Research Report Series
Lehrstuhl für Rechnertechnik und
Rechnerorganisation (LRR-TUM)
Technische Universität München

http://wwwbode.in.tum.de/

Editor: Prof. Dr. A. Bode

Vol. 34

Application-oriented evaluation of fault-tolerant systems

Max Walter

SHAKER
VERLAG
Aachen 2009

Bibliographic information published by the Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at http://dnb.d-nb.de.

Zugl.: München, Techn. Univ., Habil.-Schr., 2008

Printed in Germany.

ISBN 978-3-8322-8227-1
ISSN 1432-0169

Shaker Verlag GmbH • P.O. BOX 101818 • D-52018 Aachen
Phone: 0049/2407/9596-0 • Telefax: 0049/2407/9596-9
Internet: www.shaker.de • e-mail: info@shaker.de

Abstract

In fault-tolerant systems, redundancy in terms of hardware, software, repeated computations, or additional information is used to increase dependability. As redundancy is costly and might influence performance in a negative way, stochastic dependability models are used for a quantitative assessment of attributes like reliability, safety, and availability.

The thesis first summarizes the existing modeling concepts for fault tolerant systems, namely combinational methods (e.g. fault trees and reliability block diagrams), state-based models (e.g. Markov chains, stochastic Petri nets, and models based on a stochastic process algebra), as well as hybrid methods, which combine different kind of modeling paradigms.

Furthermore, it describes a novel, application-oriented approach for the evaluation of fault-tolerant systems. In this approach, the system is modeled using a high-level, application specific input model, which is automatically transformed into a lower-level formal model. Using existing software packages, the formal model is in turn transformed into a mathematical model which can be analyzed numerically. The results of this evaluation are presented within the scope of the high-level input model, though.

The novel approach is able to calculate the overall system's dependability from information given on the components it is built of, including information on the components themselves (e.g. MTTF- and MTTR-values), information on which combination of component failures imply a system failure (i.e. the redundancy structure of the system), as well as information on inter-component dependencies like failures with a common cause, failure propagation, different kind of redundancy strategies, non-dedicated repairmen and so on.

After first describing the basic design principles, the thesis also describes four specific tools which have been implemented according to these principles. The Simple but Extensive, Structured Availability Modeling Environment (OpenSESAME) was developed for the evaluation of High-Availability systems, The Safety Modeling Environment (SafeME) is tailored towards safety-critical systems, Information Flow Diagrams (IFD) are used to model emergency shutdown systems, and The Copula-

BAsed Reliability and Availability Modeling Environment (COBAREA) is intended for the analysis of fault-tolerant digital circuits.

In general, the evaluation of stochastic dependability models is very demanding in terms of CPU-time and memory. To alleviate this problem, the thesis also presents a novel divide-and-conquer algorithm allowing to divide large dependability models into independent parts which can be analyzed separately. The algorithm was applied in the tools mentioned above, but could be reused in similar evaluation approaches, as well.

Acknowledgements

This work would not have been possible without the support and encouragement of my supervisor Arndt Bode, chair of the "Lehrstuhl für Rechnertechnik und Rechnerorganisation" where I have been working under excellent conditions during the last years.

Markus Siegle, the second supervisor of this thesis, has also been abundantly helpful has an advisor for this thesis, and has assisted me in numerous ways, especially by introducing me to the area of stochastic process algebras and by providing access to the tool CASPA.

I would also like to thank Helmut Seidl for being my "Fachmentoratsvorsitzender". During multiple discussions, he gave me many advice concerning the more practical and organizational aspects of this habilitation.

The work presented in this thesis is based on previously published contributions to conference proceedings and scientific journals (see Appendix B on page 155). I therefore thank all the co-authors of these papers, namely Hicham Belhadaoui, Arndt Bode, Günter Graf, Sebastian Esch, Wolfgang Karl, Kai Lampka, Markus Leberecht, Philipp Limbourg, Olaf Malassé, Michael Pock, Markus Siegle, and Carsten Trinitis, for their indirect contributions to this book.

Moreover, I would like to thank the students which have contributed to my research in terms of a diploma or master thesis, namely Michael Borgwardt, Avni Islamaj, Nils Nitsch, Martin Pichler, Michael Pock, and Qi Zhu.

I express my gratitude to the fault-tolerance community and am very glad that I was able to discuss my work with excellent scientists such as Jean-François Aubry, Fevzi Belli, Marc Bouissou, Gregory Buchheit, Salvatore Distefano, Klaus Echtle, Irene Eusgeld, Felix Freiling, Bernhard Fechner, Karl-Erwin Großpietsch, Jörg Keller, Hans-Dieter Kochs, Erik Mähle, Winfried Schneeweiss, and Peter Sobe.

I would especially like to thank Christoph Lindemann and his team for providing access to the software tool DSPNexpress.

My current and former colleagues at the "Lehrstuhl für Rechnertechnik und Rechnerorganisation" were always available for discussions concerning computer science

and other interesting topics. Dear Georg Acher, Kai Bader, Florian Bernstein, Andrea Bör, Rolf Borgeest, Renate Brunnhuber, Rainer Buchty, Jan-Thomas Czornack, Detlef Fliegl, Karl Fürlinger, Ivan Gergintchev, Michael Gerndt, Helmar Göttsch, Stephan Graf, Hans Hacker, Houssam Haitof, Beate Hinterwimmer, Jürgen Jeitner, Wolfgang Karl, Elfriede Kelp, Edmond Kereku, Tilman Küstner, Tobias Klug, Steffi Lämmle, Tianchao Li, Robert Lindhof, Alexandra Linke, Thomas Ludwig, Peter Luksch, Harald Meier, Martin Mairandres, Hamza Mehammed, Barbara Nishnik, Jürgen Obermeier, Michael Ott, Josef Niedermeier, Bruno Piochacz, Hans Pongratz, Sebastian Pätzold, Sabine Rathmayer, Peter Saiko, Andreas Schmidt, Martin Schulz, Karl-Heinz Seubert, Alexandros Stamatakis, Daniel Stodden, Jie Tao, Klaus Tilk, Carsten Trinitis, Josef Weidendorfer, Roland Wismüller and Nik Wurm: it has been a pleasure and honor working with you, thanks a lot!

Last but not least I thank my lovely wife Kamila for all the support she gave me while I have been working on this thesis.

München, November 2008,

Max Walter

Contents

1 Introduction **1**
1.1 Fault-tolerant systems 1
1.2 Model-based evaluation 3
1.3 Contributions and structure of this thesis 5

I Foundations **7**

2 Combinational methods **9**
2.1 Dependability measures 9
2.2 Basic probability calculus 11
2.3 Graphical representations 14
2.4 Quantitative evaluation 16
2.5 Limitations 24

3 State-based methods **25**
3.1 Markov chains 25
3.2 Stochastic Petri nets 27
3.3 Models based on stochastic process algebras 32
3.4 Limitations 35

4 Hybrid methods **41**
4.1 Hybrid hierarchic models 41
4.2 Model transformation 42

II The application-oriented approach **51**

5 Basic design principles **53**
5.1 Intuitive user interfaces 53
5.2 Separation of different system aspects 54
5.3 Specific user-interfaces 55
5.4 Hourglass design 55

5.5 Usage of formal state-based models and existing analysis tools 56
5.6 Automated model decomposition 56

6 The simple but extensive, structured availability modeling environment (OpenSESAME) 59
6.1 Introduction 59
6.2 Input model 60
6.3 Transformation process 68
6.4 Case studies 76
6.5 Summary of Chapter 6 87

7 The safety modeling environment (SafeME) 89
7.1 User interface 90
7.2 Extensions of the intermediate model 91
7.3 Case study: fault tolerant temperature control system 93
7.4 Summary of Chapter 7 101

8 Information flow diagrams 105
8.1 Introduction 105
8.2 An emergency stop system 105
8.3 Information flow diagram of the example 108
8.4 Definition of IFD-nodes 110
8.5 Rules of DEC-nodes 112
8.6 Model analysis 112
8.7 Summary of Chapter 8 114

9 The copula-based reliability and availability modeling environment (COBAREA) 115
9.1 Introduction 115
9.2 Formal problem statement 116
9.3 Overview of the solution process 119
9.4 Implementation 121
9.5 Evaluation of the example systems 127
9.6 Summary of Chapter 9 128

10 Solving submodels independently 131
10.1 Introduction 131
10.2 Formal problem statement 132
10.3 Efficient implementation 134
10.4 Evaluation & comparison 137
10.5 Summary of Chapter 10 141

III Conclusion & outlook **143**

11 Conclusion **145**

12 Outlook **149**

Bibliography **157**

List of Figures

1.1 Non-functional properties. 4

2.1 Ccdf of a negative exponential distribution. 9
2.2 Alternate states of a repairable system. 10
2.3 Fault tree of a TMR-system. 15
2.4 Reliability block diagram of a TMR-system. 15
2.5 A non-series-parallel reliability block diagram. 16
2.6 The construction of the EED and its attribution with a Boolean term. 18
2.7 Binary decision diagram (BDD). 21
2.8 Numerical evaluation of a BDD. 22
2.9 Approximative common cause failure model using a fault tree. 24

3.1 State space of a single component. 25
3.2 State space of a system comprising two components. 26
3.3 A Petri net equivalent to the state machine from Fig. 3.2. 30
3.4 Initial marking and two subsequent markings. 31
3.5 Decomposition of an RBD into a main diagram and a lower-level diagram. 36
3.6 Petri net with shared places. 37
3.7 Interrelations between modules. 38

4.1 Gates of a dynamic fault tree. 44
4.2 Markov chains as used in a BDMP. 45
4.3 A fault tree as used in a BDMP. 46
4.4 Generic DRBD dependency between driver and target. 47
4.5 DRBD dependency edge modeling failure propagation. 48
4.6 DRBD dependency edge modeling load sharing. 48
4.7 DRBD model of a system with cold-standby redundancy. 49

5.1 A top level fault tree referring to two second level trees. 53
5.2 Non-hierarchic fault tree including a failure propagation. 54
5.3 Hierarchic model equivalent to the one from Fig. 5.2, using an FDD. 55
5.4 The hourglass design. 56

5.5 Decomposable and non-decomposable DFT. 57

6.1 A reliability block diagram with two terminal pairs. 61
6.2 Component arrays used in an OpenSESAME block diagram. 62
6.3 Hierarchic block diagram as used in OpenSESAME. 63
6.4 Four components forming a standby-redundant system. 63
6.5 Failure dependency diagrams (FDD). 64
6.6 Imperfect coverage: failure dependency diagram and block diagram. . 65
6.7 Modularization of a strongly linked FDD. 65
6.8 Component arrays used in an OpenSESAME FDD. 66
6.9 Generic model of a k-out-of-N:G system with blocking failure propagation. 67
6.10 Overview on the transformation process of OpenSESAME. 68
6.11 Data flow diagram of OpenSESAME's transformation process. 69
6.12 A model with two sets of inter-dependent components. 70
6.13 Binary decision diagram of the system shown in Fig. 6.12. 71
6.14 UML object diagram of the component-oriented data structures. . . . 72
6.15 Petri net of a component as generated by OpenSESAME. 73
6.16 Structure of a typical distributed web server. 77
6.17 Reliability block diagram of the distributed web server. 78
6.18 Failure dependency diagram of the web server. 79
6.19 Schematic drawing of the adjunct processor. 80
6.20 Redundancy structure of the adjunct processor. 81
6.21 Modified block diagram including fail-over times. 82
6.22 A fault-tolerant water supply system. 84
6.23 Reliability block diagram of the water supply system. 85
6.24 Failure dependency diagram of the water supply system. 85
6.25 Unavailability and down-time of the water supply system. 86

7.1 Generic model of a safety-critical system. 89
7.2 UML class diagram of the intermediate model. 92
7.3 Safety-critical temperature control system. 92
7.4 Petri subnet of an undesirable event as generated by SafeME. 100
7.5 Petri net of a component as generated by SafeME. 102

8.1 Emergency stop system of a chemical reactor. 106
8.2 Partial fault trees of the emergency stop system. 108
8.3 An information flow diagram. 109
8.4 A fine grain model as used in the IFD-approach. 111

9.1 Reliability block diagram of the majority voting unit. 117

9.2 Majority voting unit with correlated neighbors. 118
9.3 Computer network with correlated links. 119
9.4 Overview of the solution process of COBAREA. 120
9.5 Algorithm for conjunction term creation as used in COBAREA. . . . 123
9.6 Algorithm including counting the appearance of terms. 125
9.7 Dependency diagram as used in COBAREA. 126
9.8 Dependency graph for the network example from Fig. 9.3. 127

10.1 A multiple binary decision diagram. 134
10.2 BDD minimization rules. 135
10.3 Breadth-first search in an EED. 137
10.4 A set of benchmark RBDs taken from the literature. 138
10.5 RBD of a parallel system comprising k series system. 139
10.6 Evaluation results for the proposed algorithm. 140

List of Tables

3.1 Graphical representation of Petri net components. 29

6.1 Exemplary component table with four different component types. . . 60
6.2 Component table of the distributed web server. 76
6.3 Evaluation results for the web server (no dependencies). 78
6.4 Evaluation results for the web server (including dependencies). 79
6.5 Component table of the adjunct processor. 79

7.1 Component table of the temperature control system. 95
7.2 External faults of the temperature control system. 96

9.1 Evaluation results for the network example. 128
9.2 Evaluation results for the IC-based majority voter. 129

10.1 Resources needed for the analysis of the benchmark RBDs. 139
10.2 Evaluation results for the example `ParSer(k,n)`. 140
10.3 Computation times for the example `Web(k,n)`. 141

Abbreviations & notation

A	(steady-state) availability
$A(t)$	transient availability at time t
AP	adjunct processor
APP	application
AdvancedTCA	advanced telecom computing architecture
ASU	actor/sensor unit
BDD	binary decision diagram
BDMP	Boolean-logic driven Markov process
COBAREA	The Copula-based Reliability and Availability Modeling Environment
CT	conjunction term
CU	control unit
DB	database
DEC-node	decision node (in an IFD)
DFT	dynamic fault tree
DNF	disjunctive normalform
DNS	domain name service
DRBD	dynamic reliability block diagram
EED	edge expansion diagram
ESDU	emergency shutdown unit
FT	fault tree
FDD	failure dependency diagram
FDS	failure dependency source
FDT	failure dependency target
FMEA	failure mode effects analysis
FP	failure propagation
FPIP	failure propagation input place
FPOP	failure propagation output place
FRU	field replaceable unit
GSPN	generalized stochastic Petri net
GUI	graphical user interface
IC	intergrated circuit
ID	interrelation diagram

IFD	information flow diagram
IO or I/O	input/output
IRS	interrelation source
IRT	interrelation target
HTTP	hyper-text transport protocol
HSC	hotswap controller
λ	failure rate
LAN	local area network
MIL-HDBK	military handbook
MTBF	mean time between failures
MTTF	mean time to failure
MTTR	mean time to repair
μ	repair rate
NOF	number of failures
P	Petri net place
PN	Petri net
$R(t)$	reliability at time t
RAID	redundant array of independent disks
RBD	reliability block diagram
RTB	rear transition board
SBC	single board computer
SafeME	The Safety Modeling Environment
SESAME	Simple but Extensive Structured Availability Modeling Environment
SN	Siemens Norm
SPN	stochastic Petri net
SS7	signalling system 7
ST-node	standard IFD node
T	Petri net transition
TMR	triple modular redundancy
U	(steady-state) unavailability
$U(t)$	transient unavailability
UML	unified modeling language
WD-node	watchdog node (in an IFD)
$\vee$	Boolean OR operator
$\wedge$	Boolean AND operator
$\bar{}$	Boolean NOT operator