

A Practical Guide to SysML

The Systems Modeling Language



Morgan Kaufmann OMG Press

Morgan Kaufmann Publishers and the Object Management Group™ (OMG) have joined forces to publish a line of books addressing business and technical topics related to OMG's large suite of software standards.

OMG is an international, open membership, not-for-profit computer industry consortium that was founded in 1989. The OMG creates standards for software used in government and corporate environments to enable interoperability and to forge common development environments that encourage the adoption and evolution of new technology. OMG members and its board of directors consist of representatives from a majority of the organizations that shape enterprise and Internet computing today.

OMG's modeling standards, including the Unified Modeling Language™ (UML®) and Model Driven Architecture® (MDA), enable powerful visual design, execution and maintenance of software, and other processes—for example, IT Systems Modeling and Business Process Management. The middleware standards and profiles of the Object Management Group are based on the Common Object Request Broker Architecture® (CORBA) and support a wide variety of industries.

More information about OMG can be found at <http://www.omg.org/>.

Related Morgan Kaufmann OMG Press Titles

UML 2 Certification Guide: Fundamental and Intermediate Exams

Tim Weilkiens and Bernd Oestereich

Real-Life MDA: Solving Business Problems with Model Driven Architecture

Michael Guttman and John Parodi

Systems Engineering with SysML/UML: Modeling, Analysis, Design

Tim Weilkiens

A Practical Guide to SysML: The Systems Modeling Language

Sanford Friedenthal, Alan Moore, and Rick Steiner

Building the Agile Enterprise: With SOA, BPM and MBM

Fred Cummins

Business Modeling: A Practical Guide to Realizing Business Value

Dave Bridgeland and Ron Zahavi

Architecture Driven Modernization: A Series of Industry Case Studies

Bill Ulrich

A Practical Guide to SysML

The Systems Modeling Language

Sanford Friedenthal

Alan Moore

Rick Steiner



AMSTERDAM • BOSTON • HEIDELBERG • LONDON
NEW YORK • OXFORD • PARIS • SAN DIEGO
SAN FRANCISCO • SINGAPORE • SYDNEY • TOKYO

Morgan Kaufmann Publishers is an imprint of Elsevier



Acquiring Editor: Rachel Roumeliotis
Development Editor: Robyn Day
Project Manager: A. B. McGee
Designer: Kristen Davis

Morgan Kaufmann is an imprint of Elsevier
225 Wyman Street, Waltham, MA 02451, USA

© 2012 Elsevier Inc. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or any information storage and retrieval system, without permission in writing from the publisher. Details on how to seek permission, further information about the Publisher's permissions policies and our arrangements with organizations such as the Copyright Clearance Center and the Copyright Licensing Agency, can be found at our website: www.elsevier.com/permissions.

This book and the individual contributions contained in it are protected under copyright by the Publisher (other than as may be noted herein).

Notices

Knowledge and best practice in this field are constantly changing. As new research and experience broaden our understanding, changes in research methods or professional practices, may become necessary. Practitioners and researchers must always rely on their own experience and knowledge in evaluating and using any information or methods described herein. In using such information or methods they should be mindful of their own safety and the safety of others, including parties for whom they have a professional responsibility.

To the fullest extent of the law, neither the Publisher nor the authors, contributors, or editors, assume any liability for any injury and/or damage to persons or property as a matter of products liability, negligence or otherwise, or from any use or operation of any methods, products, instructions, or ideas contained in the material herein.

Library of Congress Cataloging-in-Publication Data

Application submitted

British Library Cataloguing-in-Publication Data

A catalogue record for this book is available from the British Library.

ISBN: 978-0-12-385206-9

Printed in the United States of America

11 12 13 14 15 10 9 8 7 6 5 4 3 2 1

Working together to grow
libraries in developing countries

www.elsevier.com | www.bookaid.org | www.sabre.org

ELSEVIER

BOOK AID
International

Sabre Foundation

For information on all MK publications visit our website at www.mkp.com

Contents

Preface	xvii
Acknowledgments.....	xxi
About the Authors	xxiii

PART I INTRODUCTION

CHAPTER 1 Systems Engineering Overview.....	3
1.1 Motivation for Systems Engineering.....	3
1.2 The Systems Engineering Process	4
1.3 Typical Application of the Systems Engineering Process	5
1.4 Multidisciplinary Systems Engineering Team	9
1.5 Codifying Systems Engineering Practice through Standards	10
1.6 Summary	13
1.7 Questions	14
 CHAPTER 2 Model-Based Systems Engineering.....	 15
2.1 Contrasting the Document-Based and Model-Based Approach.....	15
2.1.1 Document-Based Systems Engineering Approach.....	15
2.1.2 Model-Based Systems Engineering Approach.....	16
2.2 Modeling Principles.....	21
2.2.1 Model and MBSE Method Definition	21
2.2.2 The Purpose for Modeling a System.....	21
2.2.3 Establishing Criteria to Meet the Model Purpose.....	22
2.2.4 Model-Based Metrics.....	25
2.2.5 Other Model-Based Metrics	26
2.3 Summary	27
2.4 Questions	27
 CHAPTER 3 Getting Started with SysML.....	 29
3.1 SysML Purpose and Key Features	29
3.2 SysML Diagram Overview.....	29
3.3 Introducing SysML-Lite	31
3.3.1 SysML-Lite Diagrams and Language Features.....	31
3.3.2 SysML-Lite Air Compressor Example.....	34
3.3.3 SysML Modeling Tool Tips	38
3.4 A Simplified MBSE Method	44
3.5 The Learning Curve for SysML and MBSE.....	47
3.6 Summary	48
3.7 Questions	48

CHAPTER 4	An Automobile Example Using the SysML Basic Feature Set.....	51
4.1	SysML Basic Feature Set	51
4.2	Automobile Example Overview	51
4.2.1	Problem Summary	52
4.3	Automobile Model.....	52
4.3.1	Package Diagram for Organizing the Model	53
4.3.2	Capturing the <i>Automobile Specification</i> in a Requirement Diagram	55
4.3.3	Defining the <i>Vehicle</i> and Its External Environment Using a Block Definition Diagram	57
4.3.4	Use Case Diagram for <i>Operate Vehicle</i>	58
4.3.5	Representing <i>Drive Vehicle</i> Behavior with a Sequence Diagram	60
4.3.6	Referenced Sequence Diagram to <i>Turn On Vehicle</i>	60
4.3.7	<i>Control Power</i> Activity Diagram.....	62
4.3.8	State Machine Diagram for <i>Drive Vehicle States</i>	64
4.3.9	<i>Vehicle Context</i> Using an Internal Block Diagram	64
4.3.10	<i>Vehicle Hierarchy</i> Represented on a Block Definition Diagram	67
4.3.11	Activity Diagram for <i>Provide Power</i>	69
4.3.12	Internal Block Diagram for the <i>Power Subsystem</i>	69
4.3.13	Defining the Equations to Analyze Vehicle Performance	73
4.3.14	Analyzing Vehicle Acceleration Using the Parametric Diagram	75
4.3.15	Analysis Results from Analyzing Vehicle Acceleration.....	75
4.3.16	Defining the <i>Vehicle Controller</i> Actions to Optimize Engine Performance	77
4.3.17	Specifying the <i>Vehicle</i> and Its Components.....	78
4.3.18	Requirements Traceability	79
4.3.19	View and Viewpoint.....	81
4.4	Model Interchange	82
4.5	Summary	82
4.6	Questions	83

PART II LANGUAGE DESCRIPTION

CHAPTER 5	SysML Language Architecture	87
5.1	The OMG SysML Language Specification.....	87
5.2	The Architecture of the SysML Language	88
5.2.1	The General-Purpose Systems Modeling Domain	89
5.2.2	The Modeling Language (or Metamodel)	90
5.2.3	The System Model (or User Model)	91
5.2.4	Model Interchange	92
5.3	SysML Diagrams	93
5.3.1	Diagram Frames.....	94
5.3.2	Diagram Header	95

5.3.3	Diagram Description.....	96
5.3.4	Diagram Content.....	96
5.3.5	Additional Notations.....	99
5.4	The Surveillance System Case Study	100
5.4.1	Case Study Overview.....	100
5.4.2	Modeling Conventions.....	100
5.5	Organization of Part II.....	101
5.5.1	OCSMP Certification Coverage and SysML 1.3	101
5.6	Questions	102
CHAPTER 6	Organizing the Model with Packages.....	103
6.1	Overview	103
6.2	The Package Diagram	104
6.3	Defining Packages Using a Package Diagram	104
6.4	Organizing a Package Hierarchy.....	106
6.5	Showing Packageable Elements on a Package Diagram	107
6.6	Packages as Namespaces	109
6.7	Importing Model Elements into Packages	109
6.8	Showing Dependencies between Packageable Elements.....	112
6.9	Specifying Views and Viewpoints.....	114
6.10	Summary	115
6.11	Questions.....	116
CHAPTER 7	Modeling Structure with Blocks	119
7.1	Overview	119
7.1.1	Block Definition Diagram	120
7.1.2	Internal Block Diagram	121
7.2	Modeling Blocks on a Block Definition Diagram.....	121
7.3	Modeling the Structure and Characteristics of Blocks Using Properties.....	123
7.3.1	Modeling Block Composition Hierarchies Using Part Properties.....	123
7.3.2	Modeling Relationships between Blocks Using Reference Properties.....	130
7.3.3	Using Associations to Type Connectors between Parts.....	132
7.3.4	Modeling Quantifiable Characteristics of Blocks Using Value Properties	137
7.4	Modeling Flows	142
7.4.1	Modeling Items That Flow	143
7.4.2	Flow Properties	143
7.4.3	Modeling Flows between Parts on an Internal Block Diagram.....	144
7.5	Modeling Block Behavior	147
7.5.1	Modeling the Main Behavior of a Block	148
7.5.2	Specifying the Behavioral Features of Blocks.....	148
7.5.3	Modeling Block-Defined Methods	150
7.5.4	Routing Requests Across Connectors	151

7.6	Modeling Interfaces Using Ports.....	152
7.6.1	Full Ports.....	153
7.6.2	Proxy Ports.....	154
7.6.3	Connecting Ports.....	157
7.6.4	Modeling Flows between Ports	165
7.6.5	Using Interfaces with Ports	165
7.7	Modeling Classification Hierarchies Using Generalization.....	167
7.7.1	Classification and the Structural Features of a Block.....	169
7.7.2	Classification and Behavioral Features	170
7.7.3	Modeling Overlapping Classifications Using Generalization Sets.....	171
7.7.4	Modeling Variants Using Classification	172
7.7.5	Using Property-Specific Types to Model Context-Specific Block Characteristics.....	173
7.7.6	Modeling Block Configurations as Specialized Blocks.....	173
7.8	Modeling Block Configurations Using Instances	176
7.9	Deprecated Features	178
7.9.1	Flow Ports	179
7.10	Summary	180
7.11	Questions.....	182
CHAPTER 8	Modeling Constraints with Parametrics.....	185
8.1	Overview	185
8.1.1	Defining Constraints Using the Block Definition Diagram.....	185
8.1.2	The Parametric Diagram.....	186
8.2	Using Constraint Expressions to Represent System Constraints.....	187
8.3	Encapsulating Constraints in Constraint Blocks to Enable Reuse	188
8.3.1	Additional Parameter Characteristics	188
8.4	Using Composition to Build Complex Constraint Blocks.....	190
8.5	Using a Parametric Diagram to Bind Parameters of Constraint Blocks	191
8.6	Constraining Value Properties of a Block.....	193
8.7	Capturing Values in Block Configurations	195
8.8	Constraining Time-Dependent Properties to Facilitate Time-Based Analysis.....	195
8.9	Using Constraint Blocks to Constrain Item Flows	197
8.10	Describing an Analysis Context.....	198
8.11	Modeling Evaluation of Alternatives and Trade Studies.....	200
8.12	Summary	202
8.13	Questions.....	203
CHAPTER 9	Modeling Flow-Based Behavior with Activities	205
9.1	Overview	205
9.2	The Activity Diagram.....	206
9.3	Actions—The Foundation of Activities	208
9.4	The Basics of Modeling Activities.....	209

9.4.1	Specifying Input and Output Parameters for an Activity	209
9.4.2	Composing Activities Using Call Behavior Actions	211
9.5	Using Object Flows to Describe the Flow of Items between Actions	213
9.5.1	Routing Object Flows	213
9.5.2	Routing Object Flows from Parameter Sets	216
9.5.3	Buffers and Data Stores	219
9.6	Using Control Flows to Specify the Order of Action Execution	220
9.6.1	Depicting Control Logic with Control Nodes	220
9.6.2	Using Control Operators to Enable and Disable Actions	222
9.7	Handling Signals and Other Events	224
9.8	Structuring Activities	225
9.8.1	Interruptible Regions	225
9.8.2	Using Structured Activity Nodes	226
9.9	Advanced Flow Modeling	228
9.9.1	Modeling Flow Rates	228
9.9.2	Modeling Flow Order	229
9.9.3	Modeling Probabilistic Flow	230
9.10	Modeling Constraints on Activity Execution	231
9.10.1	Modeling Pre- and Post-conditions and Input and Output States	231
9.10.2	Adding Timing Constraints to Actions	233
9.11	Relating Activities to Blocks and Other Behaviors	234
9.11.1	Linking Behavior to Structure Using Partitions	234
9.11.2	Specifying an Activity in a Block Context	236
9.11.3	Relationship between Activities and Other Behaviors	239
9.12	Modeling Activity Hierarchies Using Block Definition Diagrams	240
9.12.1	Modeling Activity Invocation Using Composite Associations	240
9.12.2	Modeling Parameter and Other Object Nodes Using Associations	240
9.12.3	Adding Parametric Constraints to Activities	242
9.13	Enhanced Functional Flow Block Diagram	243
9.14	Executing Activities	243
9.14.1	The Foundational UML Subset (fUML)	244
9.14.2	The Action Language for Foundational UML (Alf)	245
9.14.3	Primitive Actions	246
9.14.4	Executing Continuous Activities	247
9.15	Summary	248
9.16	Questions	249
CHAPTER 10	Modeling Message-Based Behavior with Interactions	251
10.1	Overview	251
10.2	The Sequence Diagram	252
10.3	The Context for Interactions	252
10.4	Using Lifelines to Represent Participants in an Interaction	254
10.4.1	Occurrence Specifications	255

10.5	Exchanging Messages between Lifelines	256
10.5.1	Synchronous and Asynchronous Messages	256
10.5.2	Lost and Found Messages	258
10.5.3	Weak Sequencing	259
10.5.4	Executions	259
10.5.5	Lifeline Creation and Destruction	261
10.6	Representing Time on a Sequence Diagram	261
10.7	Describing Complex Scenarios Using Combined Fragments	264
10.7.1	Basic Interaction Operators	265
10.7.2	Additional Interaction Operators	266
10.7.3	State Invariants	268
10.8	Using Interaction References to Structure Complex Interactions	270
10.9	Decomposing Lifelines to Represent Internal Behavior	270
10.10	Summary	273
10.11	Questions	274
CHAPTER 11	Modeling Event-Based Behavior with State Machines	277
11.1	Overview	277
11.2	State Machine Diagram	278
11.3	Specifying States in a State Machine	278
11.3.1	Region	278
11.3.2	State	280
11.4	Transitioning between States	281
11.4.1	Transition Fundamentals	281
11.4.2	Routing Transitions Using Pseudostates	284
11.4.3	Showing Transitions Graphically	287
11.5	State Machines and Operation Calls	287
11.6	State Hierarchies	288
11.6.1	Composite State with a Single Region	289
11.6.2	Composite State with Multiple (Orthogonal) Regions	290
11.6.3	Transition Firing Order in Nested State Hierarchies	292
11.6.4	Using the History Pseudostate to Return to a Previously Interrupted State	293
11.6.5	Reusing State Machines	295
11.7	Contrasting Discrete and Continuous States	297
11.8	Summary	299
11.9	Questions	300
CHAPTER 12	Modeling Functionality with Use Cases	303
12.1	Overview	303
12.2	Use Case Diagram	303
12.3	Using Actors to Represent the Users of a System	304
12.3.1	Further Descriptions of Actors	305

12.4	Using Use Cases to Describe System Functionality	305
12.4.1	Use Case Relationships	307
12.4.2	Use Case Descriptions.....	309
12.5	Elaborating Use Cases with Behaviors.....	310
12.5.1	Context Diagrams.....	310
12.5.2	Sequence Diagrams	310
12.5.3	Activity Diagrams	311
12.5.4	State Machine Diagrams	313
12.6	Summary	314
12.7	Questions	315

CHAPTER 13 Modeling Text-Based Requirements and Their Relationship to Design 317

13.1	Overview	317
13.2	Requirement Diagram	318
13.3	Representing a Text Requirement in the Model	320
13.4	Types of Requirements Relationships	322
13.5	Representing Cross-Cutting Relationships in SysML Diagrams	322
13.5.1	Depicting Requirements Relationships Directly	323
13.5.2	Depicting Requirements Relationships Using Compartment Notation	324
13.5.3	Depicting Requirements Relationships Using Callout Notation.....	324
13.6	Depicting Rationale for Requirements Relationships	325
13.7	Depicting Requirements and Their Relationships in Tables.....	326
13.7.1	Depicting Requirement Relationships in Tables	326
13.7.2	Depicting Requirement Relationships as Matrices.....	327
13.8	Modeling Requirement Hierarchies in Packages	328
13.9	Modeling a Requirements Containment Hierarchy.....	328
13.9.1	The Browser View of a Containment Hierarchy	329
13.10	Modeling Requirement Derivation	329
13.11	Asserting That a Requirement is Satisfied	331
13.12	Verifying That a Requirement is Satisfied.....	332
13.13	Reducing Requirements Ambiguity Using the Refine Relationship.....	335
13.14	Using the General-Purpose Trace Relationship.....	338
13.15	Reusing Requirements with the Copy Relationship.....	338
13.16	Summary	339
13.17	Questions	340

CHAPTER 14 Modeling Cross-Cutting Relationships with Allocations..... 343

14.1	Overview	343
14.2	Allocation Relationship	343
14.3	Allocation Notation.....	345
14.4	Types of Allocation.....	347

14.4.1	Allocation of Requirements	347
14.4.2	Allocation of Behavior or Function	347
14.4.3	Allocation of Flow	348
14.4.4	Allocation of Structure	348
14.4.5	Allocation of Properties	348
14.4.6	Summary of Relationships Associated with the Term “Allocation”	349
14.5	Planning for Reuse: Specifying Definition and Usage in Allocation	349
14.5.1	Allocating Usage	350
14.5.2	Allocating Definition	351
14.5.3	Allocating Asymmetrically	351
14.5.4	Guidelines for Allocating Definition and Usage	351
14.6	Allocating Behavior to Structure Using Functional Allocation	352
14.6.1	Modeling Functional Allocation of Usage	354
14.6.2	Modeling Functional Allocation of Definition	354
14.6.3	Modeling Functional Allocation Using Allocate Activity Partitions (Allocate Swimlanes)	357
14.7	Connecting Functional Flow with Structural Flow Using Functional Flow Allocation	358
14.7.1	Options for Functionally Allocating Flow	358
14.7.2	Allocating an Object Flow to a Connector	358
14.7.3	Allocating Object Flow to Item Flow	359
14.8	Modeling Allocation between Independent Structural Hierarchies	361
14.8.1	Modeling Structural Allocation of Usage	362
14.8.2	Allocating a Logical Connector to a Physical Structure	362
14.8.3	Modeling Structural Allocation of Definition	363
14.9	Modeling Structural Flow Allocation	364
14.10	Evaluating Allocation across a User Model	366
14.10.1	Establishing Balance and Consistency	366
14.11	Taking Allocation to the Next Step	366
14.12	Summary	367
14.13	Questions	367

CHAPTER 15 Customizing SysML for Specific Domains **369**

15.1	Overview	369
15.1.1	A Brief Review of Metamodeling Concepts	370
15.2	Defining Model Libraries to Provide Reusable Constructs	373
15.3	Defining Stereotypes to Extend Existing SysML Concepts	374
15.3.1	Adding Properties and Constraints to Stereotypes	376
15.4	Extending the SysML Language Using Profiles	379
15.4.1	Referencing a Metamodel or Metaclass from a Profile	380
15.5	Applying Profiles to User Models in Order to Use Stereotypes	381
15.6	Applying Stereotypes when Building a Model	382
15.6.1	Specializing Model Elements with Applied Stereotypes	384

15.7	Summary	388
15.8	Questions	389

PART III MODELING EXAMPLES

CHAPTER 16	Water Distiller Example Using Functional Analysis	393
16.1	Stating the Problem – The Need for Clean Drinking Water	393
16.2	Defining the Model-Based Systems Engineering Approach	394
16.3	Organizing the Model	394
16.4	Establishing Requirements	396
16.4.1	Characterizing Stakeholder Needs	396
16.4.2	Characterizing System Requirements	399
16.4.3	Characterizing Required Behaviors	400
16.4.4	Refining Behavior	406
16.5	Modeling Structure	409
16.5.1	Defining Distiller’s Blocks in the Block Definition Diagram	409
16.5.2	Allocating Behavior	412
16.5.3	Defining the Ports on the Blocks	414
16.5.4	Creating the Internal Block Diagram with Parts, Ports, Connectors, and Item Flows	414
16.5.5	Allocation of Flow	417
16.6	Analyze Performance	417
16.6.1	Item Flow Heat Balance Analysis	417
16.6.2	Resolving Heat Balance	420
16.7	Modify the Original Design	420
16.7.1	Updating Behavior	420
16.7.2	Updating Allocation and Structure	421
16.7.3	Controlling the Distiller and the User Interaction	425
16.7.4	Developing a User Interface and a Controller	426
16.7.5	Startup and Shutdown Considerations	427
16.8	Summary	429
16.9	Questions	429
 CHAPTER 17	 Residential Security System Example Using the Object- Oriented Systems Engineering Method	 431
17.1	Method Overview	431
17.1.1	Motivation and Background	431
17.1.2	System Development Process Overview	432
17.1.3	OOSEM System Specification and Design Process	435
17.2	Residential Security Example Overview	437
17.2.1	Problem Background	437
17.2.2	Project Startup	437

17.3	Applying OOSEM to Specify and Design the Residential Security System.....	438
17.3.1	Setup Model	439
17.3.2	Analyze Stakeholder Needs	444
17.3.3	Analyze System Requirements	453
17.3.4	Define Logical Architecture.....	465
17.3.5	Synthesize Candidate Physical Architectures.....	472
17.3.6	Optimize and Evaluate Alternatives	501
17.3.7	Manage Requirements Traceability	507
17.3.8	OOSEM Support to Integrate and Verify System	513
17.3.9	Develop Enabling Systems	515
17.4	Summary	518
17.5	Questions	519

PART IV TRANSITIONING TO MODEL-BASED SYSTEMS ENGINEERING

CHAPTER 18 Integrating SysML into a Systems Development Environment..... 523

18.1	Understanding the System Model's Role in the Broader Modeling Context	523
18.1.1	The System Model as an Integrating Framework	523
18.1.2	Types of Models and Simulations.....	523
18.1.3	Using the System Model with Other Models.....	526
18.2	Tool Roles in a Systems Development Environment.....	530
18.2.1	Use of Tools to Model and Specify the System.....	530
18.2.2	Use of Tools to Manage the Design Configuration and Related Data....	531
18.2.3	Use of Tools to View and Document the Data.....	534
18.2.4	Verification and Validation Tools.....	535
18.2.5	Use of Project Management Tools to Manage the Development Process.....	535
18.3	An Overview of Information Flow between Tools.....	535
18.3.1	Interconnecting the System Modeling Tool with Other Tools.....	535
18.3.2	Interface with Requirements Management Tool	536
18.3.3	Interface with SoS/Business Modeling Tools.....	538
18.3.4	Interface with Simulation and Analysis Tools.....	538
18.3.5	Interface with Verification Tools.....	539
18.3.6	Interface with Development Tools.....	539
18.3.7	Interface with Documentation & View Generation Tool	540
18.3.8	Interface with Configuration Management Tool.....	540
18.3.9	Interface with Project Management Tool	542
18.4	Data Exchange Mechanisms	542
18.4.1	Considerations for Data Exchange	542
18.4.2	File-Based Exchange.....	544
18.4.3	API-based Exchange	546
18.4.4	Performing Transformations	547

18.5	Data Exchange Applications.....	548
18.5.1	SysML to Modelica (bidirectional transformation).....	548
18.5.2	Interchanging SysML Models and Ontologies.....	552
18.5.3	Document Generation from Models (unidirectional transformation).....	552
18.6	Selecting a System Modeling Tool.....	553
18.6.1	Tool Selection Criteria.....	553
18.6.2	SysML Compliance.....	554
18.7	Summary	554
18.8	Questions	555
CHAPTER 19	Deploying SysML into an Organization.....	557
19.1	Improvement Process	557
19.1.1	Monitor and Assess.....	558
19.1.2	Plan the Improvement	559
19.1.3	Define Changes to Process, Methods, Tools, and Training.....	559
19.1.4	Pilot the Approach	560
19.1.5	Deploy Changes Incrementally	561
19.2	Summary	563
19.3	Questions	563
	Appendix A.....	565
	References.....	595
	Index	599

Preface

Systems engineering is a multidisciplinary approach for developing solutions to complex engineering problems. The continuing increase in system complexity is demanding more rigorous and formalized systems engineering practices. In response to this demand, along with advancements in computer technology, the practice of systems engineering is undergoing a fundamental transition from a document-based approach to a model-based approach. In a model-based approach, the emphasis shifts from producing and controlling documentation about the system, to producing and controlling a coherent model of the system. Model-based systems engineering (MBSE) can help to manage complexity, while at the same time improve design quality and cycle time, improve communications among a diverse development team, and facilitate knowledge capture and design evolution.

A standardized and robust modeling language is considered a critical enabler for MBSE. The Systems Modeling Language (OMG SysML™) is one such general-purpose modeling language that supports the specification, design, analysis, and verification of systems that may include hardware, software, data, personnel, procedures, and facilities. SysML is a graphical modeling language with a semantic foundation for representing requirements, behavior, structure, and properties of the system and its components. It is intended to model systems from a broad range of industry domains such as aerospace, automotive, health care, and so on.

SysML is an extension of the Unified Modeling Language (UML), version 2, which has become the de facto standard software modeling language. Requirements were issued by the Object Management Group (OMG) in March 2003 to extend UML to support systems modeling. UML 2 was selected as the basis for SysML because it is a robust language that addresses many of the systems engineering needs, while enabling the systems engineering community to leverage the broad base of experience and tool vendors that support UML. This approach also facilitates the integration of systems and software modeling, which has become increasingly important for today's software-intensive systems.

The development of the language specification was a collaborative effort between members of the OMG, the International Council on Systems Engineering (INCOSE), and the AP233 Working Group of the International Standards Organization (ISO). Following three years of development, the OMG SysML specification was adopted by the OMG in May 2006 and the formal version 1.0 language specification was released in September 2007. Since that time, new versions of the language have been adopted by the OMG. This book is intended to reflect the SysML v1.3 specification, which was close to finalization at the time of this writing. It is expected that SysML will continue to evolve in its expressiveness, precision, usability, and interoperability through further revisions to the specification based on feedback from end users, tool vendors, and research activities. Information on the latest version of SysML, tool implementations of SysML, and related resources, are available on the official OMG SysML web site at <http://www.omg.sysml.org>.

BOOK ORGANIZATION

This book provides the foundation for understanding and applying SysML to model systems as part of a model-based systems engineering approach. The book is organized into four parts: Introduction, Language Description, Modeling Examples, and Transitioning to Model-Based Systems Engineering.

Part I, Introduction, contains four chapters that provide an overview of systems engineering, a summary of key MBSE concepts, a chapter on getting started with SysML, and a sample problem to highlight the basic features of SysML. The systems engineering overview and MBSE concepts in Chapters 1 and 2 set the context for SysML, and Chapters 3 and 4 provide an introduction to SysML.

Part II, Language Description, provides the detailed description of the language. Chapter 5 provides an overview of the language architecture, and Chapters 6 through 14 describe key concepts related to model organization, blocks, parametrics, activities, interactions, states, use cases, requirements, and allocations, and Chapter 15 describes the language extension mechanisms to further customize the language. The ordering of the chapters and the concepts are not based on the ordering of activities in the systems engineering process, but are based on the dependencies between the language concepts. Each chapter builds the readers' understanding of the language concepts by introducing SysML constructs: their meaning, notation, and examples of how they are used. The example used to demonstrate the language throughout Part II is a security surveillance system. This example should be understandable to most readers and has sufficient complexity to demonstrate the language concepts.

Part III, Modeling Examples, includes two examples to illustrate how SysML can support different model-based methods. The first example in Chapter 16 applies to the design of a water distiller system. It uses a simplified version of a classic functional analysis and allocation method. The second example in Chapter 17 applies to the design of a residential security system. It uses a comprehensive object-oriented systems engineering method (OOSEM) and emphasizes how the language is used to address a wide variety of systems engineering concerns, including black-box versus white-box design, logical versus physical design, and the design of distributed systems. While these two methods are considered representative of how model-based systems engineering using SysML can be applied to model systems, SysML is intended to support a variety of other model-based systems engineering methods as well.

Part IV, Transitioning to Model-Based Systems Engineering, addresses how to transition MBSE with SysML into an organization. Chapter 18 describes how to integrate SysML into a systems development environment. It describes the different tool roles in a systems development environment, and the type of data that are exchanged between a SysML tool and other classes of tools. The chapter also describes some of the types of data exchange mechanisms and applications, and a discussion on the criteria for selecting a SysML modeling tool. Chapter 19 is the last chapter of the book, and describes how to deploy MBSE with SysML into an organization as part of an improvement process.

Questions are included at the end of each chapter to test readers' understanding of the material. The answers to the questions can be found on the following Web site at <http://www.elsevierdirect.com/companions/9780123852069>.

The Appendix contains the SysML notation tables. These tables provide a reference guide for SysML notation along with a cross reference to the applicable sections in Part II of the book where the language constructs are described in detail.

USES OF THIS BOOK

This book is a "practical guide" targeted at a broad spectrum of industry practitioners and students. It can serve as an introduction and reference for practitioners, as well as a text for courses in systems modeling and model-based systems engineering. In addition, because SysML reuses many UML

concepts, software engineers familiar with UML can use this information as a basis for understanding systems engineering concepts. Also, many systems engineering concepts come to light when using an expressive language, and as such, this book can be used to help teach systems engineering concepts. Finally, this book can serve as a primary reference to prepare for the OMG Certified System Modeling Professional (OCSMP) exam (refer to <http://www.omg.org/ocsmpl/>).

HOW TO READ THIS BOOK

A first-time reader should pay close attention to the introductory chapters including Getting Started with SysML in Chapter 3, and the application of the basic feature set of SysML to the Automobile Example in Chapter 4. The introductory reader may also choose to do a cursory reading of the overview sections in Part II, and then review the simplified distiller example in Part III. A more advanced reader may choose to read the introductory chapters, do a more comprehensive review of Part II, and then review the residential security example in Part III. Part IV is of general interest to those interested in trying to introduce SysML and MBSE to their organization or project.

The following recommendations apply when using this book as a primary reference for a course in SysML and MBSE. An instructor may refer to the course on SysML that was prepared and delivered by The Johns Hopkins University Applied Physics Lab that is available for download at <http://www.jhuapl.edu/ott/Technologies//Copyright/SysML.asp>. This course provides an introduction to the basic features of SysML so that students can begin to apply the language to their projects. This course consists of eleven (11) modules that use this book as the basis for the course material. The course material for the language concepts is included in the download, but the course material for the tool instruction is not included. Using this course as an example course that introduces the language concepts, the instructor can create a course that includes both the language concepts and tool instruction on how to create and update the modeling artifacts using a selected tool. A shorter version of this course is also included on The Johns Hopkins site which has been used as a full day tutorial to provide an introductory short course on SysML. Refer to the End-User License Agreement included with the download instructions on The Johns Hopkins site for how this material can be used.

A second course on the same website summarizes the Object-Oriented Systems Engineering Method (OOSEM) that is the subject of Chapter 17 in Part III of this book. This provides an example of an MBSE method that can be tailored to meet the needs of specific applications.

An instructor may also require that the students review Chapters 1 and 2, and then study Chapter 3 on Getting Started with SysML. The student should also review the simplified MBSE method in Chapter 3, and create a system model of similar complexity to the Air Compressor example in the chapter. The student may want to review the tool section in the chapter to begin to familiarize themselves with a SysML modeling tool. The student should then study the automobile example in Chapter 4, and recreate some or all of the model in a modeling tool. Alternatively, if a modeling tool is not used, the students can use the Visio SysML template available for download on the OMG SysML website (<http://www.omgsysml.org>).

After working through this example, the instructor may choose to introduce one chapter from Part II during each following lecture to teach the language concepts in more depth. In an introductory course, the instructor may choose to focus on the SysML basic feature set, which is highlighted

throughout each chapter in Part II. The notation tables in the appendix can be used as a summary reference for the language syntax.

This second edition is also intended to be used to prepare for the OMG Certified Systems Modeling Professional (OCSMP) exams to become certified as a model user or model builder. The book can be used in a similar way as described above. For the first two levels of certification, the emphasis is on the basic SysML feature set. The automobile example in Chapter 4 covers most of the basic feature set of SysML, so this is an excellent place to start. In addition, each chapter in Part II shades the paragraphs that represent the basic feature set. In addition, the notation tables in the Appendix include shaded rows for the notational elements that are part of the SysML basic feature set. The unshaded rows constitute the remaining features that reflect the full feature set which is the covered in the third level of OCSMP certification.

CHANGES FROM PREVIOUS EDITION

This edition is intended to update the book content to be current with version 1.3 of the SysML specification, which was in the final stages of completion at the time of this writing. The changes for each SysML specification revision with change bars are available from the OMG website at http://www.omg.org/technology/documents/domain_spec_catalog.htm#OMGSysML. This update also includes marking of the basic feature set in Part II to differentiate it from the full feature set, and other changes to support preparation for the OCSMP exams. In addition, several other changes were made to this book to improve the quality and readability of the text and figures, and to incorporate additional relevant content. Some of the more significant changes are summarized below.

Chapter 3 is added in Part I and called Getting Started with SysML, to provide an introduction to a simplified variant of the language called SysML-Lite, as well as an introduction to a generic SysML modeling tool, and simplified MBSE method. The Automobile Example in Chapter 4 (previously Chapter 3) was revised to focus on the basic feature set of SysML, and is consistent with requirements for the OCSMP level 1 and 2 exams. Chapter 7 (previously Chapter 6) on blocks includes a significant rewrite to address the changes to ports and flows introduced in SysML v1.3. Chapter 9 (previously Chapter 8) on activities includes a new section on the Semantics of a Foundational Subset for Executable UML Models (fUML) which specifies execution semantics for activity diagrams. Chapter 18 (previously Chapter 17) on Integrating SysML into a Systems Development Environment, has been significantly rewritten to update existing sections and introduce new sections. The new sections include discussions on configuration management, auto-generation of documentation, a more elaborated discussion on transformations, and a summary of the SysML to Modelica Transformation specification. The modeling methods in Part III, include both the distiller example using functional analysis methods in Chapter 16 (previously Chapter 15) and the residential security example using the object-oriented systems engineering method (OOSEM) in Chapter 17 (previously Chapter 16). These chapters have been significantly refined to improve the conciseness and understandability of the methods and the quality of the figures.

Acknowledgments

The authors wish to acknowledge the many individuals and their supporting organizations who participated in the development of SysML and provided valuable insights throughout the language development process. The individuals are too numerous to mention here but are listed in the OMG SysML specification. The authors wish to especially thank the reviewers of this book for their valuable feedback; they include Conrad Bock, Roger Burkhart, Jeff Estefan, Doug Ferguson, Dr. Kathy Laskey, Dr. Leon McGinnis, Dr. Øystein Haugen, Dr. Chris Paredis, Dr. Russell Peak, and Bran Selic. The authors also wish to thank Joe Wolfrom and Ed Seidewitz, who contributed to the review of the second edition, and to Joe Wolfrom as the primary author of the Johns Hopkins University Applied Physics Lab course material on SysML and OOSEM referred to above.

SysML is implemented in many different tools. For this book, we selected certain tools for representing the examples but are not endorsing them over other tools. We do wish, however, to acknowledge some vendors for the use of their tools for both the first and second edition, including Enterprise Architect by Sparx Systems, No Magic by Magic Draw, and the Microsoft Visio SysML template provided by Pavel Hruby.

About the Authors

Sanford Friedenthal is an industry leader in model-based systems engineering (MBSE) and an independent consultant. Previously, as a Lockheed Martin Fellow, he led the corporate engineering effort to enable Model-Based Systems Development (MBSD) and other advanced practices across the company. In this capacity, he was responsible for developing and implementing strategies to institutionalize the practice of MBSD across the company, and provide direct model-based systems engineering support to multiple programs.

His experience includes the application of systems engineering throughout the system life cycle from conceptual design through development and production on a broad range of systems. He has also been a systems engineering department manager responsible for ensuring that systems engineering is implemented on programs. He has been a lead developer of advanced systems engineering processes and methods, including the Object-Oriented Systems Engineering Method (OOSEM). Sandy also was a leader of the industry team that developed SysML from its inception through its adoption by the OMG.

Mr. Friedenthal is well known within the systems engineering community for his role in leading the SysML effort and for his expertise in model-based systems engineering methods. He has been recognized as an International Council on Systems Engineering (INCOSE) Fellow for these contributions. He has given many presentations on these topics to a wide range of professional and academic audiences, both within and outside the US.

Alan Moore is an Architecture Modeling Specialist at The MathWorks and has extensive experience in the development of real-time and object-oriented methodologies and their application in a variety of problem domains. Previously at ARTiSAN Software Tools, he was responsible for the development and evolution of Real-time Perspective, ARTiSAN's process for real-time systems development. Alan has been a user and developer of modeling tools throughout his career, from early structured programming tools to UML-based modeling environments.

Mr. Moore is an active member of the Object Management Group and chaired both the finalization and revision task forces for the UML Profile for Schedulability and Performance and Time, and was a co-chair of the OMG's Real-time Analysis and Design Working Group. Alan also served as the language architect for the SysML Development Team.

Rick Steiner is an Engineering Fellow at Raytheon and a Raytheon Certified Architect. He has focused on pragmatic application of systems engineering modeling techniques since 1993 and has been an active participant in the International Council on Systems Engineering (INCOSE) model-based systems engineering activities.

He has been an internal advocate, consultant, and instructor of model-driven systems development within Raytheon. Rick has served as chief engineer, architect, and lead system modeler for several large-scale electronics programs, incorporating the practical application of the Object-Oriented Systems Engineering Method (OOSEM), and generation of Department of Defense Architecture Framework (DoDAF) artifacts from complex system models.

Mr. Steiner was a key contributor to the original requirements for SysML, the development of the SysML specification, and the SysML finalization and revision task forces. His main contribution to this specification has been in the area of allocations, sample problems, and requirements. He provided frequent tutorials and presentations on SysML and model-driven system development at INCOSE symposia and meetings, NDIA conferences, and internal to Raytheon.