

The Java Simulation Handbook – Simulating Discrete Event Systems with UML and Java

Bernd Page, Wolfgang Kreutzer
(main authors)

Coauthored by Björn Gehlsen, Johannes Göbel,
Gunnar Kiesel, Nicolas Knaak, Julia Kuck, Tim Lechler
Ruth Meyer, Gaby Neumann, Volker Wohlgemuth

Berichte aus der Informatik

**Bernd Page,
Wolfgang Kreutzer**

The Java Simulation Handbook

Simulating Discrete Event Systems
with UML and Java

Shaker Verlag
Aachen 2005

Bibliographic information published by Die Deutsche Bibliothek

Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data is available in the internet at <http://dnb.ddb.de>.

Copyright Shaker Verlag 2005

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior permission of the publishers.

Printed in Germany.

ISBN 3-8322-3771-2

ISSN 0945-0807

Shaker Verlag GmbH • P.O. BOX 101818 • D-52018 Aachen

Phone: 0049/2407/9596-0 • Telefax: 0049/2407/9596-9

Internet: www.shaker.de • eMail: info@shaker.de

Introduction

Computer simulation is an important tool for modelling and analysing a complex system. Its applications range widely; from the natural to the engineering sciences, from the social sciences to economics, from medicine to studying environmental concerns. Within computer science, simulation has been used in such diverse areas as chip design, system and communication network performance analysis, database design, and for the study of operating systems and other software. When applied to commercial problems, computer simulation offers a flexible and highly successful method to explore and optimize information and material flows in an enterprise.

Designing and implementing suitable algorithms and programs for such a wide range of applications, however, remains a challenging task; particularly since it seems highly desirable to base models on unified and transferable architectures, and since usable and powerful modelling software can be a key ingredient to the success of a simulation study.

Simulation is one of the earliest applications of computing technology. In addition to mastery of the respective domains (e.g. production, computer, telecommunication, or logistics systems), its application demands competence in a large number of areas, such as system analysis, system design, statistical methods, experimental design, software design and programming. Because of its wide ranging applications, its high degree of practical relevance, and its demand for a unique blend of skills in both system design and implementation, simulation plays also an important and easily motivated part in teaching academic computer science programmes. Good software is needed to support exploration of the whole spectrum of simulation development, from system analysis to model implementation and experimentation. Programming a simulation model, in particular, places high cognitive demands on the model designer. By offering prepackaged functionality to support specific modelling contexts, object-oriented frameworks help to master this complexity. Good modelling frameworks ensure that many *programming aspects* of a simulation study are already taken care of, so that a model designer's mind becomes free to concentrate on *problem-specific aspects* of a system under investigation.

For many years the Faculty of Informatics at the University of Hamburg has been involved in developing software for discrete event simulation frameworks, hosted in different programming languages. These frameworks package core functionality for implementing discrete event simulations and therefore leave students more time to analyse, design, experiment with, and evaluate relevant system models. Studying their architecture also aids understanding of important structural aspects and control patterns in simulation software.

Prompted by Java's increasing popularity as a programming tool for both teaching at universities and use in industrial practice, we have, for a number of years, designed and

implemented a Java-based discrete event simulation framework called DESMO-J (Discrete Event Simulation and MOdelling in Java). We plan to continue maintaining and improving DESMO-J as public domain software under the GNU Lesser General Public License. From 2000 to 2004 a close cooperation with the University of Canterbury has been supported by the Federal Ministry of Education and Research (BMBF)¹.

In drawing on the results of this cooperation, the Java Simulation Handbook presents a broad palette of important and timely topics within the context of discrete event simulation. All contributions were written by competent specialists in the respective areas from different universities, coordinated by the two main authors, who have provided the content framework for the book. The book's 16 chapters build on each other and present information in a common style. Although chapters often refer to each other's examples and discussions, each still remains reasonably self-contained. The fact that each chapter can stand on its own means that they may be read both sequentially, in the specified order, or more selectively, whenever particular information is needed. This makes the Java Simulation Handbook suitable to serve as either a textbook (e.g. for a university course) or as a reference.

The book presents its contents in four parts. The first, *Foundations*, discusses fundamental concepts of discrete event simulation. The next part, *Software*, concerns the implementation of discrete event simulation in general as well as in DESMO-J, while *Advanced Methodology* addresses current research, such as agent-based and distributed simulation. The *Applications* part then summarizes simulation's contributions to e-learning, logistics, and industrial practice. For the practitioner, it finally offers some guidelines for the successful completion of real-world simulation projects.

Systematic design and development of discrete event models is the dominant focus of the book. To support our presentation, we use the well established UML 2 (Unified Modelling Language, Version 2.0) notation for model design and the above mentioned DESMO-J Java-based simulation framework for model implementation. To give better support to the simulation domain, we also enrich UML 2 with some simulation-specific extensions.

Although this focus on model design and implementation distinguishes this handbook from many others, which favour a statistical perspective on simulation technology, the handbook also includes chapters dealing with relevant statistical aspects and other foundational issues.

It is one of this book's aspirations to teach by example. To further this purpose, it contains many examples of models, programs, and exercises, the completion of which will hopefully inspire the reader to explore further. This aspect and its associated web-based resources make the book particularly well-suited for teaching a course which emphasises the programming aspects of discrete event simulation.

¹BMBF Programme for Scientific-Technological Collaboration with New Zealand, project number NZL 00/02

The Java Simulation Handbook is supported by a number of web-based resources. Readers can access a web server at <http://www.desmoj.de> with the DESMO-J software, a web-based DESMO-J tutorial, and a link to the cooperation platform CommSy (with a CommSy project room *JavaSimulationHandbook*). It contains a simulation laboratory with many example programs and some applet-based animations. These materials were developed as part of an e-learning project, which was financed by the E-Learning Consortium Hamburg from 2003 until 2005. The handbook itself is one of the results of this project and would not exist without this support. The web-based resources are a useful add-on to the book. In addition to its role as a repository for teaching materials, the project's online cooperation platform also offers a means for discussing the book's contents. User access to the CommSy-platform is open to readers of this book using the guest account *JavaBeans* and the password *snoopy*. The same account and password allow for access to the DESMO-J laboratory on the Internet.

The intended readership of this handbook and the DESMO-J framework are students and teachers of computer science, at both academic and other, possibly more applied, tertiary educational institutions. Readers should feel confident in using object-oriented programming styles and Java. Moreover, they should be interested in acquiring core competence in system analysis, model design, and the implementation of discrete event simulation programs. Some elementary knowledge of probability theory and statistics is assumed. Since the book only uses a small range of techniques, this can be obtained rather quickly. Suitable recommendations for further reading are given under *Further Reading* at the end of those chapters where additional background information seemed to be helpful for certain readers. Some knowledge of UML may prove helpful. To provide some background, the handbook presents those parts of UML that are particularly relevant to simulation.

The authors would like to thank all contributors, without whose participation a handbook of this depth and size would not have been possible. The DESMO-J software has been developed and extended by many colleagues and students, among who the main developers of the DESMO-J core Java foundation, Tim Lechler and Sönke Claasen, deserve particular thanks. Editorial work on the manuscript and its markup in \LaTeX was done by Johannes Göbel, with assistance by Ruth Meyer. Ruth Meyer also wrote the current version of the DESMO-J tutorial. Other e-learning materials for this book were created by Alexander Bentz and Gunnar Kiesel. At the end we must thank Kirsten-Verena Bull, Frank Heitmann, and Alexander Bentz for their final corrections, which have prevented many errors and inconsistencies to slip through.

We are very grateful to the BMBF and its New Zealand counterpart, for their financial support over many years. Among other things this has enabled us to travel in order to cooperate and communicate effectively over the large physical distance between Germany and New Zealand. Ultimately, this has provided a sound scientific basis on which the Java Simulation Handbook could be built. Further thanks go to the E-Learning Consortium Hamburg, for its financial support in writing the handbook and, in particular, for assistance in developing the accompanying electronic materials.

Finally, we thank the Shaker Verlag for the professional and timely publication of this handbook in its Informatik series. We are particularly pleased with the book's electronic version, which reflects our vision of a *blended* learning environment for the teaching of simulation rather well.

Bernd Page,
University of Hamburg

Wolfgang Kreutzer,
University of Canterbury

Hamburg and Christchurch, September 2005

Contents

I	Foundations	1
1	Introduction and Basic Terms	3
1.1	Motivation and Overview	3
1.2	Systems	4
1.3	Models	5
1.4	Simulation	9
1.5	The Modelling Cycle	12
1.6	Application	17
1.7	Potential and Limitations of Simulation	18
1.7.1	Advantages of Modelling and Simulation	18
1.7.2	Limitations of Modelling and Simulation	20
	Bibliography	21
2	Basic Concepts in Discrete Event Simulation	23
2.1	Motivation and Overview	23
2.2	Discrete Event Simulation Model Components	24
2.3	Relations between Model State and Time Advance	24
2.4	Discrete Event Model Components	28
2.5	Executing a Discrete Event Simulation	29
2.6	Simulating Random Events	31
2.7	From Event to Event: A Paper and Pencil Exercise	32
	Bibliography	35
3	Object-Oriented System Development and Simulation	39
3.1	Motivation and Overview	39
3.2	History & Core Concepts of Object-Orientation	42
3.2.1	Some History	42
3.2.2	Core Concepts	43
3.3	Object-Oriented Analysis and Design	49
3.4	Object-Oriented Programming Tools	53
3.5	Summary and Perspectives	54
	Further Reading	56
	Bibliography	57

Contents

4	Simulation Model Descriptions with UML 2	59
4.1	Motivation and Overview	59
4.2	Introduction to the Unified Modelling Language	60
4.2.1	The Road to UML 2	61
4.2.2	Diagram Types and Design Principles of the UML	62
4.3	Modelling Static System Structures	65
4.4	Modelling Dynamic Behaviour	69
4.4.1	Statecharts	70
4.4.2	Activity Diagrams	77
4.5	Modelling Interactions	87
4.5.1	Sequence Diagrams of Interaction Scenarios	88
4.5.2	High Level Sequence Diagrams	90
4.5.3	Timing Diagrams	92
	Further Reading	93
	Bibliography	93
5	Discrete Event Model Design	97
5.1	Motivation and Overview	97
5.2	Dominant Discrete Event Modelling Styles	98
5.2.1	Process-Oriented Simulation Modelling	98
5.2.2	Event-Oriented Simulation Modelling	108
5.2.3	Comparison and Evaluation	117
5.3	Other Modelling Styles	129
5.3.1	Transaction-Oriented Modelling	129
5.3.2	Activity-Oriented Modelling	131
5.4	Object-Oriented Model Construction	133
5.5	Combined Modelling Styles	134
5.5.1	Embedding Events in a Process-Oriented Model	134
5.5.2	Example: A Combined Process/Event Model	135
	Bibliography	140
6	First Steps in Simulation Programming	143
6.1	Motivation and Overview	143
6.2	Java as a “Simulation Language”	144
6.2.1	General Requirements	144
6.2.2	Simulation-Specific Requirements	146
6.2.3	Advantages of Java as a Simulation Language	149
6.3	A Simple Java Class Library for Event-Oriented Simulation	149
6.3.1	Event List	150
6.3.2	Entities, Events, and Event Scheduling	152

6.3.3	Distribution Sampling, Data Collection, and Queuing	153
6.3.4	Example	156
	Further Reading	157
	Bibliography	157
7	Simulation Statistics	159
7.1	Motivation and Overview	159
7.2	Creating Random Numbers	161
7.2.1	General Approach	161
7.2.2	The Java Random Number Generator	168
7.3	Estimating an Input Distribution	170
7.4	Analysing a Simulation Experiment	173
7.4.1	Repeating a Simulation Run	173
7.4.2	Stationary Model States	174
7.4.3	Warm-up Phases and Steady States	174
7.4.4	Analysing Simulation Results	179
7.5	Observing a Simulation Experiment	185
7.5.1	Independent Replications	185
7.5.2	Batch Means	186
7.5.3	Terminating and Non-Stationary Systems	188
7.6	Sample Size and Simulation Experiments	188
7.7	Choosing Good Model Parameters	189
7.8	Conclusion	191
	Further Reading	192
	Bibliography	192
8	Validation, Verification, and Testing of Simulation Models	195
8.1	Motivation and Overview	195
8.2	Foundations of Model Validation	197
8.2.1	Basic Terms	197
8.2.2	The Validation Process	199
8.2.3	A Philosophical View of Model Validation	203
8.2.4	Principles and Guidelines for Model Validation	205
8.2.5	Classification of Validation Techniques	210
8.3	Selected Validation Techniques	211
8.3.1	Conceptual Model Validation	212
8.3.2	Model Verification and Testing	215
8.3.3	Operational Validation of Model Behaviour	221
8.4	Summary	230
	Further Reading	231
	Bibliography	231

II	Software	237
9	Simulation Software	239
9.1	Motivation and Overview	239
9.2	Requirements	240
9.2.1	General Requirements	240
9.2.2	Simulation-Specific Requirements	241
9.3	Some History	243
9.4	Classification	247
9.5	Examples	251
9.5.1	Extend	252
9.5.2	eM-Plant	254
9.6	The Role of Animation	257
9.7	Criteria for Choosing Simulation Software in Practice	259
9.8	Commercial Discrete Event Modelling Tools	260
	Bibliography	261
10	DESMO-J – A Framework for Discrete Event Modelling & Simulation	263
10.1	Motivation and Overview	263
10.2	Simulation with DESMO-J	266
10.2.1	The Event-Oriented World View	268
10.2.2	The Process-Oriented World View	273
10.2.3	Combining Events and Processes in a Single Model	274
10.2.4	Some Core Model Components	276
10.3	Experimentation with DESMO-J	280
10.4	Advanced Concepts	284
10.4.1	Higher-Level Modelling Constructs	284
10.4.2	Hierarchical Modelling Constructs	291
10.4.3	Graphical Interfaces	293
10.5	Example: Modelling Container Traffic in the Baltic Sea	293
10.5.1	Model Description	293
10.5.2	An Event-Oriented Implementation	294
10.5.3	A Process-Oriented Implementation	307
10.5.4	Using Higher-Level Modelling Constructs	314
10.6	Development and Evaluation	334
	Bibliography	335

III Advanced Methodology	337
11 Multi-Agent-Based Simulation (MABS)	339
11.1 Motivation and Overview	339
11.2 Multi-Agent Systems	340
11.2.1 The Agent Metaphor	340
11.2.2 Characteristics of MAS	341
11.2.3 Architectures for Agent Design	342
11.3 Different Views of Agent-Oriented Simulation	347
11.4 Foundations of Multi-Agent-Based Simulation	348
11.4.1 Applications of MABS	349
11.4.2 Comparison of Agent-Based and Classical World Views	351
11.4.3 Components of Multi-Agent-Based Simulation Models	353
11.4.4 Conceptual Modelling Methods	357
11.4.5 Tools for Agent-Based Simulation	360
11.5 Conclusion: Potentials, Limitations, and Prospects for MABS	365
Further Reading	367
Bibliography	368
12 Parallel and Distributed Simulation	373
12.1 Motivation and Overview	373
12.2 Synchronization	374
12.3 Modes of Distribution	376
12.3.1 Parallel Simulation	377
12.3.2 Component-Based Simulation	378
12.3.3 Web-Based Simulation	380
Further Reading	382
Bibliography	383
13 Simulation-Based Optimization	387
13.1 Motivation and Overview	387
13.2 Integration of Simulation and Optimization	388
13.3 A Formal Model for Simulation-Based Optimization Problems	391
13.4 Characteristics of Simulation-Based Objective Functions	392
13.5 Genetic Algorithms	394
13.5.1 Terminology	395
13.5.2 General Strategy	395
13.5.3 Parallelization	397
Further Reading	397
Bibliography	398

IV Applications	399
14 Simulation and E-Learning	401
14.1 Motivation and Overview	401
14.2 E-Learning Foundations	403
14.2.1 Definition	403
14.2.2 Technological Development	404
14.2.3 Three Theories of Learning	407
14.2.4 Requirements for the Design and Implementation of E-Learning	410
14.3 Computer Simulation to Improve E-Learning	412
14.3.1 Simulation and Learning	412
14.3.2 Simulation and E-Learning	413
14.3.3 Simulation-Based Learning in LogEduGate	415
14.4 E-Learning to Support Simulation Learning	419
14.4.1 Using DESMO-J to Teach Simulation Courses	420
14.4.2 The DESMO-J Web Tutorial	421
14.4.3 The DESMO-J Internet Laboratory	422
14.4.4 Using Java Applets to Demonstrate Key Simulation Concepts	424
14.5 A Web Platform for Cooperative Teaching and Learning	427
14.6 Conclusions	429
Further Reading	430
Bibliography	431
15 Simulation and Logistics	435
15.1 Motivation and Overview	436
15.2 Introduction to Logistics	436
15.3 Modelling and Simulation in Logistics	440
15.3.1 Application of Simulation in Logistics	440
15.3.2 Logistics Simulation Tools	442
15.3.3 Simulation Experiments in Logistics Problem Solving	443
15.4 Knowledge Acquisition and Knowledge Sharing in Logistics Simulation	446
15.4.1 Logistics Simulation Knowledge	446
15.4.2 Cooperative Knowledge Sharing in Logistics Simulation Projects	448
15.4.3 Simulation Model as Knowledge Repository	450
15.4.4 Experiment-Based Knowledge Creation	451
15.4.5 Knowledge Management and Logistics Simulation	452
15.5 Cases in Logistics Simulation	453
15.5.1 Simulation to Support Logistics Planning: The Case of a Paper Store	453
15.5.2 Simulation to Support Modification: The Case of a Pallet Flow System	458
15.5.3 Simulation to Support Logistics Operation: The Case of an Order Picking System	461

15.6 Conclusions	465
Further Reading	467
Bibliography	468
16 Simulation in Practice	469
16.1 Motivation and Overview	469
16.2 Introduction	470
16.3 General Requirements for a Successful Simulation Study	472
16.4 Simulation Project Organization	477
16.4.1 Roles and Responsibilities in a Simulation Project Team	477
16.4.2 Simulation as a Consulting Service	478
16.4.3 Costs and Time Requirements for a Simulation Study	480
16.5 Methods of Data Collection	481
16.6 Typical Errors and Pitfalls in a Simulation Study	483
16.7 Conclusion	484
Further Reading	485
Bibliography	486
Authors	487
Index	493