Smart Innovation, Systems and Technologies

Volume 175

The Smart Innovation, Systems and Technologies book series encompasses the topics of knowledge, intelligence, innovation and sustainability. The aim of the series is to make available a platform for the publication of books on all aspects of single and multi-disciplinary research on these themes in order to make the latest results available in a readily-accessible form. Volumes on interdisciplinary research combining two or more of these areas is particularly sought.

The series covers systems and paradigms that employ knowledge and intelligence in a broad sense. Its scope is systems having embedded knowledge and intelligence, which may be applied to the solution of world problems in industry, the environment and the community. It also focusses on the knowledge-transfer methodologies and innovation strategies employed to make this happen effectively. The combination of intelligent systems tools and a broad range of applications introduces a need for a synergy of disciplines from science, technology, business and the humanities. The series will include conference proceedings, edited collections, monographs, handbooks, reference books, and other relevant types of book in areas of science and technology where smart systems and technologies can offer innovative solutions.

High quality content is an essential feature for all book proposals accepted for the series. It is expected that editors of all accepted volumes will ensure that contributions are subjected to an appropriate level of reviewing process and adhere to KES quality principles.

**** Indexing: The books of this series are submitted to ISI Proceedings, EI-Compendex, SCOPUS, Google Scholar and Springerlink ****

More information about this series at http://www.springer.com/series/8767

Sergey V. Zykov · Amitoj Singh

Agile Enterprise Engineering: Smart Application of Human Factors

Models, Methods, Practices, Case Studies

Springer

Sergey V. Zykov
Higher School of Economics
National Research University
Moscow, Russia

Amitoj Singh
Maharaja Ranjit Singh Punjab
Technical University
Bathinda, Punjab, India

ISSN 2190-3018 ISSN 2190-3026 (electronic)
Smart Innovation, Systems and Technologies
ISBN 978-3-030-40988-3 ISBN 978-3-030-40989-0 (eBook)
https://doi.org/10.1007/978-3-030-40989-0

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

To God, my teachers, and my family

Foreword

In their book, Prof. Zykov and Dr. Singh address human factors and provide valuable insights on harnessing these factors to foster large-scale software engineering, specifically in the areas of technology and knowledge transfer.

Surprisingly, many of present-day tech leads and project managers tend to neglect the impact of these mostly irrational and typically uncertain human-related factors on software development processes, treating these processes as a purely technical matter. This careless underestimation often results in what the authors call a "crisis," i.e., a situation when the project resources in general and human factors in particular go beyond the managers' control.

In such critical circumstances, agile methods seem an evident solution and an instant remedy. However, a straightforward or blunt application of these trendy and much discussed agile frameworks, without proper product design and disciplined development, would clearly lead to the same "crisis" trap, triggered with, perhaps, the same root cause.

Therefore, the book systematically approaches the problem of taming human factors in a "peeling an onion" way. It starts from discussing technical and rather formal models and methods, and then proceeds to more practical patterns and case studies. The latter include two environments, similar in purpose, yet clearly distinct in location and ownership: the Indian Chitkara and Russian Innopolis ecosystems. Importantly, these distinctions reveal a vast spectrum of varieties, including cultural and regional ones, and provide a deep insight on managing human factors, in order to promote large-scale project efficiency and their eventual success.

The broad focus of this concise book makes it a handy guide for software developers, while the case studies may serve as a troubleshooting reference in

project management. The authors are experts who have firsthand experience in software engineering. Their integrated expertise will definitely contribute to better managed large-scale software production in the future.

Alfredo Cuzzocrea, Ph.D.
Professor of Computer Engineering, DIA Department

Head, Big Data Engineering and Analytics Lab
University of Trieste, Trieste, Italy

Research Fellow
ICAR Institute
Italian National Research Council

Preface

The subject of this volume is agility in large-scale systems development. The term "systems" embraces not only software, but also IT-intensive businesses, i.e., enterprises. These are international, distributed ventures, which typically incorporate a number of diversified companies.

However, what is agility? An intuitive understanding would give a naive instantiation of a puma or cat. Interestingly, these creatures were worshiped and praised in many ancient cultures, including Egypt and Peru. In Egypt, the cat was associated with supernatural powers. In Peruvian culture, the puma was the patron of the living creatures (the snake was the same for the dead and the condor for the immortal).

In our case of large-scale systems, one of the hampering factors in their development is complexity. Commonly, this is represented in terms of the number of components multiplied by the number of their interconnections. The larger the system, the more is its complexity. Obviously, this problem of complexity dramatically complicates large-scale system management, as the Babylon Tower effect is very likely to happen.

One way to conquer this complexity factor is the "divide-and-conquer" strategy. However, if applied too formally or carelessly, this strategy alone may be insufficient. The reason is that complexity has a number of aspects, the two principals of these being technical and managerial. The first one is often manageable by means of classical optimization and concern separation methods. To manage the second one, the above-mentioned approaches would not suffice, as they are rather formal and, therefore, rigid.

Surprisingly for many technically oriented experts, in the enterprise systems, and even in the software ones, the managerial aspect of complexity is often predominant and is the most likely reason for system development crises. Naturally, after the disaster has happened, the managers tend to blame the technologies, as they often do not realize (and sometimes neglect or even ignore) the so-called human-related factors.

Therefore, in this critical event of crisis, which often originates from the human factors that hinder the information transfer between the client and the producer (and inside their teams, which are complex systems by themselves), agility comes into play. Informally, we can understand agility as adaptability, flexibility, and responsiveness. Following the agile way is particularly important in case of development crisis that manifests itself as process uncertainty of different origins (such as quality attributes, budget gaps, or team "storming," i.e., fluctuating project parameters or product requirements).

This agility way incorporates multiple aspects of communication (e.g., negotiation, teamwork, self-reflection, etc.), often referred to as "soft" skills as opposed to the "hard" skills required for the system development managers. Proper application of these human factor-based management techniques and practices usually improves project communication and, therefore, helps to manage (i.e., avoid or mitigate) development crises.

A recent use case of agility in business and technology is the digital transformation. This kind of crisis has been triggered by the Industry 4.0 paradigm and affects any present-day enterprise. In this scenario, process agility is often the critical indicator, which tells whether a business is going to succeed or die out as the ancient dinosaurs that could not accommodate to rapid environment changes. Hence, we systematically address agility in this book and provide a "survival kit" that includes a set of models, methods, practices, and case studies.

As such, the formal (i.e., technological) side of the complexity problem is discussed thereafter to a less extent, as that was the focus of our previous books, "Crisis Management for Software Development and Knowledge Transfer" (Springer, 2016) and "Managing Software Crisis: A Smart Way to Enterprise Agility" (Springer, 2018). Instead, we specifically address the human factor-based communication problems related to knowledge transfer and diversity. Therewith, we illustrate the agile practices and present case studies of large-scale and IT-intensive ecosystems, in their digital transformation crises.

We hope that this book will become an essential "first-aid kit" and successfully assist in agile enterprise management.

Moscow, Russia — Prof. Sergey V. Zykov
Bathinda, India — Dr. Amitoj Singh

Acknowledgements

I would like to thank the colleagues of mine who significantly contributed to this book. They clarified my initially vague concepts and assisted in a number of processes including translation, copyediting, diagramming, etc. These are the students who did their master/Ph.D. theses under my supervision. A few of their takeaways were transformed and included in this book as case studies on agility improvement. They are: Kamila Agayeva, Ali Akbari, John Dadzie, Dilara Dilber, Ramis Gabeydulin, Sergey Korobin, Joseph Lamptey, Nikita Morgun, Hanieh Mortazavi, Dinara Nikolaeva, Lyubove Smirnova, Sabina Supibek, Ella Tyuryumina, and Svetlana Zlobina.

I would like to thank the Springer Executive Editor Dr. Thomas Ditzinger, the Springer Senior Editor Mrs. Aninda Bose, the Springer Senior Executive for Production Mr. Ashok Kumar, and the Springer Project Coordinators for Books Production, Mr. Daniel Joseph Glarance and Mr. Ravivarman Selvaraj, for their continuous availability and prompt assistance.

In addition, I would like to express my deep appreciation and sincere gratitude to the Editors in Chief of the Springer Series in Smart Innovation, Systems and Technologies, Prof. Lakhmi C. Jain and Prof. Robert J. Howlett, for their tireless efforts in supporting my ideas. I would like to extend my thanks to the entire Chitkara team, and particularly to Dr. Archana Mantri and Mr. Sanjeev Sahni for their enthusiasm and cooperation.

Introduction

Why Agility Matters

This book focuses on agility engineering in large-scale systems. To do this in a smart way, we suggest a multifaceted approach that carefully blends models and methods, and includes patterns, principles, and practices. This approach would be incomplete if it did not address the human-related factors. Booch explained that in architecting large-scale systems (including enterprise ones—see Fig. 1), there are a number of dimensions of complexity. Of these, managerial and technical dimensions are essential. However, depending on a particular system type, one of these dimensions may dominate. The simplest system in both dimensions is an Excel spreadsheet. A typical example of a very technically complex system is a large telecommunication switch. Surprisingly for many technical experts, the enterprise systems are complex in terms of management. Of course, this does not mean that they are technically simple. However, it is their managerial complexity which is mission-critical and, therefore, it often is a root cause of system development crises. We have discussed these crises in our previous books [2, 3]; they typically result from imbalance between available resources, business requirements, and technical constraints. We stated that "in software development … a crisis is … a disproportion between client's expectations and the actual product behavior." How can this disproportion happen? Very often, this happens as a result of miscommunication in a complex system of multi-sided project participants (at a minimum, including client team, developer team, and their management teams). Lack of resilient and responsive communication with proper (timely, focused, and goal-directed) feedback often results in distorted product vision and consequently wrong product development and delivery. In case we treat an enterprise as an "adaptive socio-technical large-scale system" [4], it includes such sub-systems as social and technical. As such, the social sub-system operates by means of messaging. Consequently, "organization is communication" [2]. Therefore, the focus of this book is the managerial aspects of the agile development; these are related to the so-called human factors. To determine the complexity of the management

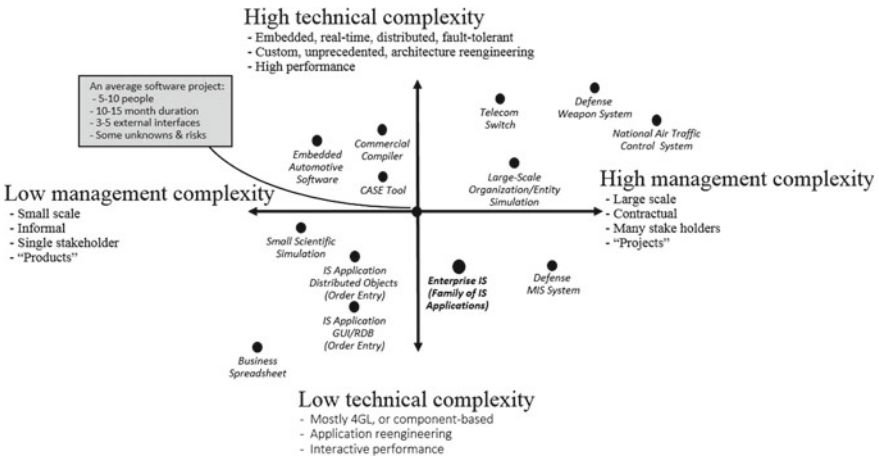

Fig. 1 Technical and managerial complexity in large-scale systems (Booch, G.: Software Architecture and the UML, 2006)

framework, we provide a brief outline of the lifecycle models and methodologies. Our idea is to provide guidelines for flexible (i.e., agile) adjusting the project parameters "on the move." As such, we analyze the model-and-methodology frameworks to explain the ways of their smart tailoring and agile adjustment by means of trade-off optimization of the managerial dimension of complexity based on the human factors. To deeper understand and master agile adjustment techniques and practices, we suggest a set of case studies, which we believe are a powerful approach to develop analytical and decision-making skills. Gill argues that case studies are especially suitable for conquering complexity in large-scale systems and better informing [5]. He also states that "… case studies … are likely to be the most rigorous form of research in many—if not most—settings where the fundamental test of our research value is its applicability to practice" [5]. Moreover, case studies address the higher levels of Bloom's taxonomy such as analysis, synthesis, and evaluation [1].

The case studies this book presents include enterprise scale systems which are either IT intensive/dependent or which focus on IT development/training. These are: Chitkara University Innovation and Research Network (India) and Innopolis Ecosystem (Russian Federation).

One mission-critical human factor that we address is diversity. We systematically address this factor in aspects such as:

- culture,
- mentality,
- production process maturity.

Therefore, we include comparative studies of the same domain in diverse conditions and analyze their similarities and comparative differences. Our case studies contain the stories of innovative ecosystems.

The domain is university training. This is generally known and, therefore, easily comprehensible. Our approach to diversity embraces locations such as India and Russia; these are combined with predominantly state vs. private ownership, and different timelines.

Our systematic approach includes a multi-context case study analysis. The case studies are multifaceted by themselves as they include multiple interpretations [5]. However, their smart application, in our view, requires comparative SWOT analysis that we include in this book. Based on that, we provide an agility checklist in the form of do's and don'ts.

This book is organized as follows. Chapter 1 summarizes the key concepts, such as agility and human factors. It outlines flexible strategies for smart software development and process improvement. Chapter 2 analyzes diversity in terms of culture, mentality, organization structure, development stage, and process maturity, relating these to system agility. It discusses large-scale software development in certain domains. Chapters 3 and 4 address Agile Knowledge Management and provide retrospective and context-specific examples, which make a solid basis for the following case studies. Chapter 5 investigates agile development at individual and team levels. Chapter 6 cross-examines the case studies on large-scale innovative and IT-intensive ecosystems. These are analyzed in terms of agility in different and diverse contexts including human-related factors. The conclusion summarizes the key findings of the book. It suggests ways for smart and agile large-scale software development.

Specifically, we consider the process and quality improvement approaches and lifecycles. Concerning the process frameworks for software development, we briefly discuss certain aspects of the "5A" model, DSDM, and DAD (Chaps. 3 and 5), and a number of other approaches such as:

- RUP (Rational Unified Process initiated by Rational Co. and later acquired/updated by IBM Corp.),
- MSF (Microsoft Solutions Framework, their de facto corporate standard),
- Scrum (a pioneering agile methodology incorporating techniques from sports and movie making),
- XP (Extreme Programming that originally failed for automobile production but was surprisingly successful in IT),
- Agile (one of the recent approaches that incorporates the merits of its predecessors).

These approaches received more coverage in [2, 3].

Our survey concludes that for each software product lifecycle and its individual stage, agility essentially assists in improving development patterns and practices. However, for every software project the developers must align this agility level with the key performance indicators, which are unique. Therefore, for the project success the developers should systematically approach this agility by means of multifaceted

and multi-contextual parameter prioritization and adjustment. This adjustment should focus on the agility improvement techniques and practices that optimize the critical lifecycle stages.

Further, concerning the case studies, we follow the approach suggested by Gill, the case method guru from Harvard Business School who successfully defended over 900 of such assignments while being a student and later developed his own case method-based teaching framework [5]. This includes:

- the story narrative,
- walk-through,
- discussion,
- takeaways.

As these case studies deal with diverse environments (in terms of culture, mentality, religious beliefs, funding sources, and a number of other aspects), we conclude that environments often critically matter as a business success factor.

The other finding is that the choice of a lifecycle pattern (such as waterfall, evolution, scrum, extreme) should be well aligned with a number of parameters, which include the functional and quality attributes of the future product, high-level project goals, and human-related factors that include multi-dimensional diversity.

This book will recommend lifecycle adjustments and process improvements that will help to establish and maintain agility in the competitive software development market. These recommendations are based on real-world case studies. This book does not contain a "silver bullet" agility recipe. Instead, it recommends patterns and practices that assist in crisis-responsive and agile large-scale software development.

References

1. Bloom, B. S. (1956). *A taxonomy of educational objectives* (2nd ed.). Addison-Wesley Longman Ltd.
2. Zykov, S. V. (2018). *Managing software crisis: A smart way to enterprise agility* (p. 153). Switzerland: Springer International Publishing.
3. Zykov, S. V. (2016). *Crisis management for software development and knowledge transfer* (p. 133). Switzerland: Springer International Publishing.
4. Gromoff, A. (2010). A study of the subject-oriented approach for automation and process modeling of a service company. *S-BPM ONE, 2010* (pp. 192–206).
5. Gill, G. T. (2011). *Case method informing with the case method: A guide to case method research, writing, & facilitation* (p. 562). Informing Science Press.

Contents

Acronyms

3D	Three dimensions
6D	Six dimensions
ACDM	Architecture-centric design method
AKM	Agile Knowledge Management
API	Application programming interface
ASD	Agile software development
ATAM	Architectural trade-off analysis method
BI	Business intelligence
CAD	Computer-aided design
CAE	Computer-aided engineering
CAM	Computer-aided manufacturing
CASE	Computer-aided software engineering
CMM(I)	Capability Maturity Model (Integration)
CMS	Content management system
CMU	Carnegie Mellon University
CoPs	Communities of practice
COTS	Commercial off-the-shelf
CRM	Customer relationship management
DAD	Disciplined agile delivery
DBMS	Database management system
ERP	Enterprise resource planning
FDD	Feature-driven development
GDP	Gross domestic product
GSD	Global Software Development
GUI	Graphical user interface
HR	Human resources
IBM	International Business Machines (Corporation)
ICT	Information and communication technologies
IEC	International Electrotechnical Commission
IP	Intellectual property

ISO	International Standards Organization
IT	Information technology
KLOC	Kilo lines of code
KM	Knowledge Management
LC	Inductor, represented by the letter L, and a capacitor, represented by the letter C
MSF	Microsoft Solutions Framework
MSIT-SE	Master of Science in Information Technology–Software Engineering
NATO	North Atlantic Treaty Organization
NPP	Nuclear power plant
PLM	Product lifecycle management
PSP	Personal software process
ROI	Return on investment
RUP	Rational unified process
SCADA	Supervisory control and data acquisition
SE	Software engineering
SEI	Software Engineering Institute
TEP	Teacher education program
XP	Extreme Programming