Abstract

Software reliability is one of the most important software quality attributes. It is an external quality attribute, which relates internally to the notion of program faults or defects. In order to assess software reliability, one of methods is to apply Software Reliability Growth Models (SGRM). More than 200 SGRM exist. However these approaches are black-box based i.e., the software system is considered as a whole and only its interactions with the outside world are modeled, without looking into its internal structure. These are mostly employed at the later stage of the software development life cycle, before which many critical design decisions are made. Identification of significant problems during implementation or operation can lead to re-engineering of large parts of the system, which has been shown to be prohibitively costly. Further they are based on unrealistic assumptions. Therefore, several recent approaches have begun to quantify software reliability at the level of architectural models, or at least in terms of high-level system structure. Some of the existing approaches are based on Markov Model. It has been experienced that mostly software fails because of ambiguity in the requirements or defect in the design. Ambiguous requirements may also lead to the design defect. So, a model should be explainable to all the stakeholders to understand all the functional requirements. Further it will give an assurance to the client that all the functional requirements of software have been taken care of. UML has proved to be a general-purpose modeling language in the field of software engineering that can be understood by all the stakeholders. UML has an ability to model the scenarios of the system.

The possibility to extend the UML to convert into a probabilistic reliability model is shown. The converted reliability model is a Markov Model, which can be easily plugged into the existing approaches of reliability prediction based on Markov Model.

The limitations of the Markov reliability Model for early prediction are also concluded. They are incapable to model the concurrency and typically limited to modeling the probability of changes in the system with the exponential distribution. Further, in these models, the reliability is the function of transition probabilities in between the states of the Markov Chain, which are either assumed or computed based on the operational profile. Analytical approach of predicting transition probabilities does not give accurate prediction. On the other hand, operational profile can be collected in the last stage of the software development life cycle or during the software operational time. Hence transition probabilities, based on operational profile does not provide early estimate of reliability.

It is shown that these limitations can be addressed by modeling a system through Petri net. We use the tool TimeNET for steady state distribution of the transition rates in between the states of Markov model, through which the transition probabilities have been derived.

Additionally, the reliability requirements of each component of a software system may not be necessarily same. Also during the system operation, its dynamics changes, which may affect the reliability of the components and hence of the overall system. Therefore, there is a requirement to have a methodology for updating the reliabilities of each component and system as well for corrective action. This motivates us to propose an approach for reliability updation of the components. Bayesian approach is used to address these issues.

All the above proposed approaches are validated with the real time safety critical system of Indian Nuclear Power Plant along with some noticeable findings.

Acknowledgements

Foremost, I would like to express my sincere gratitude to my research supervisors Prof. Anil Kumar Tripathi and Dr. Gopika Vinod for the continuous support of my Ph.D study and research, for his patience, motivation, enthusiasm, and immense knowledge. Their guidance helped me in all the time of research and writing of this thesis. Without their support and active participation in every step of the process, this thesis may never have been completed.

Besides my advisor, I would like to thank all the members of Research Program Committee and Department Post Graduate Committee for their encouragement, insightful comments, and conceptual questions.

My sincere thanks also goes to Prakash Tamboli, Scientist in NPCIL, Department of Atomic Energy, Government of India and PhD student of IIT Bombay for sparing his precious time in discussion to solve the complex mathematical problems, related to my work. I also thank Dr. Manoj Kumar, Scientist/F, Reactor Control Division, Bhabha Atomic Research Centre, Mumbai, India; for suggesting me good literature and helping me through his invaluable suggestions.

I would like to dedicate this thesis to the women in my life, my wife Pooja Singh and my son; Shaival Singh for without their love and understanding this effort could not have been accomplished. Pooja has inspired me at different times and in different ways. Pooja's constant support and motivation; her prayers for me, Shaival's good morning hugs and smiles have all helped me to persevere.

Last but not the least; I would like to thank my family: my parents Prof. Himmat Singh and Mrs. Maheshwari Singh, for giving birth to me at the first place and supporting me spiritually throughout my life; my younger brother and his wife for their love and affection; my in-laws Mr. D.R. Singh and Mrs. Bhanukumari who had taken care un-tirelessly to my son during my and Pooja's research period.

Contents

1 Introduction **1**

1.1 Need for early prediction of reliability for software systems 5

1.2 Motivation of Research 6

1.3 Objectives of the present work 8

1.3.1 Uncertainty in Markov reliability models 8

1.3.2 Uncertainty in input parameters in Markov reliability models and their limitations 9

1.3.3 Impact analysis of component reliability on the reliability of a software system 9

1.4 Scope of the Research 10

1.5 Thesis Outline 10

2 Literature Survey **12**

2.1 Introduction 12

2.2 Dependability 12

2.3 Classification of Computer Based Systems 16

2.4 Hardware and Software Reliability 17

2.5 Fundamentals of Software Reliability 18

2.5.1 Time to failure 19

2.5.2 Reliability function 19

2.5.3 Hazard Rate 20

2.6 Probability Distributions and Reliability Functions 23

2.6.1 Exponential Distribution 23
2.6.2 Weibull Distribution 23
2.6.3 Rayleigh Distribution 24
2.7 Classification of Software Reliability Models 25
2.7.1 Data-domain models 26
2.7.1.1 Fault-seeding models 26
2.7.1.2 Input-domain models 26
2.7.1.3 Time-domain models 27
2.7.1.4 Bayesian software reliability growth models 27
2.7.1.5 Other models 27
2.8 An Overview of Software Reliability Early Prediction 27
2.8.1 Limitations of existing approaches 30
2.8.1.1 Modeling Limitations 31
2.8.1.2 Analysis Limitations 31
2.8.1.3 Parameter Estimation Limitations 32
2.8.1.4 Validation Limitations 32
2.8.1.5 Optimization Limitations 32

3 Problem Formulation and Solution Strategies 33

3.1 Introduction 33
3.2 Problem Formulation 35
3.2.1 Uncertainty in Markov reliability models 35
3.2.2 Uncertainty in input parameters in Markov reliability models and their limitations 35
3.2.3 Impact analysis of component reliability on the reliability of a software system for preventive action 36
3.2.4 Updation of components reliability during testing / operational phase for corrective action 36
3.3 Solution Strategies 37
3.3.1 Strategy for dealing with the uncertainty in Markov reliability models 37

3.3.2 Strategy for dealing with the uncertainty in Markov reliability models and their limitations 38
3.3.3 Strategy for dealing with impact analysis of component reliability on the reliability of a software system for preventive action 38
3.3.4 Strategy for updation of components reliability during testing/operational phase for corrective action 39

4 UML Approach to Markov Reliability Modeling 41
4.1 Introduction 41
4.2 A Case Study 43
4.2.1 Short description of ECCS 43
4.2.2 ECCS Instrumentation 44
4.2.3 Control logic module 45
4.3 Defining attributes into scenario specifications 46
4.3.1 Basic Message sequence Chart (BMSC) 47
4.3.2 High-level Message Sequence Charts (HMSC) 48
4.3.3 Labeled Transition Systems (LTS) 48
4.4 A Technique for Early Software Reliability Prediction 49
4.4.1 Phase1: Scenario Specification 50
4.4.2 Phase2: LTS components identification 51
4.4.3 Phase3: Reliability model creation 57
4.4.4 Phase4: Reliability prediction 58
4.5 Sensitivity Analysis 61
4.6 Experimental Validation 63
4.7 Conclusion 65

5 Transition Probability Prediction in MC 66
5.1 Introduction 66
5.2 Related work 67
5.3 The proposed method for transition probability prediction 68

5.3.1 Phase1: Petri net model creation 69
5.3.2 Phase2: Model Parameter assignment 69
5.3.3 Phase3: Reachability graph creation 69
5.3.4 Phase4: Markov Chain creation 70
5.3.5 Phase5: Transition Probability estimation 71
5.4 A Case Study 72
5.4.1 ECCS Design Requirements 72
5.4.2 Test Facility 74
5.4.3 Test Facility Architecture 75
5.4.4 Communication Module in TF 76
5.5 Application of proposed framework for prediction of transition probabilities in Markov reliability model 77
5.5.1 Phase 1: Petri net model creation 77
5.5.2 Phase2: Model Parameter assignment 78
5.5.3 Phase3: Reachability graph creation 79
5.5.4 Phase4: Markov Chain creation 79
5.5.5 Phase5: Transition Probability estimation 81
5.6 Validation of our approach 82
5.6.1 Reliability estimation, based on the predicted transition probabilities 82
5.6.2 Reliability estimation, based on the operational profile data . . . 85
5.6.2.1 Phase 1: Data Collection 85
5.6.2.2 Phase 2: Data Analysis 86
5.6.2.3 Phase 3: Reliability Computation 86
5.6.2.4 Phase 4: Reliability Comparison 87
5.7 Sensitivity Analysis of Parameter assignment 89
5.8 Performance estimation 89
5.9 Conclusion 91

6 Dynamics in Modeling for Reliability Prediction 97
6.1 Introduction 97

6.2 Inference via Bayesian Networks 100
6.3 A Case Study 101
6.4 Reliability Estimate Updation 101
6.5 Experimental Validation 105
6.6 Conclusion 108

7 Conclusion and Future Research 109

7.1 Uncertainty in Markov reliability models 110
7.2 Uncertainty in input parameters in Markov reliability models and their limitations 110
7.3 Impact analysis of component reliability on the reliability of a software system for preventive action 111
7.4 Updation of components reliability during testing/ operational phase for corrective action 112
7.5 Future Work 112

Appendices 114

A Software Reliability Growth Models 115

A.1 Data Domain Models 115
A.1.1 Fault-seeding models 115
A.1.1.1 Mills's Hypergeometric Model 115
A.1.2 Input-domain models 117
A.1.2.1 Nelson Input Domain Model 117
A.1.2.2 Brown and Lipow Input Domain Model 117
A.1.2.3 Ramamoorthy and Bastani Input Domain Model 118
A.2 Time-domain models 120
A.2.1 Homogeneous Markov Models 121
A.2.2 Non-Homogeneous Markov Models 121
A.2.3 Semi-Markov Models 122
A.2.4 Jelinski-Moranda de-eutrophication model 122

A.2.5 Goel-Okumoto Imperfect Debugging Model 124

A.2.6 Schick and Wolverton Model 125

A.2.7 Finite Failure NHPP Models 125

A.2.8 State-space view of NHPP 127

A.2.9 Infinite Failure Models 131

A.2.9.1 Musa-Okumoto Logarithmic Poisson Execution Time Model132

A.2.9.2 Duane Model 132

A.2.9.3 Log-Power NHPP Model 133

A.3 Bayesian software reliability growth models 133

A.4 Other Models 136

A.4.1 Error Complexity Model 136

A.4.2 Littlewood-Verall (LV) Bayesian Model 137

A.4.3 Littlewood and Keiller (LK) Bayesian Model 138

B TimeNET TOOL 139

B.1 Introduction 139

B.2 System Requirements 140

B.3 Downloading TimeNET 140

B.4 How to Install the Tool 141

B.5 Configure a multi-user installation 142

B.6 Starting the Tool 142

B.7 Configuring the User Interface 142

B.8 Upgrading to TimeNET 4.0 142

B.8.1 Conversion of old Model Files 143

List of Figures

2.1 The dependability tree 13

2.2 Three-universe model 14

2.3 Bathtub curve for hardware reliability 17

2.4 Bathtub curve for software reliability 18

2.5 Classification of software reliability models 25

2.6 Probabilistic model for reliability analysis 28

2.7 DTMC model of an example 29

2.8 Probabilistic control flow graph of example application software 30

2.9 Classification of architecture-based software reliability models 30

3.1 Issues and proposed solutions for uncertainty in probabilistic models of software reliability and its estimates 40

4.1 Mimic diagram of ECCS 45

4.2 Architecture of valve control logic 46

4.3 Activity diagram for valve actuation 47

4.4 Sequence diagram for valve actuation 47

4.5 Framework for reliability prediction 50

4.6 Modified Activity diagram for valve actuation with transition probabilities 51

4.7 Modified Sequence diagram for valve actuation 52

4.8 Probabilistic-LTS for control logic 56

4.9 probabilistic-LTS for control logic with transitions 58

4.10 System Reliability as a function of components reliability 62

4.11 System Reliability as a function of transition reliability 62

5.1 Transition Probability prediction framework 69

5.2 Architecture of Test Facility System 75

5.3 SPN of Embedded Unit 78

5.4 Reachability graph 79

5.5 Markov chain creation 81

5.6 Reliability Computation framework 85

5.7 Throughput change for t_{com1} 90

5.8 Transition probability change for t_{com1} 90

6.1 BN representation of IC 101

6.2 Failure probability of A, B and C 108

A.1 Non-Homogenous Markov chain 122

A.2 Semi-Markov chain 122

A.3 Non-Homogenous Markov chain 128

A.4 Comparison of analytical and numerical mean value functions 131

List of Tables

4.1 Failure classification of system 42
4.2 Transition probabilities between activities of HMSC 51
4.3 Reliabilities of components of BMSC 51
4.4 Operational profile of 1 year for valve operation logic 65

5.1 Failures in NPP 73
5.2 Components of EU 75
5.3 EU places and transitions 93
5.4 EU transitions with delay 93
5.5 Throughput of the transitions 94
5.6 Markings of EU GSPN model 94
5.7 Command Messages and acknowledgement messages of Communication . 95
5.8 Operational profile Data 96
5.9 Accuracy of predicted unreliability figure of communication module for given seven stations 96

6.1 Statistical Parameters of S_A, S_B and W 106
6.2 Failure Probability of components A and B as a function of system C . . 107

List of Abbreviations

AIM	Analog Input Module
BMSC	Basic Message Sequence Chart
BN	Bayesian network
CBS	Computer-Based System
DCC	Digital Control Computer
DIM	Digital Input Module
DTMC	Discrete Time Markov Chain
DU	Display Unit
ECCS	Emergency Core Cooling System
EPROM	Erasable Programmable Read Only Memory
EU	Embedded Unit
FSP	Finite State Process
HMSC	High-level Message Sequence Chart
LAN	Local Area Network
LOCA	Loss of coolant accident
LT	Level Transmitter
LTS	Labeled Transition Systems
MC	Markov Chain

MCS	Monte Carlo simulation
NPP	Nuclear Power Plant
NT	Nitrogen Tank
RD	Rupture Disc
ROM	Relay Input Module
RT-20	Real Time-20
RTD	Resistance Temperature Detector
SDLC	Software Development Life Cycle
SRE	Software Reliability Engineering
SRGM	Software Reliability Growth Models
TF	Test Facility (System)

List of Symbols

X_i	node i in BN
X	Set of nodes in BN
$P(X)$	Joint probability
S_N	Strength of software component N
W	Applied load on the software component
$P(X_i)$	Marginal probability of X_i
$P(X = F)$	Probability that X fails
$f(x)$	probability density function of x
p_{ij}	Probability of transition from state i to state j
q_{ij}	Transition rate from state i to state j
$p_i(t)$	Probability that a component is in state i
R_{com}^{est}	Estimated Reliability of communication module using MC
R_{com}^{act}	Actual Reliability of communication module using operational profile
R_{com}^{diff}	Difference in R_{com}^{est} and R_{com}^{act}
UR_{com}^{est}	Estimated Unreliability of communication module using MC
UR_{com}^{act}	Actual Unreliability of communication module using operational profile
UR_{com}^{diff}	Difference in UR_{com}^{est} and UR_{com}^{act}