

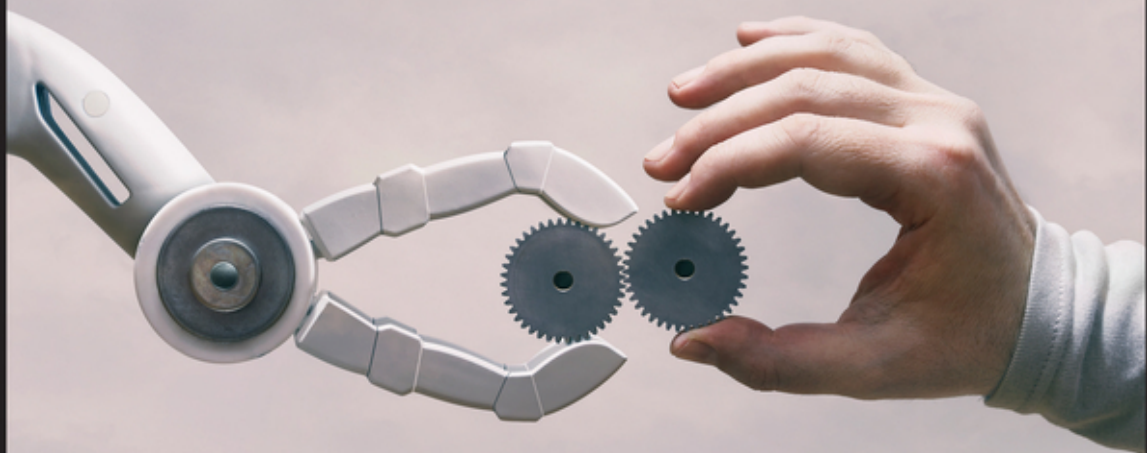
Wiley Series in Systems Engineering and Management

William Rouse, Series Editor

SECOND EDITION

MODEL-BASED SYSTEM ARCHITECTURE

TIM WEILKIENS • JESKO LAMM
STEPHAN ROTH • MARKUS WALKER



WILEY

Model-Based System Architecture

**WILEY SERIES IN SYSTEMS ENGINEERING
AND MANAGEMENT**

William Rouse, Series Editor
Andrew P. Sage, Founding Editor

A complete list of the titles in this series appears at the end of this volume.

Model-Based System Architecture

Second Edition

Tim Weilkiens

Hamburg, Germany

Jesko G. Lamm

Bern, Switzerland

Stephan Roth

Hamburg, Germany

Markus Walker

Ziefen, Switzerland

WILEY

This edition first published 2022
© 2022 John Wiley & Sons, Inc.

Edition History

John Wiley & Sons (1e, 2015)

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, except as permitted by law. Advice on how to obtain permission to reuse material from this title is available at <http://www.wiley.com/go/permissions>.

The right of Tim Weilkiens, Jesko G. Lamm, Stephan Roth, and Markus Walker to be identified as the authors of this work has been asserted in accordance with law.

Registered Office

John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, USA

Editorial Office

111 River Street, Hoboken, NJ 07030, USA

For details of our global editorial offices, customer services, and more information about Wiley products visit us at www.wiley.com.

Wiley also publishes its books in a variety of electronic formats and by print-on-demand. Some content that appears in standard print versions of this book may not be available in other formats.

Limit of Liability/Disclaimer of Warranty

In view of ongoing research, equipment modifications, changes in governmental regulations, and the constant flow of information relating to the use of medicines, equipment, and devices, the reader is urged to review and evaluate the information provided in the package insert or instructions for each medicine, equipment, or device for, among other things, any changes in the instructions or indication of usage and for added warnings and precautions. While the publisher and authors have used their best efforts in preparing this work, they make no representations or warranties with respect to the accuracy or completeness of the contents of this work and specifically disclaim all warranties, including without limitation any implied warranties of merchantability or fitness for a particular purpose. No warranty may be created or extended by sales representatives, written sales materials or promotional statements for this work. The fact that an organization, website, or product is referred to in this work as a citation and/or potential source of further information does not mean that the publisher and authors endorse the information or services the organization, website, or product may provide or recommendations it may make. This work is sold with the understanding that the publisher is not engaged in rendering professional services. The advice and strategies contained herein may not be suitable for your situation. You should consult with a specialist where appropriate. Further, readers should be aware that websites listed in this work may have changed or disappeared between when this work was written and when it is read. Neither the publisher nor authors shall be liable for any loss of profit or any other commercial damages, including but not limited to special, incidental, consequential, or other damages.

Library of Congress Cataloging-in-Publication Data applied for:

ISBN: 9781119746652

Cover Design: Wiley

Cover Image: © Ivan Bajic/Getty Images

Set in 9.5/12.5pt STIXTwoText by Straive, Chennai, India

10 9 8 7 6 5 4 3 2 1

Contents

	Foreword	<i>xv</i>
	Preface	<i>xvii</i>
	About the Companion Website	<i>xxi</i>
1	Introduction	<i>1</i>
2	An Example: The Scalable Observation and Rescue System	<i>5</i>
3	Better Products – The Value of Systems Architecting	<i>9</i>
3.1	The Share of Systems Architecting in Making Better Products	<i>9</i>
3.2	Benefits that can be Achieved	<i>10</i>
3.2.1	Benefit for the Customer	<i>10</i>
3.2.2	Benefit for the Organization	<i>12</i>
3.3	Benefits that can be Communicated Inside the Organization	<i>14</i>
3.4	Beneficial Elements of Systems Architecting	<i>15</i>
3.5	Benefits of Model-Based Systems Architecting	<i>16</i>
4	Systems, Systems of Systems, and Cyber-Physical Systems	<i>17</i>
4.1	Definition of “System”	<i>17</i>
4.1.1	System Elements	<i>19</i>
4.1.2	System Context	<i>20</i>
4.1.3	System Characteristics	<i>21</i>
4.1.4	Purpose	<i>22</i>
4.1.5	System Evolution	<i>23</i>
4.2	Definition of “System of Systems”	<i>23</i>
4.3	Definition of “Cyber-Physical System”	<i>26</i>
4.4	Composition of a “Cyber-Physical System of Systems”	<i>27</i>

5	Definition of System Architecture	31
5.1	What Is Architecture? – Discussion of Some Existing Definitions	31
5.2	Relations Between Concepts of “System,” “Architecture,” and “Architecture Description”	33
5.3	Definition of “Architecture”	35
5.3.1	Interactions	36
5.3.2	Principles	37
5.3.3	Architecture Decisions	37
5.4	Functional and Physical Architecture	37
5.5	Taxonomy of Physical Architectures	39
5.5.1	Logical Architecture	40
5.5.2	Product Architecture	41
5.5.3	Base Architecture	41
5.6	Architecture Landscape for Systems	41
5.6.1	System Architecture	42
5.6.2	System Design	43
5.6.3	Discipline-Specific Architecture and Design	44
6	Model-Based Systems Architecting	45
7	Model Governance	51
7.1	Overview	51
7.2	Model Governance in Practice	52
8	Architecture Description	57
8.1	Architecture Descriptions for Stakeholders	58
8.2	Definition of “Architecture Description”	60
8.2.1	Architecture Viewpoints	62
8.2.2	Architecture Views	65
8.2.3	Architecture Decisions	67
8.2.4	Architecture Rationales	69
8.3	How to Get Architecture Descriptions?	69
8.3.1	Model-Based Vision	69
8.3.2	Forms and Templates	71
9	Architecture Patterns and Principles	75
9.1	The SYSMOD Zigzag Pattern	76
9.2	The Base Architecture	82
9.3	Cohesion and Coupling	85
9.4	Separation of Definition, Usage, and Run-Time	87
9.5	Separate Stable from Unstable Parts	89

9.6	The Ideal System	89
9.7	View and Model	90
9.8	Diagram Layout	92
9.9	System Model Structure	93
9.10	System Architecture Principles	95
9.11	Heuristics	95
9.11.1	Heuristics as a Tool for the System Architect	95
9.11.2	Simplify, Simplify, Simplify: Strength and Pitfall	97
10	Model-Based Requirements Engineering and Use Case Analysis	99
10.1	Requirement and Use Case Definitions	99
10.2	Model-Based Requirements and Use Case Analysis from the MBSA Viewpoint	102
10.2.1	Identify and Define Requirements	103
10.2.2	Specify the System Context	104
10.2.3	Identify Use Cases	105
10.2.4	Describe Use Case Flows	109
10.2.5	Model the Domain Knowledge	110
10.3	The SAMS Method	112
10.3.1	SAMS Method Definitions	113
10.3.2	SAMS Method	114
10.4	Use Cases 2.0	117
11	Perspectives, Viewpoints and Views in System Architecture	119
11.1	Introduction	119
11.2	The Functional Perspective	121
11.2.1	SysML Modeling of Functional Blocks	123
11.2.2	Architecture Views for the System Architect	124
11.2.3	Different Architecture Views for the Stakeholders of Different Functions	124
11.3	The Physical Perspective	125
11.3.1	Logical Architecture Example	126
11.3.2	Product Architecture Example	127
11.4	The Behavioral Perspective	130
11.5	The Layered Perspective	130
11.5.1	The Layered Approach	130
11.5.2	The Layered Perspective in Systems Architecting	132
11.5.3	Relation to the Domain Knowledge Model	134
11.5.4	Architecting the Layers	136

11.5.5	SysML Modeling of Layers	136
11.6	System Deployment Perspective	142
11.7	Other Perspectives	144
11.8	Relation to the System Context	146
11.8.1	Validity of the System Boundary	146
11.8.2	Using the System Context as a Part of the Stakeholder-Specific Views	146
11.8.3	Special System Context View for Verification	147
11.9	Mapping Different System Elements Across Different Levels	148
11.9.1	Functional-to-Physical Perspective Mapping	149
11.9.2	Mapping More Perspectives	153
11.9.3	Mapping Different Levels	153
11.10	Traceability	155
11.11	Perspectives and Architecture Views in Model-based Systems Architecting	155
11.11.1	Creating Different Architecture Views in a Model-Based Approach	155
11.11.2	Using SysML for Working with Different Perspectives and Architecture Views	157
11.11.3	The Importance of Architecture Viewpoints in Model-Based Systems Architecting	159
12	Typical Architecture Stakeholders	161
12.1	Overview	161
12.2	Requirements Engineering	162
12.3	Verification	163
12.4	Configuration Management	166
12.5	Engineering and Information Technology Disciplines	167
12.6	Project and Product Management	171
12.7	Risk Managers	174
12.8	Development Roadmap Planners	174
12.9	Production and Distribution	177
12.10	Suppliers	178
12.11	Marketing and Brand Management	178
12.12	Management	180
13	Roles	185
13.1	Roles	185
13.2	The System Architect Role	186
13.2.1	Objective	186
13.2.2	Responsibilities	186

- 13.2.3 Tasks 187
- 13.2.4 Competences 188
- 13.2.5 Required Skills of a System Architect 188
- 13.2.6 Required Skills for Model-Based Systems Architecting 190
- 13.3 System Architecture Teams 190
- 13.4 System Architecture Stakeholders 192
- 13.5 Recruiting System Architecture People 192
- 13.6 Talent Development for System Architects 194

- 14 Processes 199**
 - 14.1 Systems Architecting Processes 199
 - 14.1.1 Overview 199
 - 14.1.2 Example of Generic Process Steps 201
 - 14.1.3 Example of Concrete Process Steps 202
 - 14.1.4 Validation, Review, and Approval in a Model-Based Environment 203
 - 14.2 Design Definition Process 207
 - 14.3 Change and Configuration Management Processes 207
 - 14.4 Other Processes Involving the System Architect 207

- 15 Tools for the Architect 209**

- 16 Agile Approaches 213**
 - 16.1 The History of Iterative–Incremental Approaches 214
 - 16.1.1 Project Mercury (NASA, 1958) 214
 - 16.1.2 The New New Product Development Game (1986) 215
 - 16.1.3 Boehm’s Spiral Model (1988) 216
 - 16.1.4 Lean (1945 Onwards) 217
 - 16.1.5 Dynamic Systems Development Method (DSDM, 1994) 219
 - 16.1.6 Scrum (1995) 220
 - 16.2 The Manifesto for Agile Software Development (2001) 221
 - 16.3 Agile Principles in Systems Engineering 223
 - 16.3.1 Facilitate Face-to-Face Communication 223
 - 16.3.2 Create a State of Confidence 224
 - 16.3.3 Build Transdisciplinary and Self-Organized Teams 225
 - 16.3.4 Create a Learning Organization 225
 - 16.3.5 Design, but No Big Design (Up-Front) 226
 - 16.3.6 Reduce Dependencies 227
 - 16.3.7 Foster a Positive Error Culture 228
 - 16.4 Scaling Agile 228
 - 16.5 System Architects in an Agile Environment 230

17	The FAS Method	233
17.1	Motivation	234
17.2	Functional Architectures for Systems	236
17.3	How the FAS Method Works	239
17.4	FAS Heuristics	242
17.5	FAS with SysML	244
17.5.1	Identifying Functional Groups	244
17.5.2	Modeling the Function Structure	246
17.5.3	Modeling the Functional Architecture	249
17.6	SysML Modeling Tool Support	250
17.6.1	Create Initial Functional Groups	251
17.6.2	Changing and Adding Functional Groups	254
17.6.3	Creating Functional Blocks and their Interfaces	254
17.7	Mapping of a Functional Architecture to a Physical Architecture	254
17.8	Experiences with the FAS Method	256
17.9	FAS Workshops	258
17.10	Quality Requirements and the Functional Architecture	259
17.11	Functional Architectures and the Zigzag Pattern	262
17.12	CPS-FAS for Cyber-physical Systems	263
18	Product Lines and Variants	269
18.1	Definitions Variant Modeling	270
18.2	Variant Modeling with SysML	271
18.3	Other Variant Modeling Techniques	276
19	Architecture Frameworks	279
19.1	Enterprise Architectures	280
19.2	Characteristics of System of Systems (SoS)	282
19.2.1	Emergence	283
19.3	An Overview of Architecture Frameworks	285
19.3.1	Zachman Framework™	285
19.3.2	The TOGAF® Standard	286
19.3.3	Federal Enterprise Architecture Framework (FEAF)	288
19.3.4	Department of Defense Architecture Framework (DoDAF)	289
19.3.5	Ministry of Defense Architecture Framework (MODAF)	290
19.3.6	NATO Architecture Framework (NAF)	291
19.3.7	TRAK	292
19.3.8	European Space Agency Architectural Framework (ESA-AF)	293
19.3.9	OMG Unified Architecture Framework® (UAF®)	295
19.4	System Architecture Framework (SAF)	296
	Together with Michael Leute	296

- 19.4.1 SAF and Enterprise Frameworks 296
- 19.4.2 SAF Ontology 298
- 19.5 What to Do When We Come in Touch With Architecture Frameworks 298

- 20 Cross-cutting Concerns 301**
 - 20.1 The Game-Winning Nonfunctional Aspects 301
 - 20.2 Human System Interaction and Human Factors Engineering 303
 - 20.3 Risk Management 304
 - 20.4 Trade Studies 305
 - 20.5 Budgets 306

- 21 Architecture Assessment 307**

- 22 Making It Work in the Organization 313**
 - 22.1 Overview 313
 - 22.2 Organizational Structure for Systems Architecting 314
 - 22.3 Recipes from the Authors' Experience 318
 - 22.3.1 Be Humble 319
 - 22.3.2 Appraise the Stakeholders 319
 - 22.3.3 Care About Organizational Interfaces 319
 - 22.3.4 Show that it Was Always There 321
 - 22.3.5 Lead by Good Example 321
 - 22.3.6 Collect Success Stories and Share them When Appropriate 322
 - 22.3.7 Acknowledge that Infections Beat Dictated Rollout 323
 - 22.3.8 Assign the System Architect Role to Yourself 324
 - 22.3.9 Be a Leader 324

- 23 Soft Skills 327**
 - 23.1 It's All About Communication 328
 - 23.1.1 Losses in Communication 329
 - 23.1.2 The Anatomy of a Message 330
 - 23.1.3 Factors Influencing Communication 333
 - 23.1.3.1 The Language 333
 - 23.1.3.2 The Media Used 333
 - 23.1.3.3 Spatial Distance 333
 - 23.1.3.4 Various Connotations of Words 335
 - 23.1.4 The Usage of Communication Aids and Tools 335
 - 23.2 Personality Types 338
 - 23.2.1 Psychological Types by C. G. Jung 338
 - 23.2.2 The 4MAT System by Bernice McCarthy 340

- 23.3 Team Dynamics 341
- 23.4 Diversity and Psychological Safety 342
 - 23.4.1 Project Aristotle (Google) 342
 - 23.4.2 Elements of Psychological Safety 343
- 23.5 Intercultural Collaboration Skills 344

- 24 Outlook: The World After Artificial Intelligence 347**

Appendix A OMG Systems Modeling Language 349

- A.1 Architecture of the Language 350
- A.2 Diagram and Model 352
- A.3 Structure Diagrams 353
 - A.3.1 Block Definition Diagram 354
 - A.3.2 Internal Block Diagram 357
 - A.3.3 Parametric Diagram 361
 - A.3.4 Package Diagram 362
- A.4 Behavior Diagrams 363
 - A.4.1 Use Case Diagram 364
 - A.4.2 Activity Diagram 366
 - A.4.3 State Machine Diagram 369
 - A.4.4 Sequence Diagram 371
- A.5 Requirements Diagram 372
- A.6 Extension of SysML with Profiles 374
- A.7 Next-Generation Modeling Language SysML v2 376

Appendix B The V-Model 381

- B.1 A Brief History of the V-Model or the Systems Engineering Vee 381
- B.2 A Handy Illustration but No Comprehensive Process Description 383
- B.3 Critical Considerations 385
 - B.3.1 The V-Model as Process Description 386
 - B.3.2 The V-Model Does Not Impose a Waterfall Process 386
 - B.3.3 The V-Model Accommodates Iterations 387
 - B.3.4 The V-Model Permits Incremental Development 387
 - B.3.5 The V-Model and Concurrent Engineering 388
 - B.3.6 The V-Model Accommodates Change 388
 - B.3.7 The V-Model Permits Early Verification Planning 388
 - B.3.8 The V-Model Shows Where to Prevent Dissatisfaction 388
- B.4 Reading Instruction for a Modern Systems Engineering Vee 389
 - B.4.1 The Vertical Dimension 389
 - B.4.2 The Horizontal Dimension 389
 - B.4.3 The Left Side 389

B.4.4 The Right Side 390
B.4.5 The Levels 390
B.4.6 Life Cycle Processes 390
B.4.7 The Third Dimension 390

Appendix C Glossary 391

C.1 Heritage of the Term “Glossary” 391
C.2 Terms with Specific Meaning 393

References 399

Index 417

Foreword

Contrary to popular myth, models are not new to systems engineering. Models are the way engineers analyze both problems and solutions, so systems models are as old as systems engineering itself. With the traditional focus on written specifications as the “source of truth,” models were secondary and descriptive – sometimes reflected as simple sketches, sometimes shown in formal diagrams, partially captured in analysis packages, and often trapped in the mind of the chief engineer. The transformation of systems engineering from document-centric to model-centric practices is not about the introduction of models. It is about making models explicit and moving them to the foreground where they serve as the authoritative tool for design, analysis, communication, and system specification.

Organizations today are investing heavily in representations, standards, methodologies, and technologies to transform the practice of systems engineering through model-driven paradigms. To manage the complexity of today’s problems; to keep pace with today’s rapidly evolving technologies; to capture the required knowledge regarding the problem, solution, and rationale; to respond effectively to change – all require that systems engineering join the other engineering disciplines in moving beyond document-centric techniques and embracing the power of a model-based foundation. With energy and focus over the last 10 years has come notable progress. The industry has advanced in the area of representations with the development of SysML as a standardized set of diagrams to complement traditional systems representations. Numerous books – including a frequently-cited guide by Tim Weilkiens – explain the details of using this notation to capture and communicate system designs to improve explicitness and alignment within the systems team. Alongside these representations have emerged countless standards and frameworks to help engineering teams develop high fidelity models reflecting key systems dimensions.

However, for all the industry discussion regarding SysML, representations, standards, and tools, there remains a great deal of confusion. Understanding

SysML notation and drawing SysML diagrams do not equate to doing model-based systems engineering. Nor is the use of disjoint models and simulation in systems engineering equivalent to integrated model-based systems engineering.

Effectively moving forward with the transition to model-centric techniques requires that we step back to understand the bigger picture. Diagrams and other representations do not live in isolation but are interrelated and overlapping, communicating key aspects of the system model from specific viewpoints. System architecture and detailed analytical models are not disjoint, nor is there a single grand unified model to capture all dimensions of interest for all systems problems. To move forward, we must embrace the holistic systems perspective and apply it to model-based systems engineering, seeking out the interrelationships and developing a robust toolbox of supporting practices.

In this book, Tim Weilkiens, Jesko Lamm, Stephan Roth, and Markus Walker broaden our vision and expose us to a rich set of perspectives, processes, and methods so that we can develop an effective unified framework for model-based systems architecture. Building upon the existing industry library of textbooks on SysML, this book looks beyond representation to address models, viewpoints, and views as part of a modern approach addressing requirements, behavior, architecture, and more. It connects to a larger framework of processes, methods, and tools key to enabling model-centric practices. And it looks beyond the technical space to the critical cultural dimensions, because the transformation to model-centric techniques is far less a technical challenge than one of organizational change. Addressing the broader framework, Tim, Jesko, Stephan, and Markus bring model-centric practices together to help practitioners develop cohesive system architectures – our one chance in the life of a program to manage complexity, develop resilience, and design in critical concerns such as system security.

There is no doubt that the future of systems engineering is model-based. Document-centric techniques simply are not enough as we grapple with the challenges of today and tomorrow. Those practitioners and organizations who are early adopters in developing a cohesive model-centric framework of processes, methods, and tools will certainly be at a competitive advantage – whether producing products themselves or delivering systems services for others. If, as a profession, we can transform from document-centric to model-based systems engineering and do so with the vision of enabling model-based engineering, we can help transform the larger product lifecycle delivering radical improvements in quality, cost, and time-to-market for the benefit of all.

June 2015

*David Long
President, Vitech Corporation
INCOSE President (2014 and 2015)*

Preface

Reacting to market needs on time with systems of high quality and marketable costs is a strong competitive advantage. Once a market need has been identified, multiple disciplines are involved in developing a system toward it. They need to collaborate closely and each according to a precise understanding of the own contribution to the system development. Effective communication and the creation of understanding for the whole system-of-interest are keys for the success. Organizations are facing a more and more dynamic environment and, at the same time, an increasing organizational complexity of distributed teams and stakeholders and an increasing technical complexity of more heterogeneous relationships between system components and their environment. This context requests an explicit and sustainable system architecture.

Each of the engineering disciplines contributing to system development needs specific views for obtaining the needed insight. System models enable the creation of consistent sets of stakeholder-specific views. People using them gain a fast and comprehensible understanding of the system they are developing, which can help them choose appropriate solutions for fulfilling the market needs. All the views look at the same data baseline. There is no effort to consolidate redundant data or to clarify misunderstandings of inconsistent information and the costs of resulted errors.

A system architect needs to shape the system architecture well for realizing a successful system. Multiple tasks have to be carried out, each using an effective approach. This book provides a toolbox for the architects for their daily challenges. The scope of the book is a model-based environment, that is either already established and running or planned. The book explains how to use the SysML modeling language in obtaining model-based architecture descriptions. Nevertheless, the concepts are independent of SysML and could also be performed with other modeling languages.

This book is about people, models, and better products, based on our belief that model-based systems architecting produces better products by creating communication and insight for people involved in system development. The book presents a collection of methods and approaches, which we see as ingredients for getting the system architecture work done successfully. We present model-based systems architecting, which we see as a required backbone for excellent system architecture work together with the stakeholders. We will show that involving the stakeholders means much more than running through a formalized review process.

A fundamental principle in system architecture is simplification. Without simple concepts to be communicated to the stakeholders, the system architect will not be understood and thus will fail. We advise you, dear reader, to adopt the principle of simplification and apply it to the multitude of approaches presented in the book. Feel free only to choose the most suitable approaches for your daily work and disregard the others until you are in a situation where they turn out to be the useful ones. The book is a well-stocked toolbox and not a rigid all-or-nothing process for system architects.

Our experience tells us that each organization will have a different focus area and will need different approaches. This is why we have bundled a variety of approaches we have observed being applied successfully in the industry, in the hope that you will find some pieces of information that are suitable exactly to your current activities. We have selected those approaches, which we find easy to apply in daily work and which are important for common model-based system architectures. We do not claim to provide a complete set. Every system architect loves to go to a hardware store to extend her toolbox. And from time to time she has to discard one of her tools when it is no longer appropriate.

The book addresses system architects and their managers as well as engineers who are involved or interested in systems architecting. It is the first comprehensive book that combines the emergent discipline systems architecting with model-based approaches with SysML and puts together puzzle pieces to a complete picture. Highlighted puzzle pieces are:

- functional architectures and the Functional Architecture for Systems (FAS) method by Lamm and Weilkiens to derive the architecture from common-use case analysis
- the integration of the concept of layered architectures from the software discipline in the context of system architectures
- the modeling of system variants
- the whole picture of different architecture kinds like functional, logical, and product architectures and their relationships
- a brief description of SysML and
- a summary of the history of the V-model and recent thinking about it in the appendix

As a typical reader of this book, you may have no time to read all chapters in sequential order. Therefore, we have made the chapters as independent from each other as we could, in order to enable you to read them individually or out of a dedicated sequence when you like inspiration about a certain topic. You can find an on-demand reference about particular topics and get inspiration for directly using the presented approaches in your daily business. The topics are demonstrated using a fictitious robot-based solution for virtual exhibition or other robot-based telepresence tours as an example system.

We like to write texts using gender-fair language. On the other hand, we avoid cluttering the flow of reading by using always both genders in the same sentence. Therefore, we have only used one gender where it was not appropriate to use gender-neutral language. Feel free to replace “he” by “she” and “she” by “he” or whatever is appropriate.

We like to thank the “FAS” and “MkS” working groups of GfSE, German chapter of INCOSE, as well as the “Viewpoints” working group of the same chapter in collaboration with Swiss Society for Systems Engineering (SSSE), Swiss chapter of INCOSE. The work in these groups has provided us with new ideas that can now be found in this book. We thank NoMagic for their support in working with the Cameo tool family that we used to create the SysML models and diagrams we used in multiple chapters of this book. We also thank Erik Solda for allowing us to use the robot example, Martin Ruch for contributing ideas about the assessment of organizational interfaces, and all the colleagues at work who have influenced our way of thinking, helped us with foreign languages in both reading and writing or recommended literature and web links that are today part of the foundations of this book. We furthermore thank numerous people who provided us with advice after we had shown or explained them little fragments of this book to listen to a second opinion.

We like to thank all the supporters of MBSE who believe that MBSE enables the successful development of complex systems – in particular, David Long, who is a great expert of MBSE from the very beginning and has written the foreword.

Finally, we like to thank Brett Kurzman, editor at Wiley, Alex Castro, Kathleen Pagliaro, Bhargavi Natarajan, Sarah Lomore, and Viniprammia Premkumar for their support with the first and second editions of this book.

September 2021

*Tim Weilkiens, Jesko G. Lamm, Stephan Roth, Markus Walker
Contributors: Matthias Dänzer, Oliver C. Eichmann,
Ralf God, Michael Leute, Sylvia Melzer*

About the Companion Website

This book is accompanied by a companion website:

www.mbse-architecture.com

The website includes:

- High resolution version of all the figures in the book.

1

Introduction

Model-based system architecture (MBSA) combines the two key technologies model-based and systems architecting. Both are major parts of the future state of systems engineering [123].

Many systems result from an evolutionary development. They are driven by their parts and do not emerge from the architecture. The parts could be anything that, in combination, is assembled to a human-made purposeful system. System architecture is followed by a complete system. Often system architecture is referred to as the architecture from the perspective of a software architecture in combination with the hardware or the architecture of software-intensive systems [43]. We understand system architecture more holistically and also consider systems without any software, even though systems without any software are rarely handled with systems engineering processes and MBSA concepts like described in this book. A system architecture is always present. In today and future systems engineering, it is crucial to apply explicit systems architecting for the success of the system project [123]. Chapter 5 defines the term “system architecture” within its context.

Studies clearly show that systems architecting is critical for the performance and success of the system [68]. This is particularly evident for projects that require significant architectural work or rework. Due to more and more dynamic and complex markets and environments, the system architecture must more and more support the changing requirements and requests for radical changes. Chapter 3 lists the benefits of systems architecting.

A system architecture is about establishing solutions that are in line with the directions that guide the organization and checked for feasibility by the corresponding experts, about designing interfaces that are agreed from both sides, and about ensuring that the people who should know the architecture of a system have a common understanding of it. MBSA uses models for enabling the creation of healthy communication around the architecture of the system and for ensuring that the architecture is validated from different points of view. Models are a key

tool to be capable of developing complex systems on time and in a feasible quality. Chapter 6 defines the term “model” and MBSA and discusses related terms.

Models are more than graphics. There are even models without any graphical representations. Just the graphics is not modeling but drawing. To create a model, you need the concrete syntax, the abstract syntax, and the semantics, which you find in a modeling language. We use the international standard Systems Modeling Language (SysML) as a language for the system requirements and architecture models. Appendix A gives an overview about SysML, including an outlook on the next-generation modeling language SysML v2. Although we extensively use SysML in this book, our methods and concepts are independent of SysML and could also be implemented by any other modeling language.

The system architect is the one in charge of shaping the system architecture. This is a big responsibility and a big challenge. Organizations developing systems should carefully select people who are allowed to architect the system – and these people’s work results will be tightly monitored by stakeholders everywhere in the organization. Chapter 22 describes how systems architecting could be embedded in an organization, and Chapter 12 discusses the interfaces to the stakeholders of systems architecting. In particular, Chapter 10 introduces the adjacent discipline requirements engineering that closely collaborates with the systems architecting. The SYSMOD zigzag pattern presented in Chapter 9 shows the relationship between requirements and architecture and clearly demonstrates the need for a close collaboration. Artifacts of the model-based requirements and use case analysis are important inputs for the system architects, especially to elaborate a functional architecture using the so-called Functional Architectures for Systems (FAS) method.

Chapter 17 is a comprehensive presentation of the FAS method. Functional architectures are built of functions only and are independent of the physical components that implement the functions. The functional architecture is more stable than a physical architecture that depends on the frequently changing technologies. The architecture principle to separate stable from unstable parts is covered in Chapter 9 about architecture patterns and principles.

Besides the functional architecture, we define and discuss further system architecture kinds. The base architecture that fixes the preset technologies and adjusts the scope for innovation, the logical architecture that specifies the technical concepts and principles, and the product architecture that finally specifies the concrete system. All three architecture kinds are physical architectures. The layered architecture is an additional aspect to these architecture kinds and is presented in Chapter 11.

Another additional aspect is the modeling of variants. Variability is increasingly important. The markets are no longer satisfied by commodity products. The market requests customized products that fit personal demands of the customers.

Additionally, global markets with different local environments and policies require different configurations of a system. Chapter 18 presents a model-based concept to specify different product configurations and gives a brief introduction to model-based product line engineering (MBPLE).

The architecture concepts are presented with a consistent example system. The “Virtual Tour” system (VT system) provides virtual visits by driving with camera-equipped robots through a real exhibition. The system is easy to understand and, at the same time, sufficiently complex to demonstrate the systems architecting concepts. The VT system is also part of a rescue and observation system to illustrate a system of systems and cyber-physical systems. The systems are introduced in Chapter 2.

The system architect who thinks that his job is to make a diagram and save it on a shared network drive will most probably fail. Same for the system architects who think they are the bosses of the development staff and can instruct the other engineers. It is neither an archaeological job nor a chief instructor job. Systems architecting is collaborative work that requires communication and soft skills. A basis for a good communication is a common language and media to transport the information. Chapter 8 covers the artifacts of the architecture documentations. In Chapter 19, we extend our scope to system of systems and architecture frameworks.

Typically, engineers are focused on the technology challenges of their job. As mentioned, communication and more general soft skills are getting more and more important capabilities. The engineering disciplines are growing together. For instance, that could be seen by the modern discipline mechatronic. And the worldwide humankind is growing together due to the internet other communication and transportation technologies. In consequence, an engineer has an increasing number of communication relationships. She is no longer successful when she only manages her technology tasks. It is also important to collaborate well with team members, stakeholders, communities, and so on. Chapter 23 gives an introduction about soft skills for engineers.

2

An Example: The Scalable Observation and Rescue System

We need an example system for the demonstration of various techniques to be presented in this book. The example shall be based on one single system with one single purpose, but extensible to be explored in scenarios involving the interaction of multiple systems for a purpose different from the one of the original single system.

Our single-purpose system is based on the very old idea of telepresence robots (e.g. [249]). The concrete system was inspired by an organization called “The Workers.” They created a robot system that is called “After Dark” [36], because it is intended to be operated at night, when it is dark – and when almost any museum in town is closed. The system comprises robots that are driving through a closed museum. They carry a lamp to shed light and a camera to capture pictures. When sending the captured pictures to a remote user, the resulting offering is a virtual museum tour (VMT). The described system was demonstrated on 23 August 2014 [139]: After Dark’s robots were driving through the gallery “Tate Britain”, and people worldwide could watch the streamed camera images. A similar virtual museum tour offering based on a robot was started in The Mob Museum, Las Vegas, in 2016 [258], and the same technology was at least considered for several art museums [41].

Inspired by these systems, we present the “Virtual Museum Tour system” (VMT). Its central subsystem will be a robot as shown in Figure 2.1, intended for realizing a remote user’s telepresence in a museum. The presented robot is based on some really existing prototype that was created many years ago during a leisure activity by two students (Erik Solda and Jesko G. Lamm) at Aachen university, Germany. To get back from this historic robot prototype to the example system considered here, please imagine the shown fictitious museum tour robot to be an industrial product with today’s technologies onboard: It will use latest artificial intelligence (AI) to be able to navigate autonomously in a museum. But of course the system also comprises servers to control such robots, cloud services to offer

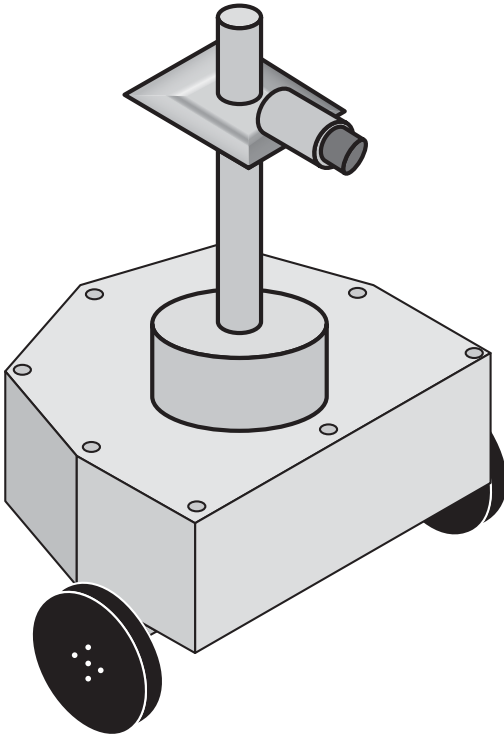


Figure 2.1 The museum tour robot.

onboarding to people worldwide, and apps for mobile devices to schedule virtual museum tours and watch the corresponding video streams.

A storyboard [152] in Figure 2.2 explains the system’s main use case: Currently, John is controlling a museum robot to drive it through a museum of arts. He has to write a report about modern art as a homework for school, and he has not had time to go to the museum during its opening hours. John types “Andy Warhol” on his smartphone and the robot starts driving to the pop arts division of the museum. Once there, it stops in the middle of a room. John now selects a painting showing a soup can. The robot moves toward the painting and stops in front of it. The camera on the robot now transmits a picture of the painting to John’s smartphone. A little notification box on the smartphone displays the title of the painting. John needs to analyze the artist’s way of working in more detail. Via commands entered on his smartphone, he moves the camera down. Then, he zooms in on a particular area of the painting. Now he can see the necessary details via the video stream on his smartphone. This enables John to complete his homework for school.

Unlike the initially mentioned systems at Tate Britain and The Mob Museum, our own example system is purely fictitious and also the extensions to be presented