

Preface

Imagine this happens to you: Your manager tells you “Agile is the future! Let’s go Scrum.” He *forces* you to replace the existing development process with Scrum. What would you do? Would you send your developers to a Scrum training course immediately?

It is true that more companies are embracing agile as part of their development process in order to increase speed, accelerate learning, and deliver value rapidly. And many of these companies are applying Scrum. But it is also true that evolution does not follow the principle: “Progressive dinosaurs are the future! Let’s go bird.”¹

Evolving the ways through which software-intensive products and services are developed is a challenging endeavor that needs to be done carefully. Where do you start? What do you have to consider?

This book will help you better understand the different aspects and challenges of evolving development processes. It addresses difficult problems, such as how to implement processes in highly regulated domains or where to find a suitable notation for documenting processes. This book emphasizes the need to consider *Software Process Evolution* as an important means for catching up with rapid changes in technical and market environments. It provides insights that might help you manage process evolution. It gives plenty of tips, e.g., how to cope with the threat of disruption from a process perspective. In addition, it provides many examples and cases on how to deal with software evolution in practice.

Why a Book on Managing Process Evolution?

Many organizations need to transform their business to the next level. In order to benefit from leading-edge technologies, catch up with the digital transformation, and continuously innovate and renew business models, companies have to quickly

¹Quote taken from a tweet from David Evans.

adapt and change the ways they develop products and services. As software is the key to this transformation, the ways in which modern software is developed need to change accordingly.

Another important driver for process evolution is the need to mitigate software risks. Basically, a considerable share of software risk is process-based [3]. For example, there have been several incidents which could have been avoided with appropriate coding standards and tools. Although these standards and tools are widely available, they are either not applied or not appropriately applied in many situations. Because this is normally caused by the way work and people are organized and work is carried out it is a software process issue. Companies need to find ways to ensure that process models are properly defined and, furthermore, are appropriately applied while not hindering the creativity of, e.g., designers or developers. To do this effectively, defining and deploying adequate software processes usually requires fostering the evolution of existing processes and their underlying models towards ones that suit better.

Today, there exists a variety of software processes ranging from generic and domain-specific standards, from agile methods to comprehensive process engineering frameworks. Since software processes may contain up to hundreds or even thousands of elements, the management of a software process is a demanding task and, therefore, many companies install whole departments dealing with software process improvement and management. In practice, especially in large organizations, we can observe some interesting gaps:

- Development teams tend to apply agile methods while the hosting organization focuses on “classic” structured development processes [5, 6].
- Implemented development processes in projects differ from what has been defined [4].
- Evolving software technologies and platforms require a parallel evolution of software processes to accommodate the rapid changes. However, this co-evolution does not appropriately take place.

One main reason for these gaps is different mindsets. For instance, program managers and quality assurance people need planned and directed processes for certification, budgeting, and compliance business. Developers need flexibility and processes which support creative work. Business managers need processes that allow for fast results and flexible feature delivery. Moreover, due to technology evolution, business evolves. This requires that emerging markets must be addressed, new technologies should be adopted, and globally distributed development becomes more and more important.

Apart from the big “global players,” process evolution is also highly important for small and medium-sized companies. Such companies typically neither have comprehensive process models nor process engineering groups, and often have to trust in a common understanding of principles and applied practices. However, these principles and practices need to be continuously validated against higher level goals (such as business strategies) and potentially changed in order to secure and maintain the company’s position in the market place [2]. One example for such a

change is the increasing focus on value-delivery [1]. Regardless of the company size, a major challenge that companies face is to provide all stakeholders with flexible processes that:

- Are driven by the needs of the different stakeholders,
- Have clear links to higher level goals of an organization,
- Provide interfaces that are compatible with organizational structures,
- Are supported by tools for modeling, enactment, analyses, and evolution,
- Can be tailored to individual project goals and characteristics,
- Offer adaptability and elasticity to accommodate and support technological and organizational innovations and evolutions.

This book focuses on the design, development, management, governance, and application of evolving software processes that are aligned with changing business objectives, such as expansion to new domains or moving to global production. In the context of evolving business, it addresses the complete software process life-cycle, from initial definition of a product to systematic improvement.

Who Should Read This Book?

This book is aimed at anyone interested in understanding and organizing software development tasks in an organization. The experiences and ideas in this book are useful for both those who are unfamiliar with software process improvement and want to get an overview of the different aspects of the topic, and those experts with many years of experience. In particular, the present book addresses researchers and Ph.D. students in the area of Software & Systems Engineering and Information Systems, who study advanced topics of organizing and managing (software development) projects and process improvements projects. Furthermore, the book addresses practitioners, consultants, and coaches involved in software-related change management and software process improvement projects, and who want to learn about challenges and state-of-the-art techniques and experiences regarding their application to problems in different application domains.

How is the Book Organized?

This book is organized in three parts (Fig. 1). Part 1 focuses on software business transformation, its challenges, and addresses the questions about which process(es) to use and adapt, and how to organize process improvement programs. In Chap. 1, Tony Wasserman discusses short lifecycle projects and how “low-ceremony processes” help shorting project iterations. In this context, in Chap. 2, Diebold and Zehler discuss the “right” degree of agility in rich software processes—how to find and how to achieve this. The challenge of implementing agile software

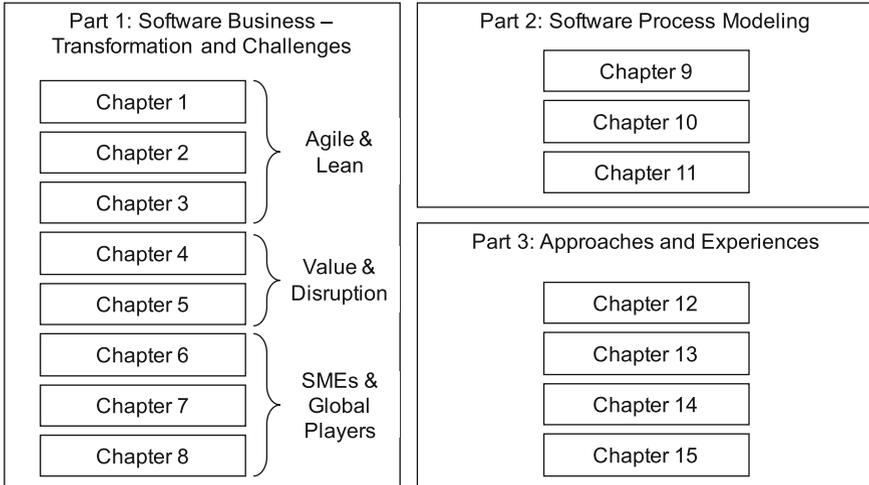


Fig. 1 Overview of the book and chapter outline

development approaches is further discussed by Houston and Rosemergy in Chap. 3, who report an agile transformation of a globally distributed company. As many companies jump to Agile processes hoping for the benefits promised, determining value and value creation is crucial. In Chap. 4, Christof Ebert discusses the principles of value-driven process management and reports experiences. Another perspective is taken by Andreas Rösel, who describes how concepts of design thinking can be applied to disruptive improvements in Chap. 5. Oisín Cawley discusses the trials and tribulations of Global Software Engineering processes in the course of business evolution with a particular focus on regulated software and system development in Chap. 6. The respective Software Process Improvement challenges, approaches, and standards for very small entities and small- to medium-sized companies are presented in Chap. 7 by Mary-Luz Sánchez-Gordón and her colleagues. In their systematic literature review, they give a comprehensive overview of the different improvement approaches and models and show how they find their way into international standards. Standards and their role are also key to the Space business, as presented in Chap. 8, where Christian Prause and his colleagues describe how software processes in the German Space Administration evolve and how they are tailored to the projects.

Part 2 of the book is focused on process modeling. This part starts with Chap. 9 by Dumas and Pfahl, who discuss the appropriateness of the Business Process Model and Notation (BPMN) for software processes modeling. In Chap. 10, Fazal-Baqaie and Engels present an approach to modeling evolving software processes by utilizing method engineering principles. The adaptation of case management techniques for the purpose of improving process model flexibility is demonstrated by Marian Benner-Wickner as his colleagues in Chap. 11.

Finally, Part 3 of the book collects approaches, experiences, and recommendations that help to improve software processes with a particular focus on specific lifecycle phases. The part starts with Chap. 12 in which Kai Petersen reports his experiences in industrial Software Process Improvement projects from the perspective of a researcher. He reports from projects and provides a collection of general lessons learned and recommendations to aid researchers and practitioners to plan and carry out improvement projects in an industry–academia collaboration. Chapter 13 in which Regina Hebig and her colleagues give insights into two large-scale industry projects and demonstrate how co-evolution is manifested and handled in such projects, thus addressing the co-evolution of software processes and model-driven engineering approaches. In Chap. 14, S.M. Didar Al Alam and his colleagues present an approach that helps companies to improve the release readiness of their software products. They show how bottleneck factors that hinder fast releases can be detected and they apply their concept to different Open-Source Software projects. Finally, Jesse Yli-Huumo and colleagues take a broader perspective in Chap. 15, discussing how process evolution affects technical debt. They illustrate their findings with three large-scale software projects.

We wish you an interesting and enjoyable reading experience. A collection such as this book would not be possible without the help of many persons. We would especially like to thank the authors for their insightful articles and their excellent collaboration. In addition, we would like to thank Ralf Gerstner from Springer, who supported us efficiently in completing organizational and contract issues.

Odense, Denmark
 Reutlingen, Germany
 Limerick, Ireland
 Clausthal-Zellerfeld, Germany
 Nanjing, China
 January 2016

Marco Kuhrmann
 Jürgen Münch
 Ita Richardson
 Andreas Rausch
 He Zhang

References

1. Bosch, J.: Speed, data, and ecosystems: The future of software engineering. *IEEE Softw.* **33**(1), 82–88 (2016)
2. Fagerholm, F., Guinea, A.S., Mäenpää, H., Münch, J.: Building blocks for continuous experimentation. In: *Proceedings of the International Workshop on Rapid Continuous Software Engineering, RCoSE*, pp. 26–35. ACM, New York (2014)
3. Neumann, P.G., et al.: Column: Risks to the Public. *ACM SIGSOFT Softw. Eng. Note* **40**(6), 14–19 (2015)
4. Parnas, D.L., Clements, P.C.: A Rational Design Process: How and Why to fake it. *IEEE Trans. Software Eng.* **12**(2), 251–257 (1986)

5. Theocharis, G., Kuhmann, M., Münch, J., Diebold, P.: Is Water-Scrum-Fall reality? On the use of agile and traditional development practices. In: Proceedings of the International Conference on Product-Focused Software Process Improvement. Lecture Notes in Computer Science, vol. 9459, pp. 149–166. Springer, Heidelberg (2015)
6. Vijayarathy, L., Butler, C.: Choice of software development methodologies - do project, team and organizational characteristics matter? *IEEE Software* (99), 1ff. (2015)



<http://www.springer.com/978-3-319-31543-0>

Managing Software Process Evolution
Traditional, Agile and Beyond - How to Handle Process
Change

Kuhrmann, M.; Münch, J.; Richardson, I.; Rausch, A.;
Zhang, H. (Eds.)

2016, XXVII, 332 p. 73 illus., 7 illus. in color., Hardcover
ISBN: 978-3-319-31543-0