

## 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.

## VIII Acknowledgments

### **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

<b>List of Contributors .....</b>	<b>XV</b>
-----------------------------------	-----------

<b>1 Software Project Management: Setting the Context .....</b>	<b>1</b>
---	----------

*Günther Ruhe and Claes Wohlin*

1.1 Motivation .....	1
1.2 Characteristics of Software Projects and why Software Project Management is Difficult.....	2
1.3 Ten Knowledge Areas of Software Project Management.....	5
1.4 The Book's Coverage of the PMBOK Knowledge Areas .....	17
1.5 The Multi-disciplinary Nature of Project Management .....	19
1.6 The Future of Software Engineering .....	20
1.7 Software Project Management - Past and Future .....	22
1.8 This Book .....	23

<b>Part I — Fundamentals .....</b>	<b>27</b>
------------------------------------	-----------

<b>2 Rethinking Success in Software Projects: Looking Beyond the Failure Factors .....</b>	<b>29</b>
--	-----------

*Darren Dalcher*

2.1 The Extent of Software Project Failures .....	29
2.2 Beyond Simple Success Measures .....	32
2.3 Rethinking Project Success .....	36
2.4 Towards Multiple Levels of Success .....	38
2.5 Mapping Success.....	39
2.6 Illustrative Examples.....	42
2.7 The Impact of Time .....	43
2.8 Measuring Success .....	44
2.9 Conclusions .....	49

X Contents

**3 Cost Prediction and Software Project Management..... 53**

*Martin Shepperd*

3.1 Introduction .....	53
3.2 A Review of State of the Art Techniques.....	54
3.3 A Review of Cost Estimation Research.....	57
3.4 The Interaction between People and Formal Techniques.....	60
3.5 Practical Recommendations .....	64
3.6 Follow up Sources of Information.....	67

**4 Human Resource Allocation and Scheduling for Software Project Management..... 75**

*Constantinos Stylianou and Andreas S. Andreou*

4.1 Introduction .....	75
4.2 Human Resource Allocation and Scheduling Approaches .....	77
4.3 The Implication of Software Development Personality Types.....	93
4.4 Further Research Trends and Challenges .....	101
4.5 Concluding Remarks .....	102

**5 Software Project Risk and Opportunity Management..... 109**

*Barry Boehm*

5.1 Introduction .....	109
5.2 The Duality of Risks and Opportunities.....	110
5.3 Fundamentals of Risk-Opportunity Management .....	111
5.4 Risk and Opportunity Management Methods, Processes, and Tools .....	117
5.5 Top-10 Risk Item Tracking.....	120
5.6 Risk-Balanced Activity Levels.....	121
5.7 Summary and Conclusions.....	121

**Part II — Supporting Areas..... 125**

**6 Model-based Quality Management of Software Development Projects..... 127**

*Jens Heidrich, Dieter Rombach and Michael Kläs*

6.1 Introduction .....	127
6.2 Selecting the Right Quality Models .....	131
6.3 Building Custom-tailored Quality Models.....	139
6.4 Specification and Application of Quality Models .....	146

6.5 Strategic Usage of Quality Models .....	152
6.6 Conclusions and Future Work .....	155

**7 Supporting Project Management through Integrated Management of System and Project Knowledge..... 161**

*Barbara Paech, Alexander Delater and Tom-Michael Hesse*

7.1 Introduction .....	161
7.2 Our Vision: Integrated System and Project Knowledge Management.....	165
7.3 Literature Review .....	169
7.4 Integrating System and Project Knowledge Using Work Items .....	172
7.5 Integrating System and Project Knowledge Using Decisions.....	180
7.6 Research Issues on Integrating System and Project Knowledge .....	190
7.7 Conclusions and Outlook .....	193

**8 A Framework for Implementing Product Portfolio Management in Software Business..... 199**

*Erik Jagroep, Inge van de Weerd, Sjaak Brinkkemper and Ton Dobbe*

8.1 Introduction .....	199
8.2 Research Approach.....	202
8.3 Theory-building Case Study and Evaluation.....	207
8.4 Software Product Portfolio Management Implementation Framework ...	209
8.5 Maturity Matrix for PPM .....	216
8.6 Theory-testing Case Study .....	218
8.7 Implications .....	222
8.8 Conclusions and Future Research .....	223

**9 Managing Global Software Projects ..... 229**

*Christof Ebert*

9.1 Introduction .....	229
9.2 Foundations .....	231
9.3 Benefits and Challenges .....	232
9.4 Global Software Development.....	238
9.5 Work Organization .....	241
9.6 Risk Management in Global Software Projects .....	245
9.7 Trends and Conclusions.....	248

XII Contents

**10 Motivating Software Engineers Working in Virtual Teams  
across the Globe ..... 255**

*Sarah Beecham*

10.1 Introduction .....	255
10.2 Motivation Theory .....	257
10.3 Software Engineer Characteristics .....	261
10.4 Software Engineer Motivation in GSD - A Case Study .....	264
10.5 Motivation Factors and GSD Guidelines .....	269
10.6 Theory and Practice of GSD Motivation .....	272
10.7 Summary and Conclusions .....	277

**Part III — New Paradigms ..... 283**

**11 Agile Project Management ..... 285**

*Tore Dybå, Torgeir Dingsøy and Nils Brede Moe*

11.1 Introduction .....	285
11.2 Software Project Management .....	286
11.3 Self-managing Software Teams .....	289
11.4 Team Leadership .....	291
11.5 Feedback and Learning .....	294
11.6 Principles of Agile Project Management .....	300
11.7 Conclusions .....	303

**12 Distributed Project Management ..... 309**

*Darja Šmite*

12.1 Introduction .....	309
12.2 Ten Misconceptions in Distributed Software Development .....	313
12.3 Conclusions .....	326

**13 Management and Coordination of Free/  
Open Source Projects..... 331**

*Ioannis Stamelos*

13.1 Introduction .....	331
13.2 F/OSS Management .....	336
13.3 Current Challenges in F/OSS Management .....	340
13.4 Future Open Source Management Techniques .....	343
13.5 Conclusions .....	348



**14 Inner Source Project Management..... 353**

*Martin Höst, Klaus-Jan Stol and Alma Oručević-Alagić*

14.1 Introduction .....353  
 14.2 Inner Source .....355  
 14.3 A Framework for Understanding Project Management in Inner Source 361  
 14.4 Case Studies .....366  
 14.5 Discussion and Future Work.....373

**Part IV — Emerging Techniques..... 379**

**15 Search-based Software Project Management..... 381**

*Filomena Ferrucci, Mark Harman and Federica Sarro*

15.1 Introduction .....381  
 15.2 Search-based Software Engineering.....382  
 15.3 Search-based Software Project Management .....384  
 15.4 Possible Directions for Future Work on Search-based  
 Project Management .....396  
 15.5 Conclusions .....399

**16 Social Media Collaboration in Software Projects ..... 409**

*Rachel Harrison and Varsha Veerappa*

16.1 Introduction .....409  
 16.2 Interactions in Software Projects.....410  
 16.3 Social Aspects of Software Projects.....412  
 16.4 Importance of Social Media in Software Projects.....412  
 16.5 Pilot Study .....413  
 16.6 The Future of Social Media in Software Projects .....427  
 16.7 Summary and Conclusions .....428

**17 Process Simulation – A Tool for Software Project  
 Managers? ..... 433**

*Dietmar Pfahl*

17.1 Purpose and Scope of Software Process Simulation .....433  
 17.2 An Illustrative Application Example .....436  
 17.3 The Gap Between State-of-Art and State-of-Practice .....446  
 17.4 Issues that Need to be Addressed .....450  
 17.5 Conclusions .....452

XIV Contents

<b>18 Occam's Razor and Simple Software Project Management .....</b>	<b>457</b>
<i>Tim Menzies</i>	
18.1 Introduction .....	457
18.2 Occam's Razor and Project Management .....	460
18.3 Speculation-Based Modeling (is Difficult).....	462
18.4 Support-Based Modeling (can be Simplified with Data Mining).....	464
18.5 Spectral Learning and Project Management .....	471
18.6 General Applications to Project Management.....	478
18.7 Discussion .....	479
<b>Index.....</b>	<b>485</b>

## 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-heidel-  
berg.de

### **Torgeir Dingsøy**

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 Tech-  
nologies  
Oxford Brookes University  
Oxford, United Kingdom  
Email: Rachel.Harrison@brookes.ac.uk

## XVI List of Contributors

### **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.lth.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 Commu-  
nication 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