed is the need for new FM techniques to validate system properties to make the SE process more effective.
- **Chapter 3** discusses SA FM approach, its applicability for analyzing functional availability property of system design especially for safety critical applications. The performance analysis of RBD, FTA and SA lead us to the conclusion that SA is suitable for analyzing the system's functional availability complementing the RBD and FTA. Application of the SA approach for design analysis strengthens the critical phases of system's engineering process.
- **Chapter 4** discusses the proposed modification of the systems engineering process with the incorporation of the SA formal method to determine the system availability metric in addition to its safety, reliability, and functionality. The SA is added in the design phase of the engineering process. The conventional process is compared with the modified proposed engineering process, and its effectiveness is demonstrated by simulating the system state analysis of safety critical systems. This approach is automated with a tool that helps designers visualize the system availability after entering the system details. The chapter discusses the addition of the availability metric to the already existing system metrics of safety, reliability, and functionality. The availability analysis using SA is the paradigm shift from static data analysis to dynamic visual simulation.
- **Chapter 5** discusses the outcome, conclusion and ideas for the future as a way forward for research work.

2

Existing FM Techniques, Methodologies And Systems Engineering

Systems Engineering, (SE), ensures a well-established process aiding the production of a high-quality system. It also helps demonstrate system behavior and functionality as desired by the project. This process evolves to make it effective and improve the quality of system (QoS). In response to the increased demand for systems-of-systems, SE practices have steadily become more formalized and in recognition of the need for improved technical costs, schedule outcomes, and better products it has become more specialized as discussed by Beer *et al.* [32] and Fieler [11]. The inception of model-based systems engineering (MBSE) and formal methods, FM, in the engineering process is a way forward in transiting from a document-centric to a model-centric engineering process. A typical SE process is either document-centric or model-centric, and consists of various phases, such as the conception phase, requirement phase, design phase, implementation phase, and maintenance phase as described in literature described by Honour [12], Teresa [15], Ramos *et al.* [33], Hoban [1], Estefan [3] and Koning *et al.* [17]. All these phases are inter-linked and transit from one to the other, through some pre-determined transition criteria. The engineering model adopted defines the transition criteria between these phases. Each of these phases has some input criteria before transiting into the phase and certain output criteria before transiting out of that particular phase.

Every phase of the engineering process performs reviews, analysis, and/or testing based on the activities planned out. Reviews at each phase enable the engineers to perform a qualitative assessment of the accuracy of requirements, from concept and requirement phases to design and implementation phases. Analysis ensures the ability to repeat the assessment t in each phase. Testing generates test scenarios against the requirement to verify implementation of the requirement; this is usually performed from the design

2. EXISTING FM TECHNIQUES, METHODOLOGIES AND SYSTEMS ENGINEERING

phase. Review and analysis is performed from the concept phase. Figure 2.1 shows the flowchart for the engineering process.

- The **conceptual phase** is the beginning of the engineering process for a project. This phase captures the idea for the project and defines the functionality of the system. The system functionality can be defined either through text or by a model. There are tools that enable capturing of this concept. In this phase of project planning of a system, the design, verification and validation, qualification, and maintenance are planned. The project plan includes development, the tests, the qualification environment, the system standard to be adopted and the artifacts/reports to be generated during each of these phases. The concept phase transits to the requirement phase.
- In the **requirement phase**, the system functionality is captured in the form of requirements and reviewed for consistency with the conceptual phase. The requirement phase transits to the design phase. Review and verification are carried out against the requirement and design.
- In the **design phase**, the requirements are translated into physical specifications. The system architecture and sub-systems required to achieve functionality are defined in this phase. The traceability of the system design to its requirements is reviewed and functionality is verified through simulation. The design phase then transits to the implementation phase.
- In the **implementation phase**, the system is built as per the design requirements, after which it is tested, and reports are generated. The implementation phase transits to the verification and validation phase.
- The **verification and validation phase** is essential, comprising reviews, analysis, and testing at every phase. The verification and the validation are based on the artifacts and reports generated. This phase leads to the deployment phase.
- The **deployment phase** begins after the output from the requirement, design, and verification phases meet predefined criteria. Deployment refers to the installation of the system and operation in the field.
- The **maintenance phase** commences after the system is installed in the field and continues through till the end of its life.

The SE process provides guidelines to ensure a reliable and high-quality product implementable in the industry. The SE process is configured by different industries and organizations for an effective process leading to a highly reliable product. According to NASA Project Management Guidelines, NPR 7120.5 [1], the relationship between systems engineering, project planning, and project control is visualized as shown in Figure 2.2.

Figure 2.2 shows project management that can be thought of as having two major areas of emphasis, both, having equal weight and importance. These areas are SE and project control. There are areas where the two cornerstones of project management overlap. In these areas, SE provides the technical aspects or inputs; whereas project control provides the programmatic, cost, and schedule inputs. The SE process consists of system design, technical management, and product realization. The various activities in SE and project control are displayed in Figure 2.3.

2. EXISTING FM TECHNIQUES, METHODOLOGIES AND SYSTEMS ENGINEERING

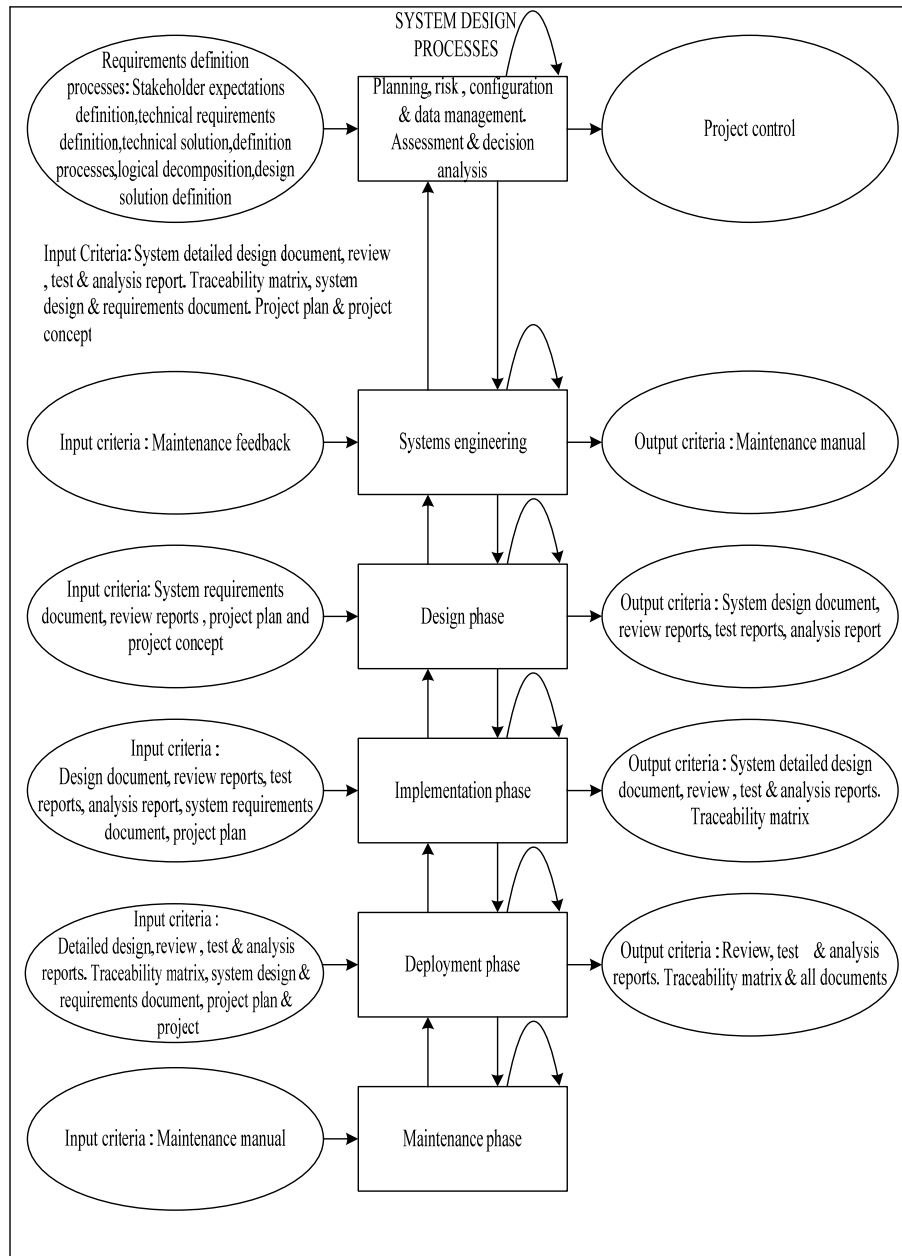


Figure 2.1: Engineering Process Flowchart

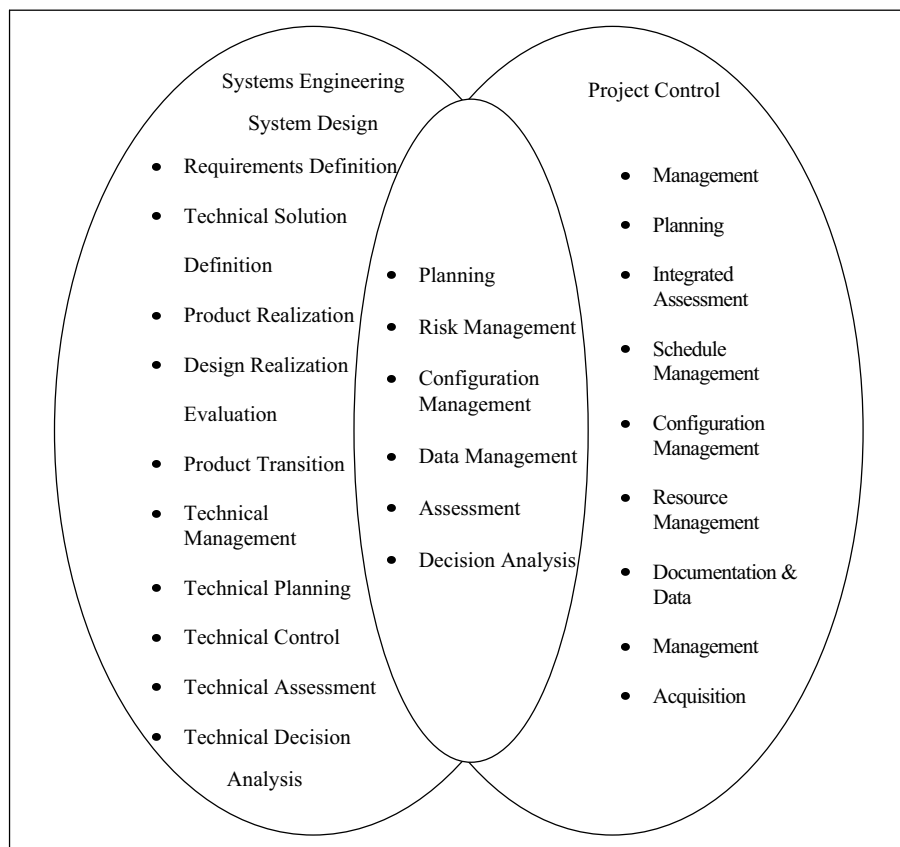


Figure 2.2: NASA Engineering Process Flowchart - 1 [1]

The four system design processes define and baseline stakeholder expectations, generate and baseline technical requirements, and convert the technical requirements into a design solution that will satisfy baseline stakeholder expectations. The product realization processes are applied to each operational/mission product in the system structure, starting from the lowest level product, working up to integrated products at the higher levels. The technical management processes are used to establish and evolve technical plans for the project, manage communication across interfaces, assess progress against plans and requirements for the system products/ services, control technical execution of the project through to completion, and aid in the decision-making process.

There is another way of depicting the SE lifecycle process. This is shown in Figure 2.4. The purpose of SE is to produce systems that satisfy customer needs, increase the probability of system success, reduce risk, and bring down total lifecycle cost [2]. The system lifecycle has seven phases: (1) discovering system requirements, (2) investigating alternatives, (3) full-scale engineering design, (4) implementation, (5) integration and test, (6) operation, maintenance, and evaluation, and (7) retirement, disposal, and replacement.

2. EXISTING FM TECHNIQUES, METHODOLOGIES AND SYSTEMS ENGINEERING

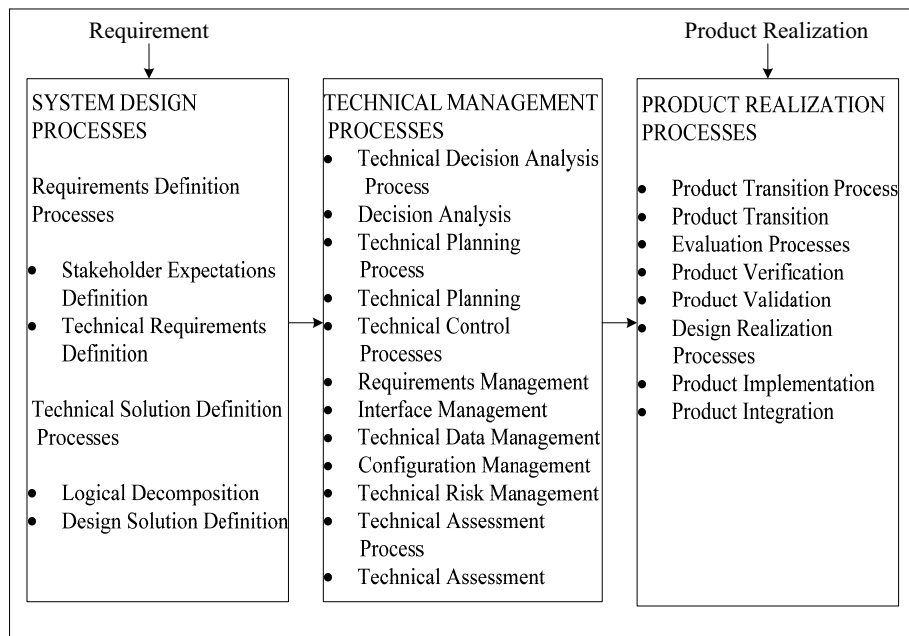


Figure 2.3: Systems Engineering Process Flowchart - 2 [1]

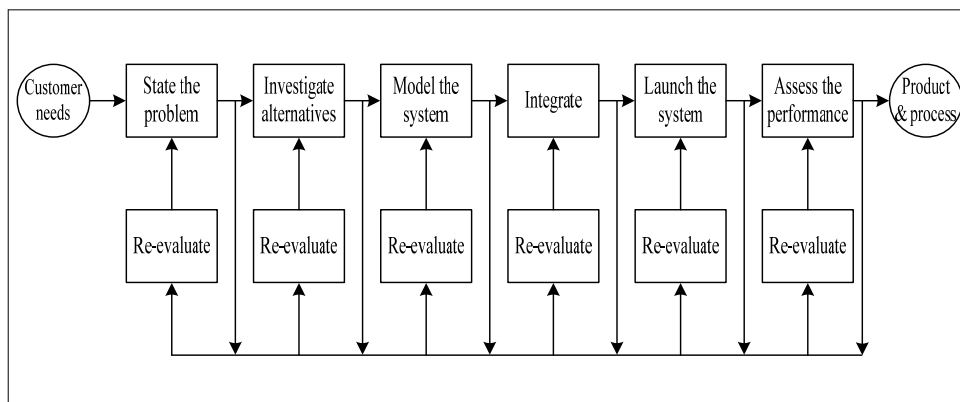


Figure 2.4: Systems Engineering Process Flowchart - 3 [2]

2.1 Systems Engineering Models

Different engineering process flavors are practiced to implement the engineering process in the industry by means of these models. Several engineering models are in place to implement the SE process, all of which play critical roles. Some of the widely-used models are *Waterfall Model*, *Prototype Model*, *V-Model*, *W- Model*, and the *Spiral Model*.

2.1.1 Waterfall Model

‘Waterfall Model’ is the simplest implementation of the engineering process where the feedback is not provided to the previous phase, each phase is transited only after the current phase is complete and correct. The model is like a top-down flow, moving from concept to maintenance and resembling a waterfall in the process. There are some transition criteria at the input and output of each phase, but no feedback. The waterfall model is shown in Figure 2.5.

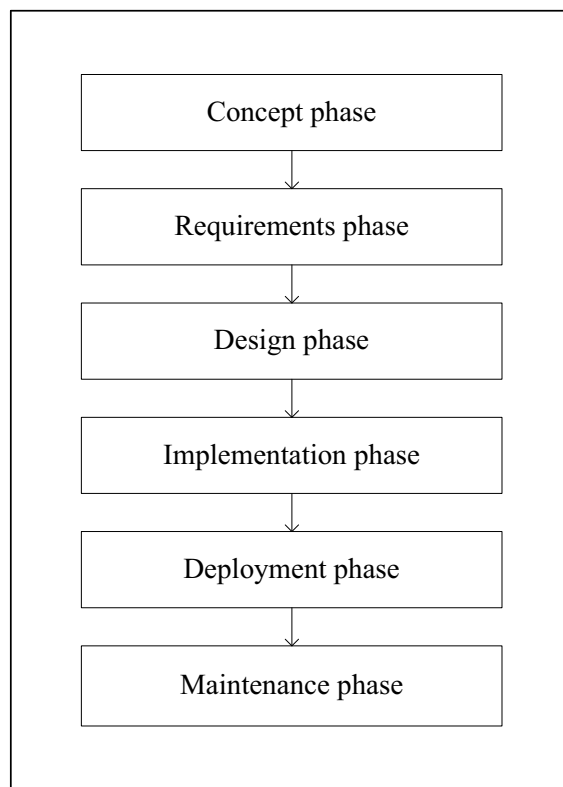


Figure 2.5: Waterfall Model

2. EXISTING FM TECHNIQUES, METHODOLOGIES AND SYSTEMS ENGINEERING

2.1.2 Prototype Model

‘Prototype Model’ of the engineering process follows the approach of complete system development as a proof of concept. The prototype system is built, tested, and then, modified according to feedback from the end-user, till the final requirements are met. This approach is used, and works best, to capture the requirements and system realization in a short time without following the SE standards. This approach is used when the project requirements are known in detail and ahead of time. It is an iterative, trial-and-error process that takes place between developers and users. The prototype model is shown in Figure 2.6.

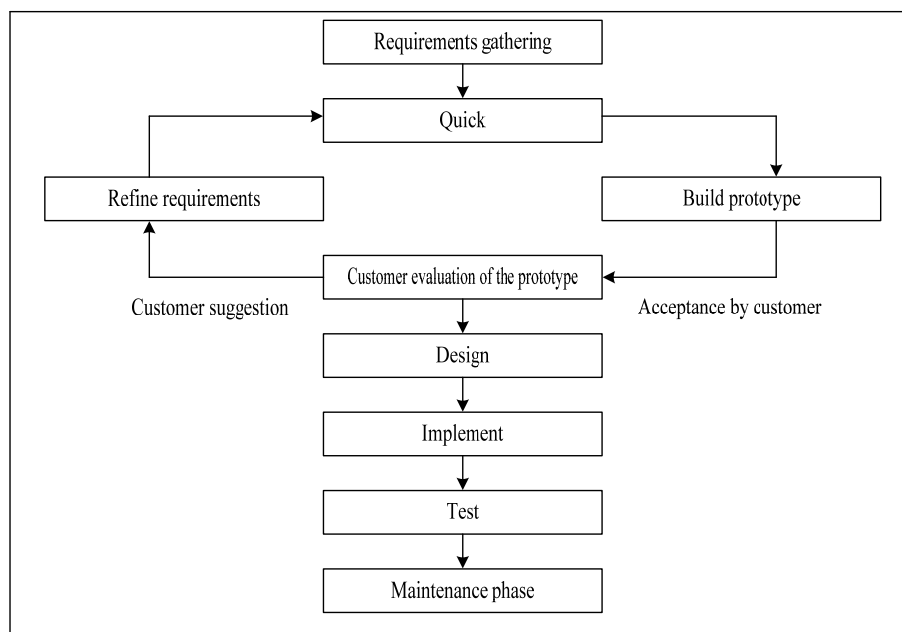


Figure 2.6: Prototype Model

2.1.3 V Model

The airborne safety-critical SE process adopts the ‘V Model’ where the verification and validation phase moves parallel to the requirement, design, and implementation phases. As seen in Figure 2.7, every phase of the process has verification and validation activity, which could be review, analysis, or testing activity.

2.1.4 W Model

'W Model' is a more refined version of the 'V Model'. This model defines the major activities of the engineering process along with the verification and validation carried out for each of these activities. In the 'V Model', it is defined at the phase level, whereas in 'W Model' it goes one step further, in terms of details, to provide better visibility and make the process more effective. This model is shown in Figure 2.8.

2. EXISTING FM TECHNIQUES, METHODOLOGIES AND SYSTEMS ENGINEERING

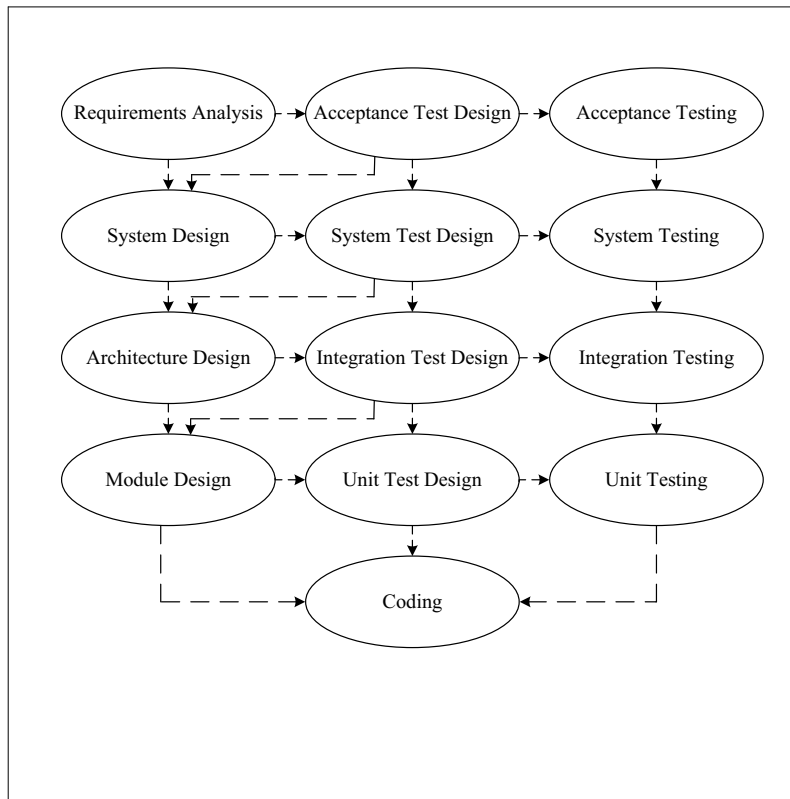


Figure 2.7: V Model

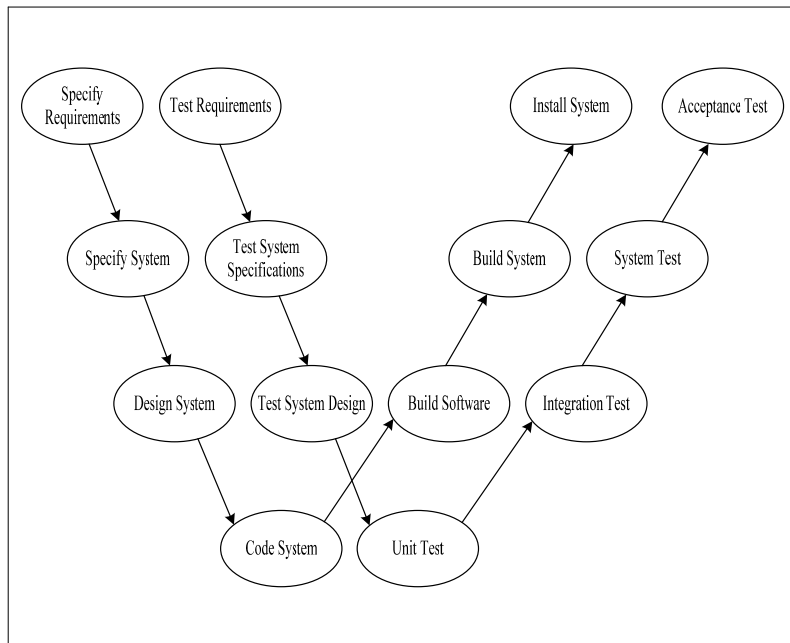


Figure 2.8: W Model

2.1.5 Spiral Model

The 'Spiral Model' combines the features of the Prototype and Waterfall models. The Spiral Model is intended for large, expensive, and complicated projects. The US military has adopted this model for its Future Combat Systems program in an effort to combine the advantages of the top-down and bottom-up concepts. The Spiral Model is displayed in Figure 2.9.

Various industry standards are defined to enable us to implement an effective systems engineering process. Some popular industry standards are the RTCA DO-178 [34] for avionics, EN 50128 for railways [35], EN60880 for nuclear power [36], IEC 61508 for industry [37], and ISO 26262 for the automotive sector [38].

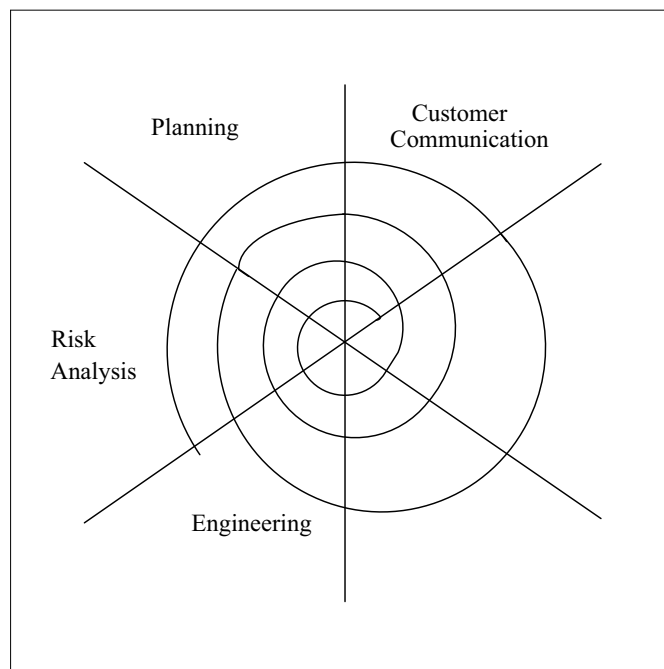


Figure 2.9: Spiral Model

2.2 Evolution of the Systems Engineering Process, Integration of FMs with SE and Industry Standards

There is a need to modify SE to catch errors early in the engineering phase and in order to reduce cost and time-to-market of the product. Leaving detection of errors for later phases of systems engineering is expensive. Modification of the engineering process, from document-centric to model-centric, is discussed by Erkkinen [14]; as an example of the evolution process. The design phase is the most critical in the SE lifecycle, as discussed by Brown *et al.* [39], since it crystallizes the system requirements into a realizable design and sets the stage for subsequent system implementation. The design phase validates the requirements captured, either through documents or models. The model-centric engineering process has an advantage over the document-centric process as the former helps visualize the design, and make it faster and better. It also helps bridge the gap between the requirement and implementation phases as discussed by Bajaj *et al.* [40].

In the design phase, designers validate all possible system designs by metrics for system properties such as functionality, safety, reliability, reach, security, maintainability, and availability; as discussed by Wang *et al.* [9], Milutinovic and Lucanin [41], and Reza *et al.* [10]. In the design phase, simulation is used to analyze system properties as discussed by Kuhn *et al.* [42], Gnad [43], Strobel and Arnold [44], Esterel [45], Xiaohui *et al.* [46], and Milutinovic and Lucanin [8]. These articles discuss the validation of functionality requirements that help demonstrate behavior of a system. Feedback from the simulation is used to improve the design and further analyze system properties. This approach bridges the gap between system requirements, implementation and the end-users. The other advantage of simulating design in the model-based approach is that design flaws are detected earlier in the process. This leads to reduced time-to-market, better quality and provides high assurance of the system [11]. The effect of errors on time-to-market and cost, depending on when they are detected, is shown in Figure 2.10. The increase in cost and time-to-market for the same error detected in the various phases of the process is clearly seen, i.e., the effect of an error detected in the requirement phase is less than that of the same error detected in the deployment phase. This graph underscores the necessity to detect errors early on in the process.

The goal of the SE process is a product that satisfies the customer's needs, has a higher probability of success, reduces risk, and brings down the total lifecycle cost. The SE standards enable designers to develop systems of high quality from the perspective of maintainability, availability, reliability, and safety (MARS). The engineering standards provide guidelines to aid designers in building an effective and high-quality system. The

2.2 Evolution of the Systems Engineering Process, Integration of FMs with SE and Industry Standards

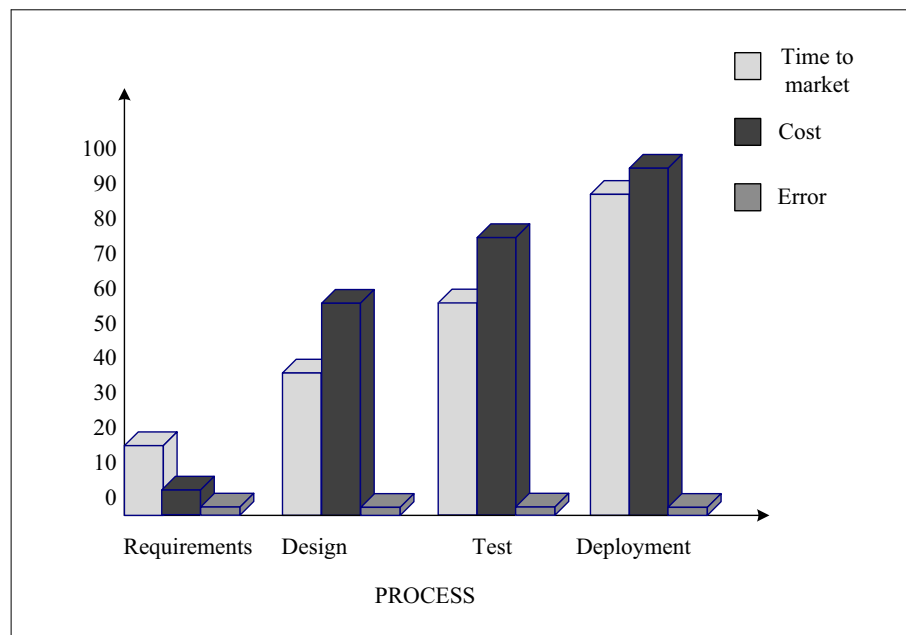


Figure 2.10: Relationship between the Engineering phases, Time-to-market, Cost and Errors

effect of faults on the project cost in the different phases of the engineering process is shown in Figure 2.11.

It is observed that if the faults are detected earlier in the process, then time-to-market and cost of the project reduce. On the other hand, if the faults are detected later in the process, then time-to-market and cost increase. Figure 2.11 shows that the early detection of faults is beneficial to the project in terms of delivery, cost, and quality.

SE standards have evolved with time, making them more effective, from DoD to MIL-STD 499 as shown in Figure 2.12. The figure shows the transition and evolution of the SE standards. Table 2.1 lists some of the popular SE standards. NASA too has recognized the importance of these industry standards with elements referenced and incorporated into the recently ratified NASA NPR 7123. In addition, the International Council on Systems Engineering (INCOSE) [3] announced a commitment toward the adoption of the 15288 standard - parts of that standard have already been integrated into INCOSE, comprehensive guide and resource for understanding the practice of systems engineering.

In practice, FM is used in every phase of SE to validate system properties. Various SE phases, such as requirements, design, implementation, and verification and validation, benefit from the application of FM, as discussed by NASA [47]. FM modifies the engineering techniques to explore and demonstrate critical system properties that are difficult to investigate through simulation. FM modifies techniques for object orientation, capture of requirements, design methodology, implementation, safety, and verification of

2. EXISTING FM TECHNIQUES, METHODOLOGIES AND SYSTEMS ENGINEERING

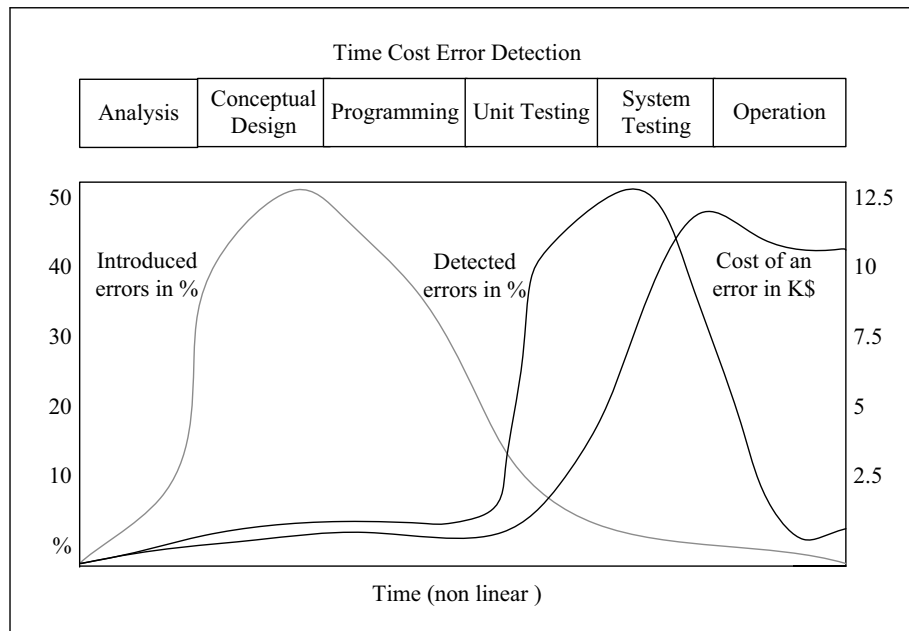


Figure 2.11: Graph Depicting the Cost and Error Relationship

Standard Name	Description
ISO/IEC 15288	Specifies a common framework for describing the lifecycle of a system
ANSI/EIA 632	Provides an integrated set of processes that aid in engineering a system
IEEE 1220	Provides a standard for managing a system
MIL-STD 499A	Military standard for engineering management
MIL-STD 499B	Military standard for systems engineering

Table 2.1: Systems Engineering Industry Standards

2.2 Evolution of the Systems Engineering Process, Integration of FMs with SE and Industry Standards

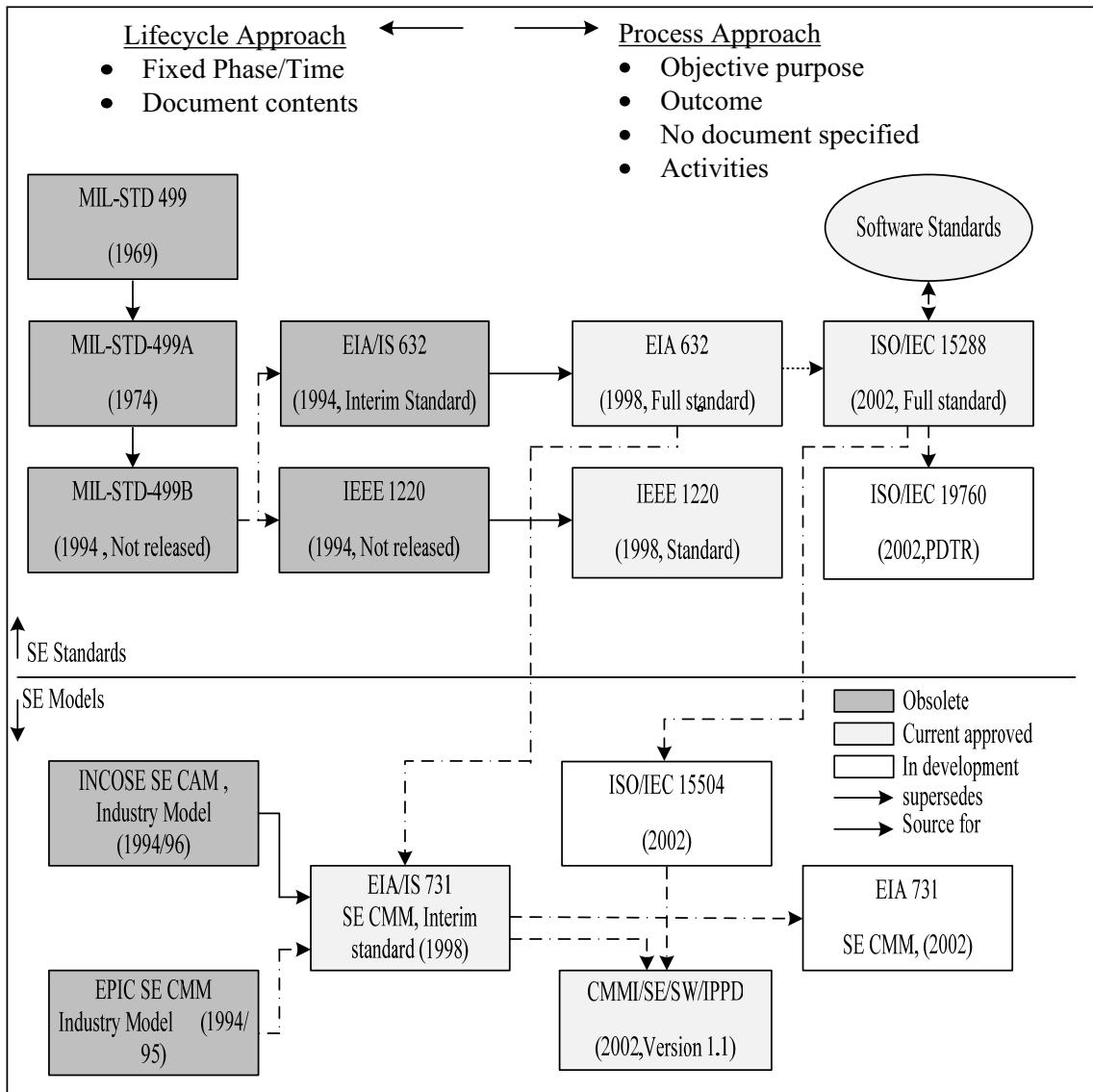


Figure 2.12: Evolution of Engineering Process [3]

2. EXISTING FM TECHNIQUES, METHODOLOGIES AND SYSTEMS ENGINEERING

systems engineering process. These modified approaches ensure a high quality of the system (QoS). To make a system more effective, SE in safety-critical applications adopts tool chains from different vendors, minimizing risk and time-to-market. There is a lot of work being carried out to simplify the inception of FM in the industry's engineering process. Industry standards help people to seamlessly introduce formal method techniques into the engineering process. The known advantages of FM in supplementing the conventional process are realized by the industry. Industry standards propose the usage of FM to achieve the same levels of quality control and assurance by other means, and also derive other benefits such as reduced costs.

The various FM levels that define the rigor required in the formalization of the SE process are discussed by Rushby in [13], [48], and [49]. Based on the applicability of the FM, they can be applied to Levels 0 or 1, or Levels 2 or 3. The engineering process where no FM is used is categorized as *Level 0*. When formal notations are used for logic-intensive system requirements, it is categorized as *Level 1*; the other requirements are written in English. *Level 2* adopts the FM and uses tools for mathematically describing the system requirements; parsing and type-checking is done at this level. *Level 3*, uses a formal specification language, automated tools for specification, model checking, and theorem proving techniques for the verification and validation phase; theorem and proof-checkers are used at this level. On a discrete level, FM's are classified as *light weight* and *heavy weight*. A *light weight* FM offers a cost-effective way of improving the quality of the system. *Light weight* FM adopts *Level 1* and *Level 2* techniques in the engineering process, while *heavy weight* FM adopts *Level 3* techniques. FM selection depends on the criticality of the system, its cost, and the time of delivery of the product.

- ***Level 0 FM*** is the approach adopted when no FM is used. The system analysis is undertaken using conventional engineering techniques; specifications are described using natural language, augmented by diagrams and pseudo-code. Testing in verification and validation phase includes either black-box or structural testing, executed manually.
- ***Level 1 FM*** is the approach adopted when system requirements are represented by formal notations derived from logic and discrete mathematics. Verification is manual and system testing is performed by means of black-box testing.
- ***Level 2 FM*** is the approach used when system requirements are represented by discrete mathematics using support tools to automate the requirements phase of the process.
- ***Level 3 FM*** is the approach when a formal specification language with an integrated development environment is used. The verification is performed formally

2.2 Evolution of the Systems Engineering Process, Integration of FMs with SE and Industry Standards

by using theorem-proving or proof-checking approaches. Advanced tools supporting formal approaches are employed to automate the process, thereby enhancing process quality.

The rigor and level of the FM depend upon the complexity and criticality of the application, the cost of the project, and the time schedule for it. The verification for Level 1 application is performed by means of informal analysis. In case of Level 2 FM, the verification is performed formally by rules of deduction. In the case of Level 3 FM, the formal approach is implemented for requirements, design, and verification.

The rigor and level of the FM can be applied to the early part of the lifecycle, which comprises of requirement capture and system specification; the late lifecycle applies FM to program verification i.e. in the implementation phase. The application, its criticality, cost, and effectiveness help in determining the rigor, level, and the phase of systems engineering where FM can be applied. FM in the latter half of the lifecycle is useful as they are easy to implement and understand, whereas, FM in the early phase is complex to understand and implement, yet very powerful.

In case the system is highly critical, a heavy weight FM is adopted, given that the cost and criticality of the system is high and delivery time is more than that entailed for a non-critical system. In case the criticality of the system is not high then, the light weight techniques are adopted to provide system assurance. Light weight FM ensure reduced costs and swift delivery of the system. There may be systems where the assurance for safety and criticality are not given high priority; under these conditions, no FM is adopted. In a nutshell, FM is:

- **1:** Applicable to all phases of the engineering lifecycle. It is seen that FM are effective if implemented in the early phases of the lifecycle, capturing faults earlier and reducing cost. This helps in achieving a higher performance from a system.
- **2:** Applicable to the entire system, sub-systems, or components. The applicability of the FM to a system, sub-system, or component is dependent on the application and its impact on safety of the system. For example, there are specific applications where FM's are applied only to a critical part of the system.
- **3:** Applicable to validate a system's properties such as safety, functionality, reliability, and availability.

The levels of FM and the scope of FM usage are shown in Table 2.2.

FM can be used to analyze system properties such as functionality, safety, reachability, security, and reliability. They can be used to formalize the system requirements,

2. EXISTING FM TECHNIQUES, METHODOLOGIES AND SYSTEMS ENGINEERING

Levels Of Formalization	Scope Of Use
Mathematical concepts and notations, informal analysis	Life cycle phases
Formalized specification languages	System components
Formal specification languages, comprehensive environment with automated model checker theorem prover	System functionality

Table 2.2: Scope of FM Use

system model, or its verification and validation. It is felt that the more critical the system functionality, the greater is the need for FM. An FM in highly-critical applications helps in ensuring system functionality by reducing the lifecycle time. FM approaches ensure system safety, which is applicable to high-integrity systems whose properties are validated for such systems.

In order to reduce the development costs, formal techniques can be substituted for certain review and test processes to achieve improved efficiencies. Usage of FM is a strictly ‘internal’ process, meaning that no external documentation of their use is submitted in support of certification. FM, is one way to gather ‘evidence for the use of the method’ as required by Section 12.3 of RTCA DO-178B. Formal specifications for some of the requirements and design documents are necessitated by RTCA DO-178B, but they retain traditional methods of review and analysis. Traditional reviews and analyses will continue to be performed, although some traditional data may be supplemented or replaced by formal specifications. Use of FM for safety-critical systems is essential because the practices are inadequate and only FM can significantly increase the assurance provided for these systems.

FM’s have limitations too. The limitation of the FM is in understanding it. For example, formal specifications, such as writing highly complex programs, need expertise. Writing formal specifications is derived from informal requirements. Improper translation may lead to erroneous specifications, resulting in to project failure. This shows that using an FM does not assure a high-quality product. Improper usage of FM may destroy the quality, reliability, and safety of the product. Other examples showing the limitation may be seen in the verification phase. An improper usage of the FM for purposes of verification may lead to unreliable products, affecting both, safety and reliability. Woodcock *et al.* [50] discusses the applicability of FM with measurable data from industries such as transport, nuclear, defense, finance, healthcare, consumer electronics, and telecommunications. FM’s are used for applications like real-time systems, parallel systems, distributed systems, hardware, and high-data volume systems. There is a trend in improving the existing methodologies for making the process effective. The effect of FM on product cost, time, and quality (CTQ), is graphically shown, proving that a properly adopted FM

2.2 Evolution of the Systems Engineering Process, Integration of FMs with SE and Industry Standards

can improve the overall quality of the product, reduce the time-to-market, and the product cost. The usage of FM in engineering processes is both, positive and negative. The authors discuss some of the industry projects, which were executed successfully using FM approaches. One of the critical limitations of formal methods is their scalability i.e., FM suffers from the state explosion problem as discussed by Liu and Adams [51].

There is a lot of work underway to improve the systems engineering for a better QoS. New methodologies are being developed and practiced for making the process more effective. Work is also being carried out through the usage of FM techniques for different applications. Introducing FM into systems engineering has brought about a lot of change, resulting in the evolution of the process, the standards, and the guidelines. Literature provided by Dondossola in [52] and [53], Bowen and Hinchey [54], Bowen and Stavri-dou [55], Gogolla [56], Bicarregui and Matthews [57], Easterbrook and Callahan [58], Qureshi [59], Rushby [48], Ameer *et al.* [60], Prasad *et al.* [61], Hsiung *et al.* [62], Heitmeyer [63], Woodcock *et al.* [64], Caldwell [65], Rushby [49], Joshi [66], Prasad *et al.* [67], Margaria and Steffen [68], National *et al.* [69], Miller and Collins [70], Baudin *et al.* [71], Vassev and Hinchey [72], Cimatti *et al.* [73], IEEE [18], Havelund *et al.* [74] and Beer *et al.* [75] discusses the application of FM.

The usefulness of FM is being realized for safety applications and the civil aerospace standard of RTCA DO-178B provides guidelines to incorporate FM verification techniques to ensure safety, effectiveness, and assurance of the safety-critical systems. This approach ensures safety by eliminating the potential design faults and increasing the quality as an alternative means of aiding in the certification process. In addition to the conventional approach, and according to the RTCA DO-178B guidelines, some activities that can be performed using the techniques of the FM are checking for the correctness of requirements, consistency between the models and the requirements, and prototype development from the specifications to demonstrate the system property and functionalities. Bowen *et al.* [4] discuss the work being done by the RTCA working group to introduce FM guidelines in the software engineering process of the next version of RTCA DO-178B, i.e., DO-178C. The RTCA SC-205/EUROCAE WG-71 is working towards making the FM an approachable one and the conventional engineering process is being modified with the FM-based approach. This modified process is shown in Figure 2.13 and is recommended by the certification workgroup. This figure shows the alternative verification techniques using formal methods, comparing them to conventional verification techniques.

With the industry standard and certification guidelines for incorporating FM in the engineering process, industries have had success stories of using FM techniques and tools to make the project a success. FM has played a critical role in verifying system safety and correctness in projects like IBM transaction processing system using Z-specification

2. EXISTING FM TECHNIQUES, METHODOLOGIES AND SYSTEMS ENGINEERING

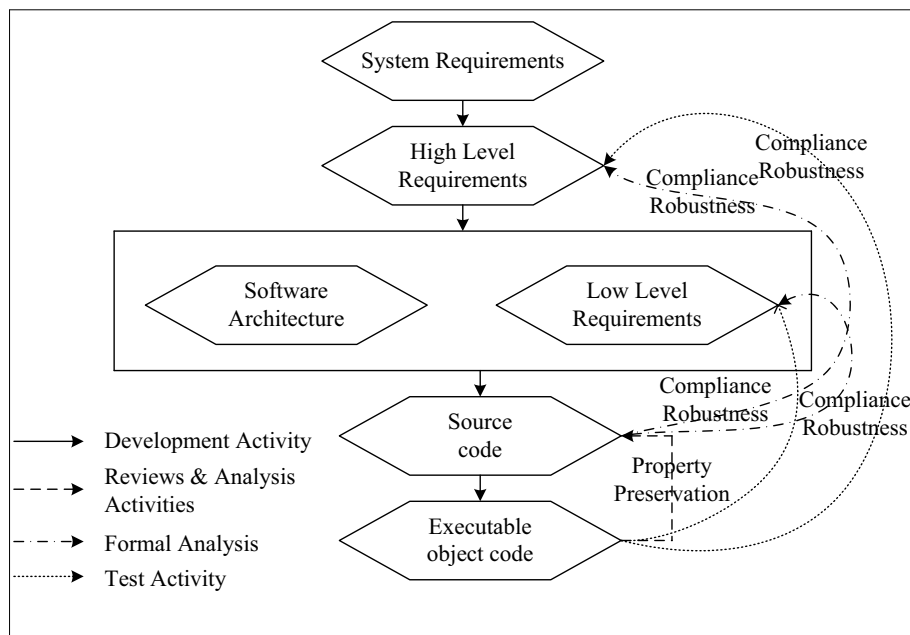


Figure 2.13: Modified Engineering Process for Certification [4]

language, development of the floating point T800 Transputer by Inmos using VDM, the British Viper microprocessor for NASA, usage of the LOTOS language and tools for specifying the model and software for the Flight Warning Computer (FWC) of the Airbus 330 and 340 aircraft, formal specification using state charts for traffic collision and avoidance system.

In his work Kunh et.al [42] describes the usefulness of FM in the verification and validation phase of the engineering process. The capability of FM to define and verify the system properties is one of the main reasons for its employment in the system development process. The FM approach in the specification phase helps improve specification precision, and analyzes, and proves system properties. This can be achieved by means of model checkers and theorem provers. Comparisons between the model checkers and the theorem provers against the notation, method of operation, output, and range of applicability are provided for the proper usage of FM in the verification and validation process. The theorem provers that are most commonly used are PVS, ACL2, HOL, Isabelle, Nuprl, Z/Eves, SCR, and the B-method. There exist tools for these theorem provers, which enable users to easily understand and execute the verification and validation of the system.

Industries have adapted FM techniques and tools for safety-critical systems. Case studies of Airbus and Rockwell Collins discuss FM techniques and tools to ensure the safety and quality of avionics systems. In the Airbus program, FM's are used in the verification and validation phase of the engineering process. This phase for Airbus is carried

2.2 Evolution of the Systems Engineering Process, Integration of FMs with SE and Industry Standards

out at the equipment, system, and aircraft level, as discussed by Laurent [5]. Figure 2.14 shows the ‘V’ model used in qualification of avionics. The verification and validation activities carried out at the aircraft level are numbered 1, 7, and 8; those carried out at system level are numbered 2, 5, and 6; those at equipment level are 3 and 4. The verification and validation are part of the conventional approach and include static and dynamic tests. These tests generate cases to verify the requirements and identify faults in the systems at different levels.

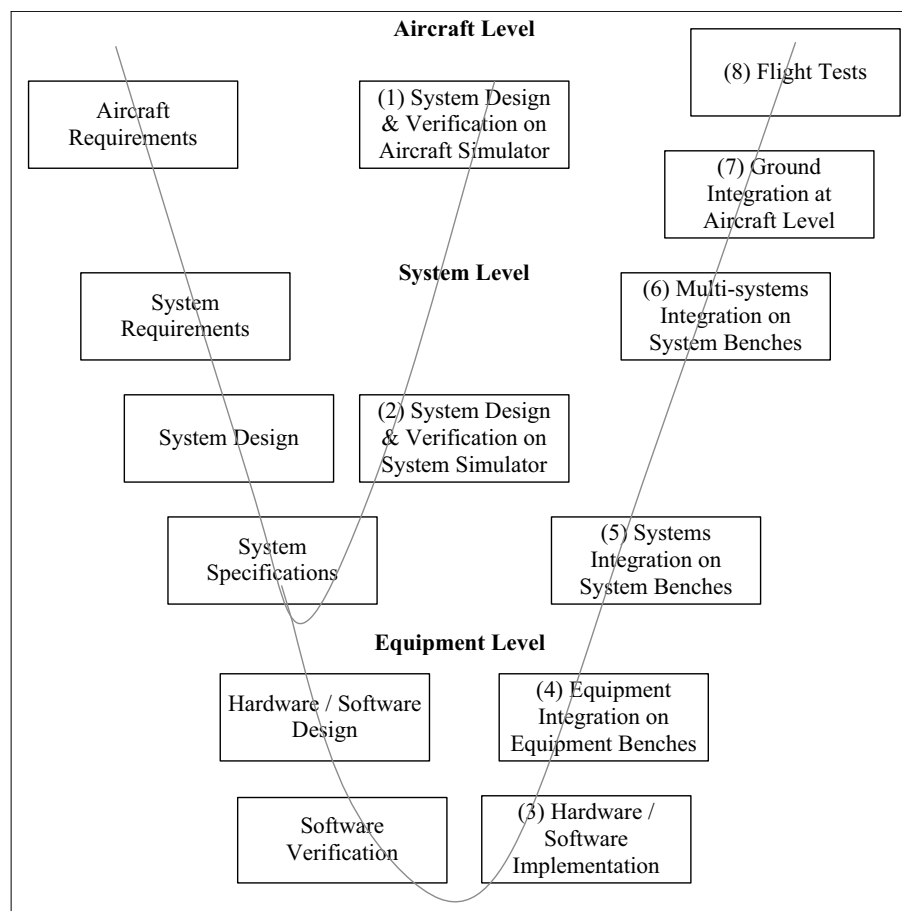


Figure 2.14: V Model Adopted in Aerospace Process [5]

This verification and validation process is modified by the usage of the SCADE formal language. The implemented and modified verification and validation process is shown in Figure 2.15, where the automation is carried out by modeling the system using the SCADE tool, which generates the test cases and the coverage analysis, making the verification and validation process effective along with reduced time and cost.

2. EXISTING FM TECHNIQUES, METHODOLOGIES AND SYSTEMS ENGINEERING

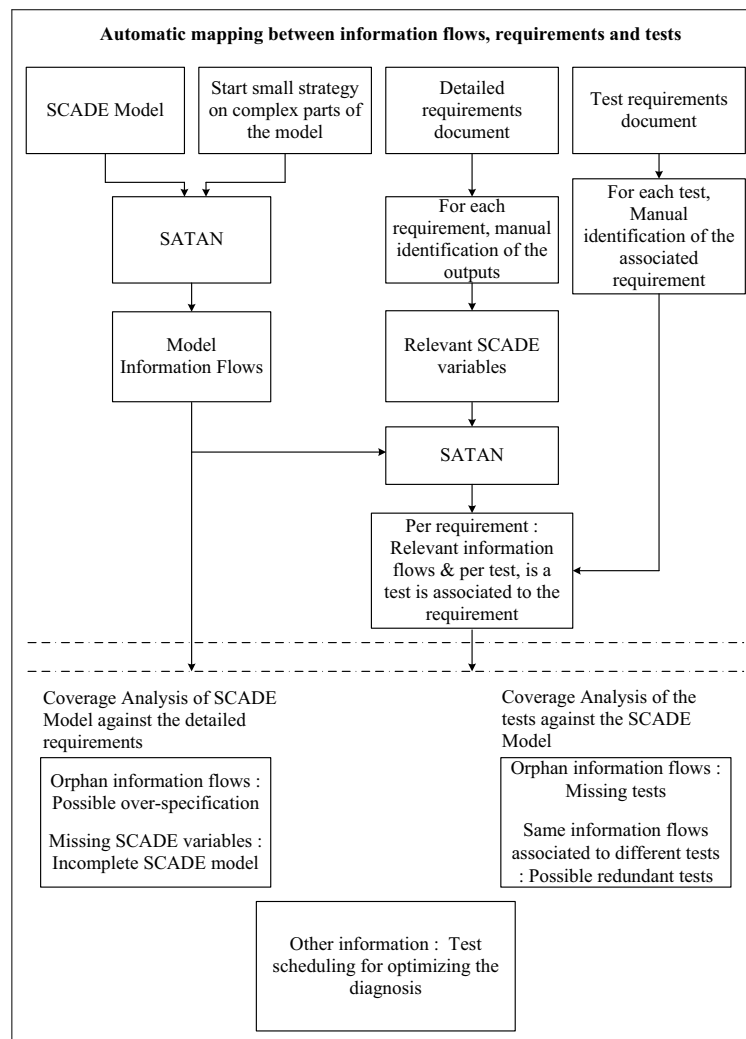


Figure 2.15: SCADe Engineering Process [5]

The research work taken up by Rockwell-Collins adopts the existing formal techniques and tools to realize the effective engineering process discussed by Miller *et al.* [6]. Figure 2.16 shows how the process developed.

2.2 Evolution of the Systems Engineering Process, Integration of FMs with SE and Industry Standards

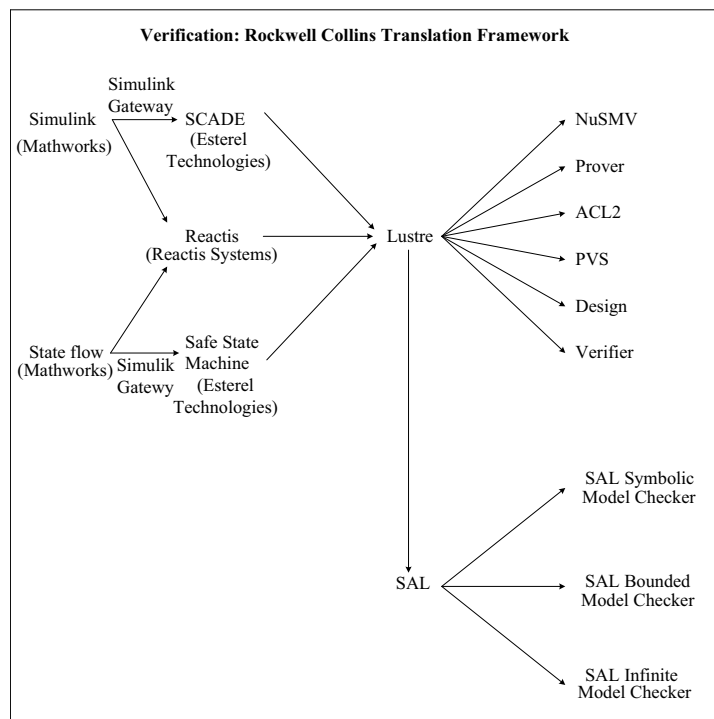


Figure 2.16: Formal Techniques and Tools to Realize an Effective Engineering Process by Miller *et al.* [6]

2. EXISTING FM TECHNIQUES, METHODOLOGIES AND SYSTEMS ENGINEERING

In this process, state charts generated with the SysML tool were verified using the NuSMV model checker. The NuSMV model checker helps in verifying the finite state concurrent systems and can generate infinite sets of possible scenario. A translation algorithm was developed to convert the SysML state chart to the NuSMV model checker interpreted form. This algorithm was developed in-house.

The usage of FM in development, verification and validation, and requirements analysis of safety applications has been discussed. Literature by Benner and Benner Associates [76], Ponsard *et al.* [77], Department Of Defense [78], FAA [79], Mitsumaki *et al.* [80], Harauz and Poon [81], Price *et al.* [82], Burks [83] and Cernes [84] discuss the systems engineering applications, standards, tools, flowcharts, and pitfalls. Cernes [84] describes the importance of systems engineering and its different phases, namely, problem definition, system design, system synthesis, system analysis and modeling, optimization, decision-making, planning for action, implementation, and verification. The paper also discusses systems engineering with respect to safety of critical applications. Laios [85] describes systems engineering in the aerospace domain. The importance of systems engineering in the development of systems such as Avionics, c31, FBW, CCV, and FBL is discussed. IEEE [18] 15288, a popular standard for systems lifecycle processes in the development of commercial systems, discusses the guidelines for implementing an effective process for systems engineering.

This thesis proposes a novel FM-based technique to determine the system's functional availability in the design phase of engineering process. The proposed work is capable of computing the system's functional availability and possesses some importance in today's engineering design world.

3

System Algebra

Formal methods, FM, are gaining popularity, in proving the accuracy and assurance of the systems during the system design phase of the systems engineering, SE, process. System designers can mathematically verify accuracy and safety assurance against the requirements with FM, generating metrics that can be quantized. The FM approach has the ability to detect errors or faults in the design early on in the life cycle. This results in finalizing on a better design within the stipulated time as discussed by Fieler [11]. The only negative aspect of using FM in real time applications is the time it takes to understand and implement them; this learning and training results in no tangible output. The positive aspect of this approach is the safety assurance FM provides in high integrity and safety-critical systems. In complex safety-critical systems, high availability is also a required feature, apart from safety. FM can assure safety and System Algebra, SA, has the capability to guarantee the functional availability of the system.

SA is an algebra-based FM, and is an effective method to analyze the system design through its behavior. The behavior of the system is determined based on the functional availability. The basic notion of SA is that every system of any significant size is created by a composition from smaller sub-systems or components. The system states are broadly classified into safe, hazardous, unsafe, and bad state. Next section provides the definition for the system states. System functionality and availability can be determined for each of these states. A system's fault-tolerance and safety is influenced by the availability of sub-systems or components that build the system. Conversely, a system may itself be a module or part of a larger system, so that its behavior (determined by its functionality) affects the overall behavior of the larger system. SA thus helps in analyzing the behavior of a system by means of functional availability.

3.1 System Algebra Preliminaries

Rao [29] proposed the concept of SA to mathematically model safety-critical systems of any complexity and analyze their safety through mathematical definitions and properties. SA views the safety-critical systems to exist in one of the three states i.e. safe, hazardous, or unsafe state. A safe state does not lead to a bad state under the normal sequence of system transitions. A hazardous state can lead to a bad state or safe state depending on subsequent system transitions. An unsafe state leads the system to a bad state under subsequent system transitions. These states are the safety states of a system as each state provides functionality along with associated safety. Any system, irrespective of its complexity, transits from the safe state to the unsafe state over time. The functionality available also changes from state to state. In a safe state, the entire functionality of the system is available and it reduces as the system transits to an unsafe state. The level of the functional availability is determined by the system design. SA can analyze a system design based on its functional availability, which can be further mapped to the project requirements.

Mathematically, a system state is defined as a poset. The poset B_S is a *bad* state, which is denoted by \perp . There is no transition from a bad system state to a safe system state. The bad state \perp is considered unique; i.e., any system state poset with multiple \perp states is equivalent to a poset with a single \perp . If there are multiple \perp states in a poset, it is possible to replace that poset with another that has only one \perp such that $y \sqsubseteq \perp$ if *any* \perp is reachable from y . A system state is considered *safe* if the bad state cannot be reached from it using any sequence of allowable system state transitions. The set of all such safe states is denoted as $\text{safe}(B_S)$.

Similarly, a system state is called *hazardous* if the bad state can be reached from it using some sequence of system state transitions, but a safe state can also be reached. The set of all hazardous states is denoted by $\text{hazardous}(B_S)$. A system state is called *unsafe* if the bad state can be reached from it using every allowable sequence of system state transitions. Thus, in a sense, the bad state is inevitable from an unsafe state. The set of all unsafe states is denoted by $\text{unsafe}(B_S)$.

Mathematically safe, hazardous and unsafe states of a safety-critical system are identified as 0, 1, and 2, respectively, in the three-value safety logic for a safety-critical system, as defined below.

- Definition 3.1.1.** (a) A state \perp in B_S is designated as *bad* if it represents a failed state of the system.
(b) A state $y \in B_S$ is *safe* if $y \not\sqsubseteq \perp$. Mathematically, it is denoted as 0.

- (c) A state $y \in B_S$ is *hazardous* if $y \sqsubseteq \perp$ and $y \sqsubseteq z$, for some $z \in \text{safe}(B_S)$. Mathematically it is denoted as 1.
- (d) A state $y \in B_S$ is *unsafe* if $y \sqsubseteq \perp$ but $y \not\sqsubseteq z, \forall z \in \text{safe}(B_S)$. Mathematically it is denoted as 2.

The system safety states for a given safety-critical system exist in a finite set of states. The transition between these safety states is defined by reachability. *Reachability* is defined as a partial order: it is reflexive, asymmetric, and transitive and there exists only a finite set of states for safety-critical systems.

SA views the system as a composition of independent sub-systems and components that can be decomposed into disjoint subsystems. These sub-systems are further decomposed till the decomposition reaches the component level. The relationship amongst the components and sub-systems is represented as series and/or parallel, or as a combinational amongst the sub-systems or components. This relationship is represented by mathematical operations; ‘direct sum’ represented as $+$ and ‘direct product’ represented as \times . The sub-systems represented in series, relate by means of ‘direct sum’ and sub-systems represented in parallel, relate by means of the ‘direct product’. The definition of ‘direct sum’ and ‘direct product’ is as follows.

Definition 3.1.2. Let B_M and B_N denote the system-state posets of M and N respectively.

The direct sum $M + N$ of the two components M and N is defined by the following partial ordering on $B_M \cup B_N$. $y \sqsubseteq z$ in $M + N$, if either

- (i) $y, z \in B_M$ and $y \sqsubseteq z$ in B_M ; or
- (ii) $y, z \in B_N$ and $y \sqsubseteq z$ in B_N .

‘Direct sum’ represents a situation where the weakest link between sub-systems has to fail first, for the system to fail.

Definition 3.1.3. The direct product $M \times N$ of two components is defined by the following partial ordering on the cartesian product $B_M \times B_N$: $(y, z) \sqsubseteq (y', z')$ if $y \sqsubseteq y'$ in B_M and $z \sqsubseteq z'$ in B_N .

‘Direct product’ represents a situation where the strongest link between the sub-systems has to fail, for the system to fail.

Operational properties of ‘direct sum’ and ‘direct product’ algebra are used to analyze the system functionality. Based on the definitions 3.1.2 and 3.1.3, we can determine the safety properties of the system given in the following theorem.

If y_M denotes the state of system M and y_N denotes the state of system N , then the safety function values proposed by Rao [29] are given by $\beta(y_M)$ and $\beta(y_N)$ respectively.

3. SYSTEM ALGEBRA

Theorem 3.1.4. For sub-systems $\beta(y_M)$ and $\beta(y_N)$, the ‘direct sum’ is represented as $\beta(y_{M+N})$ and the ‘direct product’ is represented as $\beta(y_{M \times N})$

- (i) If $\beta(y_M) = 0$ and $\beta(y_N) = 0$, then $\beta(y_{M+N}) = 0$.
- (ii) If $\beta(y_M) = 1$ and $\beta(y_N) = 0$, then $\beta(y_{M+N}) = 1$.
- (iii) If $\beta(y_M) = 1$ and $\beta(y_N) = 1$, then $\beta(y_{M+N}) = 1$.
- (iv) If $\beta(y_M) = 2$ or $\beta(y_N) = a$, then $\beta(y_{M+N}) = 2$.
- (v) If $\beta(y_M) = 0$ or $\beta(y_N) = a$, then $\beta(y_{M \times N}) = 0$;
- (vi) If $\beta(y_M) = 1$ and $\beta(y_N) = 2$, then $\beta(y_{M \times N}) = 1$;
- (vii) If $\beta(y_M) = 2$ and $\beta(y_N) = 2$, or vice versa, then $\beta(y_{M \times N}) = 2$.

Here, $a \in \{0, 1, 2\}$ stands for a “don’t care” value.

From the theorem 3.1.4 we can determine the system state based on the system composition. The three-state safety logic is represented mathematically in the following way:

- (i) $0 + 0 = 0$
- (ii) $0 + 1 = 1$
- (iii) $1 + 1 = 1$
- (iv) $2 + a = 2$
- (v) $0 \times a = 0$
- (vi) $1 \times 2 = 1$
- (vii) $2 \times 2 = 2$

The following example determines the system state for an abstract system having the SA expression as shown in equation 3.1.1. The system state is determined using the operational properties given in theorem 3.1.4.

$$S = (D^3 + 3D + 5D^2) \quad (3.1.1)$$

- (i) The safety value of $D^3 = D \times D \times D$ is given by $0 \times 1 \times 2 = 0$.
- (ii) The safety value of $3D = D + D + D$ is given by $0 + 1 + 1 = 1$.
- (iii) The safety value of $5D^2 = (D \times D) + (D \times D) + (D \times D) + (D \times D) + (D \times D)$ is given by $(0 \times 1) + (1 \times 1) + (0 \times 0) + (1 \times 1) + (1 \times 2) = 0 + 1 + 0 + 1 + 1 = 1$.

The abstract system design is such that it starts with a safe state and with failure of sub-system/ sub-systems enters the hazardous state. As seen the total safety value of the system is: $(0 + 1) \times 1 = 1$. Therefore, we see that the system as a whole is in a hazardous

state, given these sub-system / component values and the failure rates. The condition 2x2 for DxD is not considered as it will take the sub-system to an unsafe state which is not a safety value. Based on the system composition, and using the algebra properties proposed by Rao [29] for ‘direct sum’ and ‘direct product’, we can predict the fault tolerance of the system. Based on the lemma and corollaries proposed by Rao, the best-case and the worst-case fault-tolerance is determined using the following theorem:

Theorem 3.1.5. Systems $X, Y \in \mathcal{U}$, if $|X|$ and $|Y|$ are the numbers of components therein, then best-case and worst-case tolerance computations are given below.

- (i) $\beta_{best}(X \times Y) = \max\{\beta_{best}(X) + |Y|, |X| + \beta_{best}(Y)\};$
- (ii) $\beta_{worst}(X \times Y) = \beta_{worst}(X) + \beta_{worst}(Y) + 1;$
- (iii) $\beta_{best}(X + Y) = \beta_{best}(X) + \beta_{best}(Y);$
- (iv) $\beta_{worst}(X + Y) = \min\{\beta_{worst}(X), \beta_{worst}(Y)\} .$

Let \mathcal{U} is the set of all systems, then $\beta_{best} : \mathcal{U} \rightarrow \mathbb{N}$ and $\beta_{worst} : \mathcal{U} \rightarrow \mathbb{N}$ are functions giving the best- and worst-case fault-tolerances of a system. From the theorem 3.1.5 we can determine the best and the worst fault-tolerances of the system.

- (i) $\beta_{best}(A) = i$ for $A \in \mathcal{U}$, if i is the cardinality of the largest set of components of A that can fail without causing A to fail.
- (ii) $\beta_{worst}(B) = j$ for $B \in \mathcal{U}$, if $j + 1$ is the cardinality of the smallest set of components in B whose failure causes B to fail.

Thus if X^n describes the ‘direct product’ of $X \in \mathcal{U}$ with itself n times, and nX describes the ‘direct sum’ of $A \in \mathcal{U}$ with itself n times, then we can say that:

- (i) A system X^n can tolerate faults in $n - 1$ of the subsystems, where $n \in \mathbb{Z}_+$
- (ii) A system nX cannot tolerate any faults

These properties of the SA operators, ‘+’ and ‘ \times ’, help in determining the system state, functionality available at each state.

3.1.1 Mathematical Representation of a System

SA decomposes a system into disjoint sub-systems, where each sub-system can be modeled by a polynomial formed by a combination of ‘direct sum’ and ‘direct product’ methods. The system is then analyzed based on the sub-system states. The system state is a polynomial function of the sub-system states, which in turn are polynomial functions of their components. A component in a given sub-system is characterized by its failure rate or

3. SYSTEM ALGEBRA

reliability factor. The components of the sub-system can either be hardware or software. For hardware components, failure rates are defined. Software components are developed either as a fail-safe or fail-operational applications of the system, per industry standards.

The safe state of a system is characterized by reliable and predictable safe functionality. The hazardous state of a system is characterized by a partially reliable, but predictable, functionality and there is a finite probability that the system can transit to a bad state based on the input conditions. The unsafe state of a system is characterized by unreliable functionality, and the system transits to the bad state over time. The bad state of a system is characterized by system malfunction that may be catastrophic in safety-critical applications. The safe functionality also indicates complete functional availability, while predictable functionality indicates limited functional availability and unreliable functional availability indicates no functional availability.

Any system irrespective of its complexity can be modeled as a combination of series and parallel sub-systems. These sub-systems are integrated to provide the required system functionality defined by project requirements. The sub-systems that are in series are related to one another by means of ‘direct sum’ and those in parallel are related by means of ‘direct product’. The sub-systems may be composed of other sub-systems or components. Reasoning along similar lines, the series sub-systems or components are related by ‘direct sum’ and parallel sub-systems or components are related by ‘direct product’. Sometimes, the sub-systems or components have redundancy of 2, 3, or more depending on the system application. Redundant sub-systems or components are related by means of ‘direct product’. In order to generate the SA expression, a system design is decomposed into series and parallel system designs amongst the sub-systems and/ or components. With appropriate naming for sub-systems and components, the SA expression is derived from the system composition. For example, consider the example of an abstract system with SA expression as described in equation 3.1.2.

$$S = \left((M_{11}^3 \times M_{12} + M_{13}^2) + M_2^3 + (M_{31}^2 + M_{32}) + M_4^2 + M_5^2 \right) \quad (3.1.2)$$

where,

M_2	Subsystem 2 with no components and redundancy 3
M_{31}	Component of subsystem M_3 with redundancy 2
M_{32}	Component of subsystem M_3 with no redundancy
M_4 and M_5	Subsystem 4 and 5 with no components and with redundancy 2

3.2 System Behavioral Analysis using the Functional Availability Property

The above SA expression describes a system with 5 sub-systems. Each sub-system has its redundancy and the system state can be predicted from the failure rates of its sub-systems. For example, sub-system 1 has 3 components M_{11} , M_{12} and M_{13} . M_{11} has a triple redundancy so this component will continue to function until 2 level failures. In sub-system 3, M_{32} has no redundancy and hence the failure of this sub-system will fail sub-system 3 and cause a transition from a safe to a hazardous state providing limited function or to an unsafe state depending on the system design. If the system is a fail-op system, it will transit to a hazardous state providing limited system functionality. If the system is a fail-safe system, it will transit to an unsafe state causing the system to halt.

3.2 System Behavioral Analysis using the Functional Availability Property

SA proposed by Rao [29] provides denotational mathematical means that can be used to model, specify, and understand embedded systems. My research work is to interpret, analyze SA for its usage in design, development or certification of the safety critical civil aerospace systems. Adoption of SA approach in the engineering process as per the civil aerospace engineering framework is the outcome of this work.

Civil aerospace standards ARP 4654A, RTCA DO-178C, RTCA DO254, and AC25.1309-1A provide guidelines for design, development and certification of aircraft technologies. The standards also provide framework for engineering process. System functional availability property and system state machine concept have been derived from SA and used for already proven systems as case studies. The observations of these case studies were used for in-house new technology and modifying existing technologies. These technologies followed the civil aerospace engineering framework. The introduction of this analysis modified the existing engineering process.

3.2.1 Functional Availability using the System State and its Transition: Concept of System State Machine

With the proposal of the three-value system state and SA mathematical properties, it is possible to perform the static analysis of a system state and determine the availability of the function in those states. This is required in order to understand and improve the system by detecting design flaws. Also, if choices are available in state transitions, that information can be used to minimize hazard, ensure safety, and enhance functional availability of the system.

3. SYSTEM ALGEBRA

Based on the SA concept, a system can be visualized as a system state machine, SSM, similar to a finite state machine, FSM. SSM helps in analyzing the behavioral model of a system which is composed of a finite number of states: safe, hazardous and unsafe. Bad state is not considered; once in the unsafe state, the transition is irreversible and bad state is reached subsequently. The transition of the system, from one state to another, depends on an external event or a trigger. Similar to FSM, the SSM will have entry, exit, input, and transition actions. The entry action is the change in functionality of a system when it enters a state, per the system design. For a safe state, the system is designed to exhibit full system functionality. In case of a hazardous state, the system is designed to have limited functionality, and for an unsafe state, the system is designed to cease its functionality and just remain idle to ensure safety. The exit action is the change in the functionality of a system when it enters a new state. The input action is the system performance based on the current state and the input trigger causing the system to transit to another state. The transition action is the functionality required to perform during certain transition; this action is dependent on the system design and application. The SSM and allowed state transitions for a critical system can be graphically represented and visually interpreted.

This analysis is required in systems performing multiple safety-critical functions of varying criticality on a single platform. Indigenously developed Stall Warning/Aircraft Interface Computer System (SWS/AIC), Engine Indication and Crew Alerting System (EICAS) and Automatic Flight Control System (AFCS) are three examples of airborne safety-critical systems. Enhanced fatigue meter (eFM), and crack detection and warning system (CDWS) are examples of ground-based safety critical systems. Airborne safety-critical systems provide critical information to the pilot during flight and ground based critical systems provide critical information offline to flight or maintenance crew on the ground. Since these systems perform critical functionalities we need to analyze them with effective techniques.

SWS/AIC system performs stall computation and warning, aircraft warnings, pitch trim command-monitor, and warning functionalities. These functionalities are independent of each other. In the safe state, the SWS/AIC system provides all the functionalities. In the hazardous state, partial functionality is available. The partial functionalities available are stall computation and warning, aircraft warning, pitch monitor and warning or an allowed combination of these. In the unsafe state, there is no functionality available and the system is in the idle state to prevent safety issues for the interfacing system. EICAS performs the functionality of computation and display of engine parameters, monitors health of critical avionics systems, and generates the caution and warning messages. In the safe state all the functionalities are available. In the hazardous state, the partial functionalities that are available are: computation and display of engine parameters, monitor-

3.2 System Behavioral Analysis using the Functional Availability Property

ing of the avionics health, generation of messages, or an allowed combination of these. In the unsafe state, the system remains idle with failure indication on the display. AFCS, provides primary surface control, flight director, automatic pitch trim, steering commands to the pilot and co-pilot and different modes of flight operation. In the safe state, all these functionalities are available; in the hazardous state, primary control surface and pitch auto-trim, primary control surface and flight director, flight director and pitch auto-trim functionalities are available. In the unsafe state, system remains idle with failure indication. To validate the designs in accordance with the requirements, we need to understand the system behavior by determining the functional availability of the system. This behavioral analysis helps in detecting the design flaws, improving the system availability by modifying the system states and its transitions. The following cases discuss the possible transitions in a safety-critical system.

- ***Case 1: Transition from a Safe State to Hazardous State to Unsafe State***

In this case, the system starts with a safe state. A failure of a sub-system/component causes a transition to a hazardous state. The bi-directional arrow shows that if the failure is temporary, the system can transit back to the safe state. The system will continue to be in the hazardous state if the failure is permanent. The failure of the sub-systems/components will eventually effect a transition to the unsafe state, leading to a system failure. In the SSM, the sub-system/components' failure is the event that triggers a transition of system state. This transition is shown in the Figure 3.1. Most of the safety-critical systems such as AFCS, and EICAS, are designed for such a transition.

- ***Case 2: Safe State to Unsafe State***

In this case, the system starts with the safe state. Failure of a sub-system/components causes a system transition directly to the unsafe state. The system transition from safe to unsafe state, without going through the hazardous state, is in accordance with the system design. In the SSM, the sub-system/component's failure is the event triggering a transition to allowed states. This transition is shown in the Figure 3.2. Some of the off-line monitoring safety critical systems such as Electromechanical Fatigue Meter, and CDWS, are designed for such a transition.

- ***Case 3: Hazardous State to Safe State and Hazardous State to Unsafe State***

In this case, the system starts with the hazardous state. Failure of sub-systems/components forces a transition to the unsafe state. If the failure is not permanent, the system gets back to the safe state. This transition from the hazardous to the safe state is bi-directional, per system design. This design makes system functionality available

3. SYSTEM ALGEBRA

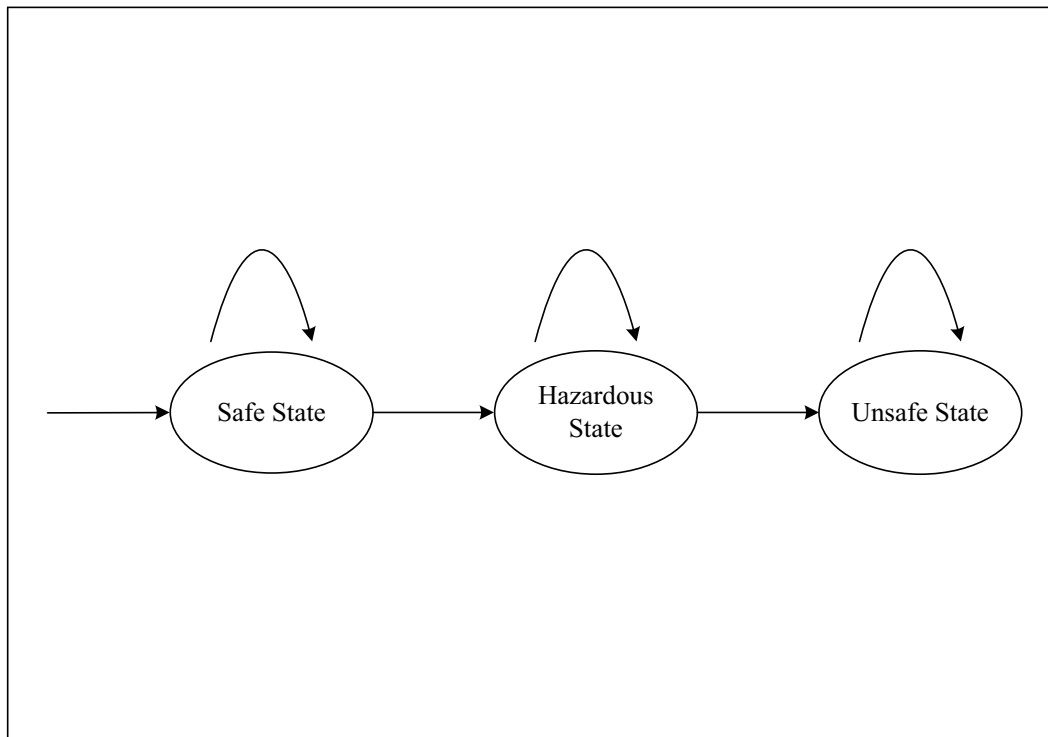


Figure 3.1: Safe State to Hazardous State to Unsafe State

for a longer period of time. The system state transition to the unsafe state is a permanent transition. In the SSM, the sub-system/component's failure is the event that triggers a transition of the system state. This transition is shown in the Figure 3.3. The concept of hazard has been well established in systems engineering for a very long time. Leveson [86] also describes the history of the evolution of this concept from the 19th century onwards. In principle hazard cannot be entirely avoided though they can be diagnosed and mitigated. There exist diagnostics to detect the hazardous state of a critical component. We consider a hazardous state as an intermediate state of the system when it transitions from a healthy (Safe) state to an unhealthy (Unsafe) state. In their work, Leveson and Stolzy [87] provide guidelines to design safety critical systems. They recommend the designer's to ensure that specifications of the system are correctly implemented, and failure leading to a mishap should be minimized or eliminated by means of robust fault tolerant design or fail-safe procedures. Since mishap cannot be eliminated, the design should reduce the exposure time of the hazardous condition. The functional availability helps in analyzing the system state at the design level of the system rather than the component level. The system analysis considers the hardware and software critical-

3.2 System Behavioral Analysis using the Functional Availability Property

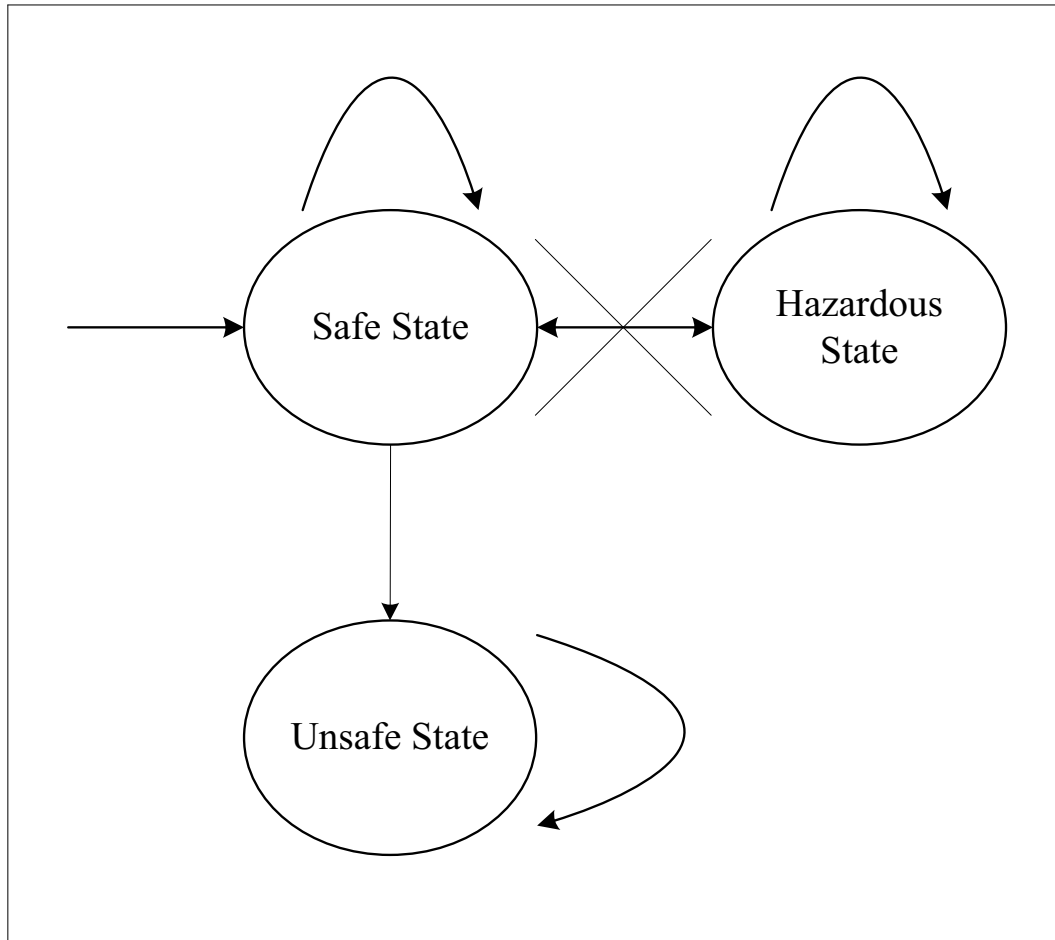


Figure 3.2: Safe State to Unsafe State

ity as per the application and aerospace standards. The SA analysis output provides design feedback to minimize the hazardous condition of the system.

- **Case 4: Unsafe State**

There are times when the system starts with an unsafe state, indicating a permanent failure. This occurs in case of a permanent failure of a sub-system leading to non-operation and unavailability of the system. This permanent failure of the system is shown in the Figure 3.4.

If a system starts with the unsafe state, then transition to any other state, i.e., safe and hazardous states, is prohibited.

The proposed SSM of a critical system, analyzed using functional availability concept, can be compared to FSM and represented as a quintuple as shown in the equation 3.2.1.

$$(\Sigma, S, s_0, \delta, F) \quad (3.2.1)$$

3. SYSTEM ALGEBRA

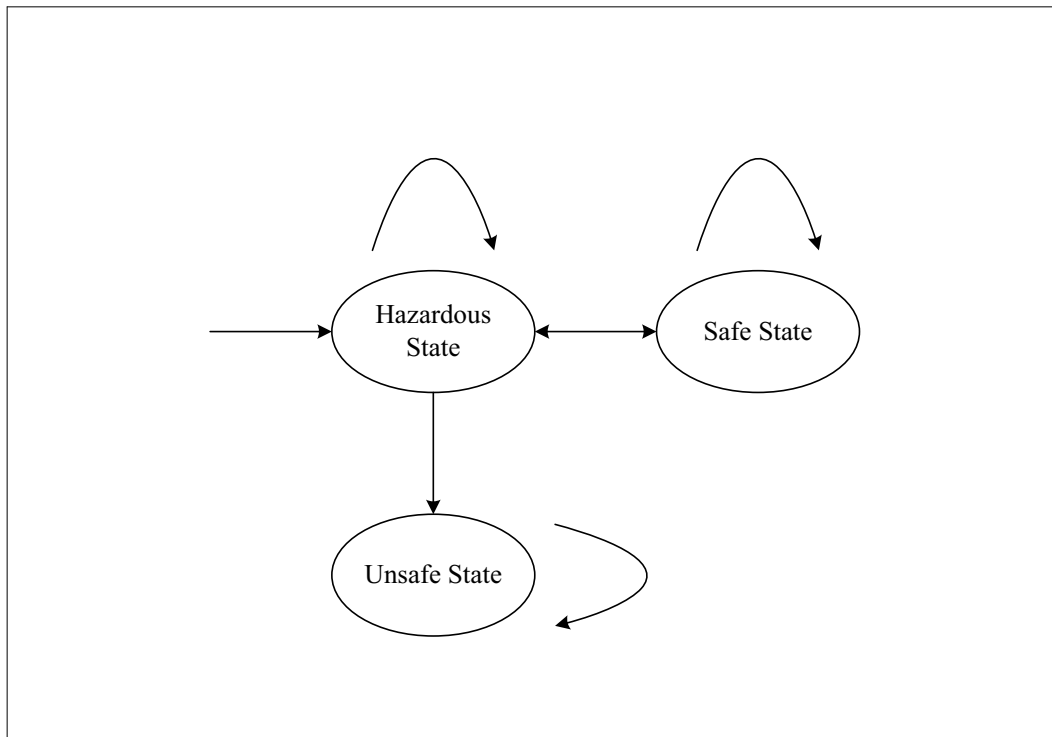


Figure 3.3: Hazardous State to Safe State to Unsafe State

where,

- Σ : Set of inputs for a given system
- S: Finite, non-empty set of states that are safe state, hazardous state, unsafe state
- s_0 : Initial state of the system that could be safe, hazardous, unsafe
- δ : State-transition function (in a non-deterministic finite state machine, i.e., it would return a set of states). This state transition could be from safe state to hazardous state to unsafe state, safe state to unsafe state, hazardous state to safe state to hazardous state to unsafe state, unsafe state, safe state, hazardous state
- F: Set of final states which can be the safe, hazardous or unsafe state.

The proposed SSM, using the system modeling concept of SA, forms the foundation for determining the functional availability property of a system. The system state, its affect on safety of the system, and available functionality are related through the functional availability concept. As discussed, the system can either be in a safe, hazardous, or unsafe

3.2 System Behavioral Analysis using the Functional Availability Property

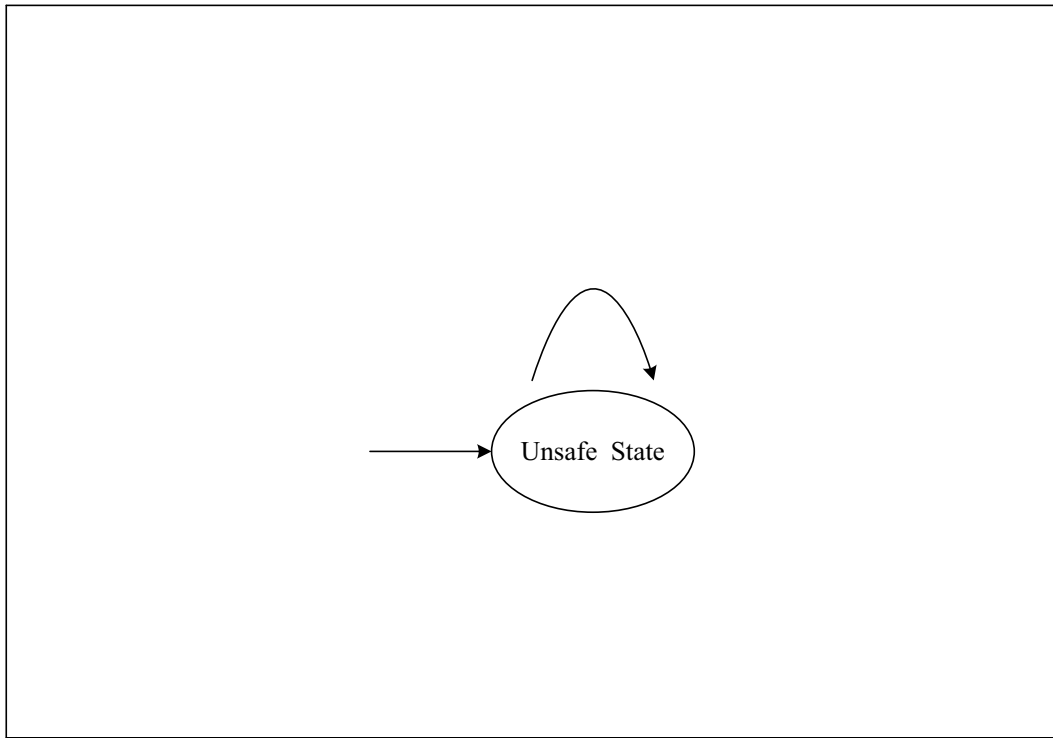


Figure 3.4: Unsafe State

state. For a system in a safe state, full system functionality is available with no impact on safety. For a system in a hazardous state, system functionality is degraded with partial effect on system safety. Finally, a system in the unsafe state has no system functionality available and its safety is not guaranteed. The system functionality based on the system state and its availability is shown in Figure 3.5. The figure shows state transitions of two abstract systems: System1 and System2.

SSM approach performs design analysis similar to failure hazard analysis. We start with single failure of each of the sub-system to analyze the effect on system state. Next we consider two sub-system failures to analyze the effect on system state. This is continued till all allowed combination of the sub-system failures is considered. The combination of the failures with the effect on system state is studied for the systems functionality and verified against the requirements to provide the design feedback. This analysis becomes important input to the failure management which is required as part of certification process of aerospace systems

As can be seen, System1 transits from safe state to hazardous state between T2 and T3. System1 is switched on at T0. At T6, System1 transits to failed state. If the failure rates of the sub-system is 10^3 per hour and each T is equal to 10^2 per hour, then System1

3. SYSTEM ALGEBRA

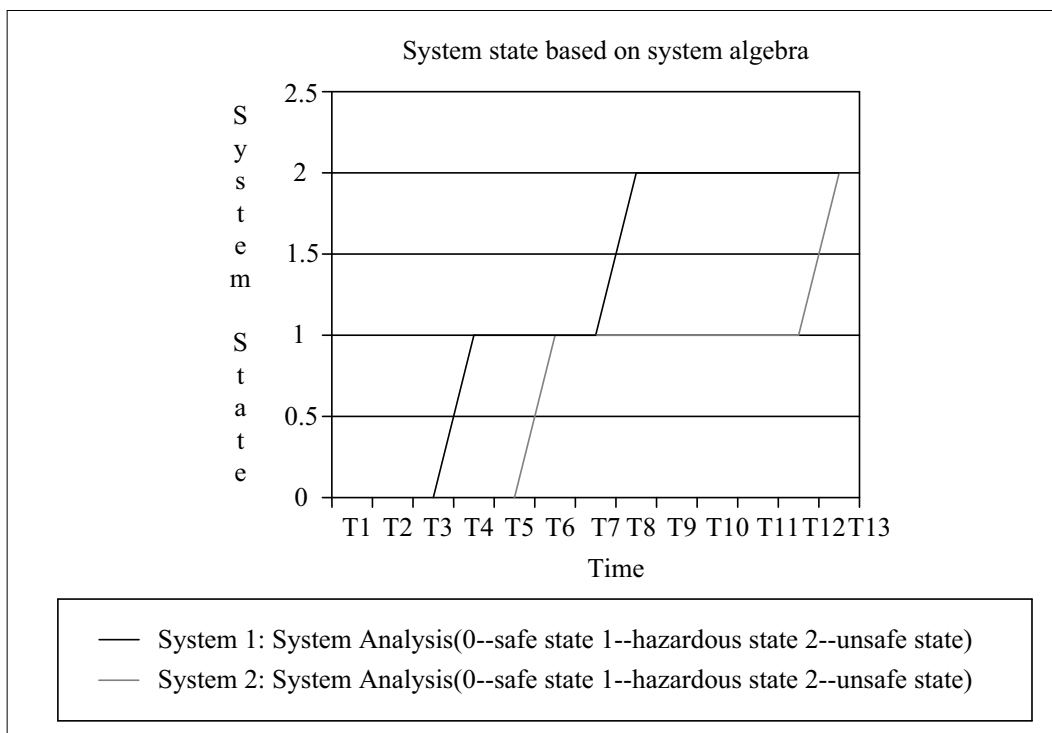


Figure 3.5: System State Transition

3.2 System Behavioral Analysis using the Functional Availability Property

will transit to a hazardous state after 10^4 hours and to unsafe state after 10^{14} hours. This means that the system enters non-functional state after 10^{14} hours. In the hazardous state, transitions will be allowed with available functionalities. SA will analyze available functionalities and provide the design feedback to maximize this availability. System2 is a more fault tolerant system that enters the hazardous state after 10^8 hours and the unsafe state after 10^{22} hours.

Well-known systems in aerospace, automobile, and medical applications are modeled using SA. These systems are analyzed for their functionality and performance against application requirements. The Jaguar FCS design is analyzed using FCS system states and their transitions. Results show that the hazardous state of the FCS system occurs at a rate of 10^9 per hour. The time depicts the failure rate of various components/ sub-systems, leading state to transition from safe to hazardous to the unsafe. Failure of the flight control system is catastrophic and as directed by FAA advisory circular [88], the failure of such system should be extremely improbable. The statement 'For quantitative analysis purposes, extremely improbable failure conditions is those having a probability on the order of 1×10^{-9} or less. Catastrophic failure conditions must be shown to be extremely improbable' puts forth the FCS availability requirements. Fielding and Meng [89] and Belcher [90] also support these design targets for the flight control systems.

The effectiveness of functional availability property based on the SA technique is demonstrated by comparing the performance results with other popular system analysis techniques such as reliability block diagram (RBD) and fault tree analysis (FTA). The relative merits of the SA model are established and discussed. The results demonstrate the capability of SA in analyzing the safety-critical systems in various application domains.

Andre [91] proposes the Safe State Machine, which uses statechart dialect with synchronous model of computation. Safe State Machine is integrated with Esterel for the SCADE toolset [92], [93] which provides a model-based embedded software design, validation, and implementation for safety critical applications such as avionics, automotive, railway, and industry applications. Safe State Machines supports notion of hierarchy, concurrency and pre-emption.

System State Machine, proposed in this work, provides graphical notation to the system modeled using SA. It's scope is limited to design analysis of embedded systems. The simulations using the System State Machine helps in understanding the system functionality and verifying it against the system requirements. It extends the classical finite state machine and provides the simulation similar to statecharts and incorporates the notion of hierarchy, orthogonality and communication between the sub-systems to determine the system state.

3.3 Verification and Validation of SA Approach for Analyzing Safety Critical Systems

SA analyzes the design of a system to reduce the probability of an unreliable state occurring when the system is operational. This technique improves the design of a system, well before implementation and test phases; as discussed by Nanda and Rao [94].

We apply SA to study and analyze safety-critical systems in aerospace, medical, and automotive domains. The system design is studied and modeled for deriving its SA expression and simulating the system availability function. The availability function of the system is analyzed using the system states and their transitions. The transition of a state is based on the failure rates of its sub-systems/components. Proven flight control systems (FCS) are analyzed by *Reverse Engineering Approach* to demonstrate the applicability of SA technique to safety-critical systems. Other critical systems are also analyzed to validate the modeling and analysis capability of SA approach.

The earliest FCS of the Jaguar aircraft is studied by Hills [95]. This FCS system is modeled using SA. The SA expression of the FCS is derived from the model, component failure rates, and its functional availability is determined. The Jaguar FCS block diagram is shown in Figure 3.6.

3.3 Verification and Validation of SA Approach for Analyzing Safety Critical Systems

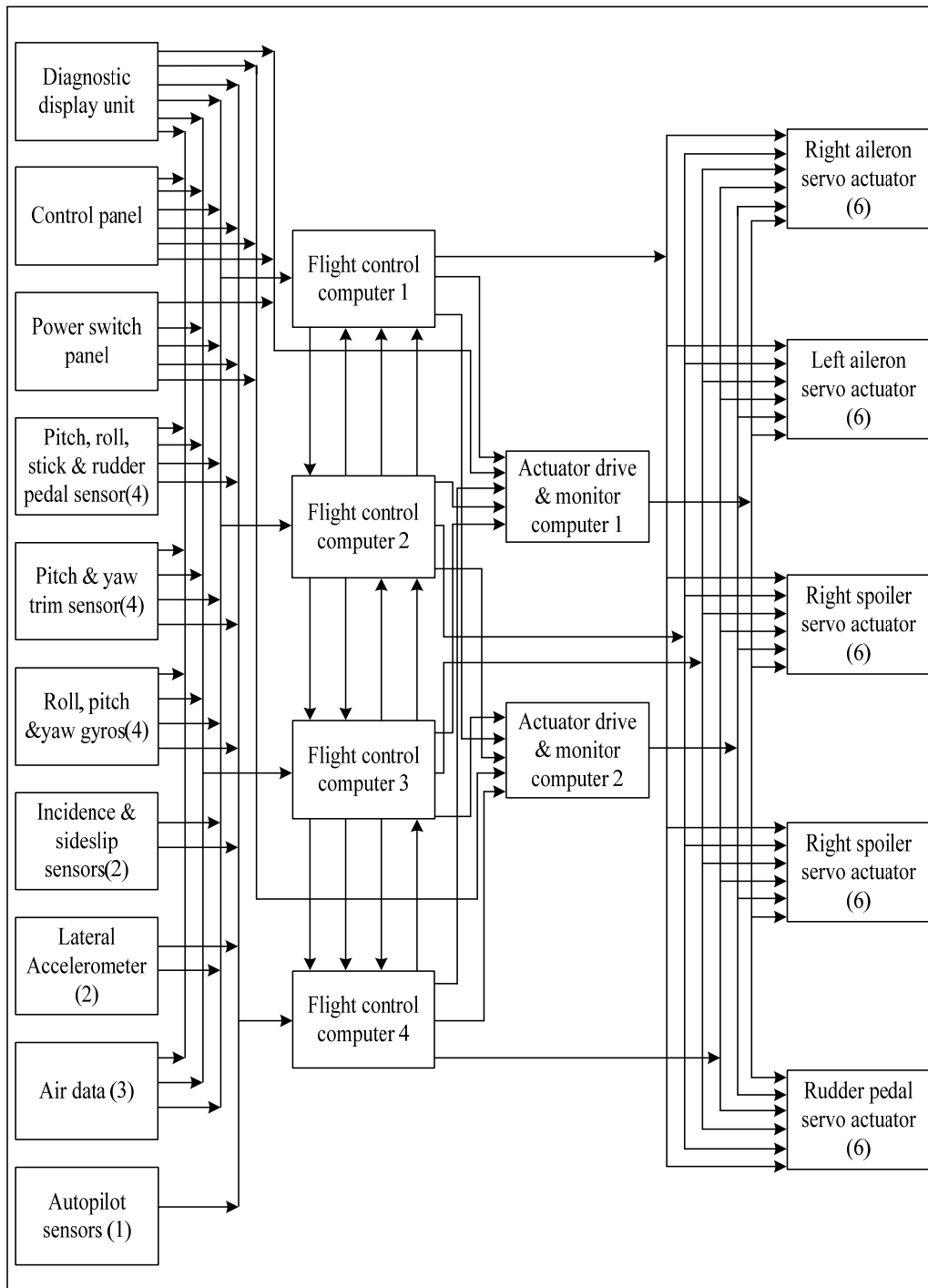


Figure 3.6: Jaguar FCS Block Diagram

3. SYSTEM ALGEBRA

The Jaguar aircraft's FCS consists of input sensors, flight control computer (FCC), and actuators. The sensors capture inputs from various flight control sensors such as air data computers, and actuators. The FCC computes control laws for driving the servo actuators to maneuver the aircraft during a flight. Servo actuators drive surfaces according to the commands given by the FCC. The redundancy of the components, input sensors, flight control computer etc., is specified in Table 3.1. The failure rates for the same are described in Table 4.3.

The FCS system is modeled by decomposing it into series and parallel combinations. Then, it is described with a simple algebraic expression using the SA approach. The failure rates and the redundancy of sub-systems and components decide system state transitions over time. The FCS system design obeys the safety design of the three-state logic. The functional availability is determined from the failure rates of the components/ sub-systems as discussed in Table 4.3. The derived SA expression is shown in the equation 3.3.1.

$$S = (I_1^3 + I_2^3 + I_3^2 + I_4^2 + I_5^2 + I_6 + I_7 + I_8 + I_9) + [C_1^4 + C_v^2] + [A_{LA}^6 + A_{RA}^6 + A_{LS}^6 + A_{RS}^6 + A_{RD}^6] \quad (3.3.1)$$

where,

- $I_1, I_2, \text{ etc.}:$ Input stages, and C_1 and C_v are computational stages
- $A_{LA}^6:$ Left aileron with a redundancy of 6
- $A_{RA}^6:$ Right aileron with a redundancy of 6
- A_{LS}^6 and $A_{RS}^6:$ Left and Right spoilers, a redundancy of 6
- $A_{RD}^6:$ Rudder pedal with a redundancy of 6

System Stage	Component or Subsystem	Redundancy
Input	Dynamic Pressure, Sensor	Triplex
Input	Static Pressure, Sensor	Triplex
Input	Lateral Acceleration, Sensor	Duplex
Input	Angle Of Attack, Sensor	Duplex
Input	Sideslip, Sensor	Duplex
Input	Autopilot, Sensor	Simplex
Input	Diagnostic Display Unit, Subsystem	Simplex
Input	Control Panel, Subsystem	Simplex
Input	Power Panel, Subsystem	Simplex
Computing	Flight Control Computer, Subsystem	Sextuplex
Output	Actuator	Sextuple

Table 3.1: Redundancy of the FCS Sub-systems and Components

3.3 Verification and Validation of SA Approach for Analyzing Safety Critical Systems

Component or Subsystem	Failure Rate	Redundancy
Dynamic Pressure	3×10^{-5}	Triplex
Static Pressure	3×10^{-5}	Triplex
Lateral Acceleration	3×10^{-5}	Duplex
Angle Of Attack	1×10^{-4}	Duplex
Sideslip	3×10^{-5}	Duplex
Autopilot	3×10^{-5}	Simplex
Diagnostic Display	1×10^{-4}	Simplex
Control Panel	1×10^{-4}	Simplex
Power Panel	1×10^{-4}	Simplex
Flight Control Computer	1×10^{-4}	Sextuplex
Actuator	2.5×10^{-4}	Sextuplex

Table 3.2: Failure Rates of FCS Sub-systems and Components

System analysis using SA simulates various component and sub-system failures to study the chain of events that cause state transitions from safe to hazardous to unsafe over time. Component failures lead to subsystem failures, and, the overall FCS state is analyzed using the derived SA expression of the system. A simulation program is developed to compute system state at any point in time and predict functional availability. The program computes the system state using the SA model and its algebraic polynomial. The Gaussian methodology is used to generate random numbers due to its accuracy and wide applicability, as discussed in the paper [96]. The failure rates of the components are randomly generated using this Gaussian distribution. These random failure rates comply with the failure rates of the sub-systems/components described earlier.

The system is analyzed sequentially by examining its state for subsystem failures at input, computational and actuator stages. At each stage, failures are simulated by analyzing the system state derived from the algebraic polynomial. The input stage failure is analyzed based on the component failure, which can either be a no-failure, an unsafe failure or a hazardous failure. Similarly, the computation stage and the actuator stage failure scenarios are also simulated. The simulation helps in analyzing the availability of the system. From the failure rates, the SA expression and system state transitions, it is observed that the Jaguar FCS system is available for a time of 10^9 hours. This time, which is around 20 years, is sufficient for the life of the fighter aircraft. Apart from predicting the functional availability of the system, SA helps in visualizing state transitions from a known state under various failure combinations derived through the SA expression of the system. The system analysis, using this technique, enables designers to propose a better system design, and helps in better understanding the system during the design phase, the accident, or when the incident is reported.

3.4 Verifying and Validating the Critical System Parameters of Similar Systems using SA

SA technique can be used to compare competing system designs by analyzing their functional availability using the SSM concept. To demonstrate this feature, FCS' functional availability of Airbus A380 and Boeing B777 are compared. The FCS systems of A380 and B777 are adapted from the model discussed by Siewiorek and Narasimhan [97]. The SA expressions of the A380 and B777 are derived from their system models. It is observed that although the functionality of both FCS' is identical, the system designs are different, which can be simulated by the functional availability of the system.

A software program is developed to compute system states from the failure rates of the components and sub-systems. The designs are simulated for different combinations of failure at sub-system levels. The system state is observed from the SA expression and is logged. It is observed from the simulation data that FCS of Boeing and Airbus transit to the unsafe state at the same time, except for two scenarios where the Boeing system transits to the unsafe state earlier. This simulation brings out two important observations about the effect of SA on system design analysis. The first observation is that two different system designs from different designers for the flight control functionality of the aircraft performed in accordance to system safety requirement. This observation helps in realizing that there exist various design options to achieve the same functionality. The second observation is that, no two designs are identical; it was observed that in certain failure combinations, the Boeing 777 system enters the failed state earlier than the Airbus A380 system. This proves that the SA technique has the capability to analyze system design against its requirements and provide valuable feedback for improving system design. The block diagram of the FCS for Boeing is shown in the Figure 3.7 and that of Airbus is shown in the Figure 3.8. Various combinations of test scenarios were generated to simulate the FCS system states. The test scenarios were based on the failure rates of the sub-systems and components. The failure rates were randomly generated by in-house software.

3.4.1 FCS of Boeing B777

The block diagram of B777 system is shown in the Figure 3.7.

3.4 Verifying and Validating the Critical System Parameters of Similar Systems using SA

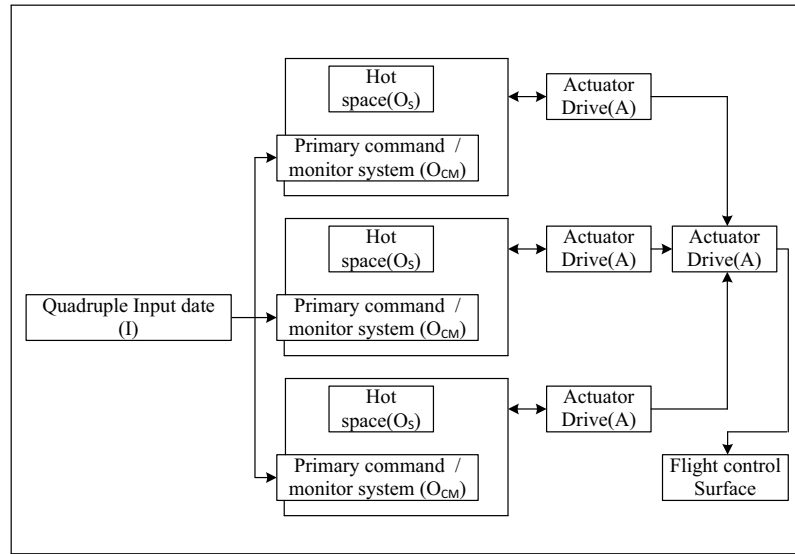


Figure 3.7: B777 FCS Block Diagram

The SA expression for the B777 is derived in equation 3.4.1.

$$\begin{aligned} \text{System} &= \text{InputStage} + \text{ComputationalStage} + \text{ActuatorStage} \\ &= (I_k^4 + O_{CM}^3 + O_S^3 + A^3) \end{aligned} \quad (3.4.1)$$

where,

- I_k : Inputs to the B777 system which define the input stage
- O_{CM}, O_S : Command/monitor and standby system that define the computational stage
- A^3 : Actuator redundancy due to hydraulic and electrical drives that define the actuator stage

3.4.2 FCS of Airbus A380

The block diagram for A380 system is shown in Figure 3.8.

3. SYSTEM ALGEBRA

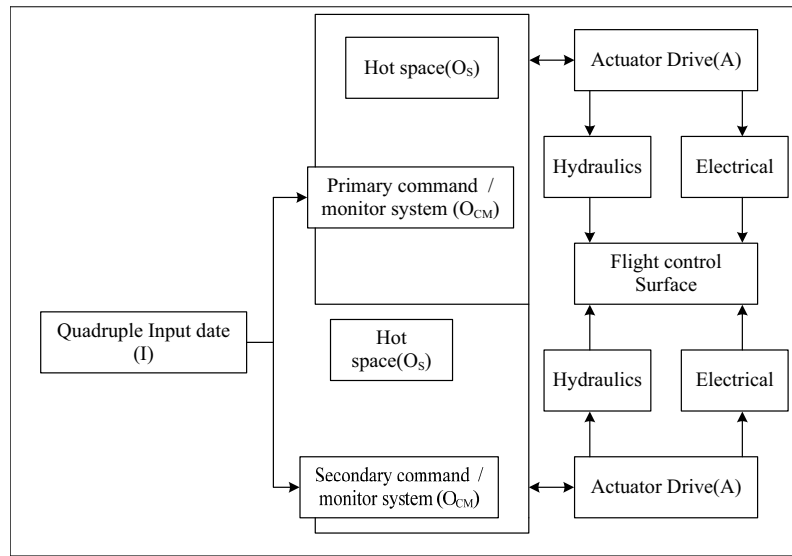


Figure 3.8: A380 FCS Block Diagram

The SA expression for the Airbus A380 is derived in equation 3.4.2.

$$\begin{aligned}
 \text{System} &= \text{InputStage} + \text{ComputationalStage} + \text{ActuatorStage} \\
 &= (I_k^4 + O_{CM}^2 + O_S^2 + A^2)
 \end{aligned}
 \tag{3.4.2}$$

where,

- I_k : Inputs to the A380 system that define the input stage
- O_{CM}, O_S : Command/monitor and standby system that define the computational stage
- A^2 : Actuator redundancy due to hydraulic and electrical drives that define the actuator stage

FCS' of B777 and A380 consist of input sensor stage, flight control computer, and the actuator stage. The signal flows from the input sensors to computation and finally to the actuators. The mathematical representation of each stage is derived from the redundancy and the overall system polynomial is computed from the input, computational, and actuator stages. FCS system is simulated given the failure rates of the sub-systems in Airbus and Boeing, assumed to be identical for this analysis. Similar failure rates are also

3.4 Verifying and Validating the Critical System Parameters of Similar Systems using SA

taken for demonstrating the capability of SA for comparing two identical safety-critical systems.

The system state of FCS for Boeing and Airbus is analyzed for failures of components in the sub-systems by generating random failed data sets. The failed data is generated based on the failure rates of the components. The state transitions from the safe to hazardous to unsafe are compared, providing insight into system design.

3.4.3 Validating System Design based on System State Transition

Various combinations of test scenarios were generated to simulate the FCS system states. The test scenarios were based on the failure rates of the sub-systems and components. The failure rates were randomly generated by in-house software.

The random testing methodology has been a proven approach for generating the test cases to validate the critical functionalities of the system. The control-law code for the indigenously developed Light Combat Aircraft was verified for its correct functionality. Work by Giri *et al.* [98], Jeppu [99] and Rajalakshmi *et al.* [100] provide the reference to use test cases based on random data for verifying the critical functionalities as part of the independent verification and validation technique. We have used this approach to simulate failures of the sub-systems randomly because random failures portray the unpredictable behavior of the system and help in detecting the design flaws. The software program written in Matlab script generates random failure rates for each of the sub-system or component used in the system. These failure rates are generated based on the specification of the sub-system or component. System state is determined based on the failure of the sub-system or component at periodic interval. This interval is decided based on the failure data generated. The system states were observed for the different combinations of failure rates of sub-systems and can be broadly categorized under various scenarios:

3.4.3.1 Scenario A: *Failure at Input Stage* – 1

Random data was generated for simulating the failures at the input stage of the FCS system. The failures at the input stage affect the output and provide the information regarding the point at which the B777 or the A380 FCS' system enters the failed state.

The system state transitions of B777 and A380 are shown in the Figure 3.9.

3. SYSTEM ALGEBRA

Input signals		System Boeing	System Airbus	Output A380	Boeing output
2.13×10^8	2.13×10^8	4.71×10^8	4.71×10^8	4.71×10^8 1.4×10^9	4.71×10^8
1.31×10^9		1.4×10^9	1.396×10^9		
1.17×10^9		5.02×10^8	5.02×10^8		
6.54×10^8		1.42×10^9	1.421×10^9		
5.88×10^8		7.11×10^6			
8.31×10^8		2.22×10^9			
7.63×10^7					
3.08×10^8					
2.59×10^8					
5.53×10^8					
2.30×10^8					
7.72×10^8					
5.93×10^8					
1.69×10^9					
6.49×10^8					
2.97×10^8					
2.86×10^8					
5.51×10^8					
1.46×10^9					
1.60×10^8					
2.42×10^8					
3.79×10^8					
5.91×10^8					
7.70×10^8					
2.06×10^9					
2.30×10^8					
1.09×10^9					
3.281×10^8					

Table 3.3: Failure at Input Stage-1

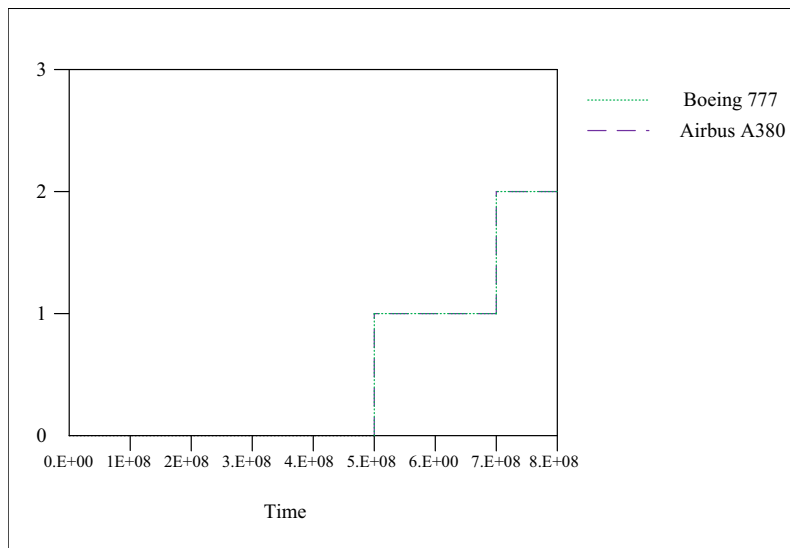


Figure 3.9: Graphical Representation of Scenario A

3.4 Verifying and Validating the Critical System Parameters of Similar Systems using SA

3.4.3.2 Scenario B: Failure at Input Stage – 2

Random data was generated for simulating the failures at the input stage of the FCS system. The failures at the input stage affect the output and provide the information regarding the point at which the Boeing or the Airbus FCS system enters the failed state.

Input signals		System Boeing	System Airbus	Output A380	Boeing output
1.79×10^8		1.22×10^9	1.22×10^9	1.31×10^9	1.31×10^9
5.54×10^8		3.60×10^8	3.60×10^8		2.70×10^8
1.11×10^9		4.73×10^8	4.73×10^8		
6.43×10^8	1.79×10^8	4.49×10^8	4.49×10^8		
1.80×10^8		1.05×10^9			
7.79×10^8		6.68×10^8			
2.04×10^8					
6.86×10^8		1.80×10^8			
1.72×10^8					
2.75×10^8					
1.35×10^8					
1.53×10^7		1.53×10^7			
7.19×10^8		7.19×10^8			
7.19×10^9					
1.56×10^9					
8.87×10^8					
6.24×10^8					
1.75×10^7					
4.68×10^8					
7.65×10^8	1.75×10^7				
5.50×10^8					
1.08×10^7					
2.12×10^8					
5.98×10^8	1.08×10^7				
1.65×10^9					
5.35×10^8					
8.27×10^8					
1.71×10^9	5.35×10^8				

Table 3.4: Failure at Input Stage-2

The system state transitions of B777 and A380 are shown in the Figure 3.10.

3. SYSTEM ALGEBRA

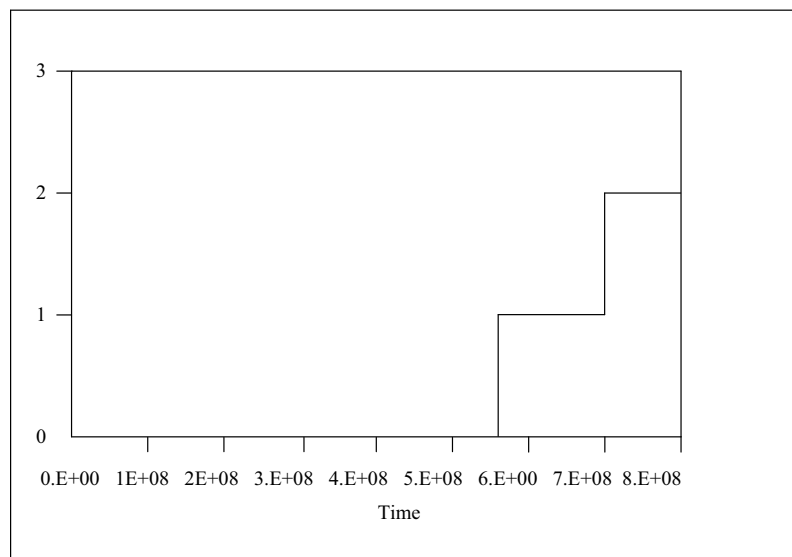


Figure 3.10: Graphical Representation of Scenario B

3.4.3.3 Scenario C: *Failure at Computational Stage*

Random data was generated for simulating the failures at the computational stage of the FCS system. The failures at the computational stage affect the output and provide the information regarding the point at which the Boeing or the Airbus FCS system enters the failed state.

The system state transitions of B777 and A380 are shown in the Figure 3.11.

3.4 Verifying and Validating the Critical System Parameters of Similar Systems using SA

Input signals	System Boeing	System Airbus	Output A380	Boeing output
1.65×10^8	8.89×10^8	8.89×10^8	1.158×10^9	1.16×10^9
8.66×10^8	1.33×10^9	1.331×10^9	3.04×10^8	
Fail	1.02×10^8	1.02×10^8		
Fail	8.36×10^8	8.36×10^8		
1.53×10^9	2.2×10^9			
4.46×10^7	1.17×10^9			
Fail				
Fail				
7.02×10^8				
1.54×10^9				
Fail				
Fail				
6.27×10^8				
4.5×10^8				
Fail				
Fail				
7.87×10^8				
9.02×10^8				
Fail				
Fail				
9.79×10^8				
7.07×10^8				
Fail				
Fail				
1.21×10^9				
9.62×10^8				
Fail				
Fail				

Table 3.5: Failure at Computational Stage

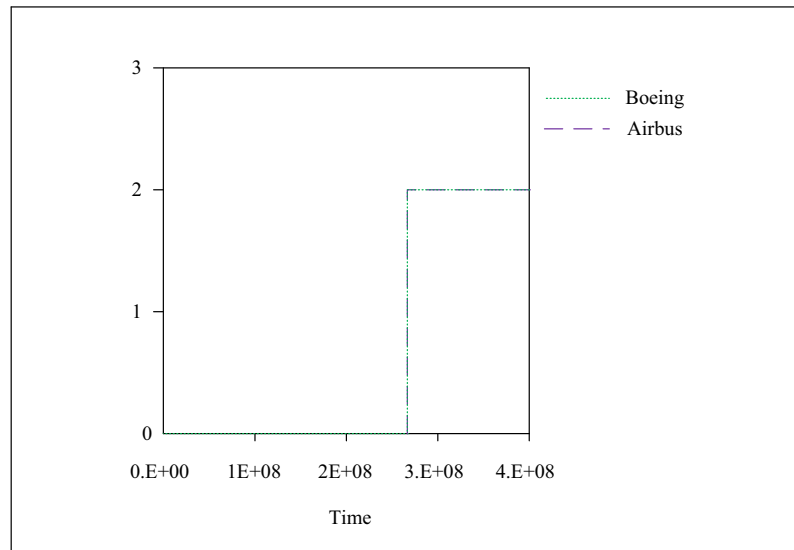


Figure 3.11: Graphical Representation of Scenario C

3. SYSTEM ALGEBRA

3.4.3.4 Scenario D: Failure at Output Stage

Random data was generated for simulating failures at the output stage.

Input signals	System Boeing	System Airbus	Output A380	Boeing output
-3.7×10^8	367356323	367356323		
-6.2×10^8	624020275	624020275	367356323	
-6.2×10^8	600765633	600765633		
38473855	38473855	38473855	38473855	38473855

Table 3.6: Failure at Output Stage

The system state transitions of B777 and A380 are shown in the Figure 3.12.

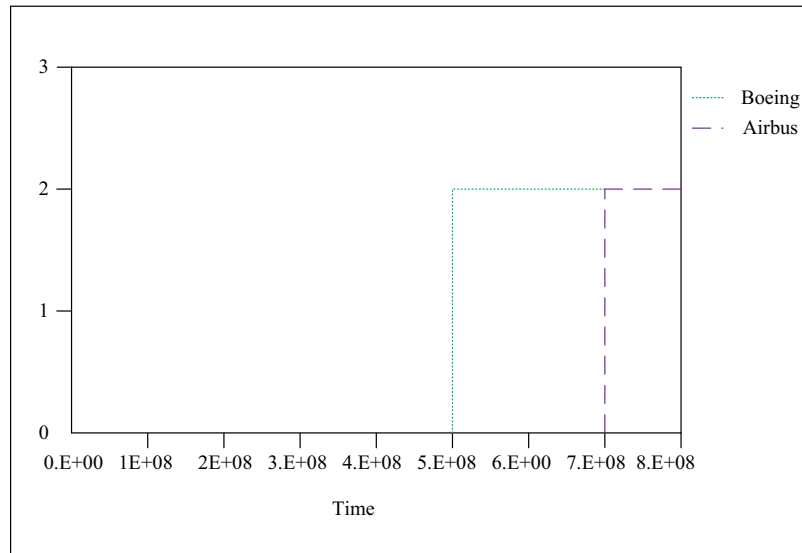


Figure 3.12: Graphical Representation of Scenario D

These simulation scenarios helped in demonstrating the capability of SA to compare critical features of similar systems by means of functional availability analysis.

3.5 System Design Analysis using SA: Forward Engineering

This section discusses experiments to validate the effectiveness of the SA approach in analyzing system design based on the system availability concept. The experiments attempt to develop techniques to analyze a design from the functional availability perspective. The case studies discussed are examples of new technology development and modification of existing technology for in-house projects such as enhanced smart fatigue meter

3.5 System Design Analysis using SA: Forward Engineering

(eFM), and stall warning & aircraft interface computer (SWS/AIC) system. For eFM system design variations using SA approach were adopted for IAF Jaguar (MK-II) aircraft. Two system designs were analyzed against the project requirements for functionality and safety. Further modification to the approved design was analyzed for re-using the eFM for indigenously developed Medium Altitude Long Endurance UAV, RUSTOM-II. For SWSAIC, system upgrade with additional functionalities was required. For this SA approach was used to generate the data flow and control flowchart as part of the impact analysis artifact as required by the Certification agency.

3.5.1 Case Study 1: Functional Availability Analysis for SWS/AIC system design upgrade

SWS/AIC system is an integral part of the CSIR-NAL developed SARAS, a 14-seater, multi-role light transport aircraft. It is the state-of-art system that alerts the pilots of the impending stall whenever the aircraft approaches stall angle of attack; it computes the aircraft warnings like takeoff, landing, overspeed, hydraulic low pressure, and pitch command, monitoring and warning.

The SWS/AIC interfaces with two Angle-of-Attack (AOA) transmitters, two ADCUs (Air Data Computer Unit), two AHRUs (Altitude Heading Reference System), cockpit control panels, pitch trim actuator, shaker actuator, EFIS (Electronic Flight Instrumentation System), EFIS panels, lamp annunciates/indicators, caution/warning system, hydraulic system, fuel system, pitch trim potentiometer and engine system. The interfaces between SWS/AIC and other systems are through ARINC-429, analog, discrete, and RS-422. Each channel of the computer shall receive signals from the dual ADCUs and AHRUs, right and left EFIS and RADALT (Radio altimeter) via ARINC 429, and dedicated SWS analog vane position sensors; discrete inputs from landing gear whether 'up and locked' or 'down and locked', PTT (Press To Test) switches; and the shaker on/off switches on the control wheel, pilot inputs for pitch trim from control wheel; hydraulic system, fuel system, pitch trim position potentiometer, engine torque from the right and left; these inputs are used to perform monitoring and warning annunciation. The ARINC-429 data is validated through sign status matrix bits. The block diagram of the SWS/AIC system is shown in Figure 3.13.

3. SYSTEM ALGEBRA

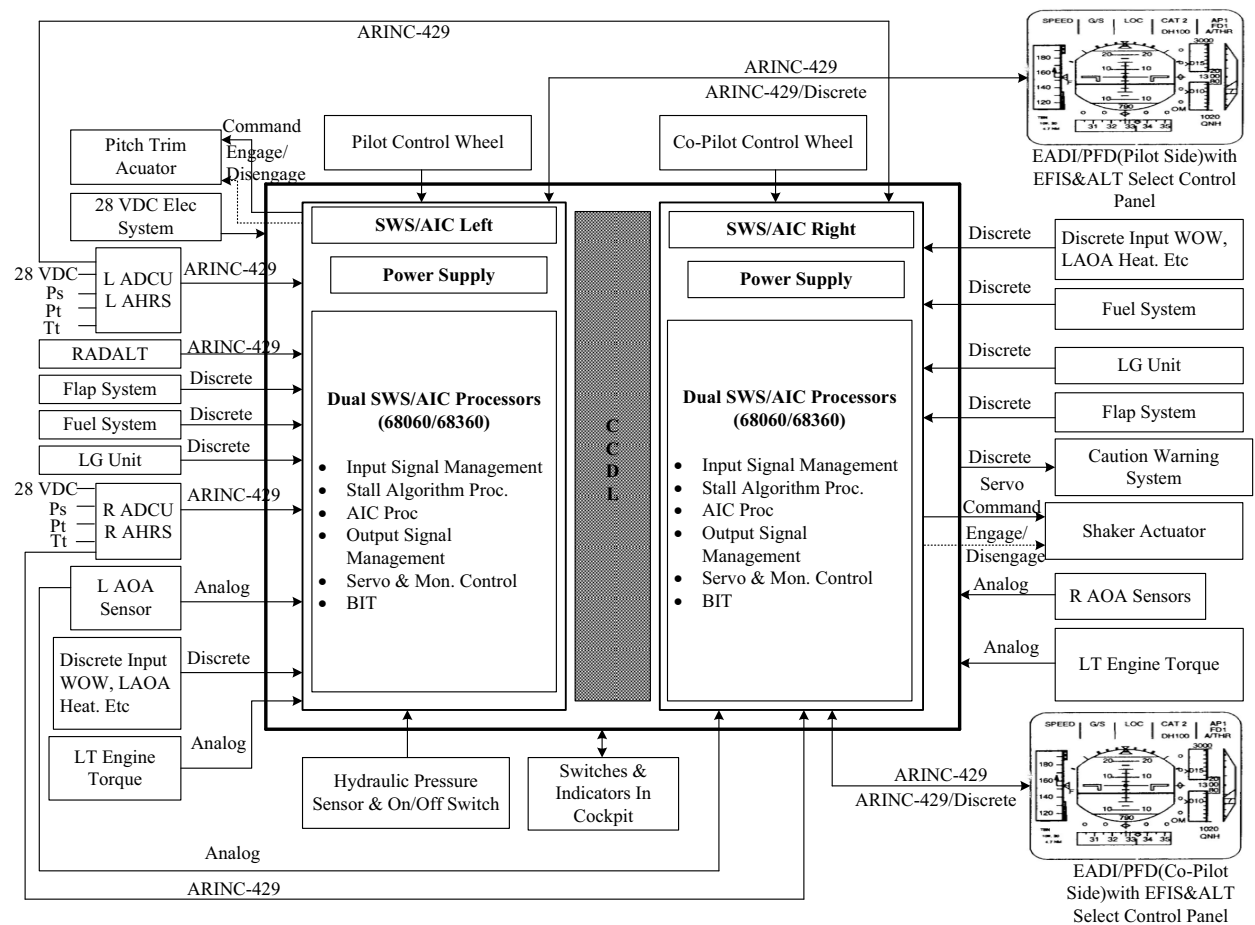


Figure 3.13: Block Diagram of SWS/AIC

3.5 System Design Analysis using SA: Forward Engineering

The SA expression derived for the system is computed based on SA properties. The algebraic expression for the SWS/AIC system is shown in equation 3.5.1.

$$\begin{aligned}
 S = & [(IA_1^2) + (IA_2^2) + (IA_3^2) + (IA_4) + (ID_1^2) + (ID_2^2) \\
 & + (ID_3^2) + (ID_4^2) + (ID_5^2) + (ID_6) + (ID_7^2) \\
 & + (ID_8^2) + (ID_9^2) + (IAR_1^2) + (IAR_2^2) \\
 & + (IAR_3) + (IAR_4) + CC + MC + OSHR + OD + OPTTRM + OAR + (PS^2)]
 \end{aligned}
 \tag{3.5.1}$$

where,

- PS: Power supply sub-system
- IA: Analog input sub-system
- IAR: ARINC input sub-system
- ID: Discrete input sub-system
- CC: Computational computer sub-system
- MC: Monitor computer sub-system
- OSHR: Shaker output
- OD: Discrete output sub-system
- OPTTRM: Pitch Trim output
- OAR: ARINC output

The failure rate for the sub-systems and components is shown in Table 3.7

Functional availability analysis is performed based on the operational relationship between sub-systems. This relationship will determine the system state and is shown in Table 3.8.

Component or subsystem	Failure Rate
Power supply	3×10^{-4}
Analog input	3×10^{-4}
ARINC input	3×10^{-4}
Discrete input	1×10^{-4}
Computation computer	3×10^{-5}
Monitor computer	3×10^{-5}
Shaker output	1×10^{-4}
Pitch trim output	1×10^{-4}
Discrete output	1×10^{-4}
ARINC output	1×10^{-4}

Table 3.7: Failure Rates of SWS/AIC Sub-systems and Components

3. SYSTEM ALGEBRA

Sub-system failure	State	Functionality
No failure	Safe	Stall computation and warning, Pitch monitor and AIC warnings
ARINC sub-system	Hazardous	Pitch monitor and AIC warnings
Analog	Hazardous	Pitch monitor and AIC warnings
Discrete	Hazardous	Stall computation and warning AIC Warnings
Computation computer	Unsafe	No functionality
Monitor computer	Unsafe	No functionality
PS	Unsafe	No functionality
Analog and Discrete	Hazardous	AIC Warnings
Analog and ARINC	Hazardous	AIC Warnings
Discrete and ARINC	Unsafe	No functionality
Analog and Discrete and ARINC	Unsafe	No functionality
Analog and Computation computer	Unsafe	No functionality
Discrete and Computation computer	Unsafe	No functionality
ARINC and Computation computer	Unsafe	No functionality
Analog and Monitor computer	Unsafe	No functionality
Discrete and Monitor computer	Unsafe	No functionality
ARINC and Monitor computer	Unsafe	No functionality

Table 3.8: Functional Availability Analysis of SWS/AIC

Based on the failure rates, we see that the system transits to the hazardous state after 10^6 hours and transits to unsafe state after 10^9 hours. With this functional availability, the SWS/AIC system gradually transits to a degraded performance without affecting the safety of the interfacing systems. These analysis were discussed with the test crew for clearance of the system availability, course of action by the pilot when the system is not available.

To improve design for longer availability, fault-tolerance of the sub-systems needs to be analyzed. The best and the worst fault-tolerance of the SWS/AIC system based on the SA definition for system fault-tolerances are shown in equations 3.5.2 and 3.5.3. The best-case and worst-case equations based on the SA approach help in understanding the systems' fault tolerance. The best-case fault tolerance provides information about the sub-system/ sub-systems of the system design whose failure does not cause the system to fail. This analysis helps in determining the system robustness. The worst-case fault tolerance provides information about the sub-system/ sub-systems of the system design whose failures causes system to fail. This analysis helps in determining the weakest link of the design which can be worked on. Failure rates and redundancy of the components affect the availability of the sub-system which in turn affects the functional availability of the system. SA based approach validates the system design based on the functional availability. The system consists of hardware and software which provides the functionality of

3.5 System Design Analysis using SA: Forward Engineering

the system and the availability under different failure scenarios

$$\begin{aligned}
 \beta_{best} \left(\sum_{i=1}^{10} S_i \right) &= \left(\sum_{i=1}^{10} \beta_{best}(S_i) \right) \\
 &= \beta_{best}(\text{IA} + \text{ID} + \text{IAR} + \text{CC} + \text{MC} + \text{OSHR} + \text{OD} + \text{OPTTRM} + \text{OAR} + \text{PS}) \\
 &= \beta_{best}(\text{IA}) + \beta_{best}(\text{ID}) + \beta_{best}(\text{IAR}) + \beta_{best}(\text{CC}) + \beta_{best}(\text{MC}) \\
 &\quad + \beta_{best}(\text{OSHR}) + \beta_{best}(\text{OD}) + \beta_{best}(\text{OPTTRM}) + \beta_{best}(\text{OAR}) + \beta_{best}(\text{PS})
 \end{aligned} \tag{3.5.2}$$

$$\begin{aligned}
 \beta_{worst} \left(\sum_{i=1}^{10} S_i \right) &= \left(\sum_{i=1}^{10} \beta_{worst}(S_i) \right) \\
 &= \beta_{worst}(\text{IA} + \text{ID} + \text{IAR} + \text{CC} + \text{MC} + \text{OSHR} + \text{OD} + \text{OPTTRM} + \text{PS} + \text{OAR}) \\
 &= \min(\beta_{worst}(\text{IA}), \beta_{worst}(\text{ID}), \beta_{worst}(\text{IAR}), \beta_{worst}(\text{CC}), \beta_{worst}(\text{MC}) \\
 &\quad , \beta_{worst}(\text{OSHR}), \beta_{worst}(\text{OD}), \beta_{worst}(\text{OPTTRM}), \beta_{worst}(\text{PS}), \beta_{worst}(\text{OAR}))
 \end{aligned} \tag{3.5.3}$$

The redundancy and failure rates of these components/ sub-systems determine the system state and available functionality. Modification of the design will modify the fault-tolerance and functional availability.

3.5.2 Case Study 2: Functional Availability Analysis for eFM system design

Enhanced fatigue meter(eFM), designed and developed by CSIR-NAL, provides a continuous monitoring of g crossings for fighter aircrafts. Continuous ‘g’ monitoring is critical to determine aircraft fatigue. eFM also keeps a record of g crossings to compute the fatigue index of the aircraft and helps in enhancing the maintenance of the aircraft. The eFM system consists of a processor module, 16-key keyboard with the processing module, seven-segment LED display, power supply, memory with data logging unit, real-time clock, accelerometer and signal conditioning module. The eFM system acquires, and processes the signal from the accelerometer, and computes the ‘g’ crossings and the fatigue index for the aircraft. The block diagram for eFM is shown in Figure 3.14.

3. SYSTEM ALGEBRA

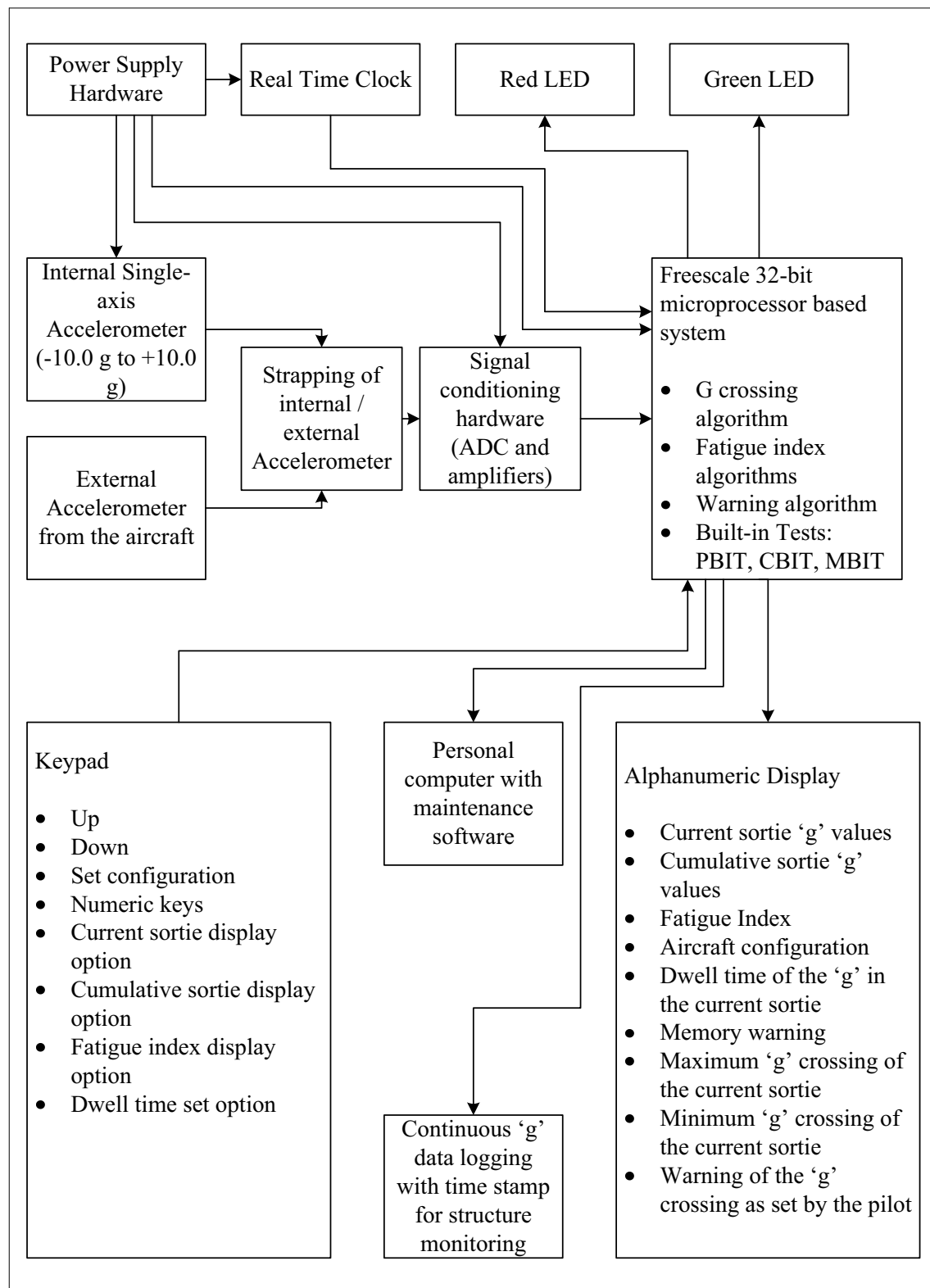


Figure 3.14: Block Diagram of Enhanced Fatigue Meter

3.5 System Design Analysis using SA: Forward Engineering

The SA expression derived for the system is computed based on SA properties. The SA expression for the eFM is shown in equation 3.5.4.

$$S = (\text{PS} + \text{ACCN} + \text{SIGCOND} + \text{CPU} + \text{KBD} + \text{DISP} + \text{RTC}) \quad (3.5.4)$$

where,

PS:	Power supply sub-system
ACCN:	Accelerometer assembly sub-system
SIGCOND:	Signal conditioning sub-system
CPU:	Processing sub-system
KBD:	Keyboard sub-system
DISP:	Display sub-system
RTC:	Real time sub-system

The worst- and the best-case fault tolerance of the system compares the systems design to the fault tolerance requirements. The best and the worst fault tolerances of the eFM system based on the SA definition is shown in equations 3.5.5 and 3.5.6.

$$\begin{aligned} \beta_{best} \left(\sum_{i=1}^7 S_i \right) &= \left(\sum_{i=1}^7 \beta_{best}(S_i) \right) \\ &= \beta_{best}(\text{PS} + \text{ACCN} + \text{SIGCOND} + \text{CPU} + \text{KBD} + \text{DISP} + \text{RTC}) \\ &= \beta_{best}(\text{PS}) + \beta_{best}(\text{ACCN}) + \beta_{best}(\text{SIGCOND}) + \beta_{best}(\text{CPU}) + \beta_{best}(\text{KBD}) \\ &\quad + \beta_{best}(\text{DISP}) + \beta_{best}(\text{RTC}) \end{aligned} \quad (3.5.5)$$

$$\begin{aligned} \beta_{worst} \left(\sum_{i=1}^7 S_i \right) &= \min(\beta_{worst}(S_i)) \\ &= \beta_{worst}(\text{PS} + \text{ACCN} + \text{SIGCOND} + \text{CPU} + \text{KBD} + \text{DISP} + \text{RTC}) \\ &= \min(\beta_{worst}(\text{PS}), \beta_{worst}(\text{ACCN}), \beta_{worst}(\text{SIGCOND}), \beta_{worst}(\text{CPU}), \\ &\quad \beta_{worst}(\text{KBD}), \beta_{worst}(\text{DISP}), \beta_{worst}(\text{RTC})) \end{aligned} \quad (3.5.6)$$

The feedback on fault-tolerance and functional availability helps engineers improvise the design for better system fault tolerance and functional availability. One way of improving fault tolerance is to add redundancy in the critical failure path of the system.

3. SYSTEM ALGEBRA

This approach may increase the system complexity, but ensures functional availability of the system for a longer time. All these options have to be justified by designers before the final design is frozen. This analysis proves that the SA approach aids designers to understand the system better and improve system design.

The outcome of this analysis was to retain the current design as the system is not an online system used for on-line monitoring, hence a less complex design was preferred.

Visualization of SSM for the eFM

The possible transitions by the system, from safe to failed state, are shown in Table 3.9. These transitions are determined from the tree diagram generated for the system from its SA expression. The tree diagram documents the paths traversed, as shown in the Figure 3.15.

<i>Safestate</i> →	1 → 4 → 6 → 7 →	<i>Failedstate</i>
<i>Safestate</i> →	1 → 4 → 6 → 8 →	<i>Failedstate</i>
<i>Safestate</i> →	1 → 4 → 6 → 9 →	<i>Failedstate</i>

Table 3.9: System State Transitions for eFM

3.5 System Design Analysis using SA: Forward Engineering

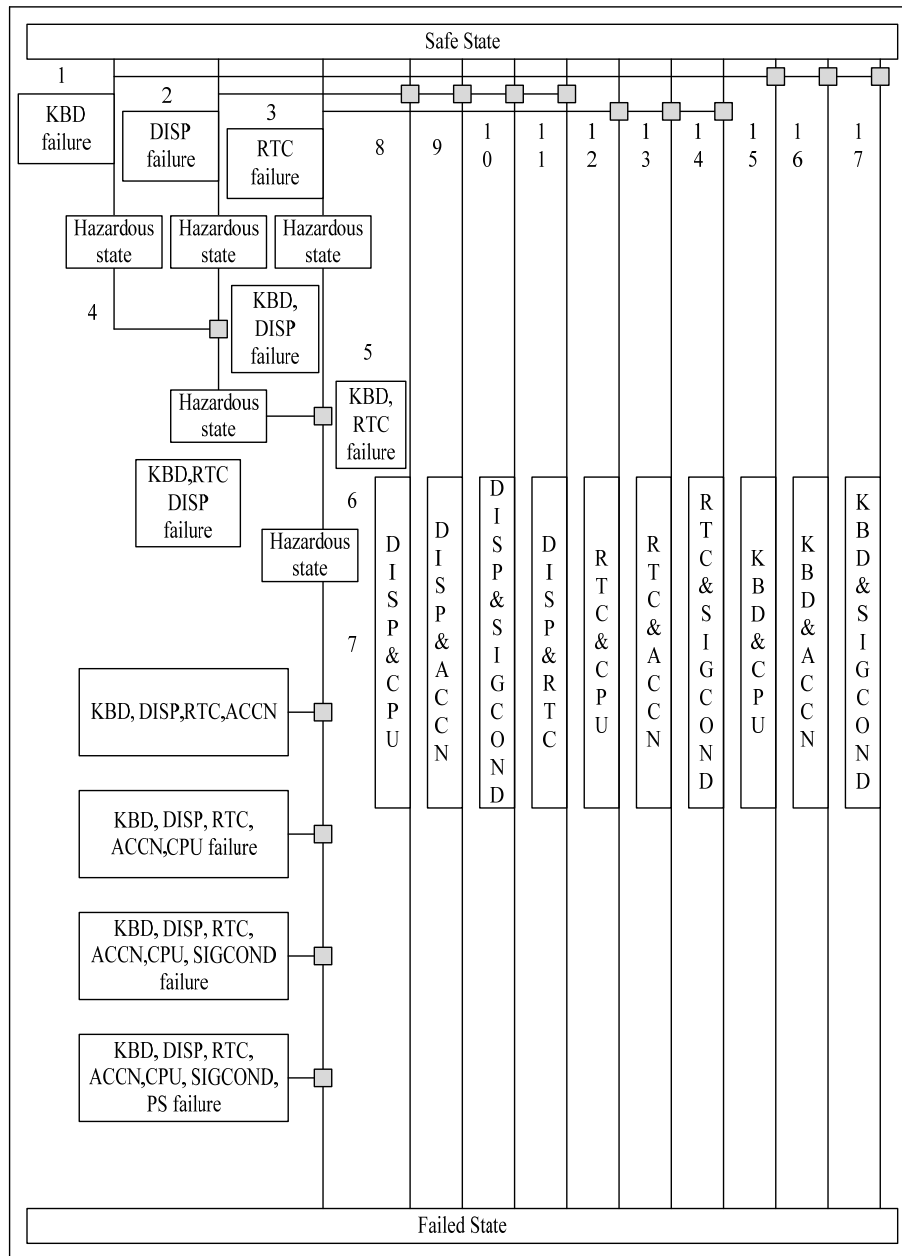


Figure 3.15: Tree Diagram of eFM

3.6 Analyzing other existing Safety System Designs using SA: Reverse Engineering

The simulation results with FCS for Jaguar, Boeing 777, and Airbus A380 proved the validity of SA for analyzing the system functional availability given the failure rates, system states, and the algebraic expression. This technique is next applied to other proven safety-critical systems, such as automobile and medical systems to analyze their functional availability property. For these case studies, the SA expression is derived from their system design, safety is analyzed by the best- and worst-case scenario and their functional availability is calculated using the algebraic expression.

These analyses are performed on the already proven system using reverse-engineering. Reverse-engineering is used to understand the system requirements. With reverse engineering, better performance for the same requirements can be achieved. With this approach, the ambiguity of design does not exist, hence we can verify the applicability of the SA technique to analyze safety-critical systems. These case-studies demonstrate the analysis capability of SA approach in the design phase. The early detection of design flaws not only improves the engineering process, but also ensures timely delivery and improves safety of the system.

3.6.1 Steer-by-wire Automobile: Automobile Domain

Steer-by-wire automobile system design was analyzed by Pimentel [101] and an architecture for the system was proposed. The proposed architecture supports three major failure modes and features several safety protocols for the steer-by-wire system. The block diagram of the system is shown in the Figure 3.16.

3.6 Analyzing other existing Safety System Designs using SA: Reverse Engineering

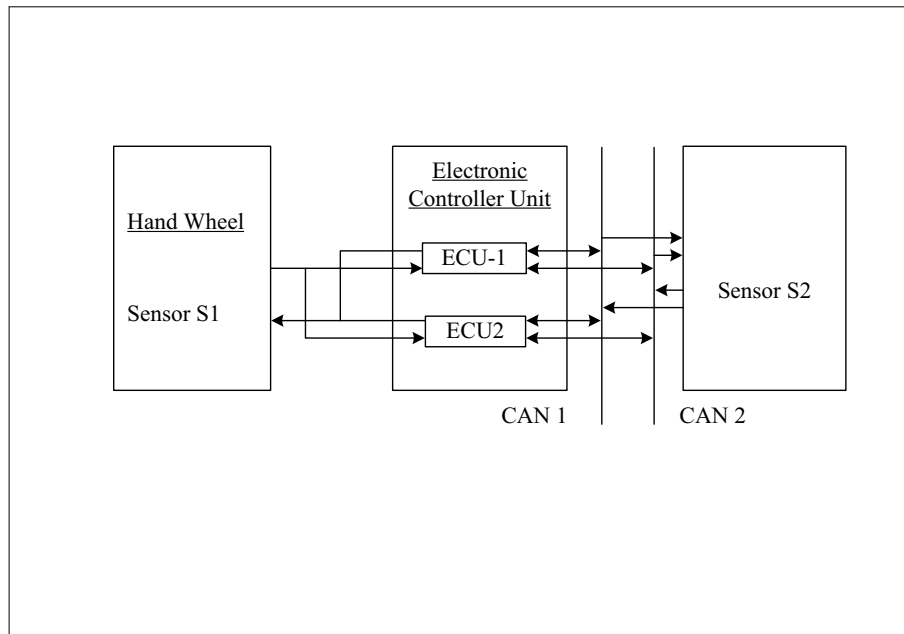


Figure 3.16: Block Diagram of Steer-by-wire Automobile

The system design duplicates sensors and actuators at hand wheel and road wheel to ensure sustained functional availability and safety of the system. This duplication forms the fault-tolerance feature of the system. There are duplicate electronic control units (ECU1 and ECU2), micro-controllers (MC3 and MC4) for the sensors and actuators, and controller area network buses (CAN1 and CAN2). This replication improves fault-tolerance as the system does not have a single point of failure. Software plays a fundamental role to implement schemes to deal with other recoverability, fault-tolerance, and fail-safe requirements. In addition to application functionality, communication and physical interface are also considered in software design. There are several options for safety-critical hardware architectures such as networked or direct IO, micro-controller-based or dedicated CAN interfaces for sensors and actuators, and degree of replicated components. The SA expression for the system after the decomposition into series and parallel components is shown in the equation 3.6.1.

$$\begin{aligned}
 S &= (\text{HW} + \text{ECU} + \text{CAN}^2 + \text{MC}) \\
 &= (S_1 + A_1 + \text{ECU}_1 + \text{ECU}_2 + \text{CAN}^2 + S_2 + A_2)
 \end{aligned}
 \tag{3.6.1}$$

where,

HW: Hand wheel sub-system

3. SYSTEM ALGEBRA

ECU ₁ and ECU ₂ :	Electronic control unit
CAN:	Controller area network
MC:	Micro-controller road wheel sub-system
S ₂ and A ₂ :	Sensors and actuators

The worst- and the best-case fault-tolerances of the system, predicted by Rao [29], are shown in equations 3.6.2 and 3.6.3

$$\begin{aligned}
 \beta_{best} \left(\sum_{i=1}^4 S_i \right) &= \sum_{i=1}^4 \beta_{best}(S_i) \\
 &= \beta_{best}((HW + S_1 + A_1) + ECU_1 + ECU_2 + CAN_2 + MC + S_2 + A_2) \\
 &= \beta_{best}(HW) + \beta_{best}(S_1) + \beta_{best}(A_1) + \beta_{best}(ECU_1) + \beta_{best}(ECU_2) \\
 &\quad + \beta_{best}(CAN_2) + \beta_{best}(MC) + \beta_{best}(S_2) + \beta_{best}(A_2)
 \end{aligned} \tag{3.6.2}$$

$$\begin{aligned}
 \beta_{worst} \left(\sum_{i=1}^4 S_i \right) &= \min(\beta_{worst}(S_i)) \\
 &= \beta_{worst}((HW + S_1 + A_1) + ECU_1 + ECU_2 + CAN_2 + MC + S_2 + A_2) \\
 &= \beta_{worst}(HW), \beta_{worst}(S_1), \beta_{worst}(A_1), \beta_{worst}(ECU_1), \beta_{worst}(ECU_2), \\
 &\quad \beta_{worst}(CAN_2), \beta_{worst}(MC), \beta_{worst}(S_2) + \beta_{worst}(A_2)
 \end{aligned} \tag{3.6.3}$$

As seen, the computational stage has a redundancy of 2 (ECU and CAN) and meets system functional availability requirements. The functional availability expression for the system is provided in the expression given in equation 3.6.4.

$$\text{Availability} = (HW, ECU, CAN, A, MC)_{Min} \tag{3.6.4}$$

The design of the steer-by-wire system ensures smooth state transition, from safe to hazardous to unsafe state based on component/ sub-system failure rates. The SA expression shows that the system is simple, and loosely coupled, and hence ensures the availability for a longer time despite high failure rate at the component level. The availability of a system with expected functionality is based on the minimum availability of sub-systems in the sequential chain. Therefore, to improve the availability of the system, the tolerance of the critical sub-systems: *HW*, *ECU*, *CAN*, *A* and *MC* can be improved.

3.6 Analyzing other existing Safety System Designs using SA: Reverse Engineering

3.6.2 Railway Intelligent Transportation System Architecture (RITS): Transport Domain

The Chinese railway authority is upgrading the railway transport system with advanced technologies to provide the railway system with the capability of higher speed, capacity, degree of safety and quality of services. The RITS architecture, discussed by Li *et.al* [102], is capable of achieving it. The block diagram of the RITS is shown in the Figure 3.17.

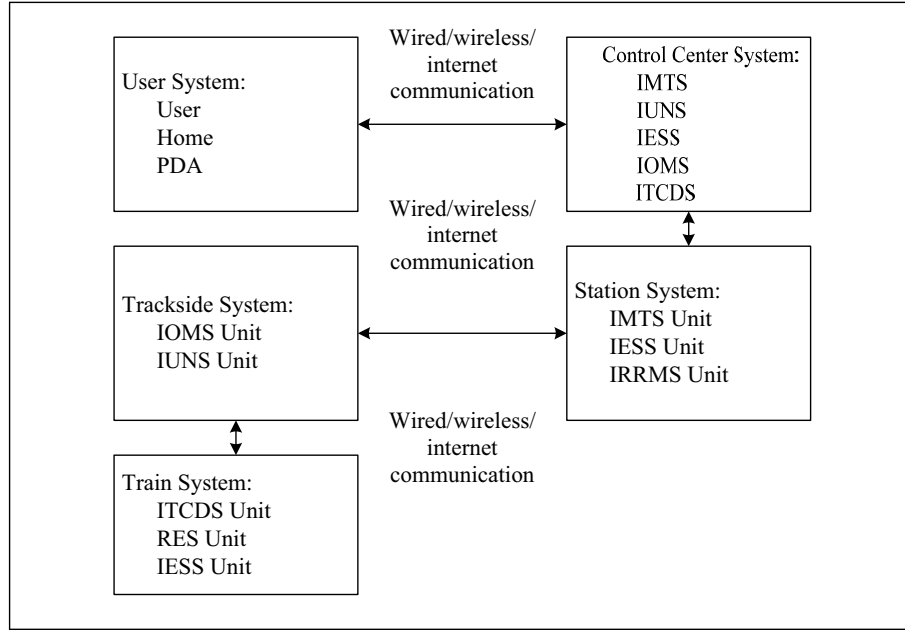


Figure 3.17: Block Diagram of RITS

The SA expression of the system is shown in equation 3.6.5.

$$\begin{aligned}
 S &= [US + CCS + SS + TrkS + TS] \\
 &= [((User) * (Home) * (PDA)) + ((IMTS) * (IUNS)) * ((IESS) * (IOMS) * (ITCDS)) \\
 &\quad * ((IRRMS) * (RES)) + ((IMTS) * (IESS) * (IRRMS) * (RES)) \\
 &\quad + ((IOMS) * (IUNS) * (IRRMS)) \\
 &\quad + ((ITCDS) * (RES) * (IESS) + ((IRRMS) * (IUNS)))]
 \end{aligned}
 \tag{3.6.5}$$

where,

- IMTS: Inter-Model Transportation System
- IUNS: Intelligent User Navigation System

3. SYSTEM ALGEBRA

IESS: Intelligent Emergency Rescue and Safety Supervision
IOMS: Intelligent railway Operation Management System
ITCDS: Intelligent Train Control and Dispatching System
IRRMS: Intelligent Railway Resource Management System
RES: Railway E-business System

The best and worst fault-tolerances of the system design computed by SA are given in equations 3.6.6 and 3.6.7.

$$\begin{aligned}\beta_{best}\left(\sum_{i=1}^5 S_i\right) &= \sum_{i=1}^5 \beta_{best}(S_i) \\ &= \beta_{best}(\text{US} + \text{CCS} + \text{SS} + \text{TrkS} + \text{TS}) \\ &= \beta_{best}(\text{US}) + \beta_{best}(\text{CCS}) + \beta_{best}(\text{SS}) + \beta_{best}(\text{TrkS}) + \beta_{best}(\text{TS})\end{aligned}\tag{3.6.6}$$

$$\begin{aligned}\beta_{worst}\left(\sum_{i=1}^5 S_i\right) &= \sum_{i=1}^5 \beta_{worst}(S_i) \\ &= \beta_{worst}(\text{US} + \text{CCS} + \text{SS} + \text{TrkS} + \text{TS}) \\ &= \beta_{worst}(\text{US}, \text{CCS}, \text{SS}, \text{TrkS}, \text{TS})_{Min}\end{aligned}\tag{3.6.7}$$

The functional availability expression for the system is shown in equation 3.6.8.

$$\text{Availability} = (\text{US}, \text{CCS}, \text{SS}, \text{TrkS}, \text{TS})_{Min}\tag{3.6.8}$$

The proposed architecture is sequential with no redundancy; hence the functional availability is the minimum availability of either of the sub-systems.

3.6.3 A Distributed Fault-Tolerant Architecture for Nuclear Reactor Control and Safety Functions: Nuclear Domain

The distributed fault tolerant system is discussed by Hecht *et al.* [103] and the block diagram is shown in the Figure 3.18. It is a simple example, but this analysis indicates the nuclear reactor indicates the efficacy of the SA technique in determining the functional availability of a system during design. Earlier also, simple examples have been used to demonstrate analytical capability. For instance, Leveson and Stolzy [104] analyze system safety using Petri Nets, giving the example of a simple railroad crossing. The railroad crossing system consists of the train, controlling device and crossing gate, and

3.6 Analyzing other existing Safety System Designs using SA: Reverse Engineering

is a manifestation of a well-known system of everyday experience, but the analyses help understand finer aspects that are not very clear otherwise. Likewise, here too we consider a relatively unsophisticated but standard system and show how it may be analyzed using our approach.

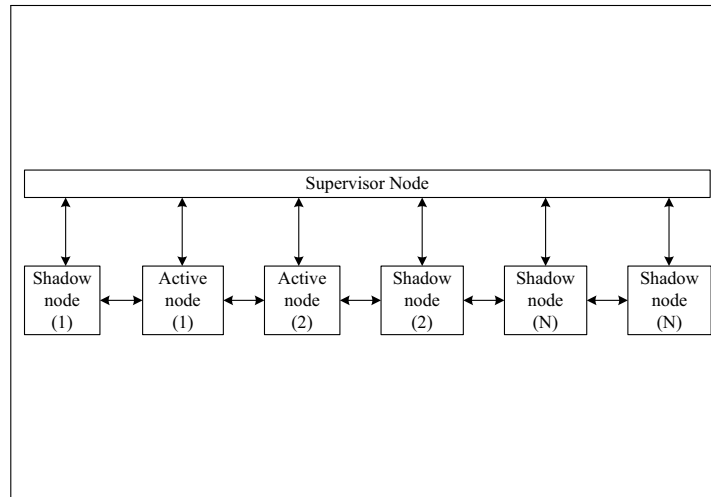


Figure 3.18: Block Diagram of Distributed Fault-tolerant Architecture for Nuclear Reactor Control and Safety Functions

The SA expression is shown in equation 3.6.9.

$$S = (\text{SupNd} + (\text{ShwdNd} + \text{ActNd})^N) \quad (3.6.9)$$

where,

- SupNd: Supervisor Node
- ShwdNd: Shadow Node
- ActNd: Active Node

As seen in the expression, the supervisor node is the point of failure as other nodes have redundancies. If the availability of the supervisor node is high, then the system is available for a longer period of time.

The best and worst fault tolerances of the system design are given in equations 3.6.10 and 3.6.11.

$$\begin{aligned}
 \beta_{best} \left(\sum_{i=1}^5 S_i \right) &= \sum_{i=1}^5 \beta_{best}(S_i) \\
 &= \beta_{best} (\text{SupNd} + (\text{ShwdNd} + \text{ActNd})^N) \\
 &= \beta_{best}[\text{SupNd}] + \beta_{best}[(\text{ShwdNd} + \text{ActNd})^N]
 \end{aligned} \quad (3.6.10)$$

3. SYSTEM ALGEBRA

$$\begin{aligned}\beta_{worst}\left(\sum_{i=1}^5 S_i\right) &= \sum_{i=1}^5 \beta_{worst}(S_i) \\ &= \beta_{worst}[\text{SupNd} + (\text{ShwdNd} + \text{ActNd})^N] \\ &= \beta_{worst}[\text{SupNd}, \text{ShwdNd}, \text{ActNd}]_{Min}\end{aligned}\tag{3.6.11}$$

The functional availability expression for the system is provided in the expression given in equation 3.6.12.

$$\text{Availability} = (\text{SupNd})_{Min}\tag{3.6.12}$$

The availability expression shows the supervisor node availability as the critical point of the system. The failure of the supervisor node makes the system transit from a safe state to an unsafe state. Hence, if there is a need to make the system available for a longer time, the fault-tolerance of the supervisor node needs to be increased. This can be done by increasing the redundancy or minimizing the failure rate of the supervisor sub-system. These decisions need to be taken by the designer to improve the functional availability of the system.

3.6.4 A Fault-Tolerant Architecture for Computer-based Railway Vehicle Brake Systems: Railway Domain

Johansson [105] discusses a fault-tolerant architecture for railway vehicle brake system; the block diagram is shown in the Figure 3.19.

3.6 Analyzing other existing Safety System Designs using SA: Reverse Engineering

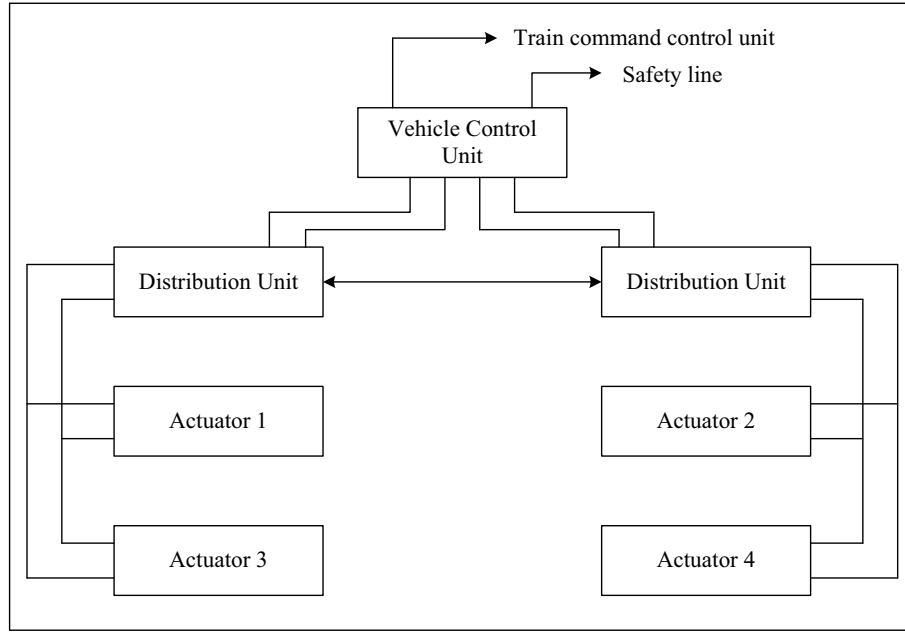


Figure 3.19: Block Diagram of Fault-tolerant Architecture for Computer-based Railway Vehicle Brake Systems

The SA expression for this system is shown in equation 3.6.13.

$$\begin{aligned}
 S &= \text{VCU} + \text{DU}^2 \\
 &= \text{VCU} + \text{A}_1^2 + \text{A}_2^2
 \end{aligned}
 \tag{3.6.13}$$

where,

VCU: Vehicle Control Unit

DU: Distribution Unit

A₁ and A₂: Actuators

The best and the worst fault-tolerances of the system are given in equations 3.6.14 and 3.6.15.

$$\begin{aligned}
 \beta_{best} \left(\sum_{i=1}^3 S_i \right) &= \sum_{i=1}^3 \beta_{best}(S_i) \\
 &= \beta_{best}(\text{VCU} + \text{A}_1 + \text{A}_2^2) \\
 &= \beta_{best}(\text{VCU}, \text{A}_1, \text{A}_2)_{Min}
 \end{aligned}
 \tag{3.6.14}$$

3. SYSTEM ALGEBRA

$$\begin{aligned}
 \beta_{worst} \left(\sum_{i=1}^3 S_i \right) &= \sum_{i=1}^3 \beta_{worst}(S_i) \\
 &= \beta_{worst}(\text{VCU} + A_1 + A_2^2) \\
 &= \beta_{worst}(\text{VCU}, A_1, A_2)_{Min}
 \end{aligned}
 \tag{3.6.15}$$

The functional availability expression for the system is given in equation 3.6.16.

$$\text{Availability} = (\text{VCU})_{Min}
 \tag{3.6.16}$$

The availability expression shows the vehicle control unit as the critical point for system availability. Its failure causes a transition from a safe state to an unsafe state.

3.6.5 Distributed Architecture Block Diagram: Railway Domain

Ghosh *et al.* [106] describes the distributed architecture for advanced train control systems. The block diagram of the system's distributed architecture is shown in Figure 3.20.

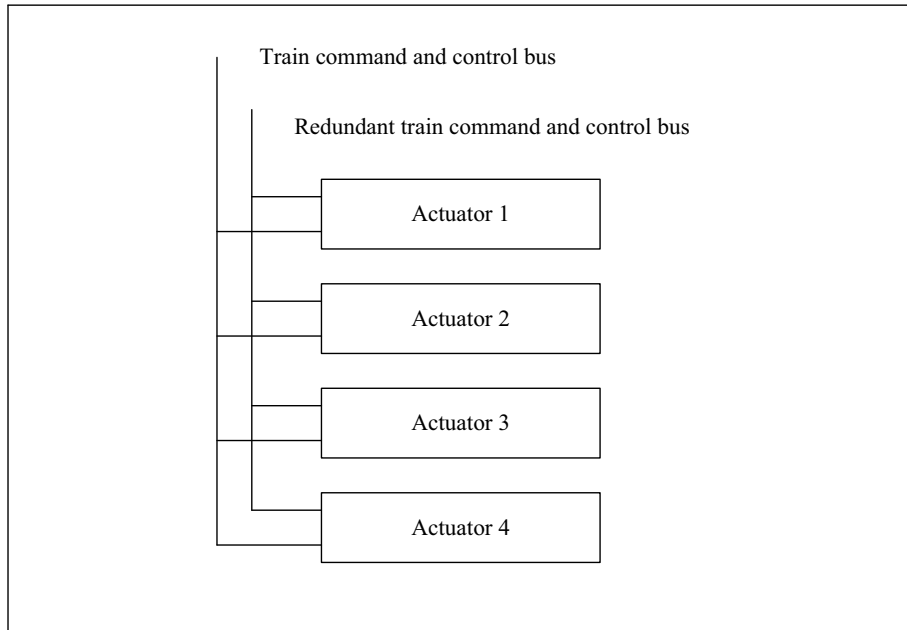


Figure 3.20: Block Diagram of Distributed Architecture

The SA expression for the system is given in equation 3.6.17.

$$S = A^4
 \tag{3.6.17}$$

3.6 Analyzing other existing Safety System Designs using SA: Reverse Engineering

where, A: Actuators

The best and the worst fault tolerances of the system are given in equations 3.6.18 and 3.6.19.

$$\begin{aligned}\beta_{best}\left(\sum_{i=1}^1 S_i\right) &= \sum_{i=1}^1 \beta_{best}(S_i) \\ &= \beta_{best}\left((A)^4\right)\end{aligned}\tag{3.6.18}$$

$$\begin{aligned}\beta_{worst}\left(\sum_{i=1}^1 S_i\right) &= \sum_{i=1}^1 \beta_{worst}(S_i) \\ &= \beta_{worst}\left(A^4\right) \\ &= \beta_{worst}[A]_{Min}\end{aligned}\tag{3.6.19}$$

The functional availability for the system is expressed in equation 3.6.20.

$$Availability = (A)_{Min}\tag{3.6.20}$$

The nuclear reactor has a quadruple redundancy in a distributed architecture. The system is available for longer and a transition from the safe to unsafe state is possible only when all the four system fails. This availability is determined based on the failure rates of the system.

3.6.6 Typical Transport Power Source: Aerospace Domain

Feiner *et al.* [107] describes the power systems of power-by-wire aircraft. The block diagram of the system is shown in the Figure 3.21.

The SA expression for the system is shown in the equation 3.6.21.

$$\begin{aligned}S &= ECU \\ &= \left(PS^2 + ES + FCS + EIS\right)\end{aligned}\tag{3.6.21}$$

where,

- PS: Power Supply
- ES: Environmental System
- FCS: Flight Control Systems
- EIS: Electrical System

3. SYSTEM ALGEBRA

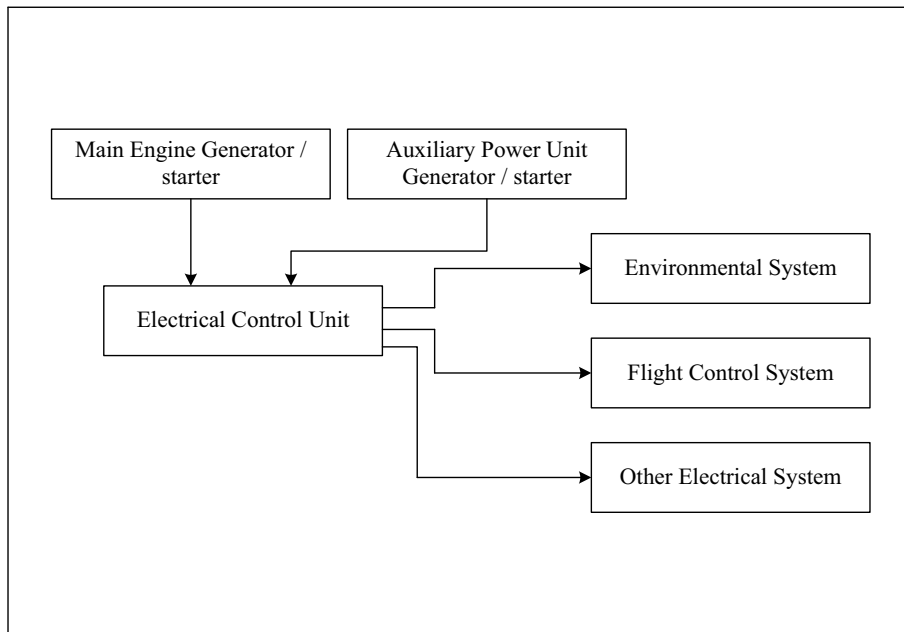


Figure 3.21: Block Diagram of Typical Power Source

The best and the worst fault tolerances of the system are given in the equations 3.6.22 and 3.6.23.

$$\begin{aligned}
 \beta_{best} \left(\sum_{i=1}^4 S_i \right) &= \sum_{i=1}^4 \beta_{best}(S_i) \\
 &= \beta_{best}(\text{ECU}) \\
 &= \beta_{best}(\text{PS}^2 + \text{ES} + \text{FCS} + \text{EIS})
 \end{aligned} \tag{3.6.22}$$

$$\begin{aligned}
 \beta_{worst} \left(\sum_{i=1}^4 S_i \right) &= \sum_{i=1}^4 \beta_{worst}(S_i) \\
 &= \beta_{worst}(\text{ECU}) \\
 &= \beta_{worst}(\text{PS}, \text{ES}, \text{FCS}, \text{EIS})_{Min}
 \end{aligned} \tag{3.6.23}$$

The functional availability expression for the system is shown in equation 3.6.24.

$$\text{Availability} = (\text{ES}, \text{FCS}, \text{EIS})_{Min} \tag{3.6.24}$$

The transition of the system from the safe to unsafe state is dependent on failures of the environmental system, flight control system and electrical system. If any of them fails, the system transits to the unsafe state.

3.6 Analyzing other existing Safety System Designs using SA: Reverse Engineering

3.6.7 Integrated Elevator Electrical Power and Control using the Power-by-wire (PBW): Aerospace Domain

Feiner [108] discusses the system and the block diagram of the system is shown in Figure 3.22.

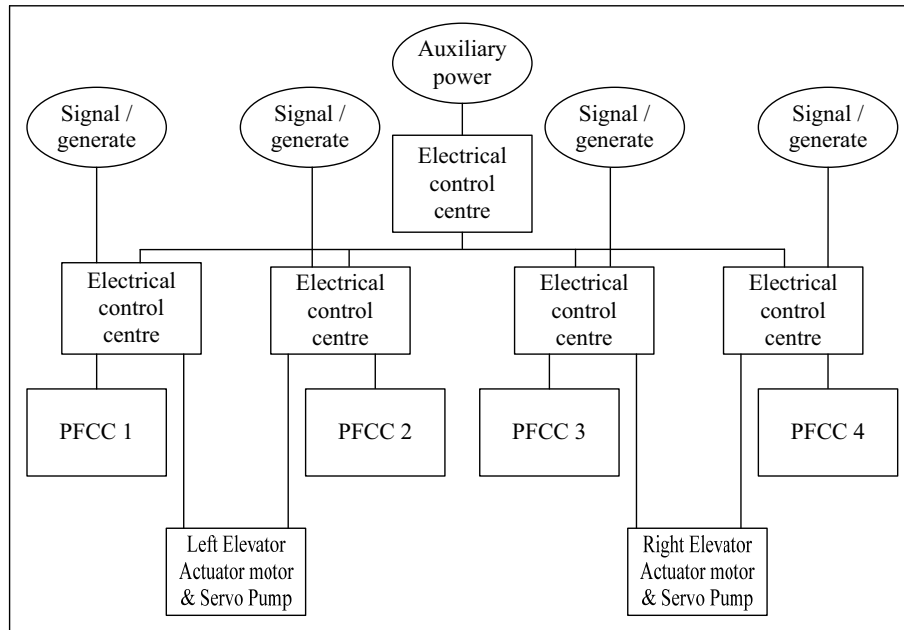


Figure 3.22: Block Diagram of Integrated Elevator Electrical Power and Control

The SA expression for the system is shown in equation 3.6.25.

$$S = ((IPC_1 + IPC_2)^4 + SIG^4 + PFCC^4 + E_{Left} + E_{Right} + PS) \quad (3.6.25)$$

where,

- PFCC*: Primary Flight Control Computer
- IPC₁, IPC₂*: Intelligent Power Controller
- IPSC*: Integrated Power Signal Cable
- E_{Left}, E_{Right}*: Engine Left And Right
- PS*: Power Supply

3. SYSTEM ALGEBRA

The best and the worst fault tolerances of the system are shown in equations 3.6.26 and 3.6.27.

$$\begin{aligned}\beta_{best}\left(\sum_{i=1}^7 S_i\right) &= \sum_{i=1}^7 \beta_{best}(S_i) \\ &= \beta_{best}\left((IPC_1 + IPC_2)^4 + SIG^4 + PFCC^4 + E_{Left} + E_{Right} + PS\right)\end{aligned}\quad (3.6.26)$$

$$\begin{aligned}\beta_{worst}\left(\sum_{i=1}^7 S_i\right) &= \sum_{i=1}^7 \beta_{worst}(S_i) \\ &= \beta_{worst}\left((IPC_1 + IPC_2)^4 + SIG^4 + PFCC^4 + E_{Left} + E_{Right} + PS\right) \\ &= \beta_{worst}\left(IPC_1, IPC_2, SIG, PFCC, E_{Left}, E_{Right}, E_{PS}\right)_{Min}\end{aligned}\quad (3.6.27)$$

The functional availability expression for the system is given in equation 3.6.28.

$$Availability = \left(E_{Left}, E_{Right}, PS\right)_{Min}\quad (3.6.28)$$

The transition of the system from a safe to an unsafe one depends on failures of the left and right engines, and the power supply. If any one of them fails the system transits to the unsafe state.

3.6.8 LCN – A Loosely Coupled Network System: Commercial Domain

Schiebe [109] describes the loosely coupled network system. This network provides high-speed data communication between the computers and its peripherals. The block diagram of the system is shown in Figure 3.23.

The SA expression for the system is shown in equation 3.6.29.

$$S = \left(COAX + MF^N + NAD^N + CTL^N + ADP^N\right)\quad (3.6.29)$$

where,

- COAX*: Co-axial Cable
- MF*: Interfaces
- CTL*: Interfaces
- ADP*: Adapter

3.6 Analyzing other existing Safety System Designs using SA: Reverse Engineering

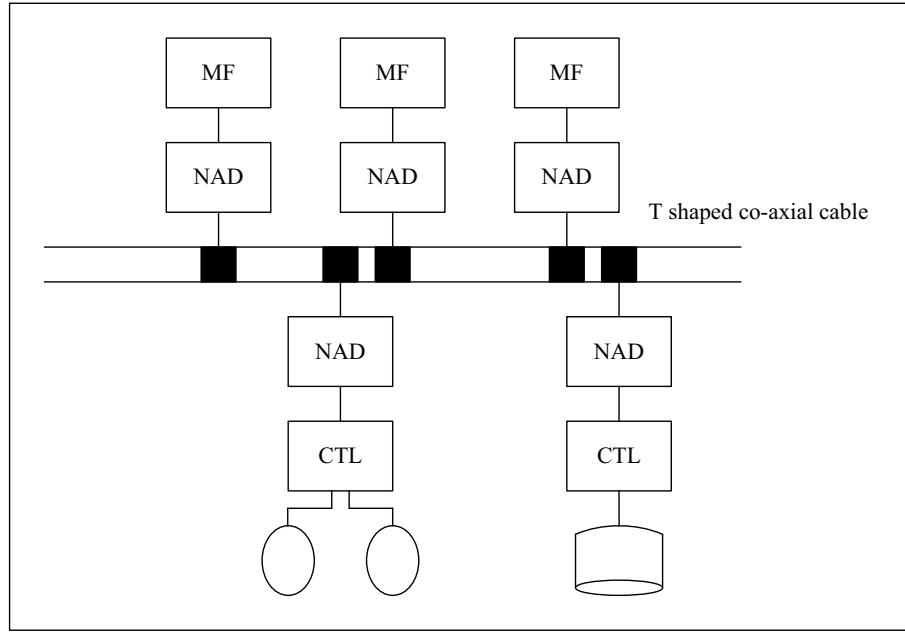


Figure 3.23: Block Diagram of LCN

The best and the worst fault tolerances of the system are shown given in equations 3.6.30 and 3.6.31.

$$\beta_{best} \left(\sum_{i=1}^5 S_i \right) = \sum_{i=1}^5 \beta_{best}(S_i) \quad (3.6.30)$$

$$= \beta_{best} \left(COAX + (MF)^N + (NAD)^N + (CTL)^N + (ADP)^N \right)$$

$$\beta_{worst} \left(\sum_{i=1}^5 S_i \right) = \sum_{i=1}^5 \beta_{worst}(S_i) \quad (3.6.31)$$

$$= \beta_{worst} \left(COAX + (MF)^N + (NAD)^N + (CTL)^N + (ADP)^N \right)$$

$$= \beta_{worst} (COAX, MF, NAD, CTL, ADP)_{Min}$$

The functional availability expression for the system is given in equation 3.6.32.

$$Availability = (COAX)_{Min} \quad (3.6.32)$$

3.6.9 System Complexity Analysis using SA: ATR72-600

The examples discussed as part of reverse-engineering demonstrate the capability of the SA technique in modeling safety-critical systems using mathematical relations of the ‘di-

3. SYSTEM ALGEBRA

rect sum' and 'direct product'. The operational properties between sub-systems determine the system state and the SSM concept determines the functional availability of the system. The analysis of the system design can be matched with project requirements to understand the design, detect design flaws, or modify the design by improving availability of the system.

The SA approach can be used to analyze a system of any complexity. This example discusses the capability of an SA approach to analyze a complex system. The ATR 72 [110] is a twin-engine turboprop short-haul regional airliner built by French-Italian aircraft manufacturer ATR. The ATR 72-600 will feature the latest technological enhancements while building upon the well-known advantages of the current aircraft, viz., high efficiency, proven dispatch reliability, low fuel burn, and operating cost. The boost function provides additional power as needed (for instance during takeoff). A glass cockpit flight deck featuring five wide LCD screens will replace the current EFIS (Electronic Flight Instrument System). In addition, a multi-purpose computer (MPC) will further enhance flight safety and operational capabilities. The ATR 72-600 consists of 30 sub-systems; the block diagram of the system is shown in Figure 3.24.

The SA expression for the system is shown in equation 3.6.33.

$$\begin{aligned}
 S = & [(ICP)^2 + (ADS)^2 + (AHRS)^2 + (RA)^2 + (ADF)^2 + (DME)^2 + (VHFNAV)^2 \\
 & + (VHFCOM)^2 + VHFCOM_1 + (HF)^2 + (ATC)^2 + ESCP + (EFCP)^2 + (MPC)^2 \\
 & + (TxCAS) + (ATR_{sys}) + Clock + MPC + MFC + (MCDU)^2 + (PFD)^2 + [CAC + CAC] \\
 & + [(FGCP) + PtchAPAU + RollAPAU + YawAPAU + (PFD)^2 + (MFD)^2 + EWD]
 \end{aligned}
 \tag{3.6.33}$$

where,

<i>ICP</i> :	Integrated core processor
<i>ADS</i> :	Automatic dependent surveillance
<i>AHRS</i> :	Attitude heading reference system
<i>RA</i> :	Right Aileron
<i>ADF</i> :	Automatic direction finder system
<i>DME</i> :	Distance measuring equipment
<i>VHFNAV</i> :	VHF Navigator
<i>VHFCOM₁, VHFCOM₂</i> :	VHF communicators
<i>HF</i> :	High frequency
<i>ATC</i> :	Air traffic controller
<i>ESCP, ESCP₂</i> :	Enhanced speed control panel

3.6 Analyzing other existing Safety System Designs using SA: Reverse Engineering

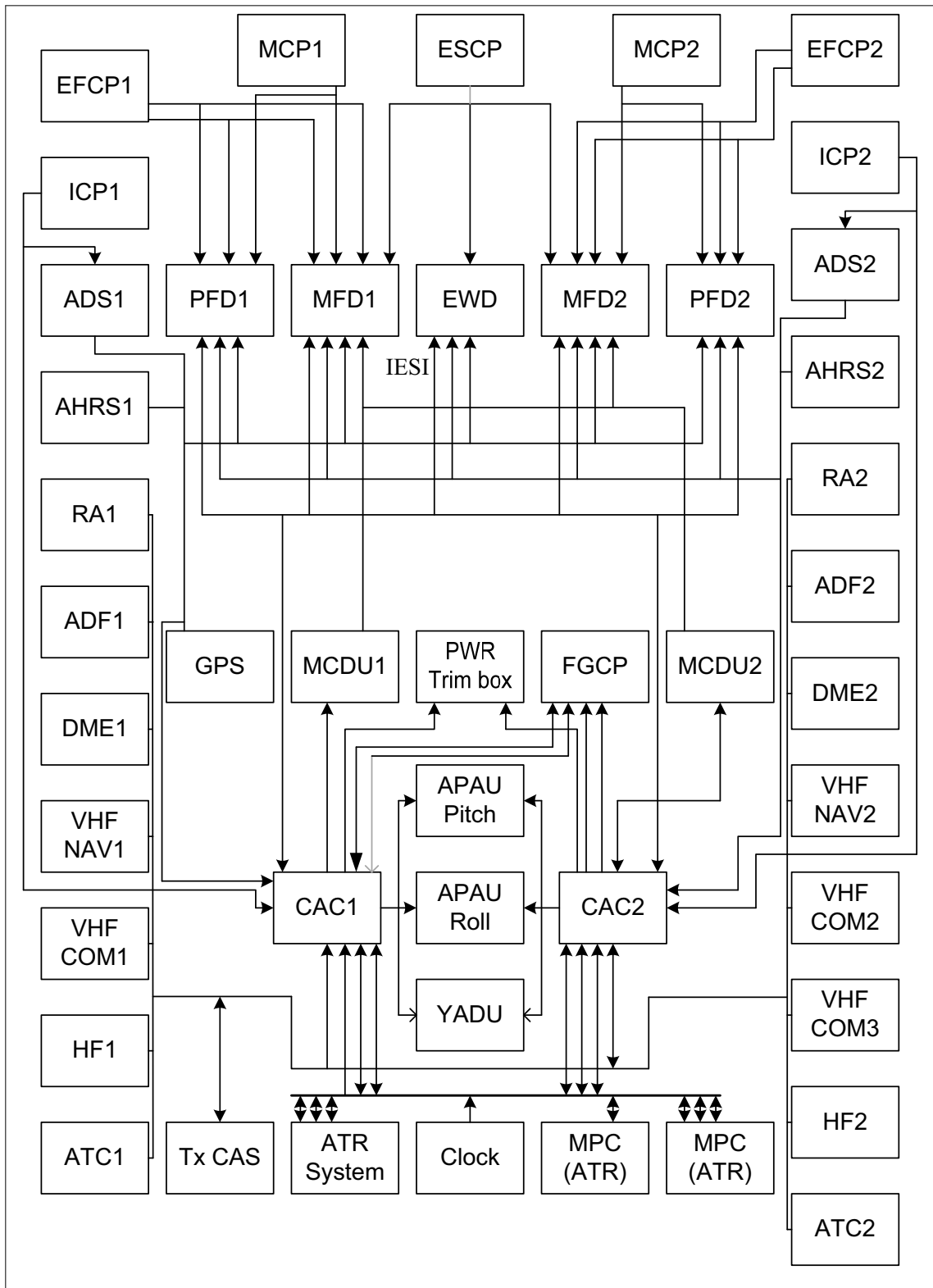


Figure 3.24: Block Diagram of ATR72-600

3. SYSTEM ALGEBRA

<i>TxCAS:</i>	Transmit computer avionics system
<i>ATRs_{sys}:</i>	Automatic thrust system
<i>Clock:</i>	Clock
<i>MPC:</i>	Multi-purpose computer
<i>MFC:</i>	Multi-function computer
<i>MCDU:</i>	Maintenance and control display unit
<i>PF_D:</i>	Primary function display
<i>CAC:</i>	Common air computer
<i>FGCP:</i>	Flight guidance control panel
<i>PtchAPAU:</i>	Pitch
<i>RollAPAU:</i>	Roll
<i>YawAPAU:</i>	Yaw
<i>PF_D:</i>	Primary function display
<i>MFD:</i>	Main function display
<i>EWD:</i>	Engine and warning display

The complexity of the system is shown in the expression. The analysis of a complex avionics system demonstrates that the SA approach can model complex systems and analyze their functional availability from system states and transitions. This unique feature of the SA, to determine a system's functional availability makes analysis of the system design effective and robust.

3.7 Other System Analysis Methods and Performance Comparison of System Algebra with RBD and FTA

We have studied system analysis methods like Reliability Block Diagram (RBD), Fault Tree Analysis (FTA), Probabilistic Safety Assessment (PSA), Hardware Reliability Analysis (HRA), Stochastic Petri Nets (SPN), and Statistical Model Checking (SMC). SA approach is comparable to RBD, FTA, PSA, HRA, SPN and SMC methods for its principle and scope. The applicability of SA for analysis can be demonstrated by comparing its performance with other popular industry system analysis techniques: Reliability Block Diagram (RBD) and the Fault Tree Analysis (FTA). These are popular industry practices for system analysis in early phases of the systems engineering life cycle process, as discussed by NASA [111]. System engineers must rely on contributions from the specialty engineering disciplines, in addition to traditional design disciplines, for functional expertise and specialized analytical methods. These specialty engineering areas include reli-

3.7 Other System Analysis Methods and Performance Comparison of System Algebra with RBD and FTA

ability, maintainability, logistics, test, production, transportation, human factors, quality assurance, and safety engineering.

In both systems analysis and engineering, the resources required for building a product are one of the decisions to be made. Reliability is defined as the probability that a device, product, or system will not fail for a given period of time under specified operating conditions. Reliability is an inherent system design characteristic. As a principal contributing factor to operating and support costs, reliability plays a key role in determining the system's cost-effectiveness.

Bazovsky [112] and Birolini [113] discuss reliability engineering as a major specialty discipline needed to build a cost-effective system. This is accomplished in the systems engineering process by actively designing features to ensure expected performance under specified operating conditions and by independent reliability analysis for design trade-offs. To build a reliable system, the design should include fault avoidance, fault tolerance, and functional redundancy. Fault avoidance specifies system design margins. Fault tolerance is the ability of a system to continue operating even after a component has failed. It is implemented through design redundancy, fault detection, and response capability. Functional redundancy allows the system to respond to component failures to meet mission requirements.

Fault Tree, discussed by Stamatelatos *et al.* [114], describes it as a graphical representation of the combination of faults and some (undesired) top events occurring as a result of those faults. It is constructed during a fault analysis, as a qualitative technique to uncover conditions that could cause a top event.

PSA is an analysis technique used during the design and operating phase of the nuclear plant. It considers the interaction between the environment, human-machine interface, and various systems in the plant. The international atomic energy agency, IAEA, provides guidelines for improving the quality of the PSAs for achieving an efficient and reliable decision making in PSA related projects. The IAEA report [115] discusses the regulatory perspective on the use of PSA, technical aspects of PSA related applications, PSA application process, and decision making aspects for projects carried out in the various countries. Guptan *et al.* [116] discuss risk based approach employed by NPCIL to improve the overall safety, reliability of the system. PSA follows a sequential approach. The first step is to assess the probability of an initiating event that may lead to a plant damage. Next step is to assess the reliability of systems designed to meet safety requirements. In the final step the sequence of events that may lead to an accident is to assessed. In comparison to PSA, SA is used only during the design of an embedded system. It is not targeted to be used in the operating phase. SA does not consider the interaction of the system with human and environment.

3. SYSTEM ALGEBRA

HRA is used to determine hardware reliability of the system. Srinivas *et al.* [117] discuss the hardware reliability assessment (HRA) of the computer based instrumentation and control (I and C) system for nuclear application in the country. The work followed a systematic approach of independent verification and validation to ensure the integrity of the system using fault tree methodology coupled with reliability prediction techniques. HRA uses the AERB Safety Guide No.AERBINPP-PHWRISGID-25 standard for generating the FTA methodology and MIL-HDBK-2 17F for Reliability Prediction of Electronic Equipment. SA, on the other hand is used for embedded systems and the design analysis of system functionality considers hardware details like failure rates and reliability as per the industry standards.

SPN have successfully been used for performance analysis of critical functionalities of the systems. SPN have the capability of analyzing the execution time, CPU throughput, I/O utilization and state-space explosion. Work by McSpadden and Lopez-Benitez, Souza *et al.*, and Singh *et al.*, discuss the capabilities of SPN. McSpadden and Lopez-Benitez [118] use SPN to validated the performance of the heterogeneous computing system for static task allocations. Singh *et al.* [119] use SPN to verify the most critical function of the Nuclear power plant i.e. the communication protocol. For verifying this functionality SPN is used over Markov chain. TimeNet 4.0 tool is used to check the throughput of the protocol and was found to be satisfactory as per the application requirements. Souza *et al.* [120] discuss the use of deterministic SPN (DSPN) for analyzing the performance of pooling mechanism of JBoss application server used for distributed systems. The DSPN was able to model the actual pooling mechanism and the results were very satisfactory. In comparison to SPN, SA does not analyze execution time, CPU throughput, I/O utilization and state-space explosion. It addresses the system level requirements and not detailed requirements for a particular function or algorithm as discussed by Singh *et al.* and Souza *et al.*

SMC is used for verifying stochastic systems, with hybrid behavior. The approach is fast and has good scalability. This feature is brought out by Zuliani *et al.* [121]. The statistical model checker with Bayesian sequential hypothesis was used to create a fault model for the oxygen sensor of the fuel control system designed in the Simulink/Stateflow environment. Legay *et al.* [122] analyze SMC and provide a tutorial for analyzing SMC for efficiency, uniformity, simplicity, and its applicability. The tutorial also provides suggestions for improving the SMC for its applicability to non-stochastic systems, development of simulation techniques, and verification of hybrid systems. Sen *et al.* [123] discuss a novel approach to verify the scholastic system as part of black box testing. The proposed approach is based on Monte Carlo simulation and statistical hypothesis testing for a grid, and tandem queuing network as a case study. VESTA was used to generate the required

3.7 Other System Analysis Methods and Performance Comparison of System Algebra with RBD and FTA

test cases. The tool uses a sequence of inter-related hypothesis testing to check for a specific property in probabilistic computation tree logic. Younes *et al.* [124] discuss the importance of using probabilistic model checker, PMC, in order to verify the behavior of systems which are stochastic in nature. PMC either uses numerical or statistical approach to verify the system behaviour. The performance of the numerical and statistical probabilistic model checkers are analyzed for their performance by empirical evaluation of published systems as a case study. Tandem queuing network, symmetric polling system, dependable workstation cluster, and grid world are evaluated and a mixed approach is suggested for an effective analysis

3.7.1 Reliability Block Diagram

System reliability is about the reliability of constituent sub-systems and components; software reliability is the reliability of the programs that are part of it. A system is a collection of components, subsystems, and/or assemblies arranged to a specific design to achieve desired functions with acceptable performance and reliability. The type of components, quantity, quality, and their arrangement within the system have a direct effect on system reliability. Software reliability engineering relies on a disciplined software engineering process to anticipate and counter unintended consequences. It depends on well-defined requirements, design and implementation. The software reliability engineering is shown in Figure 3.25.

System reliability helps in predicting performance over time, as discussed by Weibull [7], Stamatelatos *et al.* [114], Bazovsky [112], and Birolini [113]. Reliability of a system helps in predicting the availability of the system, MTBF, and MTTR. The reliability requirement of a system depends on its application. The bath-tub curve is used to analyze defects in the life-cycle. The curve consists of infant mortality phase, the useful phase and the wearout period. During the infant mortality phase, initial failure rates are typically higher than average failure rates. Initial high failure rates are an indication of low design margin, manufacturing defects, or component issues. In the useful life phase, 'MTBF' is a reciprocal of the failure rate. For example, 0.001 failures/ year signifies a 1000-year MTBF. The 'weak' units in the population are gone by this time. Failure is often due to external stress. In the wearout period, failures are due to limitations of the materials used. This can be thought of as the life of a product. There are various reliability analysis techniques for analyzing complex and critical systems. These techniques include RBD, Network Diagrams, FTA and Monte Carlo Simulation. Of these techniques, RBD is the most widely used technique. RBD analyzes system reliability and availability of complex

3. SYSTEM ALGEBRA

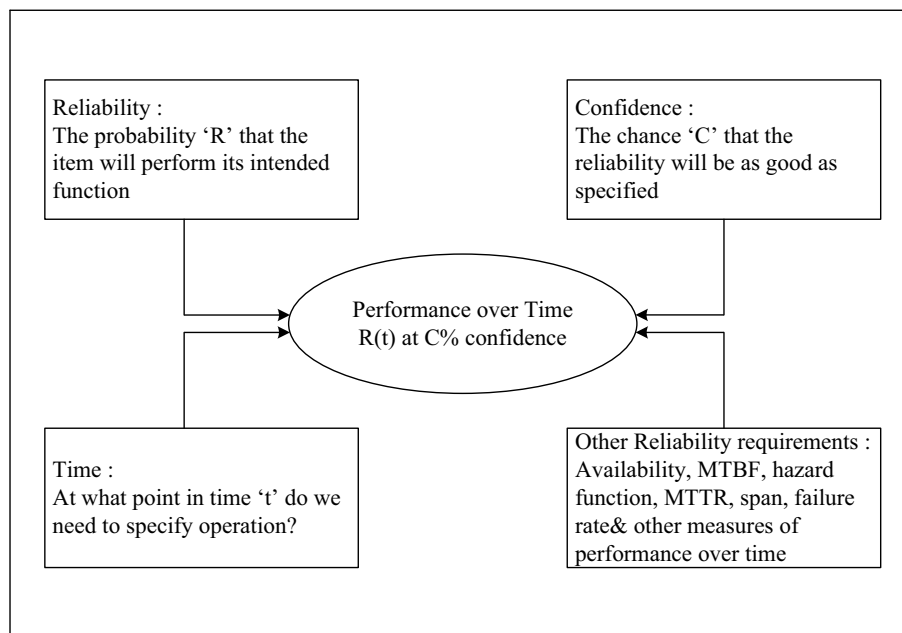


Figure 3.25: Reliability Engineering

systems using block diagrams to represent network relationships. The RBD structure defines the logical interaction of failures within a system that are required to sustain system operation.

Any system can be decomposed into one of the following configurations - series, parallel, series-parallel, k-out-of-n redundancy confirmation, complex system and standby system. Properties of reliability are:

- **1:** Reliability is a probability.
- **2:** Reliability is predicated on intended function. Generally, this is taken to mean operation without failure.
- **3:** Reliability applies to a specified period of time or relevant units. For example; the automotive industry might specify reliability in terms of miles; and the military might specify reliability of a gun for a certain number of rounds fired. A piece of mechanical equipment may have a reliability rating value in terms of cycles of use.
- **4:** Reliability is restricted to operation under stated conditions. This constraint is necessary because it is impossible to design a system for unlimited conditions. A Mars Rover will have different specified conditions from that of the family car. The operating environment must be addressed during design and testing.

Characteristics of the RBD are:

3.7 Other System Analysis Methods and Performance Comparison of System Algebra with RBD and FTA

- **1:** Based on the reliability of hardware and software components.
- **2:** Bottom-up approach.
- **3:** Used in determining the reliability, availability and common cause failure of a system.

The RBD model is shown in Figure 3.26.

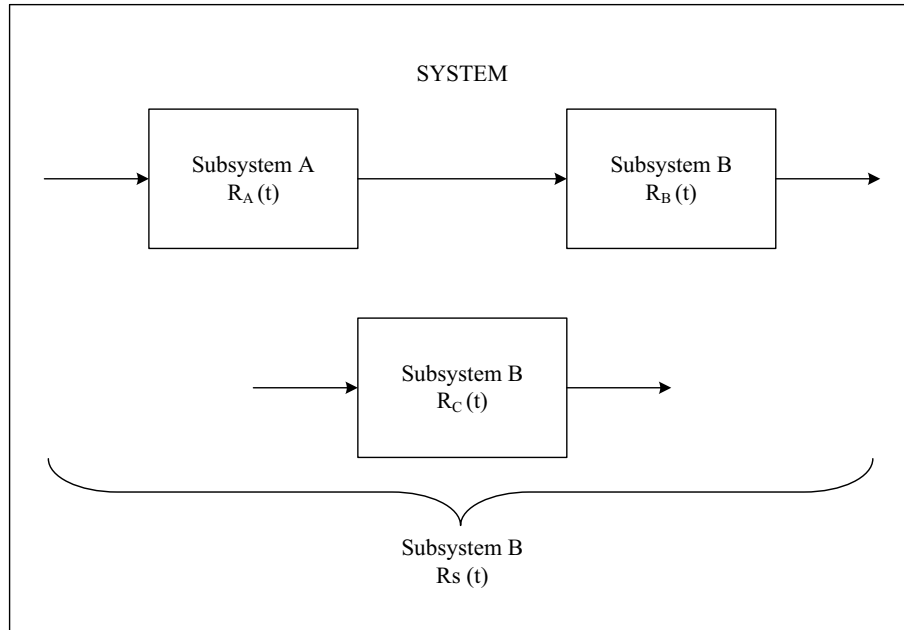


Figure 3.26: RBD Model [7]

The reliability models exist for various system configurations such as *series*, *parallel*, *standby parallel redundancy system*, *m-out-of-n active parallel redundancy system* and *series and parallel combination system*. The reliability models for the various system configurations are discussed. The reliability model of a series system is shown in Figure 3.27.

The parallel system configuration can be a simple standby parallel system or an m-out-of-n active parallel redundancy system. The standby parallel redundancy is shown in Figure 3.28.

The reliability model of m-out-of-n active parallel redundancy system is shown in Figure 3.29.

The reliability model of m-out-of-n Systems is shown in Figure 3.30.

The reliability model of a simple series-parallel system and a non-series parallel system is shown in Figure 3.31 and Figure 3.32.

The reliability models of complex systems are shown in Figure 3.33 and Figure 3.34.

3. SYSTEM ALGEBRA

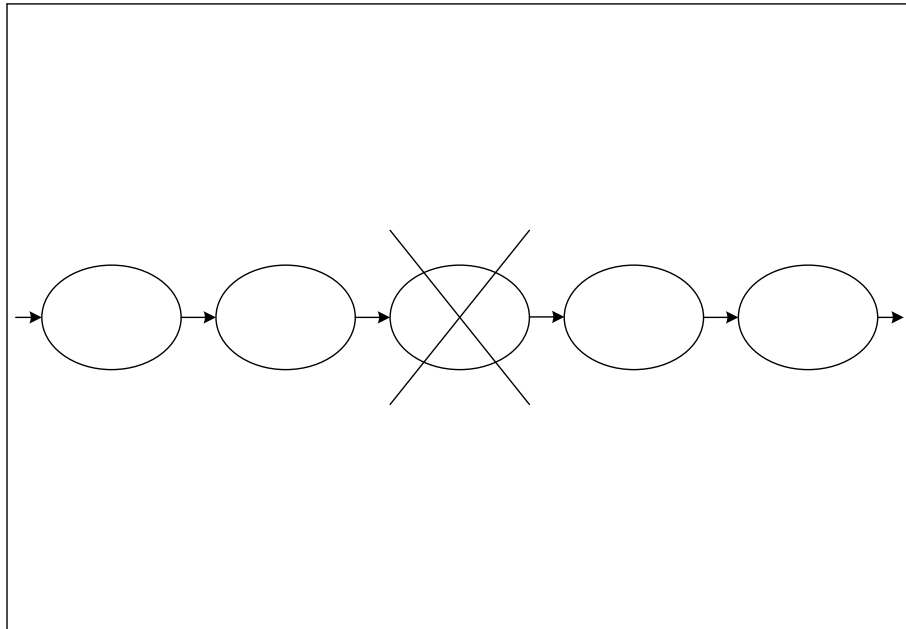


Figure 3.27: Reliability Model of the Series System [7]

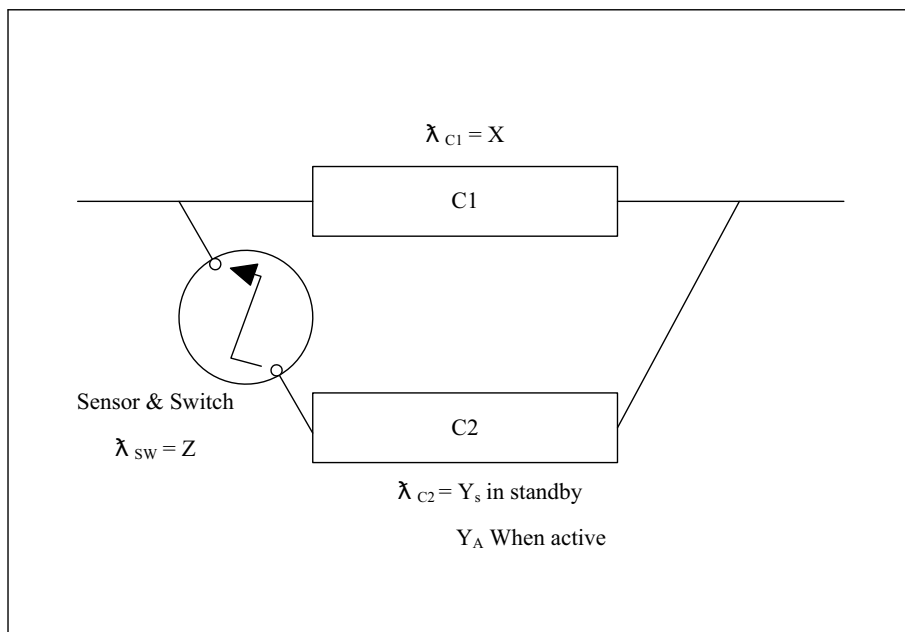


Figure 3.28: Reliability Model of Standby Parallel Redundancy [7]

3.7 Other System Analysis Methods and Performance Comparison of System Algebra with RBD and FTA

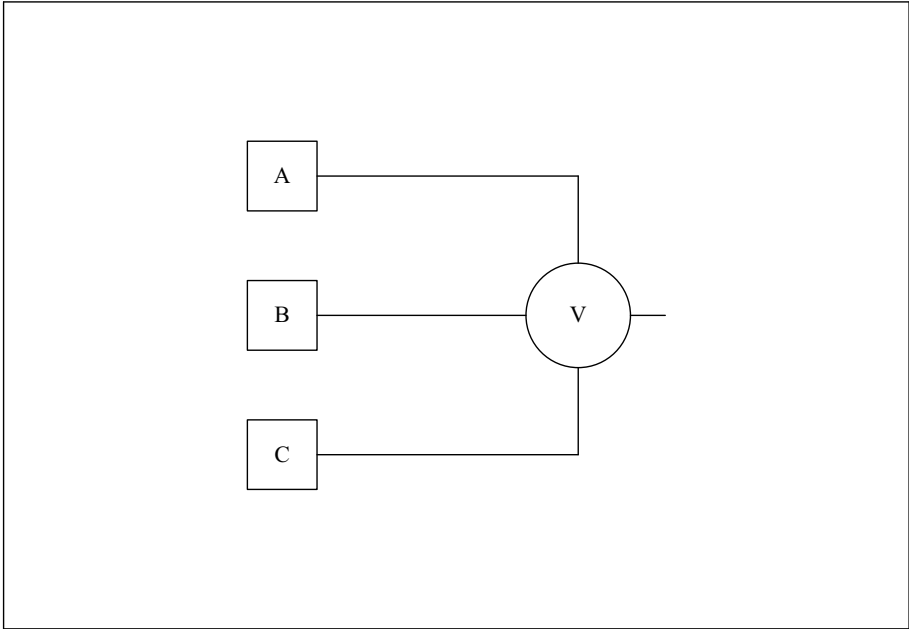


Figure 3.29: Reliability Model of m-out-of-n Active Parallel Redundancy [7]

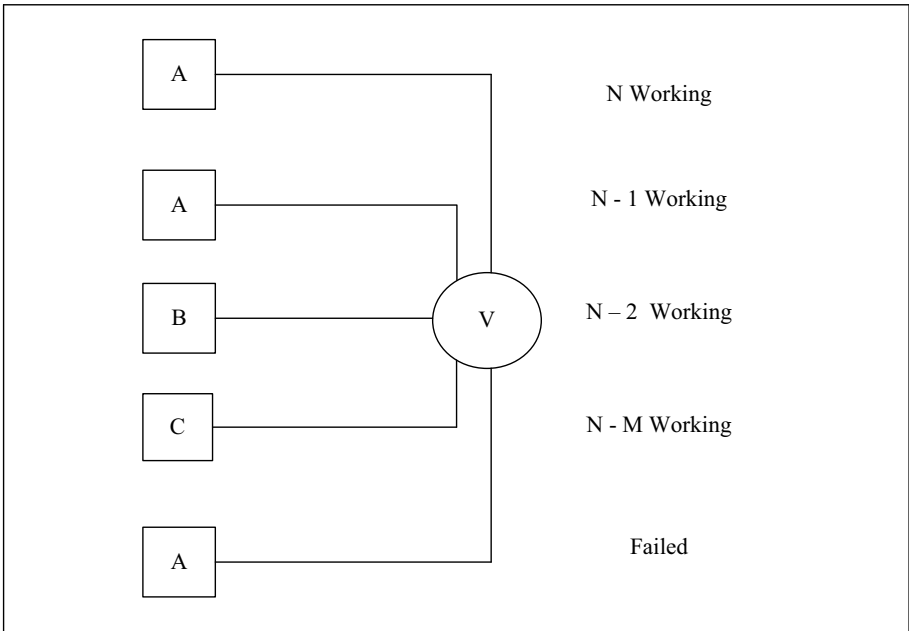


Figure 3.30: Reliability Model of m-out-of-n Systems [7]

3. SYSTEM ALGEBRA

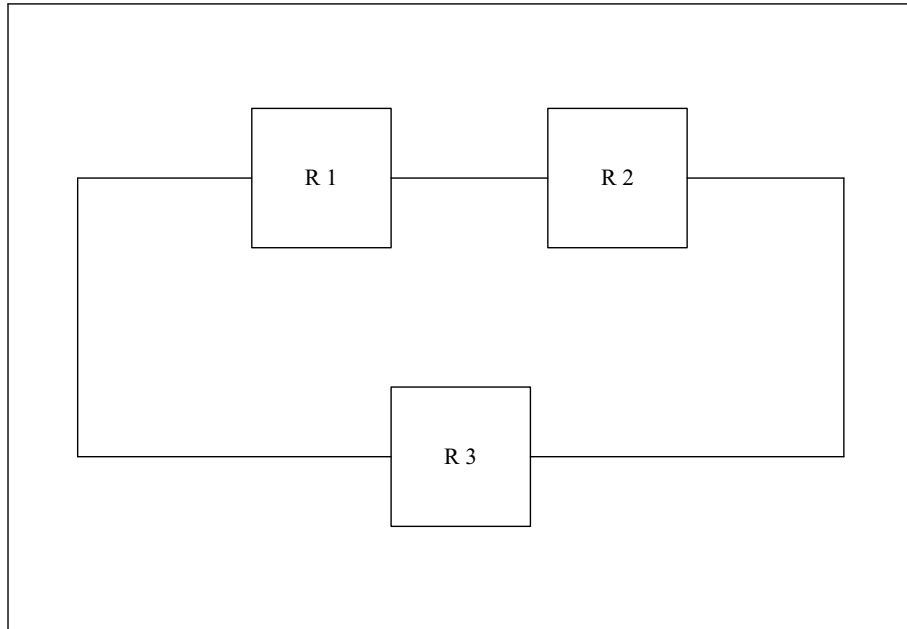


Figure 3.31: Reliability Model of Simple Series and Parallel System [7]

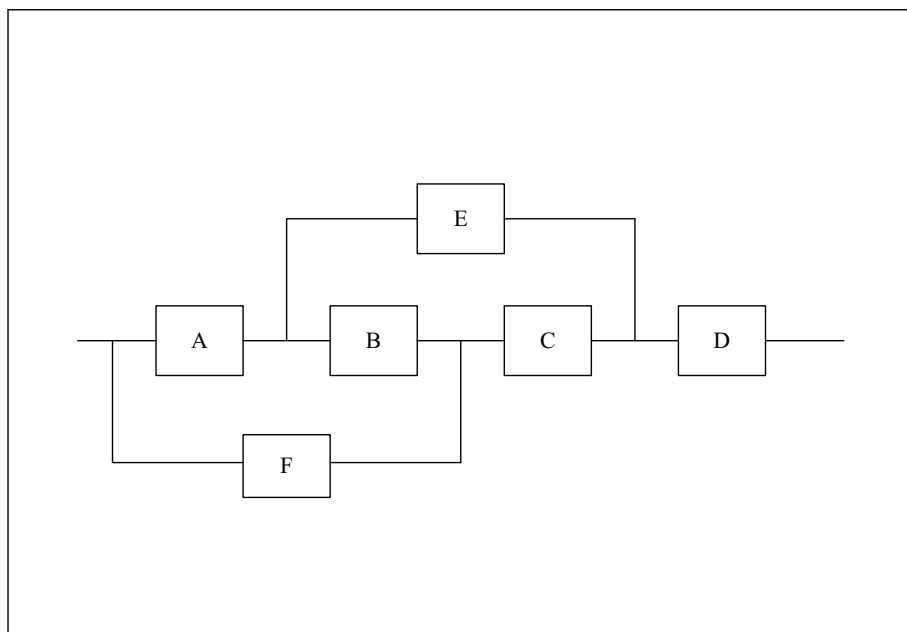


Figure 3.32: Reliability Model of Non-series Parallel System [7]

3.7 Other System Analysis Methods and Performance Comparison of System Algebra with RBD and FTA

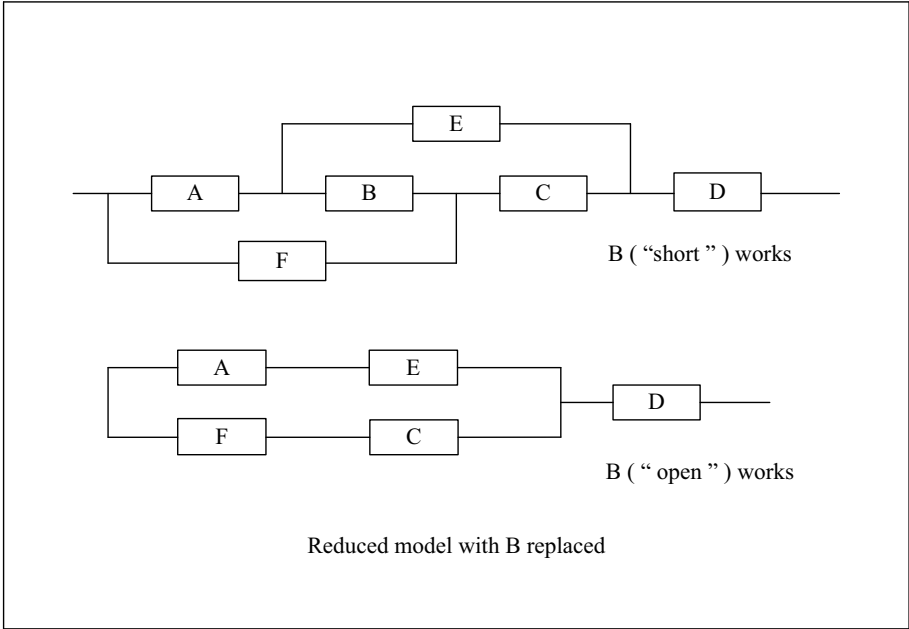


Figure 3.33: Reliability Model of Complex Systems [7]

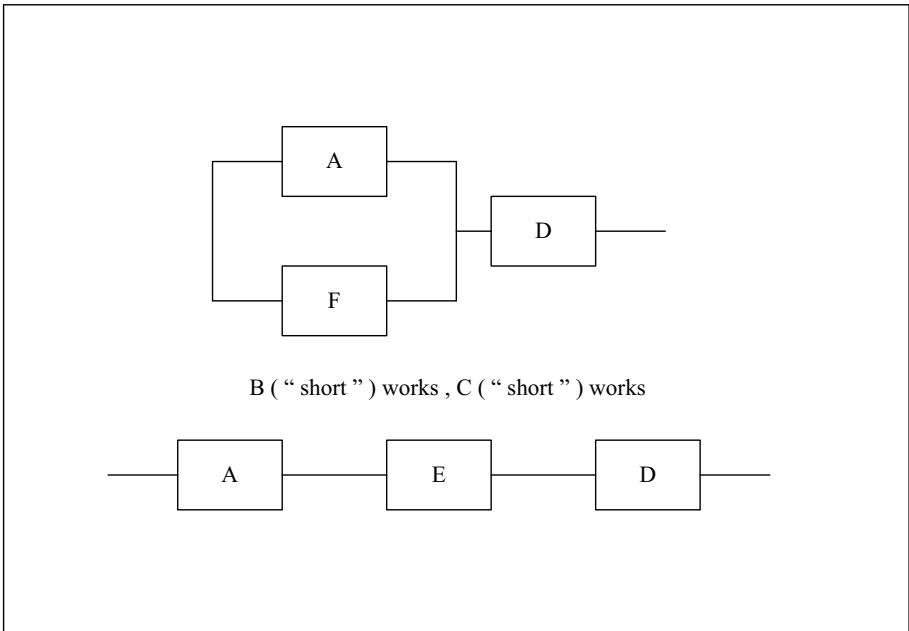


Figure 3.34: Reliability Model of Complex Systems [7]

3. SYSTEM ALGEBRA

Standard	Applicable industry
MIL-STD-785	Reliability Program for Systems and Equipment Development and Production, U.S. Department of Defense
MIL-HDBK-217	Reliability Prediction of Electronic Equipment, U.S. Department of Defense
MIL-STD-2173	Reliability Centered Maintenance Requirements, U.S. Department of Defense
MIL-HDBK-338B	Electronic Reliability Design Handbook, U.S. Department of Defense
MIL-STD-1629A	Procedures For Performing A Failure Mode, Effects And Criticality Analysis
MIL-HDBK-781A	Reliability Test Methods, Plans, and Environments for Engineering Development, Qualification, and Production, U.S. Department of Defense
IEEE 1332	IEEE Standard Reliability Program for the Development and Production of Electronic Systems and Equipment, Institute of Electrical and Electronics Engineers
DEF STAN 00-40	Reliability and Maintainability

Table 3.10: Reliability Standards

The RBD of B777 FCS is shown in Figure 3.35 and that of A380 is shown in Figure 3.36.

There exist various reliability standards such as MIL-STD-785, MIL-HDBK-217, MIL-STD-2173, MIL-HDBK-338B, MIL-STD-1629A, MIL-HDBK-781A, IEEE 1332 and DEF STAN 00-40, as shown in the Table 3.10.

3.7.2 Fault Tree Analysis

FTA is a top-down deductive approach to failure analysis, starting with a potential undesirable event (accident) called a TOP event, and determining all possible scenario that can result in a TOP event, as discussed in NASA [111] and Vesley *et al.* [125]. The analysis determines how a TOP event can be caused by individual or combined lower level failures or events. The causes of a TOP event are connected through logic gates. FTA is the most commonly used technique for causal analysis in risk and reliability studies; it identifies all possible causes of a specified undesirable TOP event. FTA leads to improved understanding of system characteristics. Design flaws and insufficient operational and maintenance procedures may be revealed and corrected during the fault tree construction. FTA is not (fully) suitable for modeling a dynamic scenario. It is one of the most important logic and probabilistic techniques used in Probability Risk Analysis and system reliability assessment today:

- **1:** Can be applied in existing systems and those being designed.
- **2:** Can be used for new designs where specific data do not exist; FTA can provide an estimate of the failure probability and important contributing factors using generic data to bracket design components or concepts.

3.7 Other System Analysis Methods and Performance Comparison of System Algebra with RBD and FTA

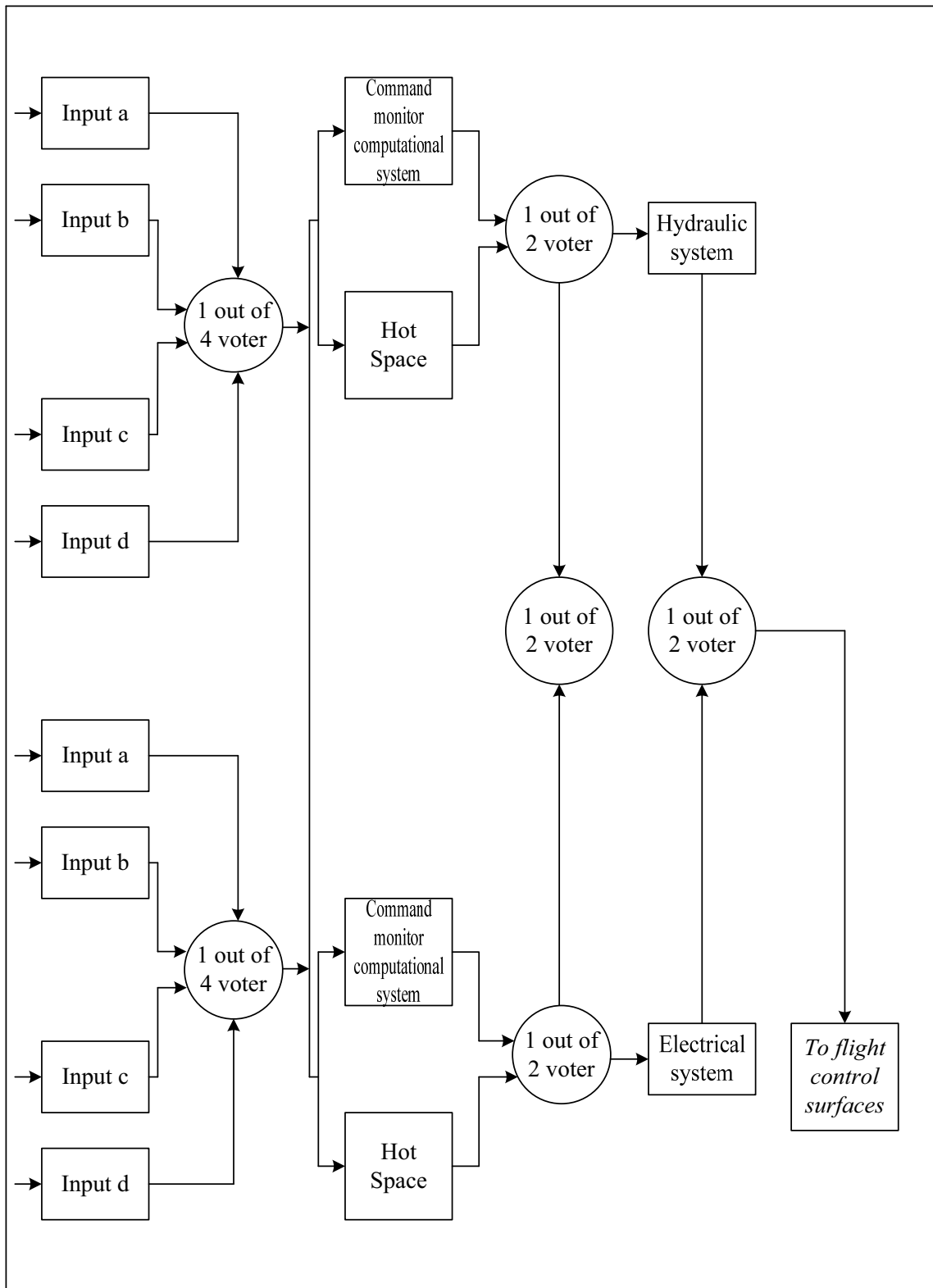


Figure 3.35: RBD of B777 FCS

3. SYSTEM ALGEBRA

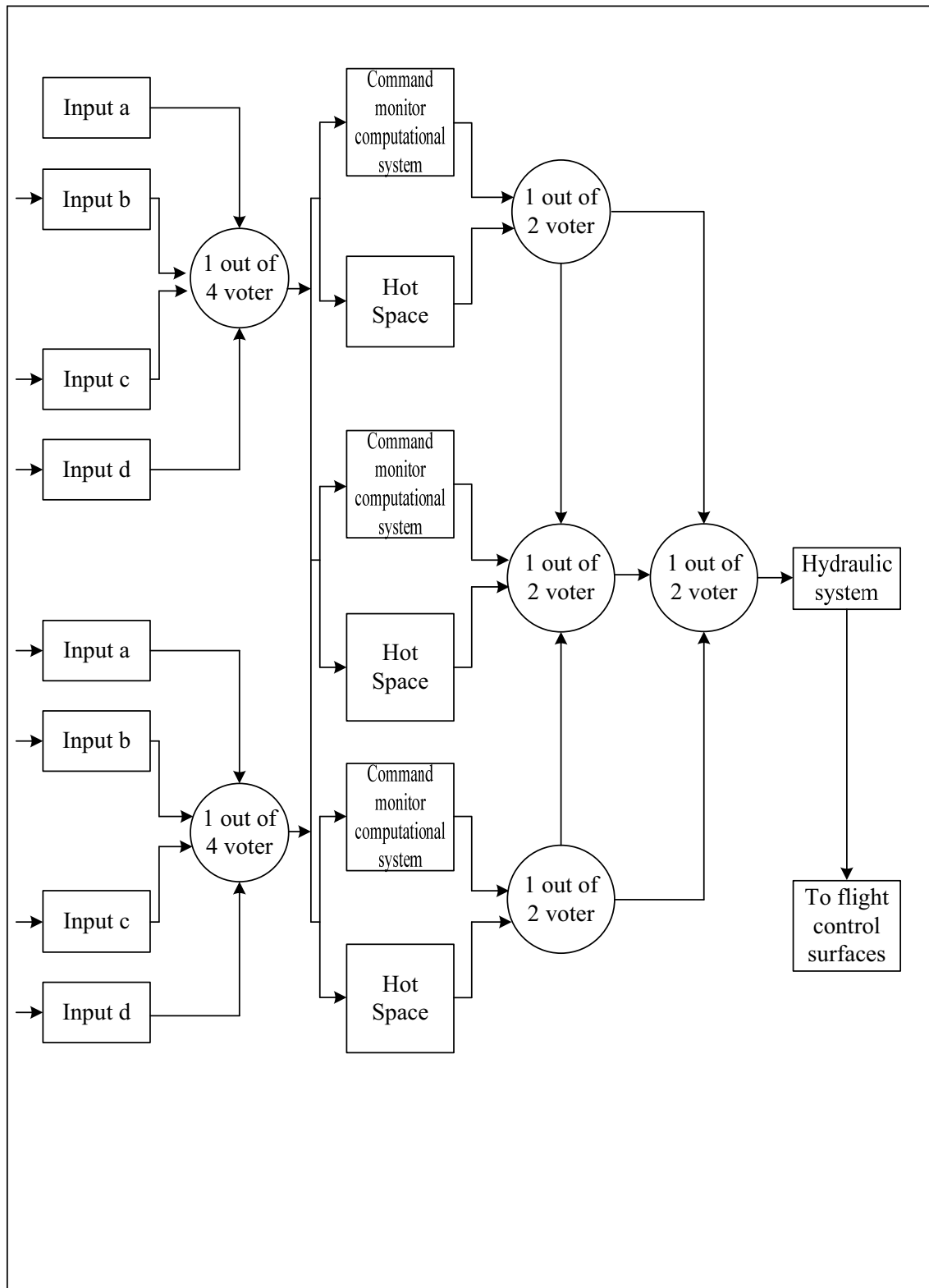


Figure 3.36: RBD of A380 FCS

3.7 Other System Analysis Methods and Performance Comparison of System Algebra with RBD and FTA

- **3:** Can be used as an important element for developing a performance-based design.
- **4:** Can be applied for an existing system. FTA can be used to identify weaknesses and to evaluate possible upgrades. It can also be used to monitor and predict behavior.
- **5:** Can be used to diagnose causes and potential corrective measures for an observed system failure.

During the systems engineering life cycle, FTA is derived once the system design is conceptualized and the FMECA (Failure mode effects, criticality analysis) of the system is available. The hardware symbols used to draw the FTA graph for some of the basic gates such as *top or intermediate event*, *AND*, *OR*, *basic event initiating fault/failure*, *fault event*, *transfer symbol*, *inhibit*, *condition symbol*, and a *normal event* are shown in Figure 3.37.

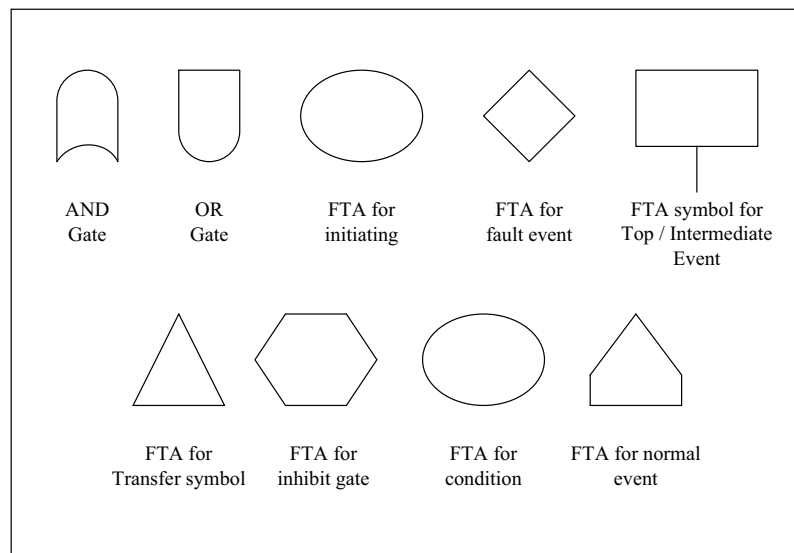


Figure 3.37: FTA Basic Gates [7]

The logic flow FTA templates for *If-Then-Else*, *While*, *For*, *Function Call*, *Go-to/Break*, *Do-while* are shown in the Figure 3.38, Figure 3.39, Figure 3.40 and Figure 3.41 respectively.

3. SYSTEM ALGEBRA

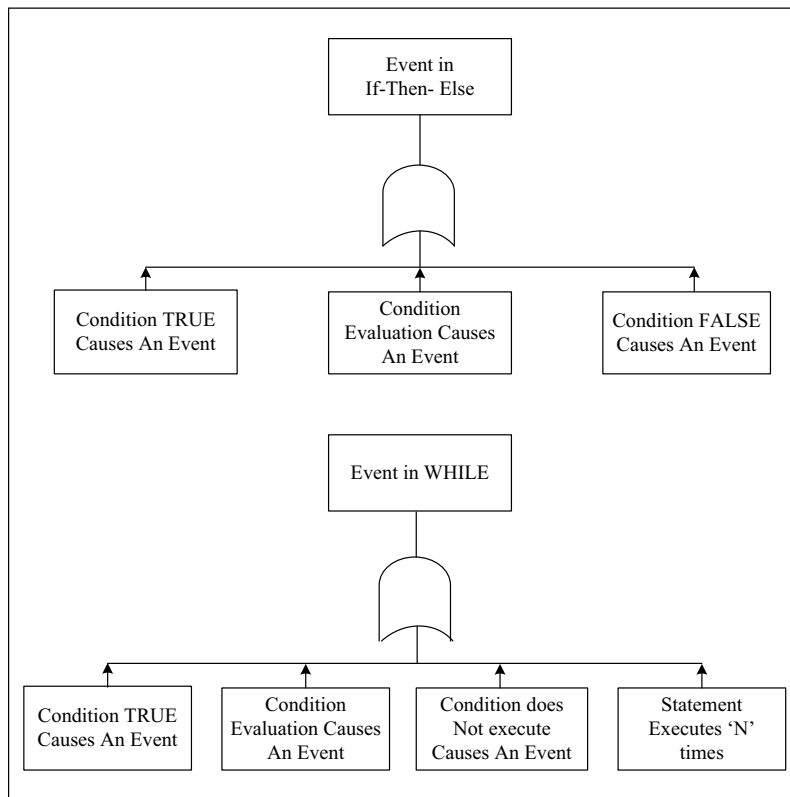


Figure 3.38: FTA Symbol for Top or Intermediate Event [7]

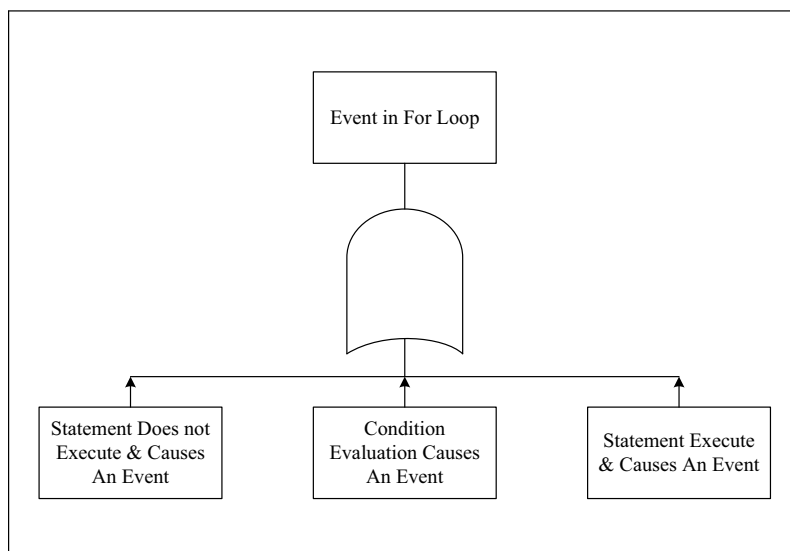


Figure 3.39: FTA for AND Gate [7]

3.7 Other System Analysis Methods and Performance Comparison of System Algebra with RBD and FTA

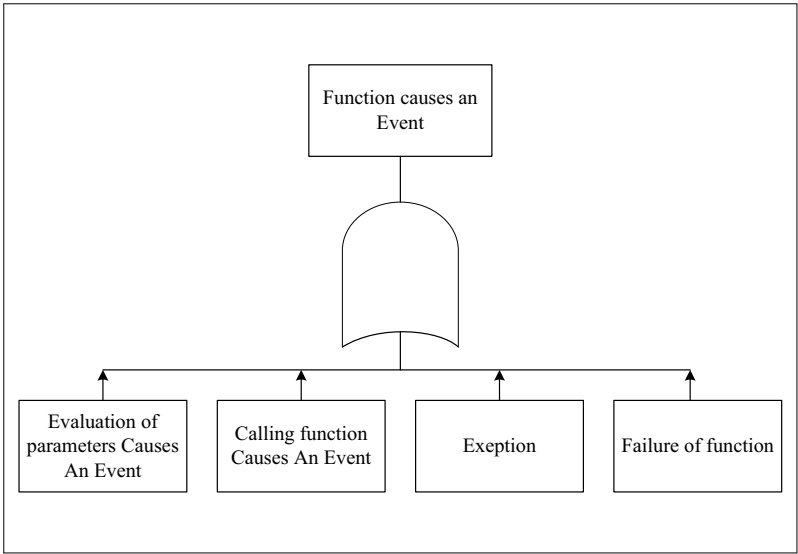


Figure 3.40: FTA for OR Gate [7]

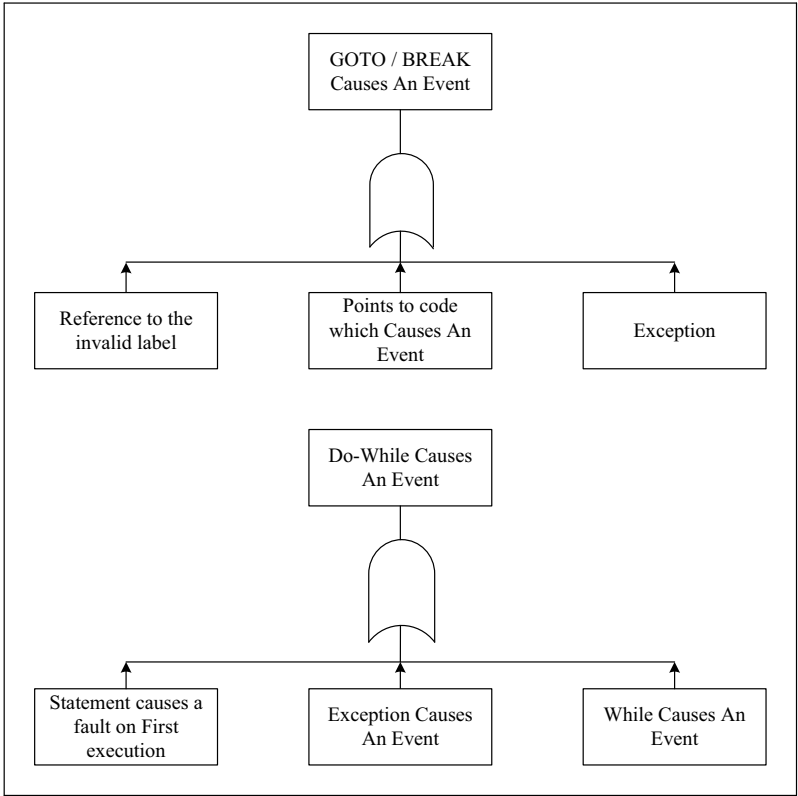


Figure 3.41: FTA for Initiating Failure [7]

3. SYSTEM ALGEBRA

There exists a relationship between RBD and FTA; RBD diagrams can be changed to FTA and vice-versa as discussed in [7]. The representation of the logic gates by means of RBD and FTA is shown in Figure 3.42 for *AND gate*, Figure 3.43 for *OR gate and voting OR gate*, and Figure 3.44 for *inhibit condition and XOR gate*. XOR is not represented in RBD since in terms of system reliability, this would imply that a two-component system would function even if both components had failed.

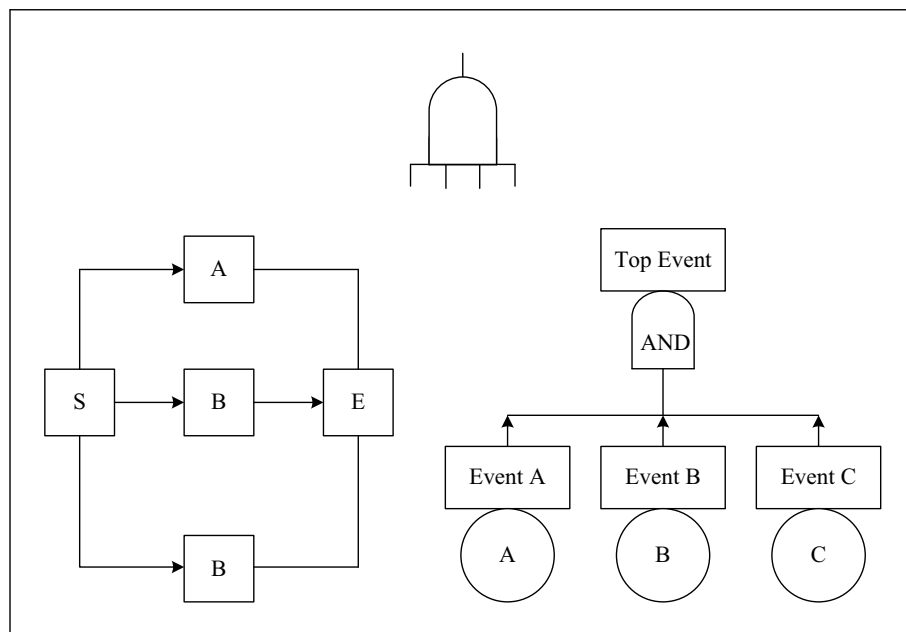


Figure 3.42: RBD, FTA Symbol for AND Gate [7]

Figure 3.45 shows the RBD generation for a FTA of a system. Similarly one can generate the FTA for an RBD diagram.

The FTA for the flight control system, using FTA logic gates and flow templates, is shown in Figure 3.46.

3.7.3 Comparison of System Algebra with other Popular System Analysis Methods

The simulation results presented and discussed validate the effectiveness of SA technique to verify critical system property, such as system functional availability, based on its states and its transition. The effectiveness of this approach is discussed by Bahil *et al.* [126], who show that system models are effective if they have observable states that can be simulated by SA. The RBD and FTA approaches are now compared with SA technique, and the merits and limitations of each technique for system analysis are discussed.

3.7 Other System Analysis Methods and Performance Comparison of System Algebra with RBD and FTA

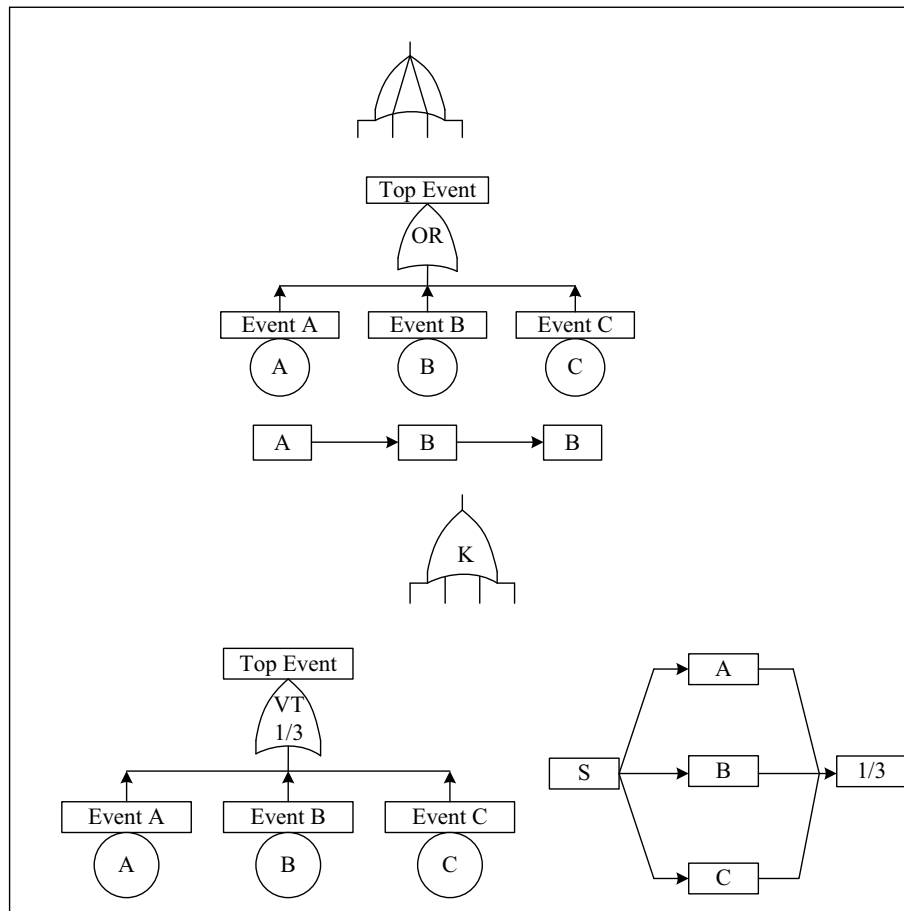


Figure 3.43: RBD, FTA Symbol for AND Gate [7]

System Algebra: This is an FM approach based on the functionality and failure rates of hardware and software components similar to RBD. It is applicable for both critical applications requiring system safety analysis as well as non-critical applications. The SA approach can be implemented during the design or verification phase of the SE life cycle. The approach determines the transition of system states under various failure scenarios. System safety is analyzed based on the SA technique based on SSM concept. It is a top-down and bottom-up approach where a complex system is decomposed to series and parallel combinations, and where the SA expression is generated by composition of the series and parallel combinations. This approach determines system fault tolerance, availability and modularity. No tool is available to generate and demonstrate the qualities of SA. The system modeling does not consider human interfaces.

Reliability Block Diagram: It is a model-based method that depends on reliability of hardware and software components. The applications include all critical and non-critical

3. SYSTEM ALGEBRA

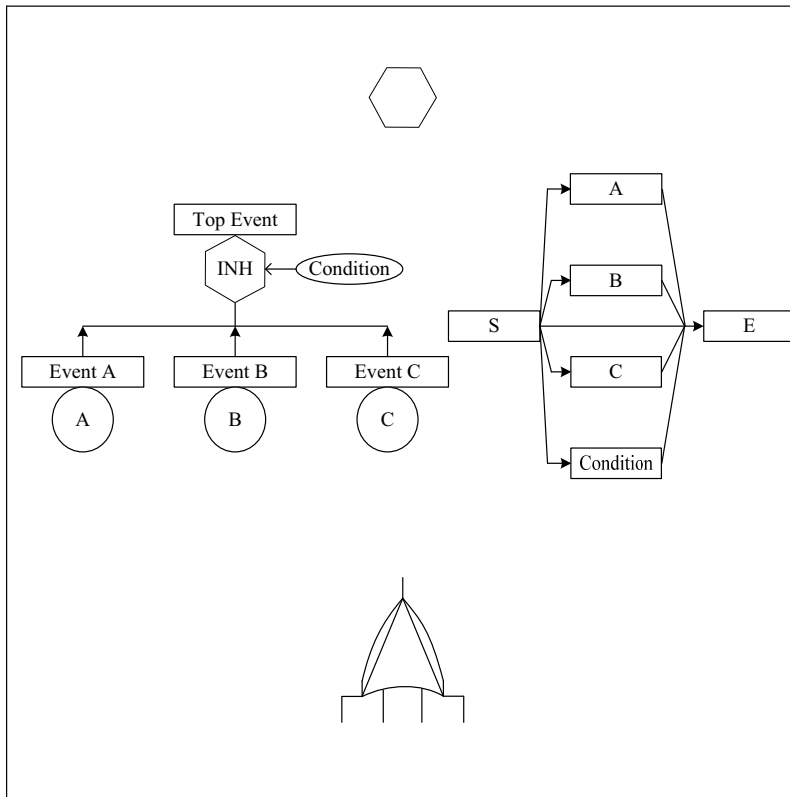


Figure 3.44: RBD, FTA Symbol for Inhibit Condition and XOR Gate [7]

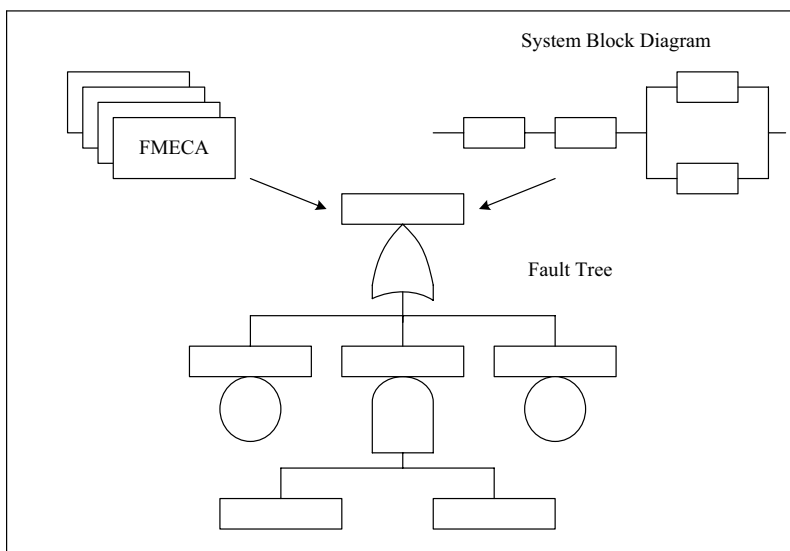


Figure 3.45: RBD Representation for FTA System [7]

3.7 Other System Analysis Methods and Performance Comparison of System Algebra with RBD and FTA

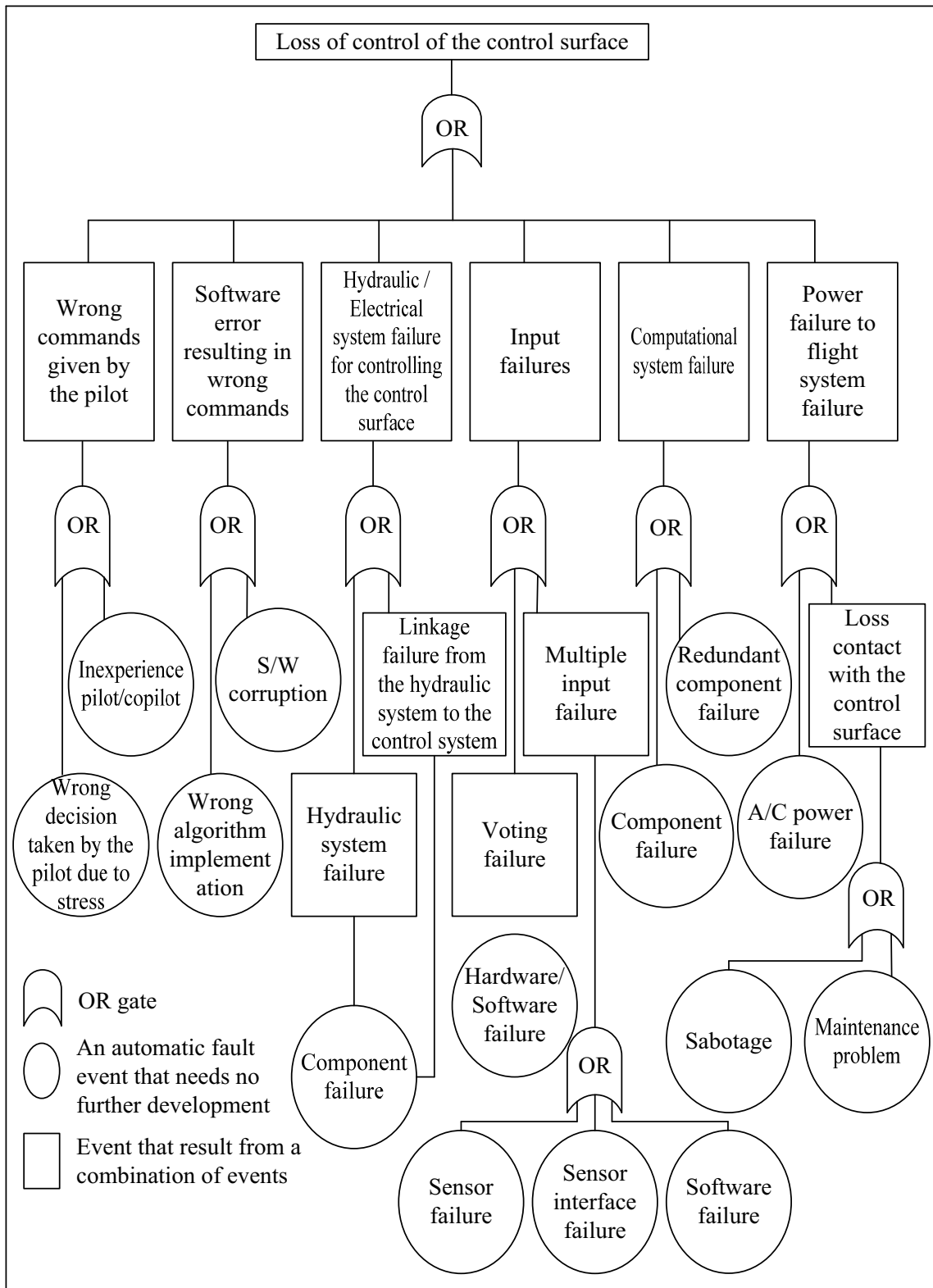


Figure 3.46: FTA of FCS System

3. SYSTEM ALGEBRA

systems. The RBD approach can be applied to the requirements phase of a systems engineering life cycle. Similar to SA, it is a top-down, bottom-up approach where a complex system is decomposed into the series and parallel combination where a complex system is decomposed to series and parallel combinations and where the RBD equation is generated by composition of the series and parallel combinations. The system is modeled using pre-defined reliability blocks. The system is represented by its functional block diagram and reliability equation. It can simulate hardware and software failures. It is used to determine reliability and common cause failures. Tools are available to estimate reliability of a system from its reliability block diagram. The system modeling does not consider human interfaces.

Fault Tree Analysis : This is a graphical technique based on the failure events of a system. It is best suited for critical applications that require system safety analysis. The FTA approach is implemented in the requirements phase of the systems engineering life cycle. Safety is analyzed based on the critical undesired events in the fault tree. It is a top-down approach that analyzes system behavior based on system functionality irrespective of its composition. The system is represented by a fault tree. The types of failure that can be simulated are hardware, software, and human interface. Tools are available to generate the fault tree of a system.

The comparison of SA with RBD and FTA shows that SA technique complements the other two methods of system analysis. SA uses the failure rates of the components or sub-systems similar to RBD, hence having no special requirement. The modeling of the system uses the top-down and bottom-up approach. RBD uses the RBD equation to compute the system reliability, where as SA derives the SA expression to depict the system. The system analysis using the SSM concept provides the functional availability information. This information is available in addition to the safety and reliability of a system determined by FTA and RBD. The analysis techniques do not provide the same information, demonstrating the complementary approach of these techniques. Together they provide more information about the system making the engineering process more effective.

4

Proposed Modified Systems Engineering Life Cycle Process in Formal Design, Verification and Validation

Engineering processes are being modified to adapt new techniques to increase effectiveness. These modifications of the engineering process are supported by industry standards. One of the popular standards in aerospace is the RTCA DO-178 standard. RTCA DO-178B was released in 1992 and with over two decades of inception of tools, model-based technology, object-oriented programming (OOP) and Formal methods, FM, RTCA DO-178C was released in June 2012 to include these advances. This standard provides guidelines to the aerospace industry in adapting new engineering processes for a safer, more reliable, and more available system. Other industry standards, such as IEC 61511 (Industry processes), IEC 61513 (Nuclear standard), CENELEC (EN 50126), and ISO 26262 (Automotive), are also adapting new techniques to improve their engineering process. The introduction of the FM-based design analysis strengthens the design phase of the engineering process by helping engineers in detecting design flaws, understanding the design requirements and improving design early on. System Algebra, SA, provides functional availability of a system by formally analyzing the design.

This research work proposes to introduce system functional availability analysis in the design phase in addition to existing system properties such as functionality, safety, security, reachability, and maintainability . The validation of these properties in the design phase helps reduce faults and enables designers to come up with better and safer designs. The introduction of the new property adds a new dimension to system analysis. This modified and effective design phase helps in improving quality while reducing time

4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION

cost for design implementation. The modified systems engineering process is validated by analyzing critical and high integrity systems used in safety-critical applications.

4.1 Evolution of Systems Engineering Process to Integrated Formal Engineering Process

The conventional engineering process is document-centric. The requirements are captured in documents and become a point of reference for all the other phases of the process. In the design phase, the system is designed and reviewed for its correct implementation. The design is manually implemented, tested, and verified to detect the faults.

Model-based development is a paradigm shift from the document and acquisition life-cycle to model-based approaches. The model-centric engineering process, also known as Model-based systems engineering, MBSE, provides unimpeded, undistorted communication between system, software, and hardware engineers to achieve the target as compared to a document-centric process. The model-centric approach bridges the gap between the requirement, design, and implementation phases. The major elements of the model-based development are discussed by Hosagrahara and Smith [127]. These elements are the model at the center, the system design-simulation, the executable specifications from models, the continuous test and verification, and the automatic code implementation. They remain the same, irrespective of the systems engineering process (either Vee or Waterfall or Iterative). The migration from document-centric (conventional) to model-centric model is shown in Figure 4.1 and includes major elements from both models of the engineering process.

In a document-centric approach, a document is at the center in contrast to the model element in a model-centric approach. Models used in the model-based development are in the designer's language of choice. The models visually capture system requirements. The system model helps designers visualize the system to be built, reason the system properties by means of simulation, and verify them. MBSE helps in better visualization of a design, enabling designers in system prototyping, analyzing system properties, and testing the system at the model level in contrast to addressing these activities at a later stage, the implementation level, in the conventional approach.

In the engineering process the system design is validated based on system properties such as functionality, safety, security, reliability, reachability, and maintainability; carried out by simulation, activity diagram, sequence diagram, statecharts, safecharts, and state machines. The system reliability analysis is performed by means of the Reliability

4.1 Evolution of Systems Engineering Process to Integrated Formal Engineering Process

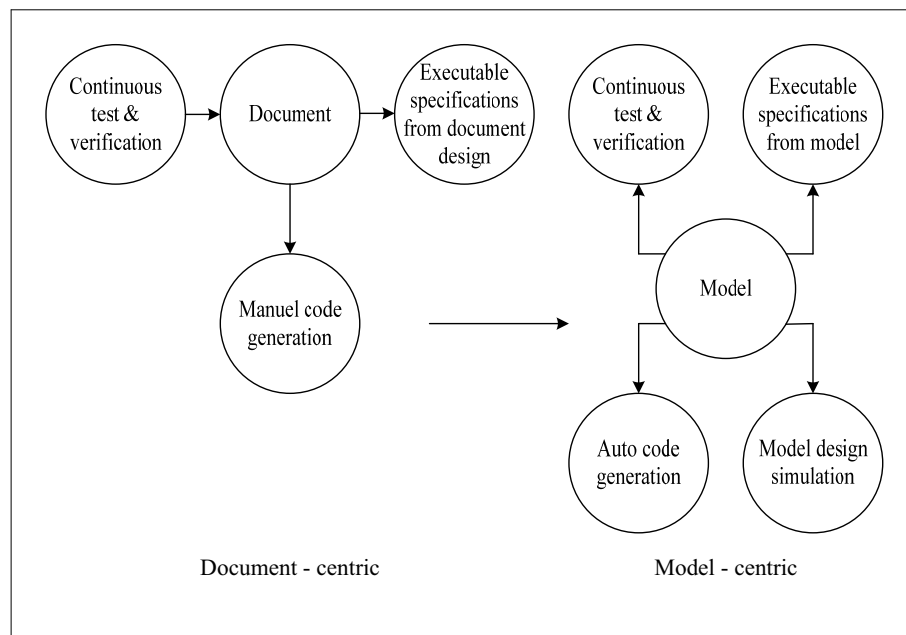


Figure 4.1: Major Elements of Model-based Design

Block Diagram (RBD). The system safety assessment process [128] consists of Functional Hazard Analysis (FHA), Preliminary System Safety Assessment (PSSA), Failure Modes and Effects Analysis (FMEA), Fault Tree Analysis (FTA), and Common Cause Analysis (CCA). The system reachability is analyzed by means of petri-nets discussed by Wang *et al.* [9].

There exist tools to compute these attributes. For example, Mathworks [127] tool *Modeling Metric Tool* is a graphical user interface (GUI) that can quantitatively measure the content of a Simulink, Stateflow model and incorporate these metrics into a development process measurement system. The tool supports model-based design in Simulink, Stateflow, structural modeling features, automated capture of input activity, time spent on various tasks, and the automated analysis and documentation of captured metrics.

Some of the other model-based leading methodologies are discussed by Estefan [3]. These methodologies adapted in the engineering process in the industry are listed in Table 4.1.

The benefit of using a model-based approach over the conventional approach is quantitatively compared by us, Nanda and Chinmayi [129], in our work to prove the effectiveness of the model-based formal approach. The proven stall warning and the aircraft interface computer system were developed by the conventional approach. The critical functionalities of the system were re-developed by a model-based approach using the

4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION

MBD Methodology	Toolset
Telelogic-Harmony SE	IBM Telelogic Tau and Telelogic Rhapsody
INCOSE Object-Oriented Systems Engineering Method (OOSEM) requirements management tools	COTS-based OMG SysML tools and associated
IBM Rational Unified Process for Systems Engineering (RUP SE)	Rational Rose product family
Vitech Model-Based System Engineering (MBSE) Methodology	CORE product suite
State Analysis (SA)	State Database

Table 4.1: Model-based Development Methodologies and Toolsets used in the Industry

Parameter	Conventional Approach	Model-Based Approach
Reusability	0 %	20%
Readability	90%	100%
Maintainability	68.33%	91.35%
Modularity	80%	100%
Reachability	50%	80%
Availability	78.2%	90%
Analyzability	20%	60%
Testability	66.6%	83.3%

Table 4.2: Metrics for Model-based Development and Conventional Engineering Process

Simulink tool. Table 4.2 shows the improvement in reusability, readability, maintainability, modularity, reachability, availability, analyzability, and testability features.

The derived metrics show the effectiveness of MBSE over the conventional engineering process. This idea is shown in Figure 4.2A and Figure 4.2B. The addition of the functional availability property in the design phase improves system analysis by 20%. Figure 4.2A shows the properties for analyzing the system design and Figure 4.2B shows the introduction of the FM-based system functional availability property to the already existing techniques. This addition is aimed at refining design analysis of the for a more effective engineering process. The effectiveness of the engineering process is determined by measuring the Measure of Performance (MOP). MOP is obtained by measuring the speed and effort for execution, and impact on safety of the system being design and developed. The metrics for the process is computed and validated by implementing different engineering frameworks for safety-critical system, SWS/AIC. With the integrated formal engineering process, an *improvement of 20%* is seen in analyzing the design phase as compared to the multi-tool, model-based framework. Further refining the system design can enhance the improvement in metrics. This is carried out as part of the research work at CSIR-NAL to develop the formal workbench.

Availability of a system is an important feature that indicates the length of time during which the system is functional. Milutinovic and Lucanin [130], Hoban [1], and Mitra *et al.* [131] help in analyzing the operational availability of a system from data as in [1], [130] or based on formal analysis as in [131]. These address the question *what time*

4.1 Evolution of Systems Engineering Process to Integrated Formal Engineering Process

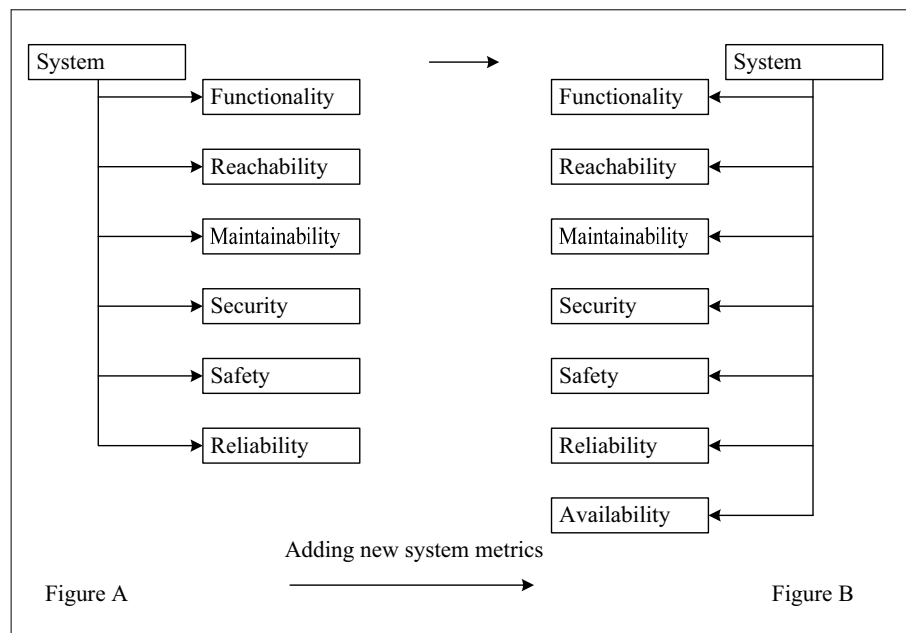


Figure 4.2: FM-based Availability Property Introduced in the System Design Analysis Attributes

to provide operational availability required for system maintenance. The SA-based system availability not only addresses the *what time* question but also *how* question, where the functional availability is described over time. The SA technique visualizes system state transitions with corresponding functionality over the life of a system. The visual demonstration of the functional availability is a step forward in analyzing the system design by its properties. This newly added property aids in designing a robust system that is better, safer, more reliable and available for a longer time. The introduction of the functional availability property in the design analysis of a system is a *paradigm shift* in system performance analysis. During performance analysis, a system is checked for its operational availability. Operational availability is computed based on data and functional availability from system simulation using the SSM concept. This paradigm shift is shown in Figure 4.3.

Figure 4.3 shows other system analysis techniques adapted in the systems engineering process - data analysis, graphical presentation, and system simulation. System reliability and operational availability are computed by means of system data. FTA, directed graph, and state flow for the system are graphically presented. The system functionality of a system is performed by means of simulation.

4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION

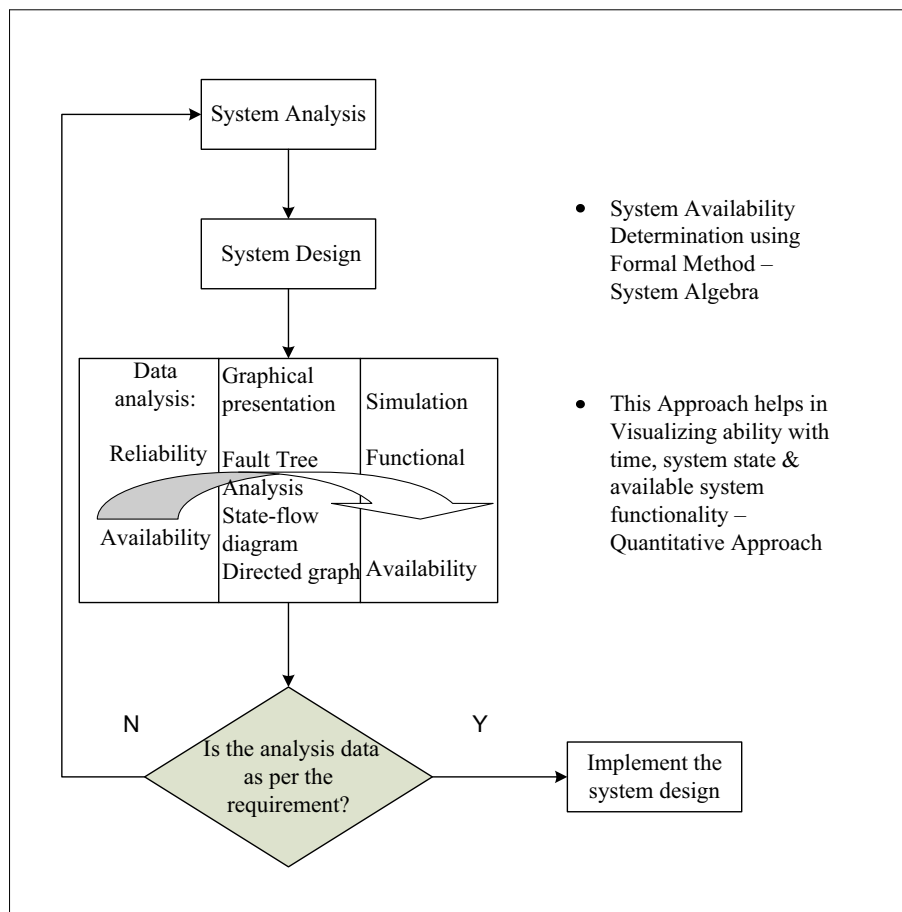


Figure 4.3: Paradigm Shift of Availability Analysis Based on Data to Based on Simulation

4.2 Proposed Novel Modified Systems Engineering

An engineering process adapts process framework comprises of various tool and techniques in different phases of the engineering life cycle. The engineering process can adopt a single tool, multiple tools or an integrated tool-set consisting of formal and informal tools. The SE process is shown in Figure 4.4.

In Figure 4.4, the engineering process comprises requirement, design, implementation, test, and deployment phases. The requirement phase captures the system requirements for the desired functionality and performance. The design phase follows the requirements phase. In the design phase, the system requirements are verified and validated. The design analyzes system properties through simulation (model analysis) or design reviews. The implementation phase follows the design phase and in this phase the system design is implemented. The implementation can be done automatically or manu-

4.2 Proposed Novel Modified Systems Engineering

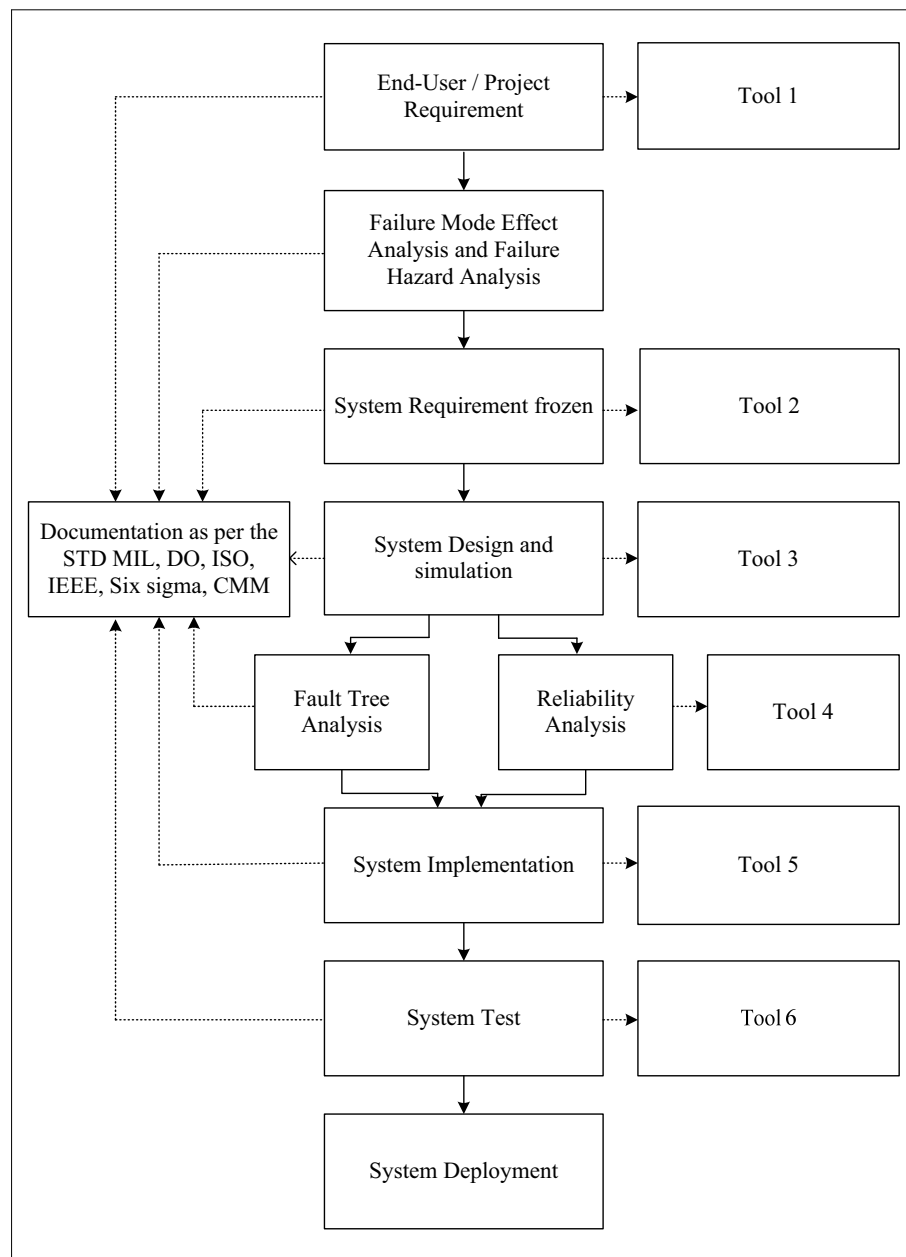


Figure 4.4: Systems Engineering Lifecycle Process

ally. In the testing phase the system design is tested for system functionality, performance, and safety against requirements. Observations from the design, implementation, and test phases are used as input for improving the design or modifying requirements. Examples of tools/techniques that are used for systems engineering are: DOORS for systems requirement capture; Clearquest for change management; SysML or Matlab/Simulink or

4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION

Rhapsody toolsets for design simulation to verify functionality at the system and/or software level. FTA for safety analysis and RBD for reliability analysis are examples of techniques employed for such analyses. We *et al.* [132], at CSIR-NAL, have justified the importance of tools used in various phases of the engineering process. The metrics act as guidelines in selecting the appropriate tool for improving software performance. System testing is performed at the software and then at the embedded level. The tests carried out at the software level, SIL (software-in-the-loop) tests; at the hardware level, PIL (processor-in-the-loop) tests; and at the system level, HIL (hardware-in-the-loop) tests. SIL and PIL can be performed using Liverpool development research association (LDRA) and Rational test real time (RTRT) tools. HIL is performed using a test rig. The design feedback can be provided from any phase (i.e. design, implementation, test and/or deployment). More effort is needed to incorporate feedback if it comes from later phases, such as during the implementation, test or deployment phases, than an earlier phase, such as the design phase. In order to minimize the discovery of design faults at later phases, the engineering processes are being equipped with effective tools for earlier phases. This leads to an evolution of the process. Documents generated as artifacts for certification in each of the engineering phases as per industry standard guidelines are, shown by dotted arrows in the figure.

The Systems Engineering Framework is well-established and like most safety standards, RTCA DO-178B defines a framework for the development process, rather than mandating a particular process or process model. RTCA DO-178C, provides the guidelines for airborne software, is next revision of RTCA DO-178B [34]. RTCA DO-178C defines framework with recommendations to use advanced techniques like formal methods, object-oriented programming and model-based design-development. These are recommended to empower the designers so that they can provide set of evidences to ensure that the system is adequately safe for a given application and operational period of time. There is always a need of novel techniques to assess the functionality, safety, security, performance, and reliability of the system. Joshi *et al.* [133], Heitmeyer [134], and Leveson and Heimdahl [135] have also worked in improving the engineering process by introduction of formal method based techniques. Joshi *et al.* [133] propose model-based safety approach to overall increase the system safety and reduce the turnaround time. This is done by performing safety analysis on extended system model which comprises of digital components (software and hardware), and mechanical components (pumps, valves, etc.). The behavior of this extended system is analyzed by incorporating fault models. These fault models provide the proof of safety properties for the extended system. Wheel brake system is taken as an example to demonstrate the effectiveness of the proposed approach. Verification of the system fault tolerance is done using NuSMV model-checker

4.2 Proposed Novel Modified Systems Engineering

and the fault tree is generated using the PVS theorem prover. Efforts are done to use these results as a certification artifact. Heitmeyer [134] discusses formal method based approach to capture system software requirements. SCR (Software Cost Reduction) formal method is used to capture the system behavior requirements precisely and unambiguously. Safety injection system is taken as an example to demonstrate the requirements capture using SCR. Heitmyer proposes an integrated SCR toolset which includes tools from requirements capture to code generation. The toolset comprises of specification editor for creating and modifying a requirements specification, consistency checker for checking the specification for well-formedness, simulator for symbolically executing the system based on the specification, model checker for analyzing the specification for application properties, dependency graph browser for displaying variable dependencies, TAME front-end to PVS, an invariant generator, a property checker Salsa, a test case generator, and a source code generator. Leveson and Heimdahl [135] describe the formal semantics of RSML specification language (Requirements State Machine Language) and propose an automated approach to analyze safety critical systems for requirements consistency and design robustness. SpecTRM-RL, which is a experimental tool is used to specify the requirements for a control system. The formal specification language is a SpecTRM-Requirements Language. The observations of the research work were to improve the specification languages used for design highly safety critical control laws of the aircrafts.

My research work introduces the SA based design analysis technique in the system design phase. The introduction into design phase is arrived by data analysis of safety critical systems designed and developed in-house. These systems are Smart fatigue Meter, Engine indication and crew alerting system, Stall warning and aircraft interface system and Automatic flight control system. We have introduced SA in the design phase of engineering process, modifying the process as shown in Figure 4.5.

The highlighted block in Figure 4.5 is the induction of the SA technique for system availability analysis in the design phase of the engineering process. The effectiveness and applicability of the SA technique has been discussed in the previous chapter with forward and reverse engineering examples. As seen in the Figure 4.5, the introduction of SA refines the design phase by detecting design flaws earlier as it is analyzed for available functions under various normal and robust scenarios. This will improve the engineering process. The SA technique complements other analysis techniques, such as FTA and RBD as discussed earlier. The technique can be inducted in the design phase of any other engineering process discussed by Nanda *et al.* [132]in their work.

The modified system design phase of the engineering process analyzes a system for functional availability in addition to reliability, functionality, and the safety against project

4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION

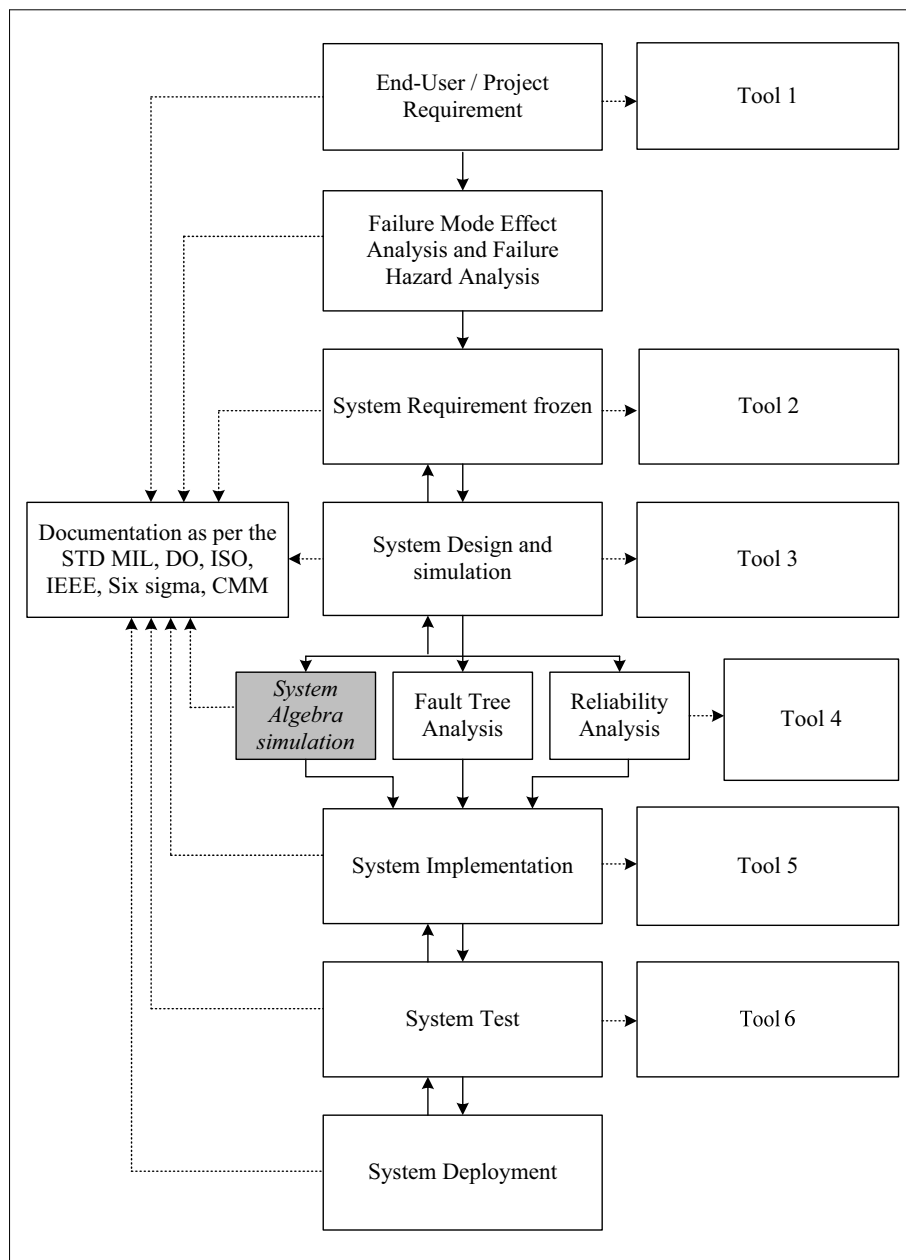


Figure 4.5: Modified Systems Engineering Life Cycle Process

requirements. This proposed process results in a faster time-to-market, prediction of system availability, reliability, functionality, security, reachability, safety, and maintainability.

The proposed technique is validated by simulating design analysis of critical systems like conventional and de-centralized clinical laboratory system, fault-tolerant architecture

4.2 Proposed Novel Modified Systems Engineering

for computer-based railway vehicle brake system, distributed fault tolerant architecture for nuclear reactor control, and the Jaguar flight control system. The availability analysis of the clinical laboratory system enabled the comparison of two different system designs. The proposed technique helped in selecting the better design with maximized availability and reduced underlying complexity. The reliability and safety were also analyzed ensuring a modular, safe, reliable system that is available for the desired time period and functionality. The design simulation helped in analyzing system availability and its complexity for new systems, like Crack Detection and Warning System (CDWS) and Fatigue Meter, and is currently being used for the system design simulation and analysis.

4.2.1 SA Analysis in the Design Phase of the Systems Engineering Lifecycle: Forward Engineering

The applicability of SA to analyze the design of the system is conducted for a new project during the design phase. The project requirements provide functional, operational, performance, safety-related, security-related, maintenance, and constraint requirements for the system to be designed. For a given requirement one can have different designs. To select the most appropriate design, according to the requirement, the techniques employed need to produce data to demonstrate the design for its intended requirements. The SA approach helps in analyzing the functional availability of the system to verify design against functional requirements.

4.2.1.1 Selection of Appropriate eFM Design Based on Functional Availability

The enhanced fatigue meter is developed by CSIR-NAL and the proposed design is discussed by Jayanthi *et al.* [136]. This example is considered again as we had to select the appropriate design in accordance to project functionality, safety, availability, cost, and time requirements.

Acceleration due to gravity or 'g' has an effect on an aircraft structure. Cyclic loads are 'g' variations the structure can withstand before entering the fatigue phase. Every aircraft structure is designed to withstand a specified 'g' cyclic load. The fatigue phase causes cracks leading to structural damage, a catastrophic phase for the aircraft. Thus, an un-monitored 'g' variation reduces the life span of an aircraft. Fatigue meter is an avionic system that detects 'g' crossings during a flight. These crossings are counted during every flight. The maintenance crew monitor these 'g' crossings and, if necessary, increase aircraft maintenance to retain the life span of the aircraft.

eFM computes the 'g' crossings of the aircraft like any other fatigue meter; in addition, it also computes peaks and troughs of the 'g' cycle that the aircraft was subjected to

4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION

during the flight. The peak and trough information, with the time of ‘g’ crossing, helps in prognostic analysis of the aircraft structure, such as fatigue index and prediction of structural failure. This critical feature provides the ‘enhanced’ adjective to the fatigue meter being developed. The enhanced fatigue meter, consists of an accelerometer that senses ‘g’ acceleration, signal conditioning, processing unit, keypad interface, and a display unit. For this purpose, two system designs were considered. Each design was to perform the system functionality and provide functional availability for a longer time.

Design I : The proposed design is shown in Figure 4.6.

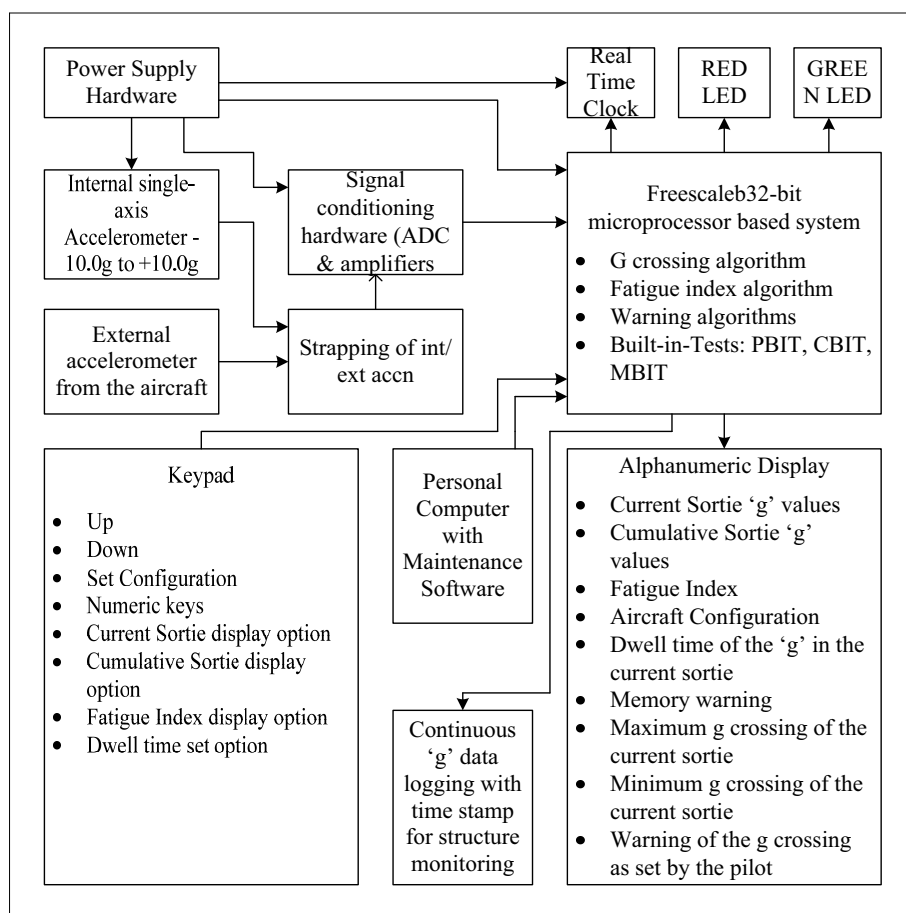


Figure 4.6: Block Diagram of Enhanced Fatigue Meter - Design I

As shown in Figure 4.6, control for the eFM functionality flows from sensor to signal conditioning, signal processing, and displaying the information on request. The system is divided into functional sub-systems, i.e., accelerometer, power supply, signal conditioner, signal processing, keyboard, display, and real-time sub-system. Each of these systems is simplex, i.e., there are no redundancies in any of the sub-systems and the control flow

4.2 Proposed Novel Modified Systems Engineering

is sequential. Hence, these sub-systems are communicating with one another by ‘direct sum’ mathematical operator. This is shown in equation 4.2.1.

$$S = (PS + ACCN + SIGCOND + CPU + KBD + DISP + RTC) \quad (4.2.1)$$

where,

PS:	Power supply sub-system
ACCN:	Accelerometer assembly sub-system
SIGCOND:	Signal conditioning sub-system
CPU:	Processing sub-system
KBD:	Keyboard sub-system
DISP:	Display sub-system
RTC:	Real time sub-system

As there are no redundancies in any of the sub-systems, components with low failure rates of the order of 10^{-3} or less are selected. Low failure-rate components increase system cost more as these components are expensive. So, there is a trade-off between functional availability, complex design and cost. The analysis of the functional availability at an interval of T which corresponds to system event used for system simulation, is shown in Table 4.3.

Table 4.3 discusses the sequence of events causing the system state to transit from safe to unsafe state. The event is a function of time and is represented as ‘T0’ to ‘T22’. The functionality available at every event is mentioned. This is one possible combination of sequential events. There can be other combinations of sequential events. The table shows the event number, failure of the sub-system or sub-systems and the status of the functional availability.

Equations 4.2.2 and 4.2.3 show the best- and worst-case fault tolerances of the design as proposed by Rao [29]. The expression provides feedback on selection of low failure-rate components for high functional availability of the system over time, as required by the project.

$$\begin{aligned}
 \beta_{best} \left(\sum_{i=1}^7 S_i \right) &= \left(\sum_{i=1}^7 \beta_{best}(S_i) \right) \\
 &= \beta_{best}(PS + ACCN + SIGCOND + CPU + KBD + DISP + RTC) \\
 &= \beta_{best}(PS) + \beta_{best}(ACCN) + \beta_{best}(SIGCOND) + \beta_{best}(CPU) + \beta_{best}(KBD) \\
 &\quad + \beta_{best}(DISP) + \beta_{best}(RTC)
 \end{aligned} \quad (4.2.2)$$

4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION

Time	Component Failure	System State	Functionality	Availability
T0		No failure	Safe	Entire functionality
T1	1×10^{-3}	KBD failure	Hazardous	User-interface non-available
T2	1×10^{-3}	DISP failure	Hazardous	User-interface non-available
T3	1×10^{-3}	RTC failure	Hazardous	No time stamp functionality
T4	1×10^{-3}	CPU failure	Unsafe	System non-functional
T5	3×10^{-5}	Accelerometer failure	Unsafe	No real data
T6	1×10^{-3}	SIGCOND failure	Unsafe	No signal conditioning functionality
T7	1×10^{-3}	Power supply failure	Unsafe	No system operational
T8		KBD and DISP failure	Hazardous	No configuration and display functionality
T9		KBD and RTC failure	Hazardous	No configuration, time-stamp functionality
T10		KBD and CPU failure	Unsafe	System non-functional
T11		KBD and ACCN failure	Unsafe	System non-functional
T12		KBD and SIGCOND failure	Unsafe	System non-functional
T13		DISP and RTC failure	Hazardous	No display and time-stamp functionality
T14		DISP and CPU failure	Unsafe	System non-functional
T15		DISP and SIGCOND failure	Unsafe	System non-functional
T16		RTC and CPU failure	Unsafe	System non-functional
T17		RTC and ACCN failure	Unsafe	System non-functional
T18		RTC and SIGCOND failure	Unsafe	System non-functional
T19		KBD, DISP and RTC failure	Hazardous	No configuration, display and time stamp functionality
T20		KBD, DISP, RTC and CPU failure	Unsafe	System non-functional
T21		KBD, DISP, RTC and ACCN failure	Unsafe	System non-functional
T22		KBD, DISP, RTC and SIGCOND failure	Unsafe	System non-functional

Table 4.3: System State Analysis using the SA Expression

$$\begin{aligned}
 \beta_{worst} \left(\sum_{i=1}^7 S_i \right) &= \min(\beta_{worst}(S_i)) \\
 &= \beta_{worst}(\text{PS} + \text{ACCN} + \text{SIGCOND} + \text{CPU} + \text{KBD} + \text{DISP} + \text{RTC}) \\
 &= \min(\beta_{worst}(\text{PS}), \beta_{worst}(\text{ACCN}), \beta_{worst}(\text{SIGCOND}), \beta_{worst}(\text{CPU}), \\
 &\quad \beta_{worst}(\text{KBD}), \beta_{worst}(\text{DISP}), \beta_{worst}(\text{RTC}))
 \end{aligned} \tag{4.2.3}$$

Equation 4.2.4 provides feedback on the weakest link in design that affects the system's functional availability.

$$\text{Availability} = (\text{PS}, \text{ACCN}, \text{SIGCOND}, \text{CPU}, \text{KBD}, \text{DISP}, \text{RTC})_{Min} \tag{4.2.4}$$

Analysis of system design using the SA approach provides the pros and cons of the design for the project in line with functionality, cost, and schedule. Advantage of this design:

- **1:** The system is functionally available for a longer time, i.e., 10^9 hours.
- **2:** The component count is simple as there are no redundancies.
- **3:** The design is simple and compact.

The disadvantage of this design is that the selection of low failure-rate components increases the cost of the system.

Design II : Adding redundancy to these units. We provide double redundancy in power supply, accelerometer, signal conditioning and signal processing sub-systems. The proposed modified design is shown in Figure 4.7.

As seen in the Figure 4.7, redundancy is provided to the critical sub-systems: power supply, accelerometer, signal conditioner, and signal processor sub-systems. The control flow still has a sequential path, but in case of a failure of any sub-system, the redundant unit takes over without affecting functional availability. The SA expression for this system is shown in equation 4.2.5.

$$S = (\text{PS}^2 + \text{ACCN}^2 + \text{SIGCOND}^2 + \text{CPU}^2 + \text{KBD} + \text{DISP} + \text{RTC}) \tag{4.2.5}$$

where,

4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION

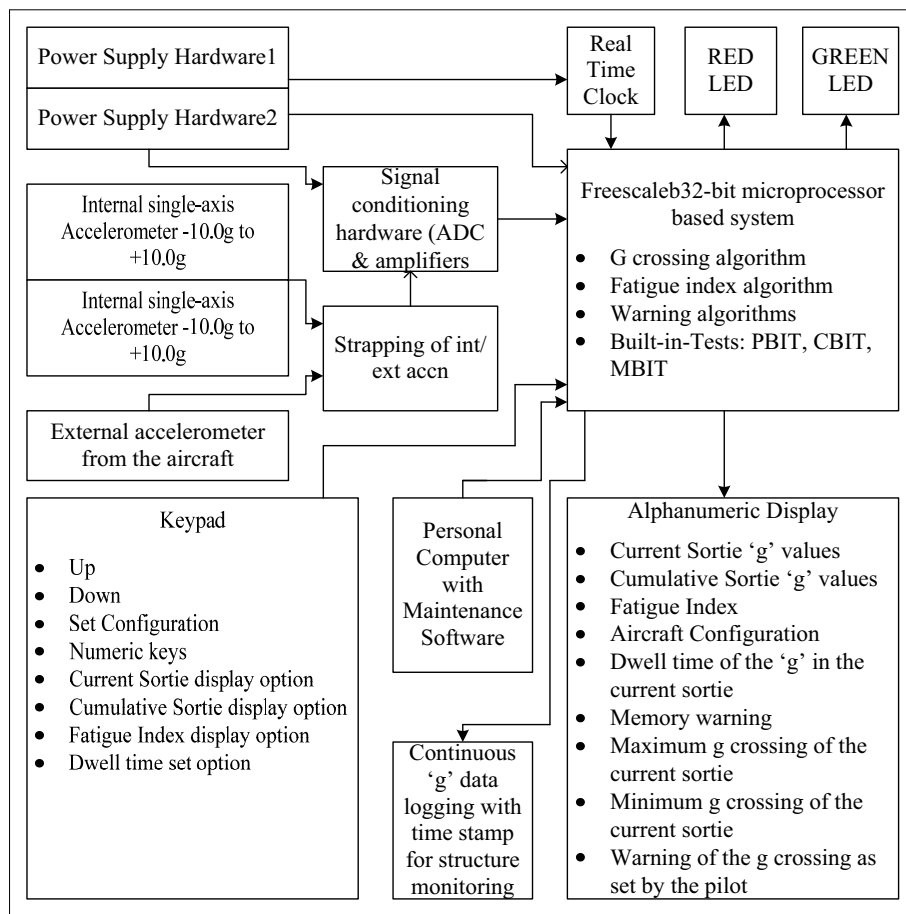


Figure 4.7: Block Diagram of Enhanced Fatigue Meter - Design II

- PS: Power supply sub-system
 ACCN: Accelerometer assembly sub-system
 SIGCOND: Signal conditioning sub-system
 CPU: Processing sub-system
 KBD: Keyboard sub-system
 DISP: Display sub-system
 RTC: Real time sub-system

Equations 4.2.6 and 4.2.7 show the best- and worst-case fault tolerances of the design. This expression provides feedback on selection of low failure-rate components for high

functional availability of the system over time, as required by the project.

$$\begin{aligned}
 \beta_{best} \left(\sum_{i=1}^7 S_i \right) &= \left(\sum_{i=0}^7 \beta_{best}(S_i) \right) \\
 &= \beta_{best}(\text{PS} + \text{ACCN} + \text{SIGCOND} + \text{CPU} + \text{KBD} + \text{DISP} + \text{RTC}) \\
 &= \beta_{best}(\text{PS}) + \beta_{best}(\text{ACCN}) + \beta_{best}(\text{SIGCOND}) + \beta_{best}(\text{CPU}) + \beta_{best}(\text{KBD}) \\
 &\quad + \beta_{best}(\text{DISP}) + \beta_{best}(\text{RTC})
 \end{aligned} \tag{4.2.6}$$

$$\begin{aligned}
 \beta_{worst} \left(\sum_{i=1}^7 S_i \right) &= \min(\beta_{worst}(S_i)) \\
 &= \beta_{worst}(\text{PS} + \text{ACCN} + \text{SIGCOND} + \text{CPU} + \text{KBD} + \text{DISP} + \text{RTC}) \\
 &= \min(\beta_{worst}(\text{PS}), \beta_{worst}(\text{ACCN}), \beta_{worst}(\text{SIGCOND}), \beta_{worst}(\text{CPU}), \\
 &\quad \beta_{worst}(\text{KBD}), \beta_{worst}(\text{DISP}), \beta_{worst}(\text{RTC}))
 \end{aligned} \tag{4.2.7}$$

Equation 4.2.8 provides feedback on the weakest link in design that affects the system's functional availability.

$$\text{Availability} = (\text{PS}, \text{ACCN}, \text{SIGCOND}, \text{CPU}, \text{KBD}, \text{DISP}, \text{RTC})_{Min} \tag{4.2.8}$$

Analysis of system design based on the SA approach provides pros and cons of the design for the project in line with functionality, cost, and schedule.

Advantage of the design is that the system is functionally available for a longer time 10^{18} hours, due to the increased redundancy of the critical sub-systems.

The disadvantage of the design is that it becomes complex and the size is no more compact.

Conclusion : 'Design I' was selected as the system design was simple, compact, and functionally available for 10^9 hours, per the project requirement.

4.2.1.2 Design of Crack Detection and Warning System (CDWS) Based on Functional Availability

The enhanced fatigue meter prevents crack propagation in an aircraft whereas CDWS detects a crack in a metallic structure like the engine of an aircraft; this is discussed

4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION

by Jayanthi and Nanda [137] in the conceptual design document. The CDWS system senses a crack with indigenously developed Giant magnetoresistive, GMR, sensors. The system consists of a probe that scans the metallic surface that is excited using a coil with a constant current source. This coil generates an eddy current over the scanned surface area. The GMR sensor picks up changes in the magnetic field pattern caused by cracks, if any. If a crack is found, a buzzer will beep, indicating the presence of a crack. The display shows visual information about the crack and its dimensions. A probe scans surface and induces eddy currents that will be sensed by the GMR sensor and processed to detect cracks. If a crack is found, its presence is transmitted to a centralized system. The block diagram of the proposed system is shown in Figure 4.8.

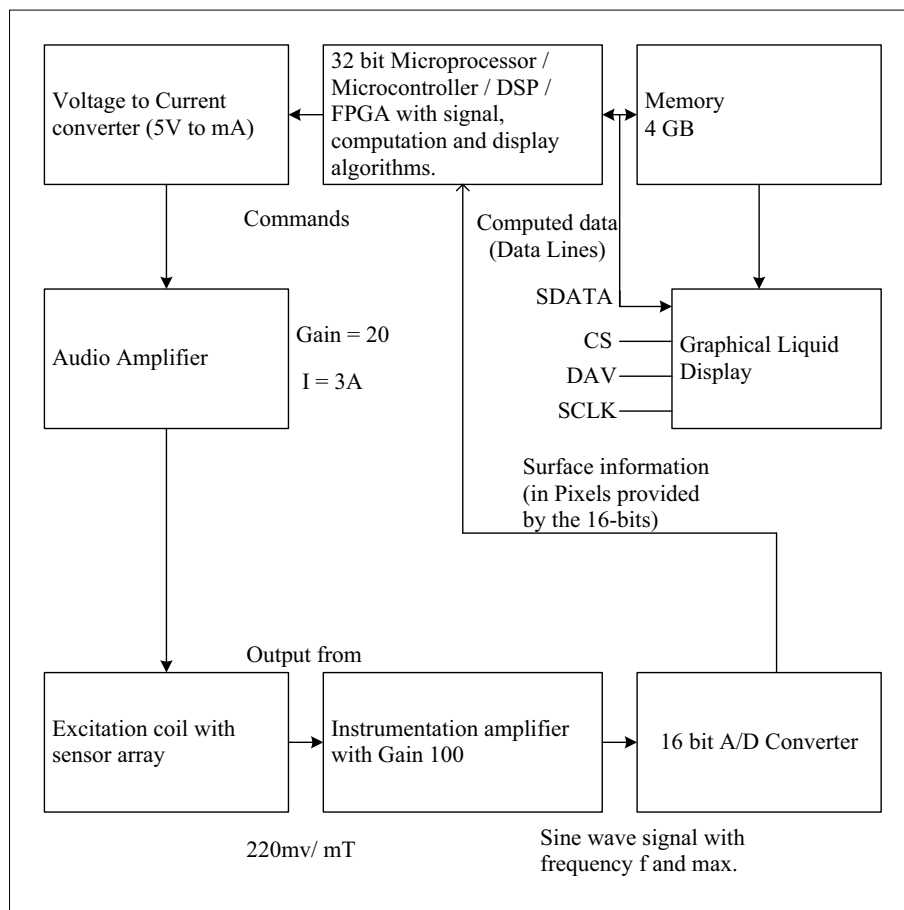


Figure 4.8: Block Diagram of Crack Detection and Warning System

As seen in Figure 4.8, CDWS control flows from sensor to signal conditioner, signal processor, and data logger to the display of the information. The system is functionally divided into sub-systems, i.e., processing unit, data-logging, data transmitter, current

4.2 Proposed Novel Modified Systems Engineering

source generator, sensor assembly, and power supply. All sub-systems are simplex, i.e., no redundancy is provided and control flow is sequential. Hence these sub-systems are communicating to one another by the ‘direct sum’ mathematical operator. The SA expression is shown in the equation 4.2.9.

$$S = ((\text{PROC} + \text{LOG} + \text{TRAN}) + \text{CRSRGEN} + \text{SENARR} + \text{PS}) \quad (4.2.9)$$

where,

PROC:	Processing unit
LOG:	Data logging
TRAN:	Data transmitter
CRSRGEN:	Current source generator
SENARR:	Sensor array
PS:	Power supply

The worst- and best-case fault tolerances of the CDWS system, as described in [29], are shown in equations 4.2.10 and 4.2.11.

$$\begin{aligned} \beta_{best} \left(\sum_{i=1}^5 S_i \right) &= \sum_{i=1}^5 \beta_{best}(S_i) \\ &= \beta_{best}(\text{PROC} + \text{LOG} + \text{TRAN} + \text{CRSRGEN} + \text{SENARR} + \text{PS}) \quad (4.2.10) \\ &= \beta_{best}(\text{PROC}) + \beta_{best}(\text{LOG}) + \beta_{best}(\text{TRAN}) + \beta_{best}(\text{CRSRGEN}) \\ &\quad + \beta_{best}(\text{SENARR}) + \beta_{best}(\text{PS}) \end{aligned}$$

$$\begin{aligned} \beta_{worst} \left(\sum_{i=1}^5 S_i \right) &= \min(\beta_{worst}(S_i)) \\ &= \beta_{worst}(\text{PROC} + \text{LOG} + \text{TRAN} + \text{CRSRGEN} + \text{SENARR} + \text{PS}) \\ &= \min(\beta_{worst}(\text{PROC}), \beta_{worst}(\text{LOG}), \beta_{worst}(\text{TRAN}), \beta_{worst}(\text{CRSRGEN}), \\ &\quad \beta_{worst}(\text{SENARR}), \beta_{worst}(\text{PS})) \quad (4.2.11) \end{aligned}$$

The system’s functional availability depends on the minimum availability of power supply, processing unit, current source generator, or sensor array. Failure of any of these

4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION

makes the system unavailable. Transition of system state from safe to unsafe under various failure scenarios is shown in equation 4.2.12. The table provides feedback on the weakest link for design that affects the functional availability of the system.

$$Availability = (PS, PROC, LOG, TRAN, CRSRGEN, SENARR)_{Min} \quad (4.2.12)$$

The CDWS system is a simple system with no redundancies in the critical failure path. The sequential path is the critical path of the system. The sequential path provides a single point failure, causing a transition from a safe to unsafe state without transiting through the hazardous state. A redundant path will enable the system to transit to the hazardous state, increasing functional availability for a longer time. After this analysis, the current design was retained as this is an offline system, and a simpler design is preferred.

The possible paths traversed by the CDWS from a safe to failed state are mentioned in Table 4.4. These transitions are determined from the tree diagram generated for the system from its SA expression. The tree diagram documents the paths traversed, as shown in the Figure 4.9.

<i>Safestate</i> →	1 → 3 → 8 →	<i>Failedstate</i>
<i>Safestate</i> →	2 → 4 →	<i>Failedstate</i>
<i>Safestate</i> →	2 → 4 → 8 →	<i>Failedstate</i>
<i>Safestate</i> →	2 → 4 → 6 →	<i>Failedstate</i>
<i>Safestate</i> →	2 → 4 → 6 → 9 →	<i>Failedstate</i>
<i>Safestate</i> →	2 → 4 → 6 → 12 →	<i>Failedstate</i>
<i>Safestate</i> →	2 → 4 → 6 → 12 → 10 →	<i>Failedstate</i>
<i>Safestate</i> →	2 → 4 → 6 → 7 →	<i>Failedstate</i>
<i>Safestate</i> →	2 → 4 → 6 → 7 → 11 →	<i>Failedstate</i>
<i>Safestate</i> →	1 → 3 → 13 →	<i>Failedstate</i>
<i>Safestate</i> →	1 → 3 → 13 → 14 →	<i>Failedstate</i>
<i>Safestate</i> →	1 → 3 → 13 → 14 → 15 →	<i>Failedstate</i>
<i>Safestate</i> →	1 → 3 → 13 → 14 → 15 → 16 →	<i>Failedstate</i>

Table 4.4: System State Transitions for CDWS

4.2 Proposed Novel Modified Systems Engineering

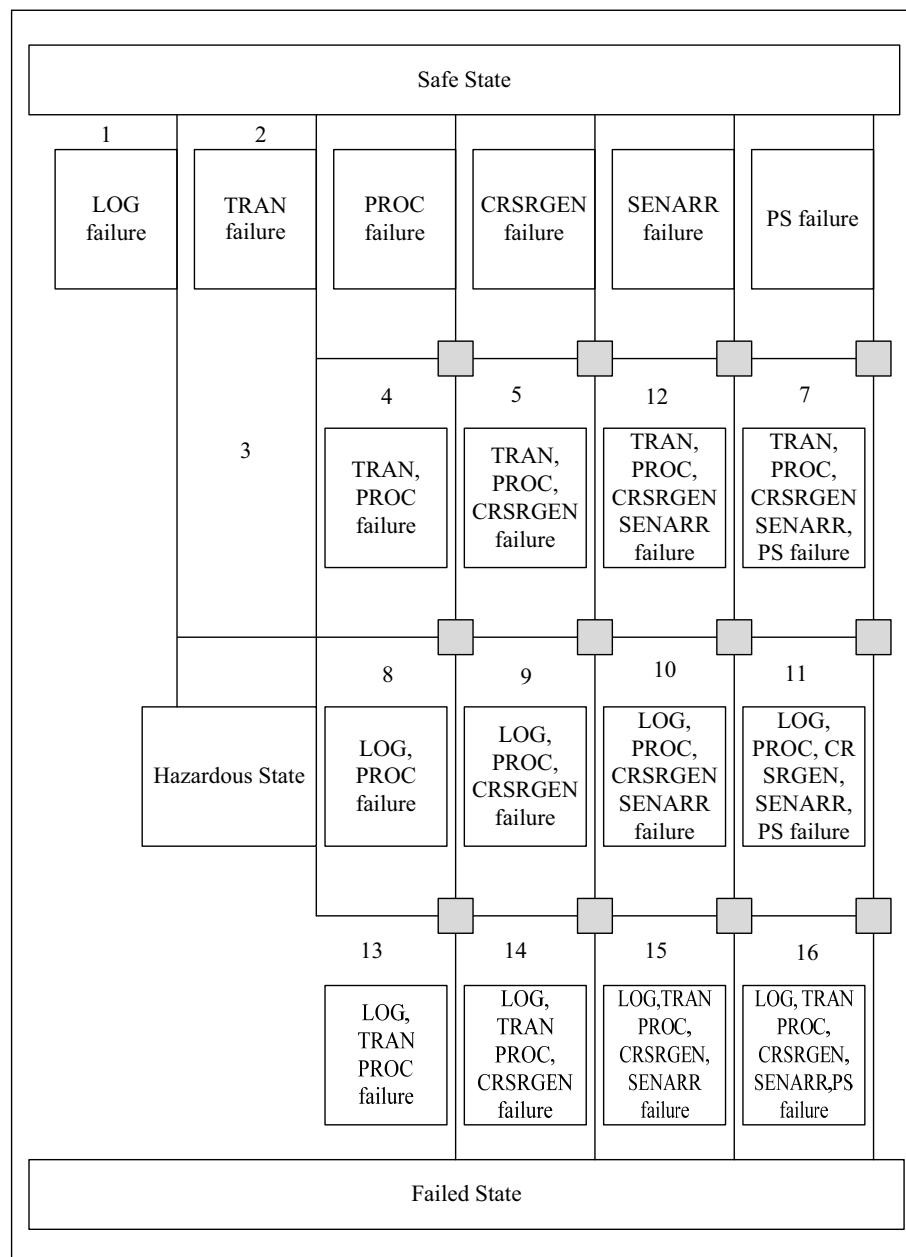


Figure 4.9: Tree diagram of CDWS

For higher functional availability, either redundancy for critical units can be provided or lower failure-rate components can be used. Similar to enhanced fatigue meter lower failure-rate components are used.

eFM and the CDWS are off-line equipments that are not used during a flight. These systems are for maintenance of an aircraft structure. The functional availability require-

4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION

ment of these equipments is less than that of critical aircraft systems providing in-flight information such as flight control system, navigation unit, surveillance system, head-up display, electronic flight bag, and display to a pilot. The functional availability requirement for these safety-critical systems is more than that of eFM and CDWS and hence these designs have low redundancy in their design, but low failure rates of components.

4.2.2 SA Analysis in the Design Phase of Systems Engineering Lifecycle: Reverse Engineering

In this approach system, analysis is done on an already proven system. The reverse-engineering approach is used to understand system requirements and then analyze its behavior from system states. Simulation validates system behavior and its functional availability. This approach helps in establishing the correctness of this approach for system analysis and its value addition to existing techniques for system design.

Case 1 : Aerospace Domain, Airbus A380 FCS

Airbus A380 flight control system (FCS) can be analyzed by modeling it using the SA approach. The SA expression for Airbus A380 is derived in equation 4.2.13. The FCS of the Airbus A380 consists of input sensors, flight control computer and actuators. The signal flows from input sensors to control computer and finally to the actuator stage. The input sensors have a quadruple redundancy, and the computational stage has dual redundancy - an active command/monitor and a standby command/monitor. The actuators, driving the aircraft surfaces, have a duplex redundancy. The SA expression of each stage is derived and the system polynomial is computed by combining those for input, computational, and actuator stages. The SA expression is shown in equation 4.2.13.

$$\begin{aligned} System &= InputStage + ComputationalStage + ActuatorStage \\ &= (I_k^4 + O_{CM}^2 + O_S^2 + A^2) \end{aligned} \quad (4.2.13)$$

where,

- I_k : Inputs to the A380 system that define the input stage
- O_{CM}, O_S : Command/ Monitor and standby system that define the computational stage
- A^2 : Actuator redundancy due to hydraulic and electrical drives that define the actuator stage

4.2 Proposed Novel Modified Systems Engineering

Table 4.5 discusses the sequence of events, causing a state transition from safe to unsafe over time. The table shows the event number, failure, system state and functionality available at that state based on failures. The failure of a sub-system or sub-systems is considered at input and/or computational stage and/or output.

The aerospace standard specifies the failure rate of sensors to be 10^{-5} hours, that of flight computer as 10^{-4} hours, and actuator as 10^{-4} hours. Using these failure rates and redundancies, the input stage failure can be calculated as 10^{-15} , i.e., once in 10^{15} hours. Similarly, a flight computer failure occurs once in 10^8 hours and an actuator failure occurs once in 10^8 hours. The flight computer has a standby redundancy of 10^8 hours, hence has a failure once in 10^{16} hours. The sequence of events shows the transition of the system from a safe to unsafe state. The system enters the hazardous state when the actuators fail, i.e. once in 10^8 hours, or 11415 years - much more than the life of an aircraft.

Operational availability uses a formula to compute system failure from the operational stage to the maintenance stage. The availability determined by this method does not provide the sequential flow of the system functionality from safe to unsafe state. It is only after the unsafe state that the system goes to the maintenance phase.

Case 2: Medical Domain, Clinical Chemistry Analyzer

A clinical laboratory system is used for analyzing blood samples. It tests a blood sample to diagnose illnesses such as jaundice, hemoglobin, bilirubin, etc., and produce a report for doctors, as discussed by Mitsumaki *et al.* [80]. There are different test options, and a doctor decides tests to be run depending on a patient's condition. The system must report accurate results to help a doctor in her diagnosis. System design analysis models the system by breaking it into simple series and parallel combinations. The SA expression for the system, thus modeled, is provided in the algorithm and the system block diagram is shown in Figure 4.10.

4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION

Time	Component Failure	Input stage state	Computational stage state	Output stage state	Functional Availability
T0	Input	Safe	Safe	Safe	Safe, Complete FCS functionality available
T1	Input	Safe	Safe	Safe	Safe, Complete FCS functionality available
T2	Input	Safe	Safe	Safe	Safe, Complete FCS functionality available
T3	Input	Safe	Safe	Safe	Safe, Complete FCS functionality available
T4	Input	Safe	Safe	Safe	Safe, Complete FCS functionality available
T5	Input	Hazardous	Safe	Safe	Hazardous, Degraded functionality available
T6	Input	Hazardous	Safe	Safe	Hazardous, Degraded functionality available
T7	Input	Hazardous	Safe	Safe	Hazardous, Degraded functionality available
T8	Input	Hazardous	Safe	Safe	Hazardous, Degraded functionality available
T9	Computational	Safe	Safe	Safe	Hazardous, Degraded functionality available
T10	Computational	Safe	Safe	Safe	Safe, Complete FCS functionality available
T11	Computational	Safe	Safe	Safe	Safe, Complete FCS functionality available
T12	Computational	Safe	Safe	Safe	Safe, Complete FCS functionality available
T13	Computational	Safe	Safe	Safe	Safe, Complete FCS functionality available
T14	Computational	Safe	Hazardous	Safe	Safe, Complete FCS functionality available
T15	Computational	Safe	Hazardous	Safe	Safe, Complete FCS functionality available
T16	Computational	Safe	Unsafe	Safe	Safe, Complete FCS functionality available
T17	Actuator	Safe	Safe	Safe	Safe, Complete FCS functionality available
T18	Actuator	Safe	Safe	Safe	Safe, Complete FCS functionality available
T19	Actuator	Safe	Safe	Safe	Safe, Complete FCS functionality available
T20	Actuator	Safe	Safe	Safe	Safe, Complete FCS functionality available
T21	Actuator	Safe	Safe	Hazardous	Hazardous, Degraded functionality available
T22	Actuator	Safe	Safe	Hazardous	Hazardous, Degraded functionality available
T23	Actuator	Safe	Safe	Unsafe	Unsafe, Functionality unavailable

Table 4.5: System State Analysis using the SA Expression

4.2 Proposed Novel Modified Systems Engineering

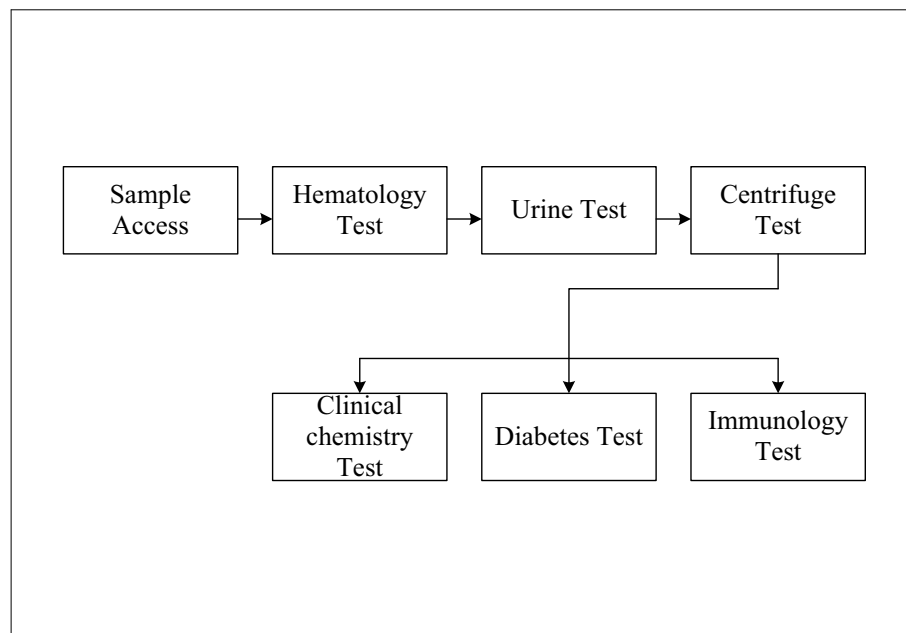


Figure 4.10: Block Diagram of Conventional Clinical Laboratory System

The SA expression derived for the system is shown in the equation 4.2.14.

$$\begin{aligned} S = & [(SA + HT + UT + CT) * CCT] \\ & + [(SA + HT + UT + CT) * DT] \\ & + [(SA + HT + UT + CT) * IMT] \end{aligned} \quad (4.2.14)$$

where,

- SA: Sample access
- HT: Hematology test
- UT: Urine test
- CT: Centrifuge test
- CCT: Clinical chemistry test
- DT: Diabetes test
- IMT: Immunology test

As seen in Figure 4.10, a clinical laboratory system consists of clinical chemistry test, diabetes test, and immunology test. For each test, blood/urine sample passes through the same process of sample access, hematology, urine, and centrifuge tests. Since the process is sequential for chemistry, diabetes, and immunology each test, is depicted by 'direct

4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION

sum' in SA expression. There are no redundancies, hence no 'direct product' operator is used in the expression.

The SA expression shows that the system is a sequential system design and a failure at *HT*, *UT*, or the *CT* stage will transit the system from a safe to unsafe state. On the other hand, if the failure is at *CCT*, *DT*, or *IMT* the system transits from a safe to hazardous to unsafe state.

The worst- and the best-case fault tolerances of the system, are shown in equations 4.2.15 and 4.2.16.

$$\begin{aligned}
 \beta_{best} \left(\sum_{i=1}^7 S_i \right) &= \sum_{i=1}^7 \beta_{best}(S_i) \\
 &= \beta_{best}([(SA + HT + UT) + CT] * CCT) + [(SA + HT + UT + CT) * DT] \\
 &\quad + [(SA + HT + UT) + CT] * IMT) \\
 &= \beta_{best}(((SA + HT + UT) + CT) * CCT) \\
 &\quad + \beta_{best}[(SA + HT + UT + CT) * DT] \\
 &\quad + \beta_{best}[(SA + HT + UT) + CT] * IMT)
 \end{aligned} \tag{4.2.15}$$

$$\begin{aligned}
 \beta_{worst} \left(\sum_{i=1}^7 S_i \right) &= \sum_{i=1}^7 \beta_{worst}(S_i) \\
 &= \beta_{worst}(((SA + HT + UT) + CT) * CCT) + ((SA + HT + UT + CT) * DT) \\
 &\quad + ((SA + HT + UT) + CT) * IMT)) \\
 &= [\beta_{worst}([(SA, HT, UT), CT], CCT) \\
 &\quad \beta_{worst}[(SA, HT, UT, CT], DT) \\
 &\quad \beta_{worst}([(SA, HT, UT), CT], IMT)]_{Min}
 \end{aligned} \tag{4.2.16}$$

If a sub-system in the sequential chain becomes non-functional, the system moves to an unsafe state directly from a safe state. For example, failure of sample access, hematology test, urine test, or centrifuge test makes the system unavailable, as the processed sample will not reach chemistry, diabetes, or immunology test stages.

The worst- and best-case fault tolerances of the system are shown in equations 4.2.17

and 4.2.18.

$$\begin{aligned}
 S_{best} &= [(SA + HT + UT + CT) * (CCT)]_{best} \\
 &= [(SA + HT + UT + CT)_{best} + (CCT)_{best} + 1] \\
 &= (SA)_{best} + (HT)_{best} + (UT)_{best} + (CT)_{best} + (CCT)_{best} + 1 \\
 &= (0 + 0 + 0 + 0) + (0) + 1 \\
 &= (0, 0, 0, 0, 0, 1)_{best} \\
 &= 1
 \end{aligned} \tag{4.2.17}$$

$$\begin{aligned}
 S_{worst} &= [(SA + HT + UT + CT) * (CCT)]_{worst} \\
 &= (SA + HT + UT + CT)_{worst} + (CCT)_{worst} + 1 \\
 &= (SA)_{worst} + (HT)_{worst} + (UT)_{worst} + (CT)_{worst} + (CCT)_{worst} + 1 \\
 &= (0 + 0 + 0 + 0) + (0) + 1 \\
 &= (0, 0, 0, 0, 0, 1)_{worst} \\
 &= 0
 \end{aligned} \tag{4.2.18}$$

The best fault tolerance for the system occurs when the system transits from a safe to hazardous state, depicted by 1. The worst fault tolerance for the system occurs when the system transits from a safe to safe state, depicted by 0.

The system availability is dependent on availability of the sub-systems in the sequential chain that, in turn, depend on failure rates of these sub-systems. The functional availability expression for the system is shown in equation 4.2.19.

$$Availability = (SA, HT, UT)_{Min} \tag{4.2.19}$$

The expression shows that the minimum availability of the units SA , HT or UT in the system determines the availability of the system, as a failure in any of them causes a transition from the safe state to unsafe state. The simplicity of the SA expression for clinical system shows a system with no redundancies.

On the other hand, an autonomous de-centralized clinical laboratory system has similar functionality, but a different system design. The system design analysis models the system by decomposing it into simple series and parallel combinations. The block diagram of the system is shown in Figure 4.11.

4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION

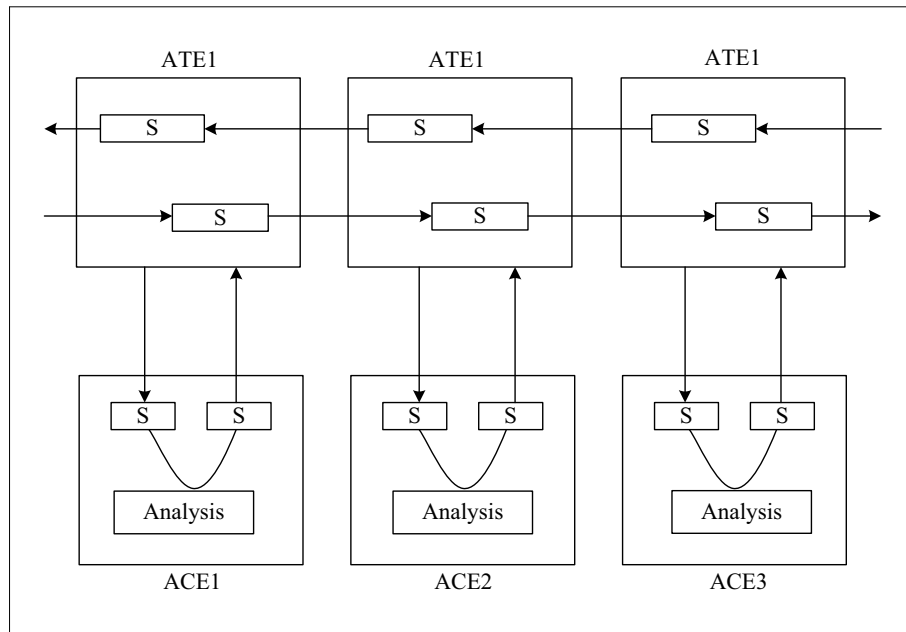


Figure 4.11: Block Diagram of Autonomous De-centralized Clinical Laboratory System

The system described by Mitsumaki *et al.* [80] consists of clinical equipment and small divided conveyers. These units are called autonomous clinical equipment (ACE) and autonomous transfer equipment (ATE). The ATE's are conveyers for transferring samples, and ACE's transfer as well as test them, as shown in Figure 4.11. The system design is modular and they autonomously decide where to transfer the samples. ATE's and ACE's form the autonomous equipment. The sub-systems are unaware of the entire system structure; they are only aware of their immediate neighbors. Each sub-system has a controller to transfer the samples. This controller is connected to a network to pass the samples across the belt. The SA expression for the system is analyzed for its state and its transitions based on the SSM concept. The fault tolerance of the system in the best and worst cases is analyzed as described earlier for conventional clinical laboratory.

The SA expression is shown in equation 4.2.20.

$$\begin{aligned}
 S &= ([ATE_{i-1} + ACE_{i-1}] + [ATE_i + ACE_i][ATE_{i+1} + ACE_{i+1}]) \\
 &= (ATE_i + ACE_i)
 \end{aligned}
 \tag{4.2.20}$$

where,

ATE1 = ATE2 = ATE3: Autonomous Transfer Equipment

ACE1 = ACE2 = ACE3: Autonomous Clinical Equipment

Each unit in the system performs a specific test and the only dependency is on the nearest neighbor. The system is thus modular where the failure of one module does not affect

4.2 Proposed Novel Modified Systems Engineering

the functionality of other modules. The single point failure is the conveyer belt and the transition from a safe to hazardous unsafe state is based on the component failure rates. The best- and worst-case fault tolerances of the system are shown in equations 4.2.21 and 4.2.22.

$$\begin{aligned}
 \beta_{best} \left(\sum_{i=1}^n S_i \right) &= \sum_{i=1}^n \beta_{best}(S_i) \\
 &= \beta_{best} (ATE_i * ACE_i) \\
 &= \beta_{best} (ATE_i) * \beta_{best}(ACE_i)
 \end{aligned}
 \tag{4.2.21}$$

$$\begin{aligned}
 \beta_{worst} \left(\sum_{i=1}^n S_i \right) &= \sum_{i=1}^n \beta_{worst}(S_i) \\
 &= \beta_{worst} (ATE_i * ACE_i) \\
 &= \beta_{worst} (ATE_i, ACE_i)_{Min}
 \end{aligned}
 \tag{4.2.22}$$

The availability expression for the system is given in equation 4.2.23.

$$Availability = (ATE, ACE)_{Min}
 \tag{4.2.23}$$

The comparison of conventional and autonomous de-centralized clinical laboratory systems prove the effectiveness of SA in analyzing the functional availability property, per requirements. As seen in the design analysis, the fault tolerance computation of two systems for the best-case and worst-case is same, but the expression for availability is different. In the conventional design, a single failure of any of the sub-systems (*SA, HT, UT and CT*) causes the system to transit to an unsafe state rather than hazardous state. In the de-centralized design, a single failure of any of the sub-systems ($[ATE_i - 1, ACE_i - 1]_{Min}$ or $[ATE_1, ACE_1]_{Min}$ or $[ATE_i + 1, ACE_i + 1]_{Min}$) causes the system to transit to a hazardous state; only the failure of all the redundant sub-systems causes the system to transit to an unsafe state. Hence, the functional availability of the de-centralized system is higher for the same failure rates.

4.2.3 Measure of Performance and Effectiveness for the Modified Engineering Process

The introduction of SA modifies the existing model-based engineering process. The SA technique analyzes design against functional requirements. The technique models a system and derives the SA expression. The analysis of system functionality with sub-system

4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION

Attribute	Metric
Abstraction	Measures the understandability and representability of a design
Modularity Complexity	Breaks the design into modules C+V; C: no. of comparisons in the module, V: no. of control variables in the module
Modifiability	Measures the changeability of the design
Stability	Ability of the system to not hang, not lose data, not disrupt system functionality, and be predictable
Sufficiency	Ensures that the design captures all the important characteristics of abstraction
Safety	Safety metric is attributed to the testability and modularity towards fault tolerance of a system
Survivability	Related to availability, reliability , reachability and maintainability
Integrity	Data integrity , safety of the design
Confidentiality	Ability of design to retain important logics without external attack

Table 4.6: Attributes to Compute in MOP in Design Phase

failures provides functional availability of the system over time. The verification of safety-critical systems proves the applicability of SA, but its effect on engineering process performance is determined by Measure of Performance (MOP). Performance is defined as the ability to do something. The technical document by INCOSE [138] defines MOP as a characterization of physical and functional attributes relating to system operation, i.e., it provides insight into performance of a specific system. Measure of Effectiveness (MOE) is the operational measure of success closely related to achievement of the mission.

We consider the analyzing capability of the technique for performance measurement using parameters such as abstraction, modularity, complexity, modifiability, stability, sufficiency, safety, maintainability, survivability, integrity, reliability, confidentiality, maturity, and cohesion. This is shown in Table 4.6.

The SA technique improves the stability, safety, and modifiability attributes of the design for the in-house systems discussed: SWS/AIC, eFM System and CDWS. The functional availability analysis, as discussed, provides effectiveness in analyzing these attributes. This improves the analyzability of the design. The analyzability of a system is defined as the effort required to detect deficiencies and modify them, which is a function (if referring ONLY to modification) of stability and availability. For SWS/AIC system, the effectiveness of the analysis is improved by 20% in the design phase. This improves the overall MOP by a factor of 20%. Thus, with the addition of SA in the design, the MOE and MOP are improved as the process follows the concept of ‘correct by construction’ in the initial phase of the process.

The proposed modified engineering process provides a formal Functional Availability analysis in order to validate the system design against project requirements for functionality, safety and criticality. The introduction of this Functional Availability technique in

Application	Domain	Number of sub-systems
Jaguar FCS	Aerospace	18
Enhanced Smart fatigue meter	Aerospace	6
Crack detection and warning system	Aerospace and Automotive	6
Clinical Laboratory system	Medical	6
Steer-by-wire controller	Automotive	6
Railway Intelligent Transportation System Architecture	Railway	6
A distributed fault tolerant architecture for nuclear reactor control and safety functions	Nuclear	3
ATR72-600	Aerospace	30

Table 4.7: System State Analysis using the SA Expression

the design phase improves the overall confidence in the system design early on, thus improving the project time schedule and reducing cost leading while increasing the overall quality of the system.

4.3 Tool Development

The verification of SA for demonstrating functional availability of a system design was performed manually with forward and reverse engineering of safety critical systems. The modeling of the system was done manually and the analysis of the system state based on the SSM concept was done using the Matlab code designed for each sub-system. The Matlab code injected the failure events and the overall system state was determined based on the SA operational properties. The effectiveness of SA for system availability analysis was performed on systems with as few sub-systems as 3 to a highly complex system, like ATR72-600 aircraft, with around 30 sub-systems. Table 4.7 mentions system complexity and domains analyzed by this approach.

Case studies are performed on in-house aerospace projects such as enhanced smart fatigue meter (eSFM), stall warning & aircraft interface computer (SWS/AIC) system, and crack detection & warning (CDWS) system. eSFM and CDWS are examples of new technology development. SWS/AIC is an example of modification of existing technology. For eSFM system design variations using SA approach were adopted for IAF Jaguar (MK-II) aircraft. Two system designs were analyzed against the project requirements for functionality and safety. Further modification to the approved design was analyzed for re-using the eFM for indigenously developed Medium Altitude Long Endurance UAV, RUSTOM-II. For CDWS, two system designs were analyzed against the project requirements for functionality. For SWS/AIC, system upgrade with additional functionalities was required. For this SA approach was used to generate the data flow and control flowchart as part of the impact analysis artifact as required by the Certification agency. The manual analysis of

4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION

these safety critical systems took time. Hence to reduce the time taken and make the analysis effective, an in-house tool is developed. The tool takes required system inputs, generates the SA expression, and simulates system state transition over time as required. Tools help in automating the process and generate data for analyzing the system's functional availability.

Woodcock *et al.* [64], carried out a survey about the trend of FM in various application domains, including transport, nuclear, defense, financial, healthcare, consumer electronics, telecom, office, and administration etc. Sixty one percent of users felt that formal techniques have helped them to successfully complete their projects. Forty nine percent felt that the techniques used were appropriate for their application, 56% felt that FM tools helped them cope with tasks, and 75% were inclined to use FM's again in future projects. These figures show that the industry is inclined to use formal methods. Tools help in automating the process; based on significant theoretical advances as formal techniques, tools can address problems of a much larger scale. FM tools used in the industry are provided in the Intelligent Knowledge Database (IKD) developed by CSIR-NAL. Some of the popular tools used in industry are listed below:

The SACEM system used the B-method in specifying the complex software. The system is an embedded hardware and software delivered in 1989 and controlled the speed of all trains along the RER Line A in Paris, involving seven billion passenger journeys, since its introduction. The specification of this complex software was constructed in B, discussed by Abrial [139], and proved interactively with auto-generated verification conditions for the code. Statistics proved that respondents have generally been positive about the utility of FM's and, in general, were satisfied with formal techniques used in their projects. The FM-based tools applied have not fully been able to live up to expectations and a majority of the respondents wish to use a similar FM technology again for new projects. Barjaktarovic [140] discusses state-of-the-art tools for FM's, types of the tools, and the industry requirement from the tools for their effectiveness. The report mentions that tools developed can either be heavy, medium, or light weight tools.

Heavy weight tools are based on logic and automated assistance in (human-guided) proof steps and are suitable for parametric verification. Their major drawback is the long learning curve. Additional research and training is needed to bring these tools within reach of application engineers. For example : Theorem provers.

The industry mostly uses model checkers and light weight tools because of their inherent benefits compared to heavy weight tools. The key benefit is clarity and not analytical power, i.e., FM's are used to provide key insights in a short period of time. For that, the notation used must be accessible to users and tools should be as automated as possible. Model checkers and other light weight tools also have the benefits of ease-of-learning,

ease-of-use, and its similarity to traditional practices. Model checkers are most commonly used for symbolic simulations, where engineers visually inspect output and recognize errors. Since no single tool can deal with all classes of problems, they need to be combined. The ease-of-use of an FM tool, and its power to analyze and provide feedback will enable its acceptance in the engineering society. It is observed that engineers are more likely to use tools that have the look and feel of tools they are familiar with, or tools that clearly help them accomplish their tasks in a timely manner.

Experience shows that engineers prefer graphical tools, which are informal tools that help automate tedious tasks, and executable specifications, preferably integrated with simulation of user interface. This allows for rapid prototyping and early review with customers. Based on stated industrial needs, we conclude that tools must address the issue of:

- **Clarity:** Notations must be accessible to engineers and have the power of expression.
- **Error finding:** Tools must identify bugs and mistakes, rather than prove correctness. Errors must be clearly identified, i.e., line number of offending code, statement and an explanation of why it is an error.
- **Effectiveness:** Tools must be fast, easy-to-use, and powerful.
- **Robustness:** Tool should be as bug-free as possible; proof, analysis, and validation should also be robust, i.e., they should not easily break when specification changes.
- **Extensibility:** Users must have the ability to customize the tool to a particular design flow.
- **Automation and methodology:** Tools should be as automated as possible; at least, users should have some methodology to guide them.
- **Reusability and change:** Proofs/modules should be reusable during frequent changes in requirements and/or specifications.
- **Scalability:** Tools must have some means of organizing large and/or growing specifications.
- **Compatibility with existing practices/tools:** Compatibility comes in two forms where: (i) the tool must be able to "fit" into the existing design flow and; (ii) the tool should resemble existing tools. Integration of graphical and textual notations for user interface should be acceptable to users.
- **Proving application specific properties:** Tools must be appropriate for the task on hand.
- **Tool support:** Tool must have good technical support, user manual, documentation, upgrades, and training. Tool must be stable.

4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION

The proposed SA simulation tool has a graphical interface, is easy to understand, and guides the designers to provide the relevant input for system analysis using the SA approach. The flowchart of the tool is shown in Figure 4.12.

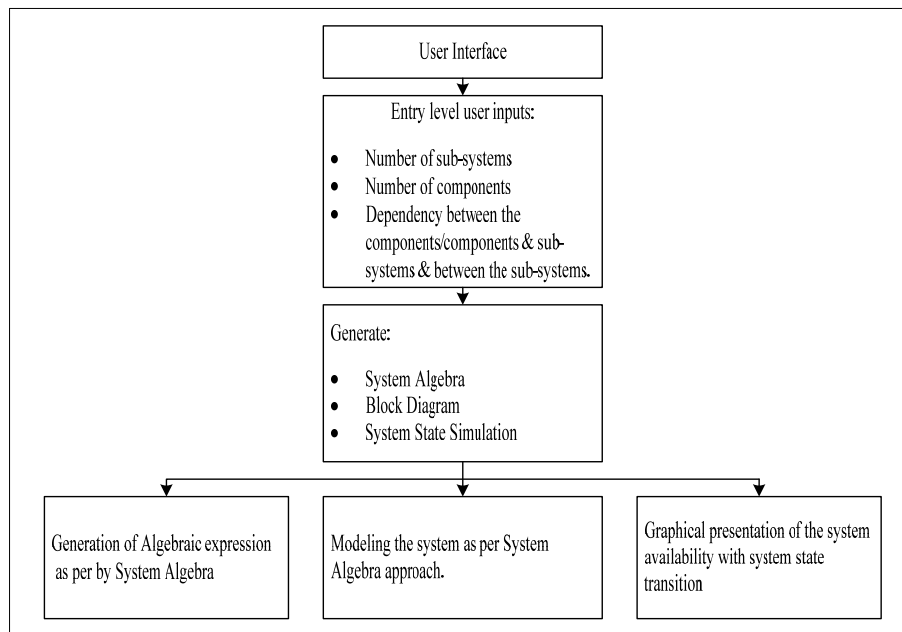


Figure 4.12: Flowchart of the Tool

Tool features are:

- **System algebra:** This shows the overall System Algebra.
- **Block diagram:** This displays the full view of how sub-systems are connected ,i.e., in parallel or in series.
- **System simulation:** This shows the different states of sub-systems during simulation. The simulation is started with the start of the timer. Based on the failure rates of sub-systems, entered, the sub-systems start failing, i.e., the software changes the colour of the failed sub-system from green to red. The initial color of all sub-systems is green, indicating the safe state of the system, providing complete functionality. When a particular sub-system fails, it changes the color to red. Depending upon the inter-relationship between the sub-systems, the SA expression value changes. This is computed based on the algebra properties. The sub-system failures are computed, system state determined based on the relationship between the sub-systems. This goes on until more than 50% of the sub-systems fail. At that time, the system enters the unsafe state, indicated by indicating the failure of all the sub-systems. This is indicated by red. The time when the system enters the

unsafe state determines the functional availability. The tool features are shown in Figures 4.13, 4.14, 4.15, 4.16, 4.17, 4.18, and 4.19.

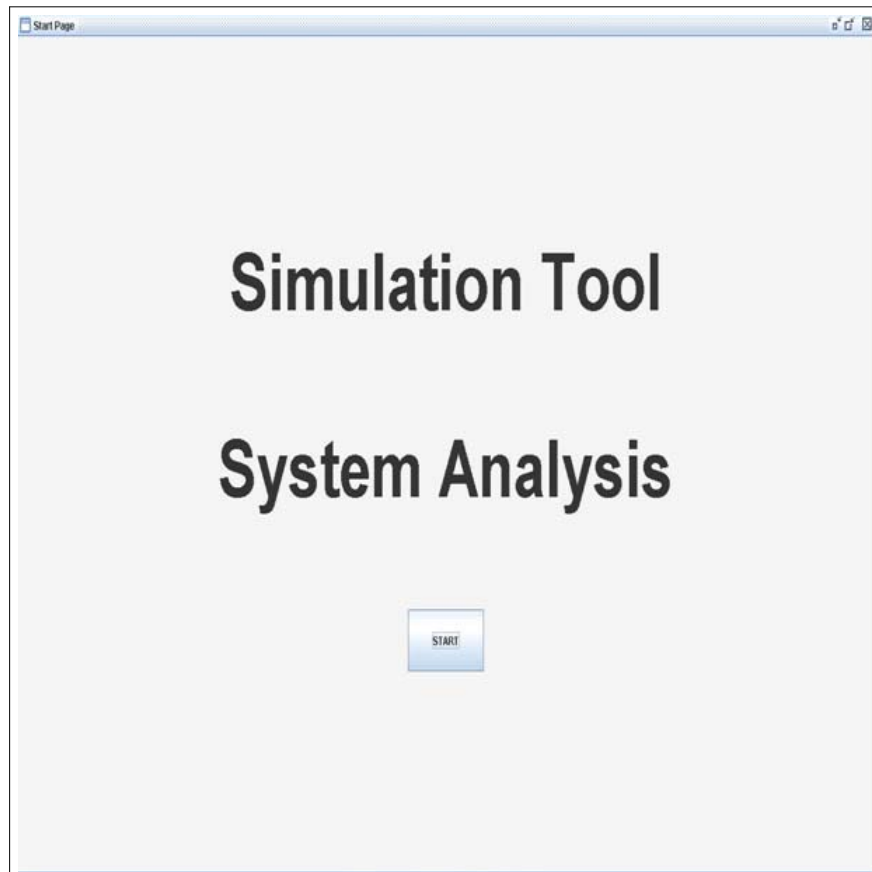
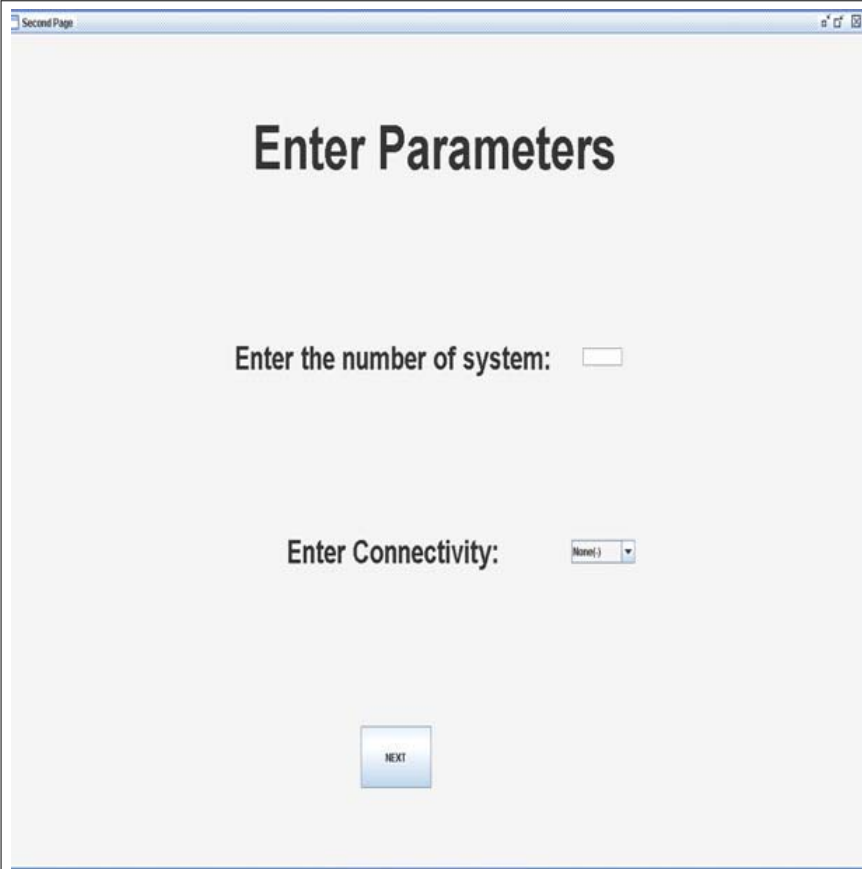


Figure 4.13: Tool : Opening Window

The opening window shows the name of the tool and start button to move to the next window. Figure 4.13 shows the first window of the tool. The tool is a simulation for system analysis using SA methodology. Figure 4.14 shows the user-interface for entering system parameters. System parameters, such as number of sub-systems and the relationship between them, are entered in this window. This information is used to generate a system block diagram.

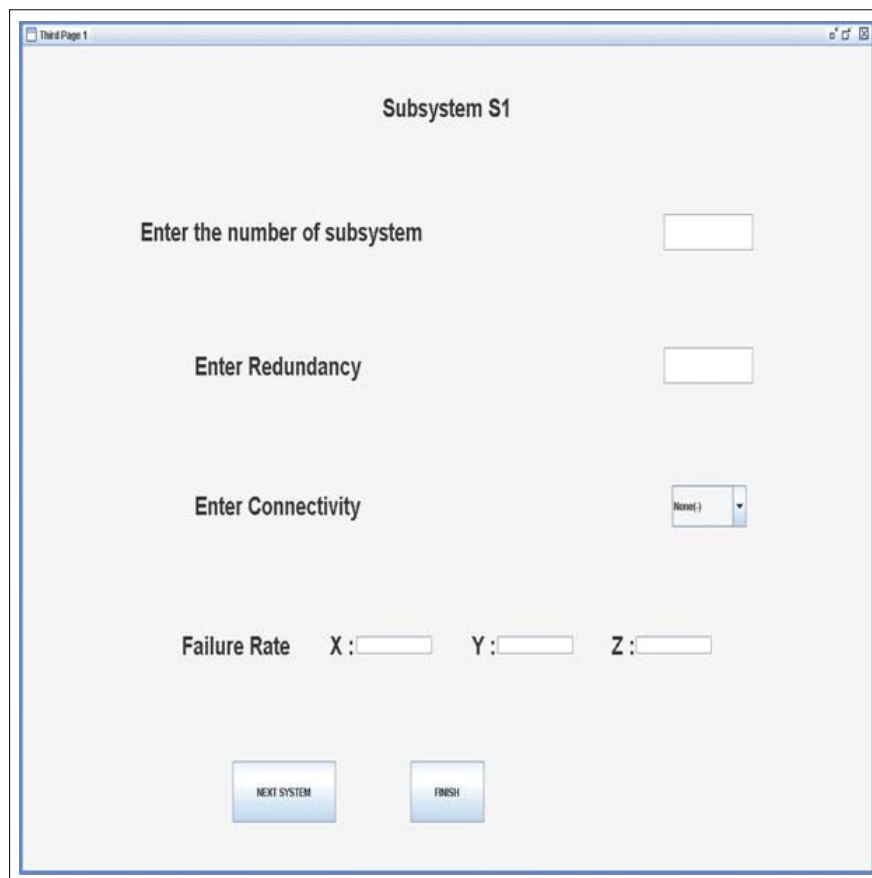
4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION



The image shows a software window titled "Second Page" with a light gray background. At the top center, the text "Enter Parameters" is displayed in a large, bold, black font. Below this, there are two input fields. The first is labeled "Enter the number of system:" followed by a small, empty white text box. The second is labeled "Enter Connectivity:" followed by a dropdown menu that currently shows "(None)". At the bottom center of the window, there is a blue button with the word "NEXT" written on it in white capital letters.

Figure 4.14: Tool : Parameter Window for System

Figure 4.15 shows the user-interface for entering sub-system parameters. For each sub-system, its redundancy, its relationship with other sub-systems, and its failure rates are entered. This information is required to generate the SA expression and system state transitions for analyzing the functional availability.



The screenshot shows a window titled "Third Page 1" containing a form for "Subsystem S1". The form includes the following elements:

- Label: "Enter the number of subsystem" followed by a text input field.
- Label: "Enter Redundancy" followed by a text input field.
- Label: "Enter Connectivity" followed by a dropdown menu currently showing "None(-)".
- Label: "Failure Rate" followed by three text input fields labeled "X:", "Y:", and "Z:".
- Two buttons at the bottom: "NEXT SYSTEM" and "FINISH".

Figure 4.15: Tool : Parameter Window for Sub-system

Figure 4.16 shows the user interface for selecting the tool options; including generation of SA expression, generation of system block diagram, and simulation of system state transitions.

4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION

Third Page 1

Subsystem S1

Enter the number of subsystem

Enter Redundancy

Enter Connectivity

Failure Rate X: Y: Z:

Figure 4.16: Tool : Feature Execution

Figure 4.17 shows the generated block diagram of a system with six sub-systems. The system is a simple system. Sub-systems 1, 2, 3, 4, and 5 have sub-systems with sequential control. The *NEXT* option provides details of the sub-systems.

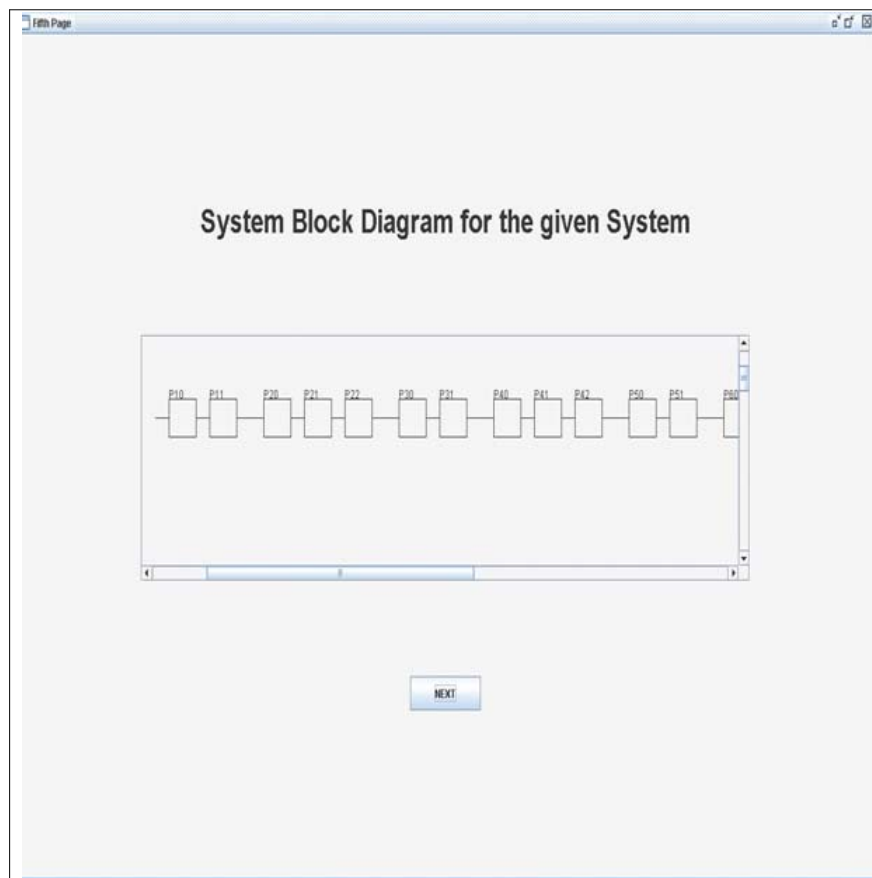


Figure 4.17: Tool : Generation of System Block Diagram

Figure 4.18 shows the SA expression for the system. The failure rates of the sub-systems are also shown in this window. These figures are used to compare the system functional availability with its reliability.

4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION

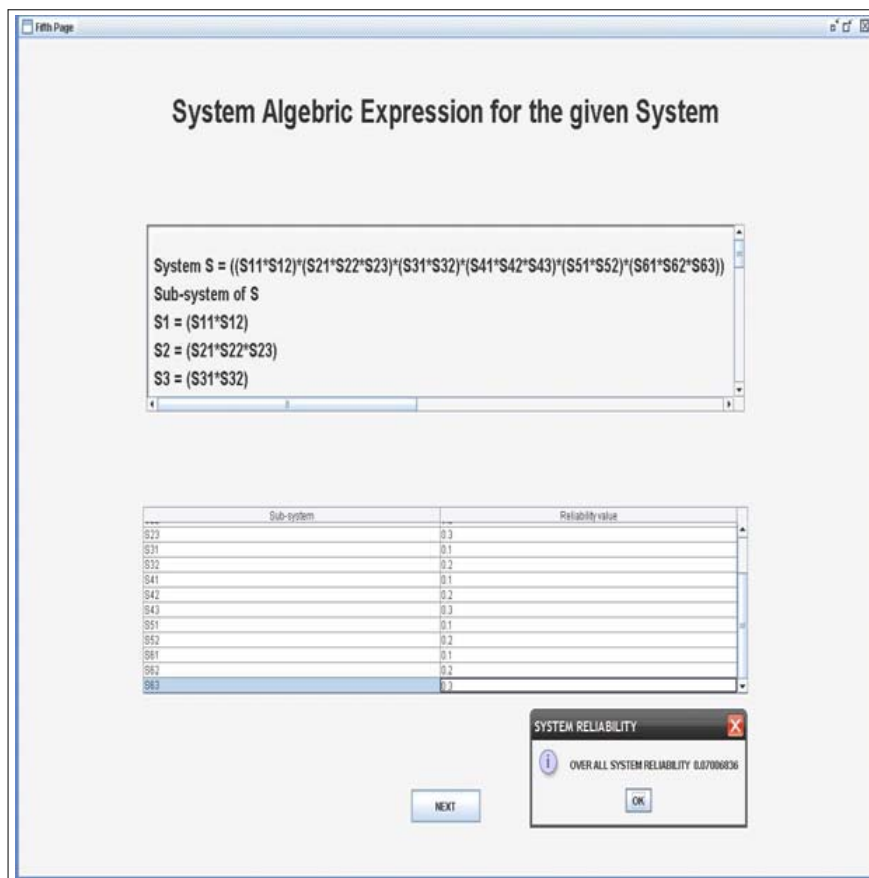


Figure 4.18: Tool : Generation of SA Expression

Figure 4.19 shows a snap-shot of system state simulation. The system state transitions from a safe state (circles in GREEN) to a hazardous state (circles in YELLOW) to an unsafe state (circle in RED) are computed from sub-system failure rates and SA expression.

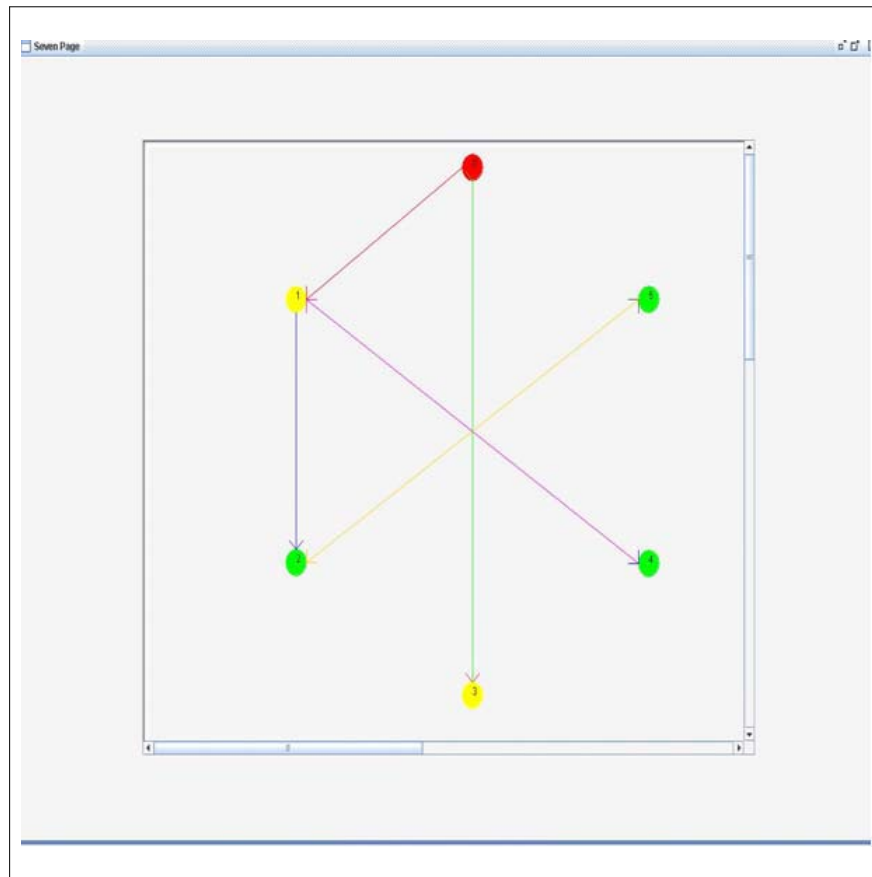


Figure 4.19: Tool : System State Simulation

The tool is developed in Java as a front- and back-end. The front-end is a user interface as described earlier and helps designers to provide the system details like:

- **1:** Number of sub-systems
- **2:** Relationship amongst the sub-systems
- **3:** Sub-system redundancy
- **4:** Sub-system failure rate

The back-end uses the SA algorithm to compute the SA expression, display the system block diagram, and perform the system simulation to determine the functional availability of the design. This simulation also helps in visualizing a system for its modularity and complexity. The ruggedization of the tool for its correct and proper operation is required to plug it to the formal workbench being developed by CSIR-NAL.

The proposed modified systems engineering with SA makes the process effective as it enables designers to analyze system properties such as safety, reliability, functionality,

4. PROPOSED MODIFIED SYSTEMS ENGINEERING LIFE CYCLE PROCESS IN FORMAL DESIGN, VERIFICATION AND VALIDATION

and availability. The tool developed enables in understanding the SA and implementing it easily. The tool needs to be developed so that it can be easily plugged to existing tools to make the process effective.

5

Conclusion And Future Work

The research work proposed a novel concept of system state machine, (SSM), for determining the functional availability property of critical complex systems. Functional availability can be more accurately determined since the concept of SSM is based on Formal Methods, FM, and System Algebra, SA. The applicability and effectiveness of SA analyzes complex critical systems by reverse-engineering. The flight control systems of defense and civil aircraft such as Jaguar, Boeing, and Airbus were analyzed to validate system design for its functional availability and safety. Other critical systems investigated - automobile, medical and railways - demonstrate the applicability and effectiveness of SA. The concept of functional availability using the SSM concept analyzes the design against its requirements and proposed application. Integrating the SA technique in the design phase of an engineering process helps in understanding, detecting design flaws early, thus improving the design and the process. Using the SA technique, one can determine the system's functional availability depending on its states; allowed state transition is the contribution to engineering process research. The development of an in-house tool and various case studies on safety-critical systems demonstrate the utility of this technique to other industrial applications.

Systems are analyzed for their functionality and performance. The functionality of a system is proven by simulation and helps in validating the intended functionality of a design. Simulation and analysis of a system is done manually or using tools without applying formal techniques. Reviews are always done manually. The functional analysis of a system ensures that it provides the required functionality. The performance of a system helps in validating its reliability, safety, and timing response. Reliability Block Diagram, RBD, Fault Tree Analysis, FTA, and dynamic techniques are used to analyze reliability, safety and timing of a system. The performance techniques are a combination of formal, semi-formal and informal techniques. Functional and performance analyses

5. CONCLUSION AND FUTURE WORK

validate overall system design, giving rise for a need to improve these techniques; the best and the most effective way is to introduce formal techniques in analyses. There is a need to introduce more formal techniques in validating the functional availability of a system during the early phase of an engineering process to understand, improve, and catch design flaws sooner. The requirement phase, specification phase and design phase are considered to make up the early phase of an engineering process.

Safety-critical systems are increasing in complexity to discharge the functionality expected. These systems are designed for high up-time, safety, and reliability. SA is capable of determining the functional availability of systems. It uses the novel concept of SSM in order to determine the functional availability property of a system. The functionality of the system at any instant is determined by its state at that instant. The system provides complete functionality if it is in a healthy state; no functionality if unhealthy or is at partial functionality if it is between these two states. SA explores this feature of a system by defining three major states: the *safe* state, the *hazardous* state, and the *unsafe* state. For a system in the safe state, complete functionality is available and its behavior meets the requirements with no impact on safety. A system in the hazardous state has limited functionality available leading to degraded performance with partial effect on system safety. A system in the unsafe state has no functionality available and its safety is not guaranteed. The safe, hazardous, and unsafe states can be further divided into sub-states depending on the functionality provided by a system. The functionality of a system is defined during its conception depending on the design requirement of a fail-operational, or fail-safe system. Over time, with the failure of sub-systems and components a system transits from one state to another. This system behavior is described by the SSM, similar to a Finite State Machine (FSM), through entry, exit, input, and transition actions. The entry action is a change in the functionality of a system when it enters a given state per system design. For the safe state, a system provides the full functionality it is designed for. In the hazardous state, a system will have limited functionality and in the unsafe state it will have no functionality and remain idle. The exit action is a change in functionality of a system when it enters a new state. The input action is the system output depending on the current state and the trigger that is causing a transition to another state. The transition action is the functionality required during a certain transition. This action is based on system design and application. The functional availability of a system, thus derived, aids designers in determining system behavior against project requirements. System analysis with SA provides a better visualization of the system design and provides a novel technique to do so. This novel analysis technique strengthens the decision-making capability of designers with a measurable or quantifiable metric of the system. This work demonstrates the ease of integrating system functional availability into an existing process, tool chain for system

analysis, and the ease of adapting a formal approach in addition to other analysis techniques for improved safety and reliability. The seamless integration of the SA approach with existing techniques, modifies the engineering process for better productivity.

This research work formalizes the design phase of an engineering process to validate functional availability of a system. This formalization is done by introducing the SA technique in the design phase; the system's functional availability is demonstrated mathematically and visually. The SA approach models (abstracts) a system, decomposing into the smallest possible independent sub-systems. This decomposition to the lowest level (atomic) is such that these sub-systems can be combined in series and parallel design to form the system again. The inter-relationship amongst the sub-systems helps in deriving the SA expression, which in turn aids in determining system complexity, fault tolerance, and the worst-case, and the best-case fault tolerances for a given system. The failure rates of the components/sub-systems in the SA expression determine the system state at any given instant.

5.1 Results & Outcome

The SA technique can be used to determine the functional availability property of the system. It is a reliable technique as it is based on FM's. All complex, safety-critical systems, which need to be designed with utmost clarity are made possible with this technique. The modeling and analysis of ground based safety critical-systems such as eFM, CDWS and of airborne safety-critical systems like FCS, SWS/AIC, EICAS, prove the effectiveness of this technique. The technique is also useful in analyzing high complex systems, such as cyber-physical and net-centric systems where multiple functions are performed on a single platform.

The effectiveness of the SA approach to characterize the functional availability of a system is validated by applying this technique to aerospace, medical, nuclear, automobile and railway domains. Proven systems were analyzed using reverse engineering approach for their availability based on system states and transitions over time. The conventional clinical laboratory system, autonomous de-centralized clinical laboratory system, Jaguar flight control system, steer by-wire controller, railway intelligent transportation system architecture, distributed fault tolerant architecture for nuclear reactor control and safety functions, fault-tolerant architecture for computer-based railway vehicle brake systems, distributed architecture block diagram, typical transport power source, and integrated elevator electrical power and control using the power by wire (PBW) were analyzed for the systems availability and fault tolerance. The introduction of this technique into the

5. CONCLUSION AND FUTURE WORK

Application	Domain	Number Of Sub-systems	Approach
Jaguar FCS	Aerospace	18	Reverse Engineering
Enhanced Fatigue Meter	Aerospace	6	Forward Engineering
Crack Detection and Warning System	Automotive and Aerospace	6	Forward Engineering
Clinical Laboratory System	Medical	7	Reverse Engineering
Autonomous Clinical Laboratory System	Medical	2	Reverse Engineering
Steer-by-wire Controller	Automotive	6	Reverse Engineering
Railway Intelligent Transportation System	Railways	6	Reverse Engineering
A distributed fault tolerant architecture for nuclear reactor control and safety functions	Nuclear	3	Reverse Engineering
A fault-tolerant architecture for computer-based railway vehicle brake systems	Railway	8	Reverse Engineering
Integrated elevator electrical power and control using the power by wire	Aerospace	10	Reverse Engineering
ATR62-70	Aerospace	30	Reverse Engineering

Table 5.1: Analysis of Systems Using SA

existing engineering life-cycle process modifies the systems engineering life cycle. The seamless integration of this approach with existing techniques makes the system functional analysis effective as it benchmarks the functional availability of a system. Table 5.1 lists the systems and their sub-systems that were modeled and analyzed using the SA. Systems with 6 to 30 sub-systems were analyzed for their functional availability.

A more complex system - having more than 25 sub-systems - can be analyzed by decomposing the complex system into smaller sub-systems. Each sub-system is further decomposed to smaller units until no further decomposition is possible. The SA expression for all sub-systems is computed to derive the same for the complete system. This approach follows for abstracting a system of any complexity.

- ***Top-down Approach***

In Top-down Approach, a system is decomposed into sub-systems. Each sub-system is further decomposed until no further division is possible. During this system decomposition, the relationship between sub-systems is also considered. This relationship helps in modeling the system. This is shown in Figure 5.1.

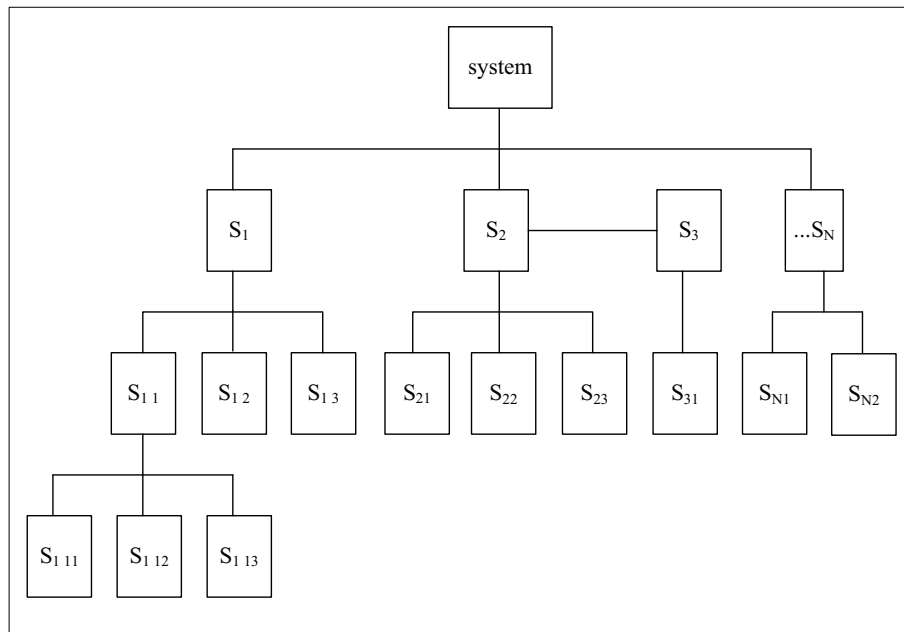


Figure 5.1: Top-Down Approach

The figure shows a system consisting of N sub-systems S_1, S_2, S_3 and S_N . The sub-systems S_1, S_2 and S_N work independently and their relationship is depicted as *direct sum*. Sub-system S_3 has a relationship with S_2 and is depicted as *direct product*. Similarly, the units under S_1, S_{11}, S_{12} and S_{13} share a *direct sum* relationship. Under S_{11} , sub-systems S_{111} and S_{113} share a *direct sum* relationship whereas S_{112} is dependent on S_{111} and is depicted as *direct product*. This analysis is carried out for all sub-systems. This relationship analysis is the key to deriving the SA expression.

- **Bottom-Up Approach**

The Bottom-up Approach follows the Top-down Approach. Every sub-system is represented by its equivalent SA expression in the Top-down approach. The sub-systems are added to the sub-system at the next higher level and thus SA expression for that higher level sub-system is derived. This is shown in Figure 5.2.

5. CONCLUSION AND FUTURE WORK

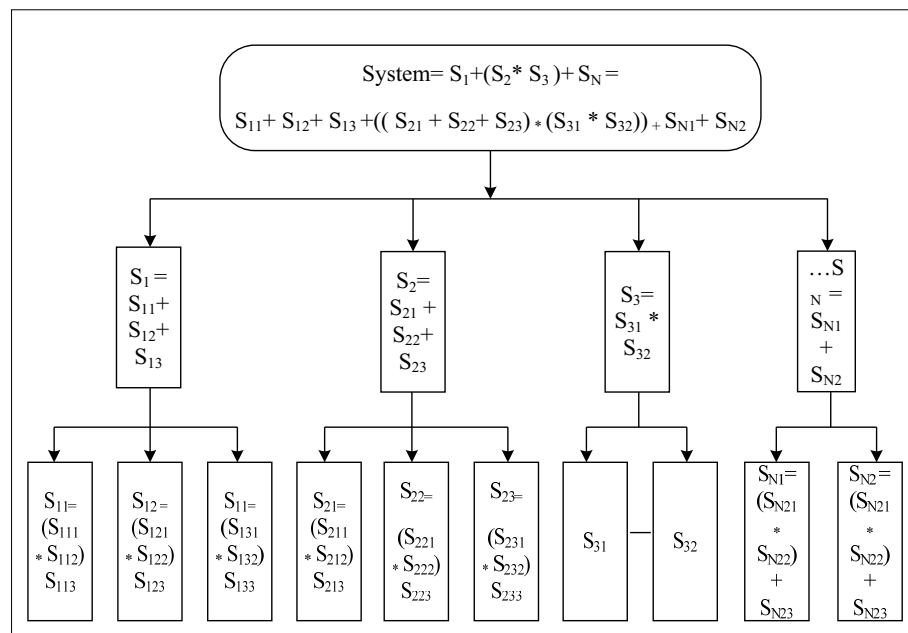


Figure 5.2: Bottom-Up Approach

The figure shows SA expression for sub-systems and the complete system. The *Top-down* and *Bottom-up* approaches using SA, enable analyzing systems of varying complexity.

The notable outcome of this research work is the analysis of the **Functional Availability** property of a system. Since a **formal approach** is used, the results provide more **assurance**. The outcome of this analysis can be used as a **benchmark** for further improvement in functional availability analysis. The SA FM approach helps in quantifying the functional availability to generate measurable values that can be related to a qualitative analysis. The tool automates system analysis by computing functional availability and depicting system states and transitions from a safe to unsafe state. The modification of the engineering process improves system analysis by detecting design flaws early in the process. The analysis of critical systems demonstrates that the SA approach can be implemented easily for system analysis.

The applicability of SA is demonstrated by integrating SA with existing techniques in the design phase of engineering process. The effectiveness of the modified engineering process is demonstrated by simulating safety-critical systems of varying complexity. Safety critical systems have been analyzed for safety and reliability based on FTA and

RBD approaches. This feature of integrating SA with existing techniques makes this approach realizable. The modified process is an integrated formal process that increases the process performance by 20%.

The integration of the SA with existing techniques creates a seamless formalized approach for the engineering process. The migration from an informal to a more formal approach allows for a better and more effective engineering process. The process automation with the tool developed in-house not only automates the technique but adds to the existing list of formal method-based tools to provide a formal workbench for design, development and verification of safety critical applications.

5.2 Future Work

Future work involves improvement of the analysis capability of SA :

- Micro-level analysis can be realized by considering more stable system states machines and the entry-exit criteria of these states into other states.
- The effects of human-machine interface could be considered to determine system functional availability. The human-machine interface will help in designing a robust system where system availability and fault tolerance will consider human interaction.
- The applicability of this approach for a prognostic approach in the system should be explored. The system analysis approach of this methodology can be used to develop software- based health management algorithms for a system.
- Tool enhancement for integrating it with existing engineering process tools. This tool development will be used to develop a formal tool workbench to provide a tool-based design and development process.

The tool will be introduced to the formal workbench to make the engineering process more reliable. The workbench introduces formal method-based tools to capture the project requirements, design the system, analyze the design, implement the system and test the system. The development of the workbench is a new initiative taken by CSIR-NAL to tackle design-development and certification of highly complex safety critical systems.

References

- [1] F. T. Hoban, "NASA systems engineering handbook," *NASA Office of Safety and Mission Assurance*, Jun. 1995, [Online]. Available : [http : //www.stanford.edu/class/cee243/NASASE.pdf](http://www.stanford.edu/class/cee243/NASASE.pdf).
- [2] A. T. Bahill and F. F. Dean, "What is systems engineering? a consensus of senior systems engineers," 2009, [Online]. Available:<http://www.sie.arizona.edu/sysengr/whatis/whatis.html>.
- [3] J. A. Estefan, "Survey of model-based systems engineering (mbse) methodologies," *INCOSE Survey of MBSE Methodologies*, vol. 1, pp. 1–80, Jun. 2008, [Online]. Available : [http : //www.incose.org/products/pubs/pdf/techdata/mttc/mbse_methodology_survey_2008 – 0610_revb – jae2.pdf](http://www.incose.org/products/pubs/pdf/techdata/mttc/mbse_methodology_survey_2008-0610_revb-jae2.pdf).
- [4] J. P. Bowen and V. Stavridou, "The industrial take-up of formal methods in safety-critical and other areas: A perspective," in *Proceedings of the First International Symposium of Formal Methods Europe on Industrial-Strength Formal Methods*, vol. 670, Apr. 1993, pp. 183–195, ISBN:3-540-56662-7.
- [5] O. Laurent, "Using formal methods and testability concepts in the avionics systems validation and verification (v&v) process," in *Third International Conference on Software Testing, Verification and Validation*, 2010, pp. 1–10.
- [6] S. P. Miller, E. A. Anderson, L. G. Wagner, and M. W. Whalen, "Formal verification of flight critical software," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2005, pp. 1–16.
- [7] weibull.com, "On-line reliability engineering resources for the reliability professional." [Online]. Available: [Online]. Available:<http://weibull.com>
- [8] D. Milutinovic and V. Lucanin, "Relation between reliability and availability of railway vehicles," *Transaction on FME*, vol. 33, pp. 135–139, Sep. 2005.
- [9] J. Wang, Y. Deng, and G. Xu, "Reachability analysis of real-time systems using time petri nets," *IEEE Transactions On Systems, Man And Cybernetics-Part B: Cybernetics*, vol. 30, pp. 725–736, Oct. 2000.
- [10] H. Reza, M. Pimple, V. Krishna, and J. Hilde, "A safety analysis method using fault tree analysis and petri nets," in *Sixth International Conference on Information Technology: New Generations*, Apr. 2009, pp. 1–6, doi:ieeecomputersociety.org/10.1109/ITNG.2009.183.
- [11] P. H. Fieler, "Model-based validation of safety-critical embedded systems," in *2010 IEEE Aerospace Conference*, Apr. 2010, pp. 1–10, doi:[10.1109/AERO.2010.5446809](http://ieeecomputersociety.org/10.1109/AERO.2010.5446809).

REFERENCES

- [12] E. C. Honour, "Understanding the value of systems engineering," 2004, [Online]. Available:<http://www.incose.org/secoe/0103/ValueSE-INC0SE04.pdf>.
- [13] J. Rushby, "Critical system properties: Survey and taxonomy," *Reliability Engineering and System Safety*, vol. 43, no. 2, pp. 189–219, 1994, doi:10.1016/0951-8320(94)90065-5.
- [14] T. Erkkinen, "Model-based design for do-178b with qualified tools," in *AIAA Modeling and Simulation Technologies Conference and Exhibit, AIAA 2009-6233*, Aug. 2009, pp. 1–13.
- [15] D. Teresa, "IEEE 1220: For practical systems engineering," *Computer*, vol. 39, pp. 92–94, May 2006.
- [16] A. L. Ramos, J. V. Ferreira, and J. Barceló, "Model-based systems engineering: An emerging approach for modern systems," *IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Review*, vol. 42, 2012, doi : 10.1109/TS MCC.2011.2106495.
- [17] H. P. de Koning, H. Eisenmann, and M. Bandecchi, "Evolving standardization supporting model based systems engineering," in *SECESA 2010*, Oct. 2010, pp. 1–31.
- [18] IEEE, "Systems and software engineering - system life cycle processes," [Online]. Available : http://webstore.iec.ch/preview/info_isoiec15288ed2.0en.pdf.
- [19] N. Dulac and N. Leveson, "An approach to design for safety in complex systems," in *INC0SE International Symposium on Systems Engineering*, Toulouse, France, Jun. 2004, pp. 137–152, doi:10.1.1.1.5227.
- [20] B. H. Sababha, O. A. Rawashdeh, and W. A. Sadeh, "A real-time gracefully degrading avionics system for unmanned aerial vehicles," in *IEEE National Aerospace and Electronics Conference (NAECON)*, Jul. 2012, pp. 171–177, doi : 10.1109/NAECON.2012.6531050.
- [21] J. Rushby, "Formalism in safety cases," *Proceedings of the Eighteenth Safety-Critical Systems Symposium*, pp. 3–17, Feb. 2010, ISBN: 978-1-84996-085-4.
- [22] S. Myrefelt, "Reliability and functional availability of hvac systems," *Proceedings of the Fourth International Conference for Enhanced Building Operations*, 2004, [Online]. Available : <http://repository.tamu.edu/bitstream/handle/1969.1/5031/ESL-IC-04-10-07.pdf>.
- [23] M. G. Richards, A. M. Ross, N. B. Shah, and D. E. Hastings, "Metrics for evaluating survivability in dynamic multi-attribute tradespace exploration," in *Journal of Spacecraft and Rockets-J SPACE-CRAFT ROCKET*, vol. 46, 2009, pp. 1049–1064.
- [24] K. S. Trivedi, D. S. Kim, A. Roy, and D. Medhi, "Dependability and security models," *Proceedings of 7th International Workshop on the Design of Reliable Communication Networks (DRCN 2009)*, Oct. 2009, [Online]. Available : <http://www.sis.pitt.edu/dtipper/3350/paper6.pdf>.
- [25] O. Akerlund, S. Nadjm-Tehrani, and G. Stalmarck, "Integration of formal methods into system safety and reliability analysis," in *In Proceedings of 17th International Systems Safety Conference*, 1999, pp. 326–336.
- [26] M. Whalen, D. Cofer, S. Miller, B. H. Krogh, and W. Storm, "Integration of formal analysis into a model-based software development process," in *Proceedings of the 12th international conference on Formal methods for industrial critical systems*, 2008, pp. 68–84.

REFERENCES

- [27] C. Chen, J. S. Dong, and J. Sun, "A formal framework for modeling and validating simulink diagrams," *Journal of Formal Aspects of Computing*, vol. 21, pp. 451–483, Mar. 1998, doi:10.1007/s00165-009-0108-9.
- [28] S. International, "Guidelines for development of civil aircraft and systems," Apr. 2012, [Online].Available :: [http : //sunnyday.mit.edu/16.355/arp4754a.pdf](http://sunnyday.mit.edu/16.355/arp4754a.pdf).
- [29] S. Rao, "A systems algebra and its applications," in *2nd Annual IEEE International Systems Conference (IEEE SysCon 2008)*, Apr. 2008, pp. 28–35, doi:10.1109/SYSTEMS.2008.4518984.
- [30] F. Brosch, B. Buhnova, H. Koziolok, and R. Reussner, "Reliability prediction for fault-tolerant software architectures," in *Seventh International ACM Sigsoft Conference on the Quality of Software Architectures (QoSA)*, Jun. 2011, pp. 75–84, doi : 10.1145/2000259.2000274.
- [31] B. Melhart and S. White, "Issues in defining, analyzing, refining, and specifying system dependability requirements," in *Seventh IEEE International Conference and Workshop on the Engineering of Computer Based Systems*, Apr. 2000, pp. 334–340, doi : 10.1109/ECBS.2000.839893.
- [32] I. Beer, B.-D. Shoham, D. Geist, R. Gewirtzman, and M. Yoeli, "Methodology and system for practical formal verification of reactive hardware," in *Proceedings of the 6th International Conference on Computer Aided Verification*, 1994, pp. 182–193, ISBN:3-540-58179-0.
- [33] A. L. Ramos, J. V. Ferreira, and J. Barceló, "Model-based systems engineering: An emerging approach for modern systems," *IEEE TransactionsonSystems, Man, andCybernetics – PartC : ApplicationsandReview*, vol. 42, 2012, doi : 10.1109/TS MCC.2011.2106495.
- [34] R. Team, "Rtca: Software considerations in airborne systems and equipment. radio technical commission for aeronautics. (rtca), european organization for civil aviation electronics (eurocae), standard document no. do-178b/ed-12b," Dec. 1992, [Online].Available : [http : //www.faa.gov/regulations_policies/advisory_circulars/index.cfm/go/document.information/documentID/698460](http://www.faa.gov/regulations_policies/advisory_circulars/index.cfm/go/document.information/documentID/698460).
- [35] E. S. for Electrotechnical Standardization, "En 50128:201: Railway applications - communication, signalling and processing systems - software for railway control and protection systems," 2012, [Online].Available : [http : //www.en-standard.eu/csn-en-50128-railway-applications-communications-signalling-and-processing-systems-software-for-railway-control-and-protection-systems/](http://www.en-standard.eu/csn-en-50128-railway-applications-communications-signalling-and-processing-systems-software-for-railway-control-and-protection-systems/).
- [36] E. Standards(EN), "En 60880-2009: Nuclear power plants - instrumentation and control systems important to safety-software aspects for computer-based systems performing category a functions," 2010, [Online].Available : [http : //www.cenelec.eu/dyn/www/f?p = 104 : 30 : 3965501406515248 ::: FS P_ORG_ID,FS P_LANG_ID : 10415,25](http://www.cenelec.eu/dyn/www/f?p=104:30:3965501406515248:::FS P_ORG_ID,FS P_LANG_ID:10415,25).
- [37] I. E. Commission, "International electrotechnical commission (iec): Functional safety of electrical/electronic/programmable electronic safety-related systems. iec 61508," 1998, [Online].Available : [http : //www.iec.ch/functionalsafety/standards/page3.htm](http://www.iec.ch/functionalsafety/standards/page3.htm).
- [38] I. O. for Standardization, "Iso/dis 26262- road vehicles functional safety," 2009, [Online].Available : [http : //www.ni.com/white-paper/13647/en](http://www.ni.com/white-paper/13647/en).
- [39] D. Brown, H. Delseny, K. Hayhurst, and V. Wiels, "Guidance for using formal methods in a certification context," in *ERTS 2010*, pp. 1–7.

REFERENCES

- [40] M. Bajaj, D. Zwemer, R. Peak, A. Phung, A. G. Scott, and M. Wilson, "SLIM: Collaborative model-based systems engineering workspace for next-generation complex systems," in *IEEE Aerospace Conference*, Mar. 2011, pp. 1–15, doi:10.1109/AERO.2011.5747539.
- [41] D. Milutinovic and V. Lucanin, "Relation between reliability and availability of railway vehicles," *Transaction on FME*, vol. 33, pp. 135–139, Sep. 2005.
- [42] D. R. Kuhn, D. Craigen, and M. Saaltink, "Practical application of formal methods in modeling and simulation," in *Proceedings of SCSC'03 (Invited)*, Jul. 2003, pp. 1–6, doi:10.1.1.6.7028.
- [43] B. Gnadt, "Model based system engineering at lockheed martin ms2," Dec. 2010, [Online]. Available : [http : //www - 950.ibm.com/...Gnadt.../IBM_HW - SW_Seminar_Gnadt_09DEC10.pdf](http://www-950.ibm.com/...Gnadt.../IBM_HW-SW_Seminar_Gnadt_09DEC10.pdf).
- [44] C. Strobel and A. Arnold, "Efficient formal model checking of sysml statecharts using rhapsody and nusmv lessons learned from practical application," *EADS Innovation Works - Systems Engineering*, Oct. 2010.
- [45] E. Technologies, "Efficient development of safe railway applications software with EN 50128 objectives using SCADE suite," Oct. 2009, [Online]. Available : [http : //www.citeulike.org/user/asilva/article/846297](http://www.citeulike.org/user/asilva/article/846297).
- [46] Y. Xiaohui, D. Ruizhong, and T. Junfeng, "The reliability design and availability analysis of a distributed storage system," *Wuhan University Journal of Natural Sciences*, vol. 11, pp. 498–510, Apr. 1919–1922.
- [47] N. Team, "Formal methods specification and analysis guidebook for the verification of software and computer systems, volume ii: A practitioner's companion,nasa- gb-o01-97, release 1.0," [Online]. Available:<http://www.csl.sri.com/papers/cvrsgnsg/cvrsgnsg.pdf>.
- [48] J. Rushby, "Formalism in safety cases," in *Proceedings of the eighteenth safety critical systems symposium*, Feb. 2010, pp. 3–17.
- [49] —, "Formal methods and the certification of critical systems," Dec. 1993, technical Report CSL-93-7.
- [50] J. Woodcock, P. G. Larsen, J. Bicarregui, and J. Fitzgerald, "Formal methods: Practice and experience," *ACM Computing Surveys*, vol. 41, pp. 19:1–19:36.
- [51] S. Liu and R. Adams, "Limitations of formal methods and an approach to improvement," in *In APSEC95: Proceedings of the Second Asia Pacific Software Engineering Conference*. IEEE Computer Society, 1995, p. 498.
- [52] G. Dondossola, "Formal methods in the development of safety critical knowledge-based components," in *European Workshop on Validation and Verification of Knowledge-Based Systems*, 1998, pp. 232–237.
- [53] —, "Formal methods: Applications and technology," in *11th International Workshop, FMICS, 5th International Workshop PDMC 2006*, Aug. 2006, pp. 1–361, ISBN: 978-3-540-70951-0.
- [54] J. P. Bowen and M. G. Hinchey, *Application of Formal Methods*. Prentice Hall International Series in Computer Science, 1995, ISBN:0-13-366949-1.

REFERENCES

- [55] J. P. Bowen and V. Stavridou, "The industrial take-up of formal methods in safety-critical and other areas: A perspective," in *LNCS: Proceedings of the First International Symposium of Formal Methods Europe on Industrial-Strength Formal Methods*, vol. 670, Apr. 1993, pp. 183–195, ISBN: 978-3-540-56662-5.
- [56] M. Gogolla, "Benefits and problems of formal methods,lncs,europe," Jun. 2004, [Online].Available : [http : //www.springer.com/cda/content/document/cda_downloaddocument/9783540220114 - t1.pdf](http://www.springer.com/cda/content/document/cda_downloaddocument/9783540220114-t1.pdf).
- [57] J. C. Bicarregui and B. Matthews, "Formal methods in practice: A comparison of two support systems for proof," in *Proceedings of the 22nd Seminar on Current Trends in Theory and Practice of Informatics*, vol. 1012, Dec. 1995, pp. 184–205, ISBN: 978-3-540-60609-3.
- [58] S. Easterbrook and J. Callahan, "Formal methods for verification and validation of partial specifications: A case study," *Journal of Systems Software*, vol. 40, pp. 199–210, Mar. 1998, doi:10.1.1.98.768.
- [59] Z. H. Qureshi, "Formal modelling and analysis of mission-critical software in military avionics systems," in *Proceedings of the eleventh Australian workshop on Safety critical systems and software*, Aug. 2006, pp. 67–77.
- [60] Y. A. Ameer, F. Boniol, and V. Wiels, "Towards a wider use of formal methods for aerospace systems design and verification," in *International journal of software tools technology transfer*, vol. 12, Feb. 2010, pp. 1–12, ISSN: 1433-2779.
- [61] M. R. Prasad, ArminBiere, and A. Gupta, "A survey of recent advances in sat-based formal verification," in *Int J Softw Tools Technol Transfer*, Feb. 2005, pp. 156–173, 7.
- [62] P.-A. Hsiung, Y.-R. Chen, and Y.-H. Lin, "Model checking safety-critical systems using safecharts," in *IEEE Transactions On Computers*, Feb. 2007, pp. 692–705, 56(5).
- [63] C. Heitmeyer, "Developing safety-critical systems:the role of formal methods and tools," in *Proceedings of the 10th AustralianWorkshop on Safety Related Programmable Systems*, Aug. 2005, pp. 25–26.
- [64] J. Woodcock, P. G. Larsen, J. Bicarregui, and J. Fitzgerald, "Formal methods: Practice and experience," in *ACM Computing Surveys*, vol. 41, Aug. 2009, pp. 19:1–19:36, doi:10.1145/1592434.1592436.
- [65] J. L. Caldwell, "Formal methods technology transfer: A view from nasa," in *ACM Computing Surveys*, vol. 12, Aug. 2009, pp. 125–137, ISSN: 0925-9856.
- [66] A. Joshi, S. P. Miller, and M. P. E. Heimdahl, "Mode confusion analysis of a flight guidance system using formal methods," pp. 1–13, 2003, [Online].Available : [http : //shemesh.larc.nasa.gov/fm/papers/ModeConfusionAnalysisUsingFormalMethods.pdf](http://shemesh.larc.nasa.gov/fm/papers/ModeConfusionAnalysisUsingFormalMethods.pdf).
- [67] M. R. Prasad, ArminBiere, and A. Gupta, "A survey of recent advances in sat-based formal verification," in *Int J Softw Tools Technol Transfer*, 2005, pp. 156–173, vol 7.
- [68] T. Margaria and B. Steffen, "Leveraging applications of formal methods,verification andvalidation," in *Proceedings of third international symposium, ISO/FASEQ*, 2008, pp. 13–15.

REFERENCES

- [69] R. K. National, D. R. Kuhn, D. Craigen, and M. Saaltink, "Practical application of formal methods in modeling and simulation," in *Proceedings of SCSC'03 (invited)*, 2003.
- [70] S. P. Miller and R. Collins, "The industrial use of formal methods: Was darwin right?" in *Proceedings of the Second IEEE Workshop on Industrial Strength*, 1998, p. 74.
- [71] C. Antoine, P. Baudin, J. M. Collart, J. Raguideau, and A. Trotin, "Using formal methods to validate c programs," in *5th International Symposium on In Software Reliability Engineering*, Nov. 1994, pp. 252–258.
- [72] E. Vassev and M. Hinchey, "Developing experimental models for nasa missions with assl," in *Workshop on Formal Methods for Aerospace (FMA)EPTCS*, 2010, pp. 88–94, doi:10.4204/EPTCS.20.10.
- [73] A. Cimatti, M. Roveri, A. Susi, and S. Tonetta, "Formalization and validation of safety-critical requirements," in *Workshop on Formal Methods for Aerospace (FMA), EPTCS*, mar 2010, pp. 68–75, doi:10.4204/EPTCS.20.7.
- [74] K. Havelund, M. Lowry, S. Park, C. Pecheur, J. Penix, W. Visser, J. L. White, and R. Technologies, "Formal analysis of the remote agent before and after flight," in *Journal of Systems Software*, Jun. 2000, pp. 199–210.
- [75] I. Beer, S. Ben-David, D. Geist, R. Gewirtzman, and M. Yoeli, "Methodology and system for practical formal verification of reactive hardware," in *Proceedings of the 6th International Conference on Computer Aided Verification*, 1994, pp. 182–193.
- [76] L. B. J. PE and L. B. Associates, "System safety analysis pitfalls," in *Proceedings of the 15th International System Safety Conference*, Aug. 1997, pp. 1–8.
- [77] C. Ponsard, P. Massonet, J. F. Molderez, A. Rifaut, A. van Lamsweerde, and H. T. Van, "Early verification and validation of mission critical systems," *Journal in Formal Methods in system design*, vol. 30, pp. 233–247, Jun. 2007.
- [78] D. O. Defense, "Systems engineering fundamentals," [Online]. Available: <http://www.dau.mil/pubs/pdf/SEFGuide>
- [79] FAA, "Systems engineering process flowchart," [Online]. Available: <http://fast.faa.gov/flowcharts/testflow/syseng.htm>.
- [80] H. Mitsumaki, T. Ikeda, and R. Kodama, "Modularized system architecture for flexible clinical laboratory systems," in *Proceedings of the Fifth International Symposium on Autonomous Decentralized Systems*, Mar. 2001, pp. 111–118.
- [81] J. Harauz and P. T. Poon, "System engineering in the twenty-first century," in *Proceedings of the 4th IEEE International Symposium and Forum on Software Engineering Standards*, May 1999, p. 246.
- [82] M. Price, S. Raghunathan, and R. Curran, "An integrated systems engineering approach to aircraft design," in *Progress in aerospace sciences*, vol. 42, Jun. 2006, pp. 331–376.
- [83] H. L. Burks, "Systems engineering tools," in *Reliability and maintainability computer-aided engineering in concurrent engineering*, Oct. 1992, pp. 241–244, doi:10.1109/RMCAE.1992.245538.
- [84] C. D. Cernes, "The role of different engineering methods," in *IEE Colloquium on Systems Engineering for Profit*, Mar. 1995, pp. 7/1–7/4, doi:10.1049/ic:19950395.

REFERENCES

- [85] B. A. Laios, "Systems engineering methods in aerospace," in *IEE Colloquium on Integrating Issues in Aerospace Control*, 1991, pp. 57–59.
- [86] N. G. Leveson, *Safeware : System Safety and Computers*. Addison-Wesley, 1995, *IS BN* : 0–201–11972–2.
- [87] N. G. Leveson and J. L. Stolzy, "Analyzing safety and fault tolerance using time petri nets," in *IEEE National Aerospace and Electronics Conference (NAECON)*, Jul. 2012, pp. 171–177, *doi* : 10.1109/NAECON.2012.6531050.
- [88] F. A. Circular, "System design and analysis," Jun. 1998, [Online].Available : [http : //www.faa.gov/documentLibrary/media/Advisory_Circular/AC25.1309 – 1A.pdf](http://www.faa.gov/documentLibrary/media/Advisory_Circular/AC25.1309-1A.pdf).
- [89] J. P. Fielding and X. Z. Meng, "Some aspects of the design of a fly-by-wire flying control system for a supersonic v/stol fighter aircraft," Mar. 1984, [Online].Available : [http : //repository.tudelft.nl/.../College_of_Aeronautics_Report_No.8413.pdf](http://repository.tudelft.nl/.../College_of_Aeronautics_Report_No.8413.pdf).
- [90] G. Belcher, D. E. McIver, and K. J. Szalai, "Validation of flight critical control systems," Dec. 1991, [Online].Available : [http : //ftp.rta.nato.int/public/PubFullText/AGARD/AR/AGARD – AR – 274/AGARD – AR – 274.pdf](http://ftp.rta.nato.int/public/PubFullText/AGARD/AR/AGARD-AR-274/AGARD-AR-274.pdf).
- [91] C. Andre, "Semantics of s.s.m (safe state machine)," Apr. 2003, [Online].Available : [http : //www.i3s.unice.fr/map/WEBS_PORTS/Documents/2003a2005/SSM – abstract.pdf](http://www.i3s.unice.fr/map/WEBS_PORTS/Documents/2003a2005/SSM-abstract.pdf).
- [92] S. Prochnow, C. Traulsen, and R. von Hanxleden, "Synthesizing safe state machines from esterel," in *ACM SIGPLAN/SIGBED conference on Language, compilers, and tool support for embedded systems*, Jun. 2006, pp. 113–124, *doi* : 10.1145/1134650.1134667.
- [93] D. Latella, I. Majzik, and M. Massink, "Towards a formal operational semantics of uml statechart diagrams," in *Third International Conference on Formal Methods for Open Object-Based Distributed Systems(FMOODS)*, Feb. 1999, pp. 1–17, *IS BN* : 0 – 7923 – 8429 – 6.
- [94] M. Nanda and S. Rao, "A formal method approach to analyze the design of aircraft flight control systems," *3rd Annual IEEE International Systems Conference*, pp. 64–69, Mar. 2009.
- [95] A. D. Hills, "Digital fly-by-wire experience," in *Fault Tolerant HardwareSoftware Architecture For Flight Critical Function,AGARD Lecture Series*, Sep. 1985, pp. 1–18.
- [96] D. B. Thomas, W. Luk, P. H. W. Leong, and J. D. Villasenor, "Gaussian random generators," in *ACM Computing Surveys*, vol. 39, Oct. 2007, pp. 1–38, *doi*:10.1145/1287620.1287622.
- [97] D. Siewiorek and P. Narasimhan, "Fault-tolerant architectures for space and avionics applications," 2005, [Online].Available : [www.http : //ti.arc.nasa.gov/projects/ishem/Papers/Siewiorek_Fault_Tol.pdf](http://ti.arc.nasa.gov/projects/ishem/Papers/Siewiorek_Fault_Tol.pdf).
- [98] S. K. Giri, A. Mishra, Y. V. Jeppu, and D. K. Karunakar, "Stress testing control law code using randomized nrt testing," in *43rd AIAA Aerospace Sciences Meeting and Exhibit*, Jan. 2005, pp. 1–9, *doi*:10.2514/MASM05.
- [99] Y. V. Jeppu, "Flight control software: Mistakes made and lessons learned," *IEEE Software*, vol. 30, pp. 67–72, Jun. 2013, *doi*:10.1109/MS.2013.42.

REFERENCES

- [100] K. Rajalakshmi, Y. V. Jeppu, and D. K. Karunakar, "Ensuring software quality-experiences of testing tejas airdata software," *Defense Science Journal*, vol. 56, pp. 13–19, Jan. 2006, [Online]. Available:<http://publications.drdo.gov.in/ojs/index.php/dsj/article/view/1863/993>.
- [101] D. J. R. Pimentel, "An architecture for a safety-critical steer-by-wire system," in *Society of Automotive Engineers, Inc.*, Mar. 2004, pp. 1–9, [Online]. Available:<http://www.sae.org/technical/papers/2004-01-0714>.
- [102] P. Li, L.-M. Jia, and A.-X. Nie, "Study on railway intelligent transportation system architecture," in *IEEE Proceedings of Intelligent Transport Systems*, Oct. 2003, pp. 1486–1489, doi:10.1109/ITSC.2003.1252729.
- [103] M. Hecht, J. Agron, H. Hecht, and K. H. Kim, "A distributed fault tolerant architecture for nuclear reactor and other critical process control applications," in *Twenty-First International Symposium on Fault-Tolerant Computing*, Jun. 1991, pp. 462–498, doi:10.1109/FTCS.1991.146702.
- [104] N. G. Leveson and J. L. Stolzy, "Safety analysis using Petri nets," *IEEE Trans. Softw. Eng.*, vol. 13, pp. 386–397, Mar. 1987, doi : 10.1109/TSE.1987.233170.
- [105] R. Johansson, "A fault-tolerant architecture for computer-based railway vehicle brake systems," in *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit*, vol. 218, May 2004, pp. 189–201.
- [106] A. K. Ghosh, V. Rana, B. W. Johnson, and J. A. Profeta, "A distributed safety-critical system for real-time train control," in *21st International Conference on Industrial Electronics, Control, and Instrumentation*, Nov. 1995, pp. 760–767, doi:10.1109/IECON.1995.483824.
- [107] L. J. Feiner, "Power electronics transforms aircraft systems," in *Conference on Idea/Microelectronics WESCON/94*, Sep. 1994, pp. 166–171, doi:10.1109/WESCON.1994.403613.
- [108] L. J. Feiner, "Power-by-wire aircraft secondary power systems," in *12th Digital Avionics Systems Conference*, Oct. 1993, pp. 439–444.
- [109] L. H. Schiebe, "A distributed safety-critical system for real-time train control," in *National computer conference*, May 1980, pp. 135–138, doi:10.1145/1500518.1500543.
- [110] "ATR 600," Jul. 2012, [Online]. Available : <http://www.atraircraft.com>.
- [111] N. Aeronautics and S. Administration, "Systems engineering handbook, sp-610s," Aug. 2007, [Online]. Available:<http://snebulos.mit.edu/projects/reference/NASA-Generic/NASA-STD-8739-8.pdf>.
- [112] I. Bazovsky, *Reliability Theory and Practice*, ser. Dover Books on Mathematics. Dover Publications, 2004, ISBN: 9780486438672.
- [113] A. Birolini, *Quality and Reliability of Technical Systems: Theory, Practice, Management*, 4th ed. Springer-Verlag, 2004, ISBN: 9783540633105.
- [114] M. Stamatelatos, W. Vesely, J. Dugan, J. Fragola, J. Minarick, and J. Railsback, "Fault tree handbook for aerospace applications," *NASA Office of Safety and Mission Assurance*, Aug. 2002.
- [115] I. International Atomic Energy Association, "Applications of probabilistic safety assessment (psa) for nuclear power plants, iaea-tecdoc-1200, issn 1011-4289," Feb. 2001, [Online]. Available : http://www-pub.iaea.org/MTCD/publications/PDF/te_1200_prn.pdf.

REFERENCES

- [116] R. K. Guptan, S. P. Dharne, and S. G. Ghadge, "Probabilistic safety assessment - risk informed approach," in *2nd International Conference on Reliability, Safety and Hazard (ICRESH)*, Dec. 2010, pp. 133–135, doi:10.1109/ICRESH.2010.5779530.
- [117] G. Srinivas, R. Guptan, S. P. Dharne, S. G. Ghadge, and U. Chandra, "Hardware reliability assessment of safety related and safety critical systems in nuclear power plants," in *2nd International Conference on Reliability, Safety and Hazard (ICRESH)*, Dec. 2010, pp. 448–454, doi:10.1109/ICRESH.2010.5779591.
- [118] A. R. McSpadden and N. Lopez-Benitez, "Stochastic petri nets applied to the performance evaluation of static task allocations in heterogeneous computing environments," in *Proceedings of sixth heterogeneous computing workshop (HCW)*, Apr. 1997, pp. 185–194, doi : 10.1109/HCW.1997.581420.
- [119] L. K. Singh, G. Vinod, and A. K. Tripathi, "Modeling and prediction of performability of safety critical computer based systems using petri nets," in *23rd International Symposium on Software Reliability Engineering Workshops*, Nov. 2012, pp. 85–94, doi : 10.1109/ISSREW.2012.41.
- [120] F. N. Souza, R. D. Arteiro, N. S. Rosa, and P. R. M. Macie, "Using stochastic petri nets for performance modelling of application servers," in *20th International symposium on parallel and distributed processing*, Apr. 2006, pp. 1–8, doi : 10.1109/IPDPS.2006.1639650.
- [121] P. Zuliani, A. Platzer, and E. C. Clarke, "Bayesian statistical model checking with application to stateow/simulink verification," *Springer Formal Methods in System Design*, vol. 43, pp. 338–367, Oct. 2013, ISSN : 0925 – 9856.
- [122] A. Legay, B. Delahaye, and S. Bensalem, "Statistical model checking: An overview," *Springer Runtime Verification:LNCS*, vol. 6418, pp. 122–135, Nov. 2010, ISBN : 978 – 3 – 642 – 16611 – 2.
- [123] K. Sen, M. Viswanathan, and G. Agha, "Statistical model checking of black-box probabilistic systems," *Springer Computer Aided Verification:LNCS*, vol. 3114, pp. 202–215, Jul. 2004, ISBN : 978 – 3 – 540 – 22342 – 9.
- [124] H. L. S. Younes, M. Kwiatkowska, G. Norman, and D. Parker, "Numerical vs. statistical probabilistic model checking: An empirical study," *Springer Tools and Algorithms for the Construction and Analysis of Systems:LNCS*, vol. 2988, pp. 46–60, Apr. 2004, ISBN : 978 – 3 – 540 – 21299 – 7.
- [125] W. E. Vesley, F. F. Goldberg, N. H. Roberts, and D. F. Haasl, "Fault tree handbook," *Systems and Reliability Research , Office of Nuclear Regulatory Research ,U.S. Regulatory Comission*, Jan. 1981.
- [126] T. Bahil, R. Botta, and J. Daniels, "The zachman framework populated with baseball models," *Journal of Enterprise Architecture*, vol. 2, pp. 50 – –68, 2006.
- [127] A. Hosagrahara and P. Smith, "Measuring productivity and quality in model-based design," *MATLAB Digest*, Mar. 2006, [Online]. Available:<http://www.mathworks.com/company/newsletters/digest/2006/mar/measuringprod.html>.
- [128] F. Team, "Faa system safety handbook," [Online].Available : [http : //www.faa.gov/library/manuals/aviation/risk_management](http://www.faa.gov/library/manuals/aviation/risk_management).
- [129] M. Nanda and C. Jamadagni, "Quantitative metrics for validating the effectiveness of the model based approach for indigenously developed sws/aic system," *International Journal of Scientific and Engineering Research*, vol. 3, pp. 1–7, Dec. 2012.

REFERENCES

- [130] D. Milutinovic and V. Lucanin, "Relation between reliability and availability of railway vehicles," *Transaction on FME*, vol. 33, pp. 135–139, Sep. 2005.
- [131] S. Mitra, N. R. Saxena, and E. J. McCluskey, "A design diversity metric and analysis of redundant systems," *IEEE Transactions on Computers*, vol. 51, pp. 498–510, May 2002.
- [132] M. Nanda, C. S. Jamadagani, J. Jayanthi, and M. V, "Quantitative metrics for improving software performance for an integrated tool platform," in *IEEE System Conference*, Apr. 2013, pp. 512–518, doi : 978 – 1 – 4673 – 3108 – 1/13.
- [133] A. Joshi, M. P. E. Heimdahl, S. P. Miller, and M. W. Whalen, "Model based safety analysis," 2005, [Online]. Available:<http://shemesh.larc.nasa.gov/fm/papers/Model-BasedSafetyAnalysis.pdf>.
- [134] C. L. Heitmeyer, "Formal methods for specifying, validating, and verifying requirements," *J. UCS*, vol. 13, pp. 607–618, May 2007, [Online]. Available : [http : //www.jucs.org/jucs_13_5/formal_methods_for_specifying/jucs_13_5_0607_0618_heimmeyer.pdf](http://www.jucs.org/jucs_13_5/formal_methods_for_specifying/jucs_13_5_0607_0618_heimmeyer.pdf).
- [135] M. P. Heimdahl and N. Leveson, "Completeness and consistency in hierarchical state-based requirements," *IEEE Transactions on Software Engineering*, vol. 22, pp. 1–15, Jun. 1996, [Online]. Available:<http://sunnyday.mit.edu/papers/mats-tse.pdf>.
- [136] J. Jayanthi, M. Nanda, L. P, and S. G. Dhage, "efm conceptual design document," in *Project Document:NAL-eFM/CDD/2011/04*, Apr. 2011, pp. 1–49.
- [137] J. Jayanthi and M. Nanda, "Cdws conceptual design document," in *Project Document:NAL-CDWS/CDD/2011*, Oct. 2011, pp. 1–29.
- [138] G. Roedler and C. Jones, "Technical measurement : A collaborative project of psm, incose and industry," 2005, [Online]. Available : [http : //www.incose.org/ProductsPubs/pdf/TechMeasurementGuide_2005_1227.pdf](http://www.incose.org/ProductsPubs/pdf/TechMeasurementGuide_2005_1227.pdf).
- [139] J.-R. Abrial, *The B-Book: Assigning Programs to Meanings*. Cambridge University Press, 1996, doi:0-521-49619-5.
- [140] M. Barjaktarovic, "The state-of-the-art in formal methods," [Online]. Available : [http : //www.cs.utexas.edu/users/csed/formal-methods/docs/StateFM.pdf](http://www.cs.utexas.edu/users/csed/formal-methods/docs/StateFM.pdf).

Index

- Aerospace domain, 11, 50, 51, 57, 58, 60, 62, 71, 75, 91, 94, 127, 133, 137, 138, 159, 161
- Andre, Charles: Semantics of S.S.M (Safe State Machine), 57
- Automotive domain, 11, 57, 58, 80, 83, 86, 88, 127, 159, 161
- Availability, 3, 4, 30, 158
- Bahil *et al.*: The Zachman framework populated with baseball models, 112
- Benchmark, 162, 164
- Best and worst-case fault tolerance, 9, 47, 74, 77, 82, 84, 85, 87, 89, 90, 92, 93, 129, 132, 135, 142, 161
- Bottom-up approach, 9, 113, 116, 162, 163
- Complex critical systems, 14
- Conception phase, 19, 20
- Critical failure path, 77
- CSIR-NAL, 71, 75, 120, 127, 148, 157, 165
- Deployment phase, 122
- Design anomalies, 8
- Design faults, 4
- Design phase, 4, 14, 19, 20, 30, 61, 80, 117, 118, 122, 125, 160, 164
- Direct product, 9, 45, 48, 94, 163
- Direct sum, 9, 45, 48, 94, 163
- Document-centric engineering process, 2, 19, 30, 118
- Effective engineering process, 3, 7, 14, 120, 165
- Engineering lifecycle, 35, 122
- Engineering models, 25
- Estefan, Jeff: Survey of Model-Based Systems Engineering Methodologies, 19
- Fail-off system, 2, 3
- Fail-operational system, 2, 3, 160
- Fail-safe system, 2, 49, 160
- Failure rates, 9, 65
- Fault tolerance, 9, 43, 74, 77, 86, 144, 161
- Fault Tree Analysis, 7, 11, 57, 96, 106, 119, 121, 124, 125, 159, 164
- FM, 7, 19, 43
- FM workbench, 12
- Formal framework, 10
- Formal methods, 3
- Formal techniques, 34, 38, 42, 159, 164
- Formal workbench, 17, 157, 165
- Forward engineering, 10, 70, 75, 127
- Functional availability, 6–9, 11–15, 42–44, 51, 53, 54, 57, 58, 60–62, 70, 73, 74, 77, 80, 82, 84, 86, 88–90, 92–94, 96, 112, 120, 121, 125, 127,

- 129, 131, 133, 135–138, 143, 145–148, 157, 159–162, 164
- Functional requirements, 3, 4
- Functionality, 1, 2, 30, 35, 117, 118, 125, 157
- Hardware Reliability Analysis, 96
- Hazardous state, 8, 43, 44, 49–51, 53–55, 57, 61, 74, 82, 136, 143, 160
- Heitmeyer:Formal Methods for Specifying, Validating, and Verifying Requirements, 125
- High integrity systems, 1, 43
- Hosagrahara and Smith:Measuring Productivity and Quality in Model-Based Design, 118
- Implementation phase, 19, 20, 118, 122
- In-house tool, 17, 61, 62, 65, 147, 148, 150, 159, 164, 165
- Informal techniques, 159
- Integrated engineering process, 11
- Joshi *et al.*:Model based safety analysis, 124
- Leveson and Heimdahl: Completeness and Consistency in Hierarchical State-Based Requirements , 125
- Leveson, Nancy:Software safety, 2
- Maintainability, 3, 30, 117, 118, 126
- Maintenance phase, 19, 20, 139
- Mathematical operations, 45, 46, 73, 94, 163
- Measure of Effectiveness , 146
- Measure of Performance, 120, 146, 165
- Medical domain, 11, 57, 58, 80, 126, 139, 143, 159, 161
- Middle-out engineering, 10
- Model-based systems engineering, 19, 118
- Model-centric engineering process, 2, 19, 30, 118
- Modified systems engineering process, 14, 118, 157, 161
- Nanda and Rao:A Formal Method Approach To Analyze The Design Of Aircraft Flight Control Systems, 58
- Non-functional requirements, 4
- Non-operational system, 3
- Nuclear domain, 84, 127, 161
- Operational availability, 4, 5, 8, 120, 121, 139
- Paradigm shift, 11, 118, 121
- Probabilistic Safety Assessment, 96
- Process automation, 17
- Random test cases, 65
- Rao,S: A Systems Algebra and Its Applications, 11, 47, 82
- Reachability, 3, 4, 30, 35, 117, 118
- Redundancy, 77, 86, 131
- Relationship, 45
- Reliability, 3, 14, 30, 35, 118, 125, 157, 160
- Reliability Block Diagram, 7, 11, 57, 96, 99, 119, 124, 125, 159, 165
- Requirement phase, 3, 19, 20, 118, 122, 160
- Reverse engineering, 10, 11, 58, 80, 93, 125, 138, 159, 161
- Rushby, John: Formalism in Safety Cases, 3
- Rushby, John:Formal Methods and the Certification of Critical Systems, 34

INDEX

- SA, 8, 10, 13, 14, 43, 44
- SA approach, 70, 78
- SA expression, 8, 9, 11, 13, 46, 48, 49, 58, 60, 61, 63, 64, 73, 77, 78, 80–83, 85, 87–89, 91, 92, 96, 113, 129, 131, 135, 136, 138, 141–144, 157, 163
- SA technique, 12, 57, 58, 62, 80, 121, 125, 159, 161
- Safe state, 8, 43, 44, 50, 51, 53–55, 61, 78, 86, 89, 90, 92, 136, 139, 143, 160, 164
- Safety, 1–3, 14, 30, 35, 43, 80, 117, 118, 125, 157, 160
- Safety critical systems, 2, 10, 11, 34, 36, 43, 50, 51, 57, 58, 159, 161, 164, 165
- Safety-critical systems, 160, 161
- Security, 1–3, 30, 35, 117, 118, 126
- Semi-formal techniques, 159
- Specification phase, 160
- State transitions, 11, 44, 51, 52, 55, 57, 58, 60, 61, 68, 70, 78, 96, 121, 136, 164
- Statistical Model Checking, 96
- Stochastic Petri Nets, 96
- System Algebra, 8
- System composition, 9, 43, 45
- System decomposition, 8, 9, 45, 60, 161, 162
- System properties, 1
- System state machine, 8, 9, 50–52, 54, 55, 62, 144, 159
- System states, 9, 11, 43, 52, 58, 65, 73, 80, 94, 138, 164
- Systems engineering, 1, 19, 29
- Top-down approach, 9, 106, 113, 116, 162
- Tree diagram, 78, 136
- Unreliable state, 58
- Unsafe state, 8, 43, 44, 49–51, 53, 55, 57, 61, 62, 74, 78, 82, 86, 89, 90, 92, 136, 139, 160, 164
- Verification and Validation, 3, 122
- Woodcock *et al.*: Formal Methods: Practice and Experience, 36