The Fraunhofer IESE Series on Software and Systems Engineering

Jens Knodel
Matthias Naab

Pragmatic Evaluation of Software Architectures

Fraunhofer IESE

Springer

The Fraunhofer IESE Series on Software and Systems Engineering

More information about this series at http://www.springer.com/series/8755

Jens Knodel · Matthias Naab

Pragmatic Evaluation of Software Architectures

Springer

Jens Knodel
Fraunhofer IESE
Kaiserslautern
Germany

Matthias Naab
Fraunhofer IESE
Kaiserslautern
Germany

ISSN 2193-8199 ISSN 2193-8202 (electronic)
The Fraunhofer IESE Series on Software and Systems Engineering
ISBN 978-3-319-34176-7 ISBN 978-3-319-34177-4 (eBook)
DOI 10.1007/978-3-319-34177-4

Library of Congress Control Number: 2016939924

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG Switzerland

About this Series

Whereas software engineering has been a growing area in the field of computer science for many years, systems engineering has its roots in traditional engineering. On the one hand, we still see many challenges in both disciplines. On the other hand, we can observe a trend to build systems that combine software, microelectronic components, and mechanical parts. The integration of information systems and embedded systems leads to so-called smart ecosystems.

Software and systems engineering comprise many aspects and views. From a technical standpoint, they are concerned with individual techniques, methods, and tools, as well as with integrated development processes, architectural issues, quality management and improvement, and certification. In addition, they are also concerned with organizational, business, and human views. Software and systems engineering treat development activities as steps in a continuous evolution over time and space.

Software and systems are developed by humans, so the effects of applying techniques, methods, and tools cannot be determined independent of context. A thorough understanding of their effects in different organizational and technical contexts is essential if these effects are to be predictable and repeatable under varying conditions. Such process–product effects are best determined empirically. Empirical engineering develops the basic methodology for conducting empirical studies and uses it to advance the understanding for the effects of various engineering approaches.

The series presents engineering-style methods and techniques that foster the development of systems that are reliable in every aspect. All the books in the series emphasize the quick delivery of state-of-the-art results and empirical proof from academic research to industrial practitioners and students. Their presentation style is designed to enable the reader to quickly grasp both the essentials of a methodology and how to apply it successfully.

Foreword

Many activities in software engineering benefit from explicit documentation of the software architecture. Such activities range from cost estimation via make-or-buy decisions to implementing software and deployment, or even the establishment of a software product line or other forms of planned reuse.

However, in many projects, the documentation of software architectures is considered as a burden and is thus avoided.

I think one of the major reasons for this reluctance to invest in software architecture documentation is the absence of feedback: Do I have a good architectural design? Is the architecture documented well enough for my project? As a result, developers are unsure whether the architecture is useful altogether. In fact, the problem of the lack of feedback in architectural design exists regardless of the documentation approach. When attempting to document the architecture, it just becomes more obvious. Anyway, the effects of bad architectural design are also well known: shift of risks to the later phases of software development, lowered productivity, and, most often, unsatisfactory quality of products. The latter, in particular, is not very surprising, as the software architecture is for many IT systems the major factor influencing the perceived quality for customers and developers alike.

Therefore, the role of early evaluation of software architectures is well understood in research. In fact, there exists a large body of proposals on how to evaluate software architectures for various concerns. Most approaches are informal, but automated formal approaches also exist. The benefits of informal approaches are a certain flexibility regarding acceptable types of architectural documentation and a broad range of potential quality impacts of the architecture. However, they require manual effort and depend to a considerable extent on the experience of the evaluation team. In contrast, formalized automated approaches require specific architectural models, but the evaluation is "objective," as it is automated. However, only specific quality concerns are answered.

This book addresses the very important field of software architecture evaluations. It is important for two reasons: First, it provides practical means for evaluating software architectures, which is relevant to the reasons given above. Second, it bundles experiences and how-to guidelines for the manual approaches, which particularly benefit from exactly such experiences and direct guidelines. This book

is an invaluable help in bundling the existing body of knowledge and enriching it strongly with real-world project expertise of the authors. It is not only a major step toward the practical applicability of software architecture evaluation but also helps to find good designs right at the beginning, as it helps to avoid design errors in all phases of a software system's lifecycle.

I wish the readers many insights from the abundant experience of the authors and a lot of fun when applying the knowledge gained.

Prof. Dr. Ralf H. Reussner
Chair Software Design and Quality, Karlsruhe Institute of Technology (KIT) and
Executive Director, Forschungszentrum Informatik (FZI), Karlsruhe, Germany

Preface

What is the Point of This Book?

Software architecture evaluation is a powerful means to mitigate risks when making decisions about software systems, when constructing them, when changing them, when considering (potential) improvements, or when reasoning about ways to migrate. Architecture evaluation is beneficial in such cases either by predicting properties of software systems before they are built or by answering questions about existing systems. Architecture evaluation is both effective and efficient: effective, as it is based on abstractions of the system under evaluation, and efficient, as it can always focus only on those facts that are relevant for answering the evaluation questions at hand.

Acknowledging the need for architecture evaluation does not necessarily mean that it has been adopted by practitioners. Although research has disseminated numerous publications about architecture evaluation, a pragmatic and practical guide on how to apply it in one's own context and benefit from it is still missing. With this book we want to share our lessons learned from evaluating software architectures. We do not aim at reinventing the wheel; rather, we present a consolidation of useful ideas from research and practice, adapting them in such a way as to make them applicable efficiently and effectively—in short, we take a pragmatic approach to evaluating software architectures. Where necessary, we will fill in gaps in existing approaches, in particular in the areas of scalability and applicability. Additionally, we aim at interrelating all aspects and techniques of architecture evaluation and creating an understandable and memorable overall picture. We will refer to examples of real architecture evaluation cases from our industrial practice (anonymized due to confidentiality reasons) and provide data on projects.

Why Read This Book?

> "The most serious mistakes are not being made as a result of wrong answers. The truly dangerous thing is asking the wrong question."

Peter Ferdinand Drucker

We think that thorough and continuous architecting is the key to overall success in software engineering, and architecture evaluation is a crucial part of architecting. Asking the right questions and knowing about the right techniques to answer is crucial for applying architecture evaluations as a valuable and pragmatic means of technical risk management in software engineering.

To date (i.e., as of February 2016), we have conducted more than 75 architecture evaluation projects with industrial customers in the past decade. In each of these projects, at least one of the authors has been directly or indirectly involved as part of our jobs as architects and consultants at the Fraunhofer Institute for Experimental Software Engineering IESE. Fraunhofer IESE is an applied research institute for software engineering located in Kaiserslautern, Germany. These projects covered a large number of different types of systems, industries, organizational constellations, technologies, modeling and programming languages, context factors, and, of course, a whole spectrum of different evaluation results. Most importantly, we collected a lot of evaluation questions that were asked and operationalized them into actions.

> "You can't control what you can't measure."

Tom DeMarco

> "Everything that can be counted does not necessarily count; everything that counts cannot necessarily be counted."

Albert Einstein

While scientific literature on architecture evaluation approaches is available, the number of publications on practical experiences is rather limited. The contribution of this book consists of the presentation and packaging of our experiences together with context factors, empirical data, example cases, and lessons learned on mitigating the risk of change through architecture evaluation. Our approach for architecture evaluation (called RATE Rapid ArchiTecture Evaluation) has evolved and been proven successful in many projects over the past decade. We will provide an

in-depth description on the ingredients of our approach, but will also tackle the field of architecture evaluation as a whole, as many of our insights and findings are independent of the approach.

After reading this book, the target audiences will be able to take their own steps in evaluating software architecture. By giving comprehensive answers to more than 100 typical questions (including questions we had, questions we heard, and questions our industrial partners had) and discussing more than 60 frequent mistakes and lessons learned, readers will take their first steps on ground paved by more than a decade of the authors' experiences.

Even more importantly, readers will learn how to interpret the results of an architecture evaluation. They will become aware of risks such as false conclusions, fiddling with data, and wrong lines of arguments in evaluations. Last but not least, readers will become confident in assessing quantitative measurement results and will learn when it is better to rely on qualitative expertise. In short, it is important to be aware what counts in architecture evaluation.

The target audience for the experience shared with this book includes both practitioners and researchers. On the one hand, we aim at encouraging practitioners to conduct architecture evaluations by showing the impact and lowering the hurdles to making first attempts on their own by providing clear guidelines, data, and examples. On the other hand, we aim at giving researchers insights into industrial architecture evaluations, which can serve as basis for guiding research in this area and may inspire future research directions.

How Should I Read This Book?

Our book is structured into three parts explaining the background of architecture evaluation, describing concrete evaluation techniques, and offering hints on how to successfully start and institutionalize architecture evaluation in practice.

Part I What is the Point of Architecture Evaluation?
Part II How to Evaluate Architectures Effectively and Efficiently?
Part III How to Apply Architecture Evaluation in Practice?

- For an **executive summary** on one page, please jump directly to question Q.117 on page 148.
- For **architects**, **developers**, or as learning material for **aspiring evaluators**, all three parts are highly relevant.
- For **managers**, mainly Part I and Part III are relevant.
- For a **quick start into an evaluation**, we recommend starting with question Q.117, reading Chaps. 3, 4, and 11, then proceeding directly to the respective checks you want to conduct in Chaps. 5–9.

In order to serve practitioners' needs in the best possible way, the entire book is structured along questions. These questions are organized in a hierarchical and uniform way to cover the big picture of architecture evaluation. For every topic, we also present frequently made mistakes we often encountered in practice and give hints on how to avoid them. Lists containing all questions, all frequently made mistakes, and the lessons learned serve to offer quick guidance for the reader.

In the following, we depict the recurring patterns that guide readers through the book. To a large extent, the chapters follow a uniform structure and are organized internally along questions. The questions are numbered consecutively. Frequent mistakes and lessons learned are visually highlighted, as shown in the following examples.

1.1 What is the point?
→ *This section summarizes the key points of the chapter's topic.*
1.2 How to do it effectively and efficiently?
→ *Here we present detailed descriptions and guidance.*
1.3 What mistakes are made frequently in practice?
→ *This section names typical pitfalls and points out how to avoid them.*

Q.001. Question

Frequently made mistake

→ Question Q.00X (*please read this question for more background information*)

Lesson learned

Acknowledgments

We would like to express our gratitude to our families, our colleagues (past and present), our customers, and the software architecture community.

We thank our families for their continuous support and encouragement in writing this book.

We had the opportunity to work and collaborate with great people at Fraunhofer IESE, where architecture evaluation has been a field of expertise for far more than 10 years now. We would like to thank our institute's directors Peter Liggesmeyer and Dieter Rombach for giving architecture evaluation a high degree of awareness and visibility. We also thank our past and present colleagues at Fraunhofer IESE for collaborating with us in software architecture projects and helping us to evolve our architecture evaluation method. The architecture evaluation projects making up this book would not have been possible without you.

We would like to thank Stefan Georg, Jens Heidrich, Thorsten Keuler, Torsten Lenhart, Mikael Lindvall, Dirk Muthig, Stefan Toth, Marcus Trapp, Carl Worms, and Stefan Zörner for their reviews and valuable feedback on earlier versions of this book. We are especially happy about the foreword by Ralf Reussner. Our special thanks go to Sonnhild Namingha for her excellent proofreading.

More than 75 customer projects were the key enabler of this book—as writing for practitioners is only possible when collaborating with practitioners. The collaboration with companies worldwide gave us lots of practical experience. Although we are not allowed to name them, we would like to thank our customers for their trust and the insights we were able to learn.

Over the years, we had many inspiring discussions with practitioners and researchers all over the world about software architecture in general and architecture evaluation in particular. We would like to thank them for stimulating ideas, shared insights, and their feedback to our ideas.

Finally, we would like to thank our publisher Springer and in particular Ralf Gerstner for his continuous support and his patience on the long journey of getting this book done—at last.

Contents

Table of Questions