Foreword

Barry Boehm

When the software field was growing up, the software being developed dealt mainly with relatively stable applications. These involved relatively stable business and scientific applications, and software involved in controlling relatively stable hardware devices. As experiences in defining requirements for hardware devices found that design solutions would often become requirements and overconstrain the solution space, the software field followed the hardware field in postponing the design until the requirements were completely and consistently defined. This led to the dominance of the sequential, top-down, requirements-first, reductionist waterfall approach used to define, develop, and manage software projects.

One of my jobs at TRW in 1976-77 was to lead a project to formalize this approach into a set of corporate software development policies and standards. These were inculcated in the company via training materials, courses, and a 40-question equivalent of the California drivers-license test that TRW software developers needed to pass. We also highlighted this material in a public relations campaign to show our mastery of software development and management.

This worked very well for a while, but by the early 1980's the assumption of stable, predetermined requirements began to lose its validity. In particular, graphic-user-interactive (GUI) terminals began to become economically viable. Users much preferred this way of operating, but our requirements engineers found that (1) it was hard to specify graphic layouts in requirement documents, and (2) it was hard to get users to define how they wanted to interact. We encountered the IKIWISI syndrome: "I can't tell you how I want it, but I'll know it when I see it."

Our more creative software engineers began to develop rapid-prototyping capabilities that potential customers found very helpful in resolving IKIWISI requirements. However, when we tried to emphasize rapid prototyping in competitive procurements, we found that we had so thoroughly brainwashed many of our senior software engineers that they would pound on the table and say, "You can't do that! It's programming before we've defined the requirements, and it violates our policies!" Further, we found that several government agencies had adopted and adapted our policies and standards as their way of doing business. And if undoing corporate policies was difficult, undoing government policies and standards was virtually impossible.

Since then, further trends have made the sequential, reductionist approach less and less viable. Requirements have become more emergent with system use. With COTS products and cloud services, their capabilities drive the system requirements rather than prespecified requirements. Time-to-market pressures and rapidly evolving products such as cell phones have made sequential definition and development processes uncompetitive in the marketplace, along with increasingly rapid changes in technology, organizations, and user preferences. Yet, many organizations cling to

the sequential, reductionist approach as a security blanket. Increasingly, they take several years to deliver a system, and then find that its technology is obsolete and that its users' needs have become much different.

Thus, the appearance of this book, Software Project Management in a Changing World, is very timely. It focuses on how people and organizations can make their processes more change-adaptive. It is good in emphasizing in its chapters on cost estimation and risk/opportunity management that unpredictable change requires probabilistic approaches, using range vs. point estimates, late-binding of product-content decisions, and evolutionary development. It has good guidance on agile project management, using principles such as minimum-critical specifications, autonomous teams, skills redundancy, and use of feedback and post-release reflection.

The book is also strong on quality management and on balancing lightweight agile methods with the use of empirical methods, using Goal-Question-Metric and Experience Factory-type approaches to management and use of project knowledge. Its chapters on global project management and global team motivation are strong on identifying and employing knowledge on personnel motivation, and on the importance of investments in team-building and trust, although the chapter on human resource allocation focuses more on algorithmic methods of project staffing.

The strong emphasis on how to make software processes more change-adaptive could have done more on how to make software products more change-adaptive. A good example is the approach in David Parnas' paper on Designing Software for Ease of Extension and Contraction. This involves identifying sources of change and encapsulating them into modules, so that change effects are largely confined to individual modules, rather than rippling through the rest of the product. This also involves identifying evolution requirements as well as current-snapshot requirements for the initial product. Other good product-adaptive approaches include open interface standards, use of design patterns and generics, judicious selection of COTS products that are change-adaptive without destabilizing their users, and emphasizing simplicity via Occam's Razor or Einstein's guidance, "Everything should be as simple as possible, but no simpler."

That said, the book is also strong in identifying sources of change in software technology and their implications for software management. These include big-data and search technology that can enhance project knowledge; and social-media technology that can enable better multi-discipline and distributed-stakeholder collaboration in software requirements negotiation, change handling, and concurrence at decision gates. Also, improved process simulation technology can be used to better understand the likely effects of alternative project decisions, and to determine the domains of applicability of various software "laws," such as Brooks' Law: Adding people to a late software project will make it later (not always true if foreseen and done early). It is also strong in identifying alternative software development methods and their management differences, such as open source, inner-source, distributed and global software development, and agile methods.

Overall, I found the book to be a pleasure to read and a valuable source of guidance on how to cope with the proliferating sources of change we all will face in the future. I hope that you will benefit from it in similar ways.

Author Biography

Barry Boehm is the TRW Professor in the USC Computer Sciences, Industrial and Systems Engineering, and Astronautics Departments. He is also the Director of Research of the DoD-Stevens-USC Systems Engineering Research Center, and the founding Director of the USC Center for Systems and Software Engineering. He was director of DARPA-ISTO 1989-92, at TRW 1973-89, at Rand Corporation 1959-73, and at General Dynamics 1955-59. His contributions include the COCOMO family of cost models and the Spiral family of process models. He is a Fellow of the primary professional societies in computing (ACM), aerospace (AIAA), electronics (IEEE), and systems engineering (INCOSE), and a member of the U.S. National Academy of Engineering.

Acknowledgments

Editing a book is a major undertaking; it may sound like it is much less work than authoring your own book. And maybe it is less work, but foremost it is different. It requires a lot of coordination and hence editors become highly dependent on the contributors, reviewers and others providing support. This book is no different.

We would like to express our gratitude to all authors contributing with their expertise to the chapters and being responsive to our comments, enquiries and requests. We are grateful to all reviewers helping us to further improve the content of the book. Maleknaz Nayebi was of tremendous support in preparing supplementary literature studies. We would also like to express our gratitude to the Springer team for their support and in particular to Ralf Gerstner for his guidance and valuable input on practical matters.

We are grateful to Professor Barry Boehm for agreeing to write the foreword, and sharing his long experience with us. He provides his personal perspective on the evolution of software development as a whole, and software project management in particular.

Last but not least, we would like to express our sincere thanks to Kornelia Streb whose input to the editing of the book has been a prerequisite to manage the progress and allowed us to focus on the content.

February 2014

Günther Ruhe and Claes Wohlin

Contents

List of Contributors

Andreas S. Andreou
Department of Electrical Engineering, Computer Engineering and Informatics
Cyprus University of Technology
Lemesos, Cyprus
Email: andreas.andreou@cut.ac.cy

Sarah Beecham
Department of Computer Science & Information Systems
Lero – The Irish Software Engineering Centre
University of Limerick
Limerick, Ireland
Email: sarah.beecham@lero.ie

Barry Boehm
University of Southern California
Los Angeles, USA
Email: barryboehm@gmail.com

Sjaak Brinkkemper
Department of Information and Computing Sciences
Utrecht University
Utrecht, The Netherlands
Email: s.brinkkemper@cs.uu.nl

Darren Dalcher
National Centre for Project Management
University of Hertfordshire
Hatfield, United Kingdom
Email: d.dalcher2@herts.ac.uk

Alexander Delater
Institute of Computer Science
University of Heidelberg
Heidelberg, Germany
Email: delater@informatik.uni-heidelberg.de

Torgeir Dingsøyr
SINTEF
Trondheim, Norway
Email: dingsoyr@idi.ntnu.no

Ton Dobbe
UNIT4
Sliedrecht, The Netherlands
Email: Ton.Dobbe@unit4.com

Tore Dybå
SINTEF
Trondheim, Norway
Email: tore.dyba@sintef.no

Christof Ebert
Vector Consulting Services GmbH
Ingersheimer Strasse 24
70499 Stuttgart, Germany
Email: christof.ebert@vector.com

Filomena Ferrucci
DISTRA
University of Salerno
Salerno, Italy
Email: fferrucci@unisa.it

Mark Harman
Software Systems Engineering Group
Department of Computer Science
University College London
London, United Kingdom
Email: mark.harman@ucl.ac.uk

Rachel Harrison
Computing and Communication Technologies
Oxford Brookes University
Oxford, United Kingdom
Email: Rachel.Harrison@brookes.ac.uk

Jens Heidrich
Fraunhofer IESE
Kaiserslautern, Germany
Email: jens.heidrich@iese.fraunhofer.de

Tom-Michael Hesse
Institute of Computer Science
University of Heidelberg
Heidelberg, Germany
Email: hesse@informatik.uni-heidelberg.de

Martin Höst
Department of Computer Science
Lund University
Lund, Sweden
Email: martin.host@cs.Ith.se

Erik Jagroep
Department of Information and Computing Sciences
Utrecht University
Utrecht, The Netherlands
Email: e.a.jagroep@uu.nl

Michael Kläs
Fraunhofer IESE
Kaiserslautern, Germany
Email: Michael.Klaes@iese.fraunhofer.de

Tim Menzies
Lane Department of Computer Science & Electrical Engineering
West Virginia University
Morgantown, West Virginia
Email: tim.menzies@gmail.com

Nils Brede Moe
SINTEF
Trondheim, Norway
Email: nilsm@sintef.no

Alma Oručević-Alagić
Computer Science
Lund University
Lund, Sweden
Email: alma.orucevic-alagic@cs.lth.se

Barbara Paech
Institute of Computer Science
University of Heidelberg
Heidelberg, Germany
Email: paech@informatik.uni-heidelberg.de

Dietmar Pfahl
Institute of Computer Science
University of Tartu
Tartu, Estonia
Email: dietmar.pfahl@ut.ee

Dieter Rombach
Technische Universität Kaiserslautern
Kaiserslautern, Germany
Email: rombach@informatik.uni-kl.de

Günther Ruhe
Department of Computer Science & Electrical Engineering
University of Calgary
Calgary, Alberta, Canada
Email: ruhe@ucalgary.ca

Federica Sarro
Department of Computer Science
University College London
London, United Kingdom
Email: f.sarro@ucl.ac.uk

Martin Shepperd
Information Systems and Computing
Brunel University
Uxbridge, United Kingdom
Email: martin.shepperd@brunel.ac.uk

Darja Smite
Blekinge Institute of Technology
Karlskrona, Sweden
Email: darja.smite@bth.se

Ioannis Stamelos
Department of Informatics
Aristotle University of Thessaloniki
Thessaloniki, Greece
Email: stamelos@csd.auth.gr

Klaas-Jan Stol
Lero, the Irish Software Engineering Research Centre
University of Limerick
Email: klaas-jan.stol@lero.ie

Constantinos Stylianou
Department of Computer Science
University of Cyprus
Lefkosia, Cyprus
Email: cstylianou@cs.ucy.ac.cy

Inge Van de Weerd
Department of Information, Logistics and Innovation
VU University Amsterdam
Amsterdam, The Netherlands
Email: i.vande.weerd@vu.nl

Varsha Veerappa
Department of Computing and Communication Technologies
Oxford Brookes University
Oxford, United Kingdom
Email: vveerappa@brookes.ac.uk

Claes Wohlin
Blekinge Institute of Technology
Karlskrona, Sweden
Email: claes.wohlin@bth.se