

# Automotive Software Architectures

Miroslaw Staron

# Automotive Software Architectures

An Introduction



Springer

Mirosław Staron  
Department of Computer Science  
and Engineering  
University of Gothenburg  
Gothenburg, Sweden

ISBN 978-3-319-58609-0      ISBN 978-3-319-58610-6 (eBook)  
DOI 10.1007/978-3-319-58610-6

Library of Congress Control Number: 2017942953

© Springer International Publishing AG 2017

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Printed on acid-free paper

This Springer imprint is published by Springer Nature  
The registered company is Springer International Publishing AG  
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

*To my family—Sylwia, Alexander, Viktoria  
and Cornelia*

# Foreword

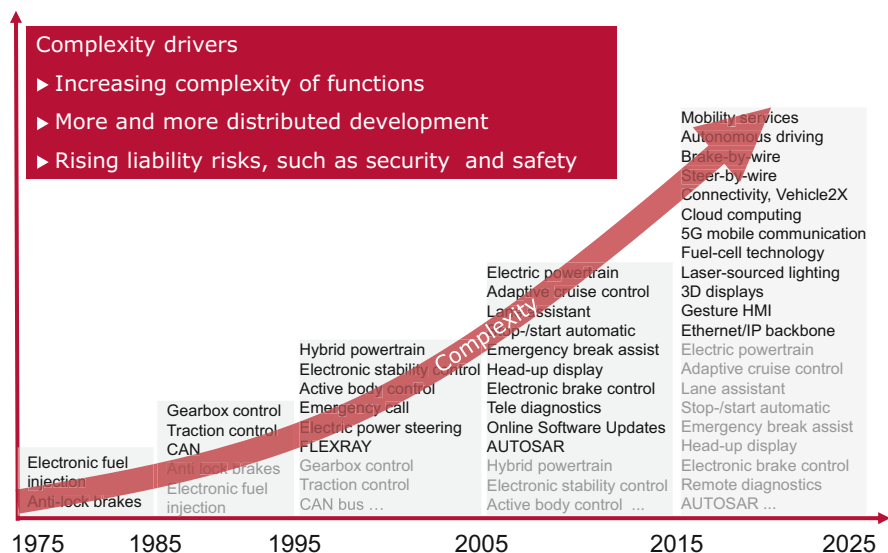
*“Without exception, our aim must be to improve the current status; and instead of being satisfied with what has been achieved, we must always strive to do our job even better.”* This is the direction which the famous automotive entrepreneur Robert Bosch gave already at the dawn of the automotive age. It is still valid today, and we are indeed never satisfied with where we are in the automotive sector.

Software and IT are the major drivers of modern cars—both literally and from a marketing perspective. Modern vehicles have more than 50 electronic control units (ECUs), with premium cars having more than 100 such embedded computer systems. Some functions, such as engine control or dynamics, are hard real-time functions, with reaction times going down to a few milliseconds. Practically all other functions, such as infotainment, demand at least soft real-time behaviors.

Today automotive software is spearheading IT innovation. Software engineering for automotive systems today encompasses modern embedded and cloud technologies, distributed computing, real-time systems, mixed safety and security systems, and, last but not least, the connection of all these elements to long-term sustainable business models. The everyday relevance of automotive software for today’s software engineers is high, and it is the focus of this book to bring this message to practitioners. Its main goal is to underline the convergence of embedded software with highly complex distributed IT systems.

Each automotive area has its own requirements for computational speed, reliability, security, safety, flexibility, and extensibility. Automotive electronic systems map functions such as braking, powertrain, or lighting controls to individual software systems and physical hardware. The resulting complexity has reached a limit that demands an architectural restart. At the same time, innovative functions such as connectivity with external infrastructures and vehicle-to-vehicle communication demand IT backbone and cloud solutions with service-oriented architectures (SOA).

Software and IT in vehicles and their environments are evolving at a fast pace. Multimodal mobility will connect previously separated domains like cars and public transportation. Mobility-oriented services such as car sharing creates completely new eco-systems and business models far away from the classic “buy your own car” approach. Autonomous driving demands highly interactive services with multi-



**Fig. 1** Software and IT advance automotive innovations and complexity

sensor fusion, far away from currently deployed functionally isolated control units. Connectivity and infotainment have transformed the car into a distributed IT system with cloud access, over-the-air functional upgrades, and high-band-width access to map services, media content, other vehicles and surrounding infrastructure. Energy efficiency evolves the classic powertrain towards high voltage hybrid and electric engines.

With so many software-driven innovations in vehicles and their environments, complexity is growing fast—and in some cases beyond what can be controlled as recent security attacks have showed. Figure 1 indicates the rapid growth of software-driven innovations along with a forecast for the near future.

To master this fast growing complexity, automotive software needs a clear architecture. Architecture evolution today is the major focus across companies, and thus the book arrives just at the right time. Architecture impacts are manifold, such as systems modeling, testing and simulation with models in the loop; the combination of several quality requirements such as safety; service-oriented advanced operating systems with secure communication platforms such as adaptive AUTOSAR (Automotive Open System Architecture); multisensor fusion and picture recognition for ADAS (Advanced Driver Assistance Systems) and autonomous driving; distributed end-to-end security for flexible remote software updates directly into the cars' firmware; connectivity of cloud technologies and IT backbones with billions of cars and their on-board devices for infotainment, online apps, remote diagnosis and emergency call processing.

This book comprehensively introduces to automotive software architecture. Authored by renowned expert Mirosław Staron it provides a guided tour through

the methodology and usage of automotive software architecture. Starting with a brief introduction to software architecture paradigms it quickly moves to current application domains, such as AUTOSAR. Architecture analysis with methods such as ATAM of SEI provide hands-on guidance specifically at the current fast paradigm change from classic networking controllers to the three-tier model of future automotive IT.

With this book Miroslaw Staron and his co-authors target both engineers and decision-makers in the automotive electronics and IT domain. They guide engineers, developers and managers along the convergence of the two worlds of IT and embedded systems. Education however has only in rare cases dedicated programs for engineering this convergence of IT and embedded systems. Business models will evolve towards flexible service-oriented architectures and eco-systems. Reference points based on industry standards such as three-tier cloud architectures, adaptive AUTOSAR, and Ethernet connectivity facilitate reuse across companies and industries. The classic functional split is replaced by a more service-oriented architecture and delivery model. Development in the future will be a continuous process which will fully decouple the rather stable hardware of the car from its functionality driven by software upgrades. Hierarchic modeling of business processes, functionality and architecture from a systems perspective allow early simulation while ensuring robustness and security. Agile service delivery models combining DevOps, micro-services and cloud solutions will allow functional changes far beyond the traditional V approach.

The techniques presented in this book are not supposed to be the ultimate truth, but provide direction in a fast evolving field. It will help you as well as your organization to grow your maturity. Our society and each of us depend on seamless mobility, and so we need to also trust these underlying systems of infrastructure and vehicles. Let's evolve the necessary technology, methods, and competences in a good direction to stay in control of automotive software and avoid the many pitfalls of classic IT systems. For this reason I wish this book all the best and good success.

As with all architecture independent of application domain, we should not forget the wisdom of another great leader, Winston Churchill, who once said: "However beautiful the strategy, you should occasionally look at the results."

Stuttgart, Germany  
February 2017

Christof Ebert

# Preface

Even since I've learned how to drive, I've been an enthusiast of cars and driving. The ability to “go places” and be in charge of the machine that can take us to these places has always been something that I've loved. When I entered the field of computer science and software engineering, the software was present in cars in very few places—basically only to control the ignition of the engine. At least that was the case in the cars I owned at the time. However, I saw a large potential for using computers in cars.

It is the ability to use more software in cars that triggered my interest in automotive software architectures. In 2015 my publisher contacted me and proposed writing a book about topics I like. I managed to convince my colleagues—Darko Durisic from Volvo Car Group, Per Johannessen from AB Volvo and Wilhelm Meding from Ericsson—to help in writing some of the chapters.

In 2017 we managed to finish the book and we hope that it will provide a solid ground for our readers in designing automotive software. We hope that by writing this book we can contribute to a more exciting, yet at the same time safer, cars. We have enjoyed writing the book and we hope that you, our reader, will enjoy reading our book.

The purpose of the book is to introduce the concept of software architecture as one of the cornerstones of software in modern cars. The book is a result of my work in the area of software engineering, with particular focus on safety systems and software measurement. Throughout my research, I've worked with multiple companies in the automotive and telecom domains and I have noticed that over time these domains became increasingly similar. The processes and tools for developing software in modern cars became very similar to those used in the development of telecommunication systems. The same is very true about software architectures—initially very different, today they are increasingly similar in terms of architectural styles, programming paradigms and architectural patterns.

The book starts with a historical overview of the evolution of software in modern cars and the description of the main challenges which drive the evolution. Chapter 2 describes the main architectural styles of automotive software and their use in car's software. In Chap. 3, the reader can find a description of software development



processes used to develop software on the car manufacturer's side. Chapter 4 introduces AUTOSAR—an important standard in automotive software. Chapter 5 goes beyond simple architecture and describes the process of detailed design of automotive software with the use of Simulink, which helps us understand how the detailed design links to the high-level design. Chapter 6 presents a method for assessing the quality of the architecture—ATAM (Architecture Trade-off Analysis Method)—and provides an example assessment. Chapter 7 presents an alternative way of assessing the architecture, namely by using quantitative measures and indicators. In Chap. 8 we dive deeper into one of the specific properties discussed in Chap. 6—safety—and can read about the important standard in that area—ISO/IEC 26262. Finally, Chap. 9 presents a set of future trends that seem to emerge today that have the potential to shape automotive software engineering in the coming years.

Gothenburg, Sweden  
January 2017

Mirosław Staron

# Acknowledgements

First and foremost, I would like to thank the co-authors of some of the chapters in this book—Darko Durisic, Per Johannessen and Wilhelm Meding. I have had the privilege of working with them for a number of years and I'm deeply thankful for their insights into the car and telecom industries.

I am greatly indebted to my family—Sylwia, Alexander, Viktoria and Cornelia—who support me in taking on challenges and see to it that I am successful. They are the most fantastic family one could imagine.

I would also like to thank my publisher—Ralf Gerstner from Springer—who has proposed the idea of the book and helped me throughout the process. Without his encouragement and practical pointers this book would have never happened.

Many thanks to dSpace GmbH, for permitting me to use images of their equipment as part of the book. I also thank Jan Söderberg from Systemite for providing me with the figures and explanations on how the SystemWeaver tool keeps the different construction artifacts together.

I am grateful to my colleagues from Volvo Car Group who have taught me about practicalities of the automotive industry. I have met many persons from the fantastic team of Volvo Cars and had many great discussions about how cars are designed today, but in particular I am indebted to Kent Niesel, Martin Nilsson, Niklas Baumann, Anders Svensson, Hans Alminger, Ilker Dogan, Lars Rosqvist, Sajed Miremari, Mikael Sjöstrand and Peter Dahlsund. I would also like to thank Mark Hirche and Malin Folke for their comments on the draft of the book.

I would also like to thank my colleagues from the research community for their help and support in both writing this book and in my research activities leading to this book. In particular I would like to thank Imed Hammouda for his feedback and comments on the ATAM evaluation chapter.

And finally I would like to thank the Swedish Innovation Agency Vinnova, the Swedish Strategic Research Foundation SSF and the Software Center for providing me with research funding that allowed me to pursue my research interests in the area of this book.

# Contents

<b>1</b>	<b>Introduction .....</b>	<b>1</b>
1.1	Software and Modern Cars .....	1
1.2	History of Software in the Automotive Industry .....	2
1.3	Trends Shaping Automotive Software Development.....	5
1.4	Organization of Automotive Software Systems .....	8
1.5	Architecting as a Discipline .....	9
	1.5.1 Architecting vs. Project Management .....	10
	1.5.2 Architecting vs. Design .....	11
1.6	Content of This Book .....	12
	1.6.1 Chapter 2: Software Architectures .....	13
	1.6.2 Chapter 3: Automotive Software Development.....	13
	1.6.3 Chapter 4: AUTOSAR Reference Model .....	14
	1.6.4 Chapter 5: Detailed Design of Automotive Software....	14
	1.6.5 Chapter 6: Evaluation of Automotive Software Architectures .....	14
	1.6.6 Chapter 7: Metrics for Software Design and Architectures.....	15
	1.6.7 Chapter 8: Functional Safety of Automotive Software .....	15
	1.6.8 Chapter 9: Current Trends in Automotive Software Development .....	15
	1.6.9 Motivating Examples in the Book .....	15
1.7	Knowledge Prerequisites .....	16
1.8	Where to Go Next.....	16
	References.....	17
<b>2</b>	<b>Software Architectures: Views and Documentation .....</b>	<b>19</b>
2.1	Introduction .....	19
2.2	Common View on Architecture in General and in the Automotive Industry in Particular.....	20
2.3	Definitions.....	22

2.4	High-Level Structures .....	23
2.5	Architectural Principles .....	24
2.6	Architecture in the Development Process.....	25
2.7	Architectural Views .....	26
2.7.1	Functional View .....	26
2.7.2	Physical System View .....	28
2.7.3	Logical View .....	29
2.7.4	Relation to the 4+1 View Model.....	31
2.8	Architectural Styles .....	32
2.8.1	Layered Architecture.....	32
2.8.2	Component-Based.....	34
2.8.3	Monolithic .....	36
2.8.4	Microkernel.....	36
2.8.5	Pipes and Filters .....	38
2.8.6	Client–Server .....	38
2.8.7	Publisher–Subscriber.....	39
2.8.8	Event-Driven .....	40
2.8.9	Middleware .....	40
2.8.10	Service-Oriented .....	42
2.9	Describing the Architectures .....	43
2.9.1	SysML .....	43
2.9.2	EAST ADL .....	45
2.10	Next Steps .....	47
2.11	Further Reading .....	47
2.12	Summary .....	47
	References.....	48
<b>3</b>	<b>Automotive Software Development .....</b>	<b>51</b>
3.1	Introduction .....	51
3.1.1	V-Model of Automotive Software Development .....	52
3.2	Requirements.....	53
3.2.1	Types of Requirements in Automotive Software Development .....	55
3.3	Variant Management.....	59
3.3.1	Configuration .....	60
3.3.2	Compilation .....	60
3.3.3	Practical Variability Management .....	61
3.4	Integration Stages of Software Development .....	62
3.5	Testing Strategies .....	63
3.5.1	Unit Testing.....	63
3.5.2	Component Testing.....	65
3.5.3	System Testing .....	66
3.5.4	Functional Testing.....	68
3.5.5	Pragmatics of Testing Large Software Systems: Iterative Testing .....	69

3.6	Construction Database and Its Role in Automotive Software Engineering .....	69
3.7	Further Reading .....	74
3.7.1	Requirements Specification Languages .....	76
3.8	Summary .....	76
	References.....	77
<b>4</b>	<b>AUTOSAR Standard .....</b>	<b>81</b>
4.1	Introduction .....	81
4.2	AUTOSAR Reference Architecture .....	83
4.3	AUTOSAR Development Methodology .....	85
4.4	AUTOSAR Meta-Model.....	90
4.4.1	AUTOSAR Meta-Modeling Environment .....	91
4.4.2	Architectural Design Based on the AUTOSAR Meta-Model .....	93
4.4.3	AUTOSAR Template Specifications .....	98
4.5	AUTOSAR ECU Middleware .....	99
4.6	AUTOSAR Evolution.....	101
4.6.1	AUTOSAR Meta-Model Evolution .....	102
4.6.2	AUTOSAR Requirements Evolution .....	107
4.7	Future of AUTOSAR .....	109
4.8	Further Reading .....	113
4.9	Summary .....	115
	References.....	115
<b>5</b>	<b>Detailed Design of Automotive Software .....</b>	<b>117</b>
5.1	Introduction .....	117
5.2	Simulink Modelling.....	118
5.2.1	Basics of Simulink .....	119
5.2.2	Sample Model of Digitalization of a Signal .....	123
5.2.3	Translating Physical Processes to Simulink.....	126
5.2.4	Sample Model of Car's Interior Heater .....	129
5.3	Simulink Compared to SysML/UML .....	136
5.4	Principles of Programming of Embedded Safety-Critical Systems.....	137
5.5	MISRA .....	138
5.6	NASA's Ten Principles of Safety-Critical Code .....	140
5.7	Detailed Design of Non-safety-Critical Functionality .....	141
5.7.1	Infotainment Applications .....	142
5.8	Quality Assurance of Safety-Critical Software .....	143
5.8.1	Formal Methods .....	144
5.8.2	Static Analysis.....	144
5.8.3	Testing .....	146
5.9	Further Reading .....	146
5.10	Summary .....	148
	References.....	148

<b>6</b>	<b>Evaluation of Automotive Software Architectures .....</b>	<b>151</b>
6.1	Introduction .....	151
6.2	ISO/IEC 25000 Quality Properties .....	152
6.2.1	Reliability .....	152
6.2.2	Fault Tolerance .....	155
6.2.3	Mechanisms to Achieve Reliability and Fault Tolerance .....	155
6.3	Architecture Evaluation Methods .....	157
6.4	ATAM .....	158
6.4.1	Steps of ATAM .....	159
6.4.2	Scenarios Used in ATAM in Automotive .....	160
6.4.3	Templates Used in the ATAM Evaluation .....	162
6.5	Example of Applying ATAM .....	164
6.5.1	Presentation of Business Drivers .....	165
6.5.2	Presentation of the Architecture .....	165
6.5.3	Identification of Architectural Approaches .....	168
6.5.4	Generation of Quality Attribute Tree and Scenario Identification .....	168
6.5.5	Analysis of the Architecture and the Architectural Decision .....	173
6.5.6	Summary of the Example .....	174
6.6	Further Reading .....	175
6.7	Summary .....	175
	References .....	176
<b>7</b>	<b>Metrics for Software Design and Architectures .....</b>	<b>179</b>
7.1	Introduction .....	179
7.2	Measurement Standard in Software Engineering: ISO/IEC 15939 .....	180
7.3	Measures Available in ISO/IEC 25000 .....	184
7.4	Measures .....	184
7.5	Metrics Portfolio for the Architects .....	186
7.5.1	Areas .....	187
7.5.2	Area: Architecture Measures .....	187
7.5.3	Area: Design Stability .....	188
7.5.4	Area: Technical Debt/Risk .....	188
7.6	Industrial Measurement Data for Software Designs .....	192
7.7	Further Reading .....	193
7.8	Summary .....	195
	References .....	196
<b>8</b>	<b>Functional Safety of Automotive Software .....</b>	<b>201</b>
8.1	Introduction .....	201
8.2	Management and Support for Functional Safety .....	203
8.3	Concept and System Development .....	204
8.4	Planning of Software Development .....	208

8.5	Software Safety Requirements .....	209
8.6	Software Architectural Design .....	210
8.7	Software Unit Design and Implementation .....	212
8.8	Software Unit Testing .....	214
8.9	Software Integration and Testing .....	216
8.10	Verification of Software Safety Requirements.....	217
8.11	Examples of Software Design .....	218
8.12	Integration, Testing, Validation, Assessment and Release .....	219
8.13	Production and Operation .....	219
8.14	Further Reading .....	220
8.15	Summary .....	220
	References .....	221
<b>9</b>	<b>Current Trends in Automotive Software Architectures .....</b>	<b>223</b>
9.1	Introduction .....	223
9.2	Autonomous Driving .....	224
9.3	Self-* .....	225
9.4	Big Data .....	226
9.5	New Software Development Paradigms .....	227
	9.5.1 Architecting in the Age of Agile Software Development .....	228
9.6	Other Trends .....	229
9.7	Summary .....	230
	References .....	231
<b>10</b>	<b>Summary .....</b>	<b>233</b>
10.1	Software Architectures in General and in the Automotive Software: A Short Recap .....	233
10.2	Chapter 2: Software Architectures .....	233
10.3	Chapter 3: Automotive Software Engineering.....	234
10.4	Chapter 4: AUTOSAR.....	235
10.5	Chapter 5: Detailed Design of Automotive Software .....	235
10.6	Chapter 6: Evaluation of Automotive Software Architectures....	236
10.7	Chapter 7: Metrics for Software Designs and Architectures.....	236
10.8	Chapter 8: Functional Safety of Automotive Software.....	236
10.9	Chapter 9: Current Trends .....	237
10.10	Closing Remarks .....	237

<http://www.springer.com/978-3-319-58609-0>

Automotive Software Architectures

An Introduction

Staron, M.

2017, XIX, 237 p. 150 illus., 107 illus. in color.,

Hardcover

ISBN: 978-3-319-58609-0