Ita Richardson
Pekka Abrahamsson
Richard Messnarz (Eds.)

LNCS 3792

Software Process Improvement

12th European Conference, EuroSPI 2005
Budapest, Hungary, November 2005
Proceedings

Ita Richardson Pekka Abrahamsson
Richard Messnarz (Eds.)

Software Process Improvement

12th European Conference, EuroSPI 2005
Budapest, Hungary, November 9-11, 2005
Proceedings

Springer

Volume Editors

Ita Richardson
University of Limerick, Department of Computer Science and Information Systems
National Technological Park Castletroy, Limerick, Ireland
E-mail: ita.richardson@ul.ie

Pekka Abrahamsson
VTT Technical Research Centre of Finland
PO Box 1100, Kaitoväylä 1, 90571 Oulu, Finland
E-mail: pekka.abrahamsson@vtt.fi

Richard Messnarz
International Software Consulting Network GmbH Research Office
Schieszstattgasse 4, 8010 Graz, Austria
E-mail: rmess@iscn.com

Library of Congress Control Number: 2005935531

CR Subject Classification (1998): D.2, K.6, K.4.2

ISSN 0302-9743
ISBN-10 3-540-30286-7 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-30286-5 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11586012 06/3142 5 4 3 2 1 0

Lecture Notes in Computer Science 3556

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

Preface

This volume is intended for SPI (software process improvement) managers and researchers, quality managers, and experienced project and research managers. The papers constitute the research proceedings of the 12th EuroSPI (European Software Process Improvement, www.eurospi.net) conference held in Budapest, 9–11 November 2005, Hungary. Conferences have been held in 1994 in Dublin, 1995 in Vienna (Austria), 1997 in Budapest (Hungary), 1998 in Gothenburg (Sweden), 1999 in Pori (Finland), 2000 in Copenhagen (Denmark), 2001 in Limerick (Ireland), 2002 in Nuremberg (Germany), 2003 in Graz (Austria), and 2004 in Trondheim (Norway). EuroSPI established an experience library (library.eurospi.net) which will be continuously extended over the next years and will be made available to all attendees. EuroSPI also created an umbrella initiative for establishing a European Qualification Network in which different SPINs and national initiatives join mutually beneficial collaborations.

From 2005, through EuroSPI partners and networks, in collaboration with the European Union (supported by the EU Leonardo da Vinci Programme), a certification body will be created for the IT and services sector so as to offer SPI knowledge and certificates to industry, establishing close knowledge transfer links between research and industry. The biggest value of EuroSPI lies in its function as a European knowledge and experience exchange mechanism between SPI research institutions and industry.

September 2005

Dr. Richard Messnarz
General Chair EuroSPI
www.eurospi.net

Organization Committee

EuroSPI 2005 is organized by the EuroSPI partnership (www.eurospi.net), internationally coordinated by ISCN, and locally supported by the John von Neumann Computer Society.

Program Committee

Conference Chair	Richard Messnarz (ISCN, Ireland)
Scientific Program Chair	Ita Richardson (University of Limerick, Ireland)
Scientific Program Chair	Pekka Abrahamsson (VTT, Finland)
Industrial Program Chair	Nils Brede Moe (SINTEF, Norway)
Industrial Program Chair	Risto Nevalainen (STTF, Finland)
Tutorial Chair	Richard Messnarz (ISCN, Ireland)
Exhibition Chair	Stephan Goericke (ISQI, Germany)
Organizing Chair	Miklos Biro (Corvinus University, Hungary)
Organizing Chair	Adrienne Clarke (ISCN, Ireland)

Local Committee

Local Organizer	John von Neumann Computer Society, www.njszt.hu

Additional Scientific Reviewers

V. Ambriola
A. Aurum
A. Beriozko
S. Biffl
M. Biro
C. Bunse
M. Ciolkowski
K. Cox
K.C. Desouza
H. Duncan
T. Daughtrey
T. Dyba
J. Pries-Heje
S. Hope
K. H. Kautz
D. Landes
M. Lindvall
P. McQuaid
M. Mueller
J. Muench
J. Niere
M. Oivo
E. Ostolaza
K. Molokken-Ostvold
G. Ruhe
P. Runeson
M. Shepperd
K. Siakas
K. Schneider
T. Stalhane
T. Varkoi
C. Wohlin

Lecture Notes in Computer Science

For information about Vols. 1–3691

please contact your bookseller or Springer

Vol. 3807: M. Dean, Y. Guo, W. Jun, R. Kaschek, S. Krishnaswamy, Z. Pan, Q.Z. Sheng (Eds.), Web Information Systems – WISE 2005 Workshops. XV, 275 pages. 2005.

Vol. 3806: M. Kitsuregawa, E.J. Neuhold, A.H. H. Ngu, J.-Y. Chung, Q.Z. Sheng (Eds.), Web Information Systems – WISE 2005. XXI, 771 pages. 2005.

Vol. 3792: I. Richardson, P. Abrahamsson, R. Messnarz (Eds.), Software Process Improvement. VIII, 215 pages. 2005.

Vol. 3791: A. Adi, S. Stoutenburg, S. Tabet (Eds.), Rules and Rule Markup Languages for the Semantic Web. X, 225 pages. 2005.

Vol. 3789: A. Gelbukh, Á. de Albornoz, H. Terashima-Marín (Eds.), MICAI 2005: Advances in Artificial Intelligence. XXVI, 1198 pages. 2005. (Subseries LNAI).

Vol. 3785: K.-K. Lau, R. Banach (Eds.), Formal Methods and Software Engineering. XIV, 496 pages. 2005.

Vol. 3784: J. Tao, T. Tan, R.W. Picard (Eds.), Affective Computing and Intelligent Interaction. XIX, 1008 pages. 2005.

Vol. 3781: S.Z. Li, Z. Sun, T. Tan, S. Pankanti, G. Chollet, D. Zhang (Eds.), Advances in Biometric Person Authentication. XI, 250 pages. 2005.

Vol. 3780: K. Yi (Ed.), Programming Languages and Systems. XI, 435 pages. 2005.

Vol. 3779: H. Jin, D. Reed, W. Jiang (Eds.), Network and Parallel Computing. XV, 513 pages. 2005.

Vol. 3777: O.B. Lupanov, O.M. Kasim-Zade, A.V. Chaskin, K. Steinhöfel (Eds.), Stochastic Algorithms: Foundations and Applications. VIII, 239 pages. 2005.

Vol. 3775: J. Schönwälder, J. Serrat (Eds.), Ambient Networks. XIII, 281 pages. 2005.

Vol. 3773: A.S. Cortés, M.L. Cortés (Eds.), Progress in Pattern Recognition, Image Analysis and Applications. XX, 1094 pages. 2005.

Vol. 3772: M. Consens, G. Navarro (Eds.), String Processing and Information Retrieval. XIV, 406 pages. 2005.

Vol. 3770: J. Akoka, S.W. Liddle, I.-Y. Song, M. Bertolotto, I. Comyn-Wattiau, W.-J. van den Heuvel, M. Kolp, J.C. Trujillo, C. Kop, H.C. Mayr (Eds.), Perspectives in Conceptual Modeling. XXII, 476 pages. 2005.

Vol. 3768: Y.-S. Ho, H.J. Kim (Eds.), Advances in Mulitmedia Information Processing - PCM 2005, Part II. XXVIII, 1088 pages. 2005.

Vol. 3767: Y.-S. Ho, H.J. Kim (Eds.), Advances in Mulitmedia Information Processing - PCM 2005, Part I. XXVIII, 1022 pages. 2005.

Vol. 3766: N. Sebe, M.S. Lew, T.S. Huang (Eds.), Computer Vision in Human-Computer Interaction. X, 231 pages. 2005.

Vol. 3765: Y. Liu, T. Jiang, C. Zhang (Eds.), Computer Vision for Biomedical Image Applications. X, 563 pages. 2005.

Vol. 3764: S. Tixeuil, T. Herman (Eds.), Self-Stabilizing Systems. VIII, 229 pages. 2005.

Vol. 3762: R. Meersman, Z. Tari, P. Herrero (Eds.), On the Move to Meaningful Internet Systems 2005: OTM Workshops. XXXI, 1228 pages. 2005.

Vol. 3761: R. Meersman, Z. Tari (Eds.), On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE, Part II. XXVII, 653 pages. 2005.

Vol. 3760: R. Meersman, Z. Tari (Eds.), On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE, Part I. XXVII, 921 pages. 2005.

Vol. 3759: G. Chen, Y. Pan, M. Guo, J. Lu (Eds.), Parallel and Distributed Processing and Applications - ISPA 2005 Workshops. XIII, 669 pages. 2005.

Vol. 3758: Y. Pan, D. Chen, M. Guo, J. Cao, J. Dongarra (Eds.), Parallel and Distributed Processing and Applications. XXIII, 1162 pages. 2005.

Vol. 3756: J. Cao, W. Nejdl, M. Xu (Eds.), Advanced Parallel Processing Technologies. XIV, 526 pages. 2005.

Vol. 3754: J. Dalmau Royo, G. Hasegawa (Eds.), Management of Multimedia Networks and Services. XII, 384 pages. 2005.

Vol. 3753: O.F. Olsen, L.M.J. Florack, A. Kuijper (Eds.), Deep Structure, Singularities, and Computer Vision. X, 259 pages. 2005.

Vol. 3752: N. Paragios, O. Faugeras, T. Chan, C. Schnörr (Eds.), Variational, Geometric, and Level Set Methods in Computer Vision. XI, 369 pages. 2005.

Vol. 3751: T. Magedanz, E.R. M. Madeira, P. Dini (Eds.), Operations and Management in IP-Based Networks. X, 213 pages. 2005.

Vol. 3750: J.S. Duncan, G. Gerig (Eds.), Medical Image Computing and Computer-Assisted Intervention – MICCAI 2005, Part II. XL, 1018 pages. 2005.

Vol. 3749: J.S. Duncan, G. Gerig (Eds.), Medical Image Computing and Computer-Assisted Intervention – MICCAI 2005, Part I. XXXIX, 942 pages. 2005.

Vol. 3747: C.A. Maziero, J.G. Silva, A.M.S. Andrade, F.M.d. Assis Silva (Eds.), Dependable Computing. XV, 267 pages. 2005.

Vol. 3746: P. Bozanis, E.N. Houstis (Eds.), Advances in Informatics. XIX, 879 pages. 2005.

Vol. 3745: J.L. Oliveira, V. Maojo, F. Martín-Sánchez, A.S. Pereira (Eds.), Biological and Medical Data Analysis. XII, 422 pages. 2005. (Subseries LNBI).

Vol. 3744: T. Magedanz, A. Karmouch, S. Pierre, I. Venieris (Eds.), Mobility Aware Technologies and Applications. XIV, 418 pages. 2005.

Vol. 3740: T. Srikanthan, J. Xue, C.-H. Chang (Eds.), Advances in Computer Systems Architecture. XVII, 833 pages. 2005.

Vol. 3739: W. Fan, Z.-h. Wu, J. Yang (Eds.), Advances in Web-Age Information Management. XXIV, 930 pages. 2005.

Vol. 3738: V.R. Syrotiuk, E. Chávez (Eds.), Ad-Hoc, Mobile, and Wireless Networks. XI, 360 pages. 2005.

Vol. 3735: A. Hoffmann, H. Motoda, T. Scheffer (Eds.), Discovery Science. XVI, 400 pages. 2005. (Subseries LNAI).

Vol. 3734: S. Jain, H.U. Simon, E. Tomita (Eds.), Algorithmic Learning Theory. XII, 490 pages. 2005. (Subseries LNAI).

Vol. 3733: P. Yolum, T. Güngör, F. Gürgen, C. Özturan (Eds.), Computer and Information Sciences - ISCIS 2005. XXI, 973 pages. 2005.

Vol. 3731: F. Wang (Ed.), Formal Techniques for Networked and Distributed Systems - FORTE 2005. XII, 558 pages. 2005.

Vol. 3729: Y. Gil, E. Motta, V. R. Benjamins, M.A. Musen (Eds.), The Semantic Web – ISWC 2005. XXIII, 1073 pages. 2005.

Vol. 3728: V. Paliouras, J. Vounckx, D. Verkest (Eds.), Integrated Circuit and System Design. XV, 753 pages. 2005.

Vol. 3726: L.T. Yang, O.F. Rana, B. Di Martino, J. Dongarra (Eds.), High Performance Computing and Communications. XXVI, 1116 pages. 2005.

Vol. 3725: D. Borrione, W. Paul (Eds.), Correct Hardware Design and Verification Methods. XII, 412 pages. 2005.

Vol. 3724: P. Fraigniaud (Ed.), Distributed Computing. XIV, 520 pages. 2005.

Vol. 3723: W. Zhao, S. Gong, X. Tang (Eds.), Analysis and Modelling of Faces and Gestures. XI, 4234 pages. 2005.

Vol. 3722: D. Van Hung, M. Wirsing (Eds.), Theoretical Aspects of Computing – ICTAC 2005. XIV, 614 pages. 2005.

Vol. 3721: A. Jorge, L. Torgo, P.B. Brazdil, R. Camacho, J. Gama (Eds.), Knowledge Discovery in Databases: PKDD 2005. XXIII, 719 pages. 2005. (Subseries LNAI).

Vol. 3720: J. Gama, R. Camacho, P.B. Brazdil, A. Jorge, L. Torgo (Eds.), Machine Learning: ECML 2005. XXIII, 769 pages. 2005. (Subseries LNAI).

Vol. 3719: M. Hobbs, A.M. Goscinski, W. Zhou (Eds.), Distributed and Parallel Computing. XI, 448 pages. 2005.

Vol. 3718: V.G. Ganzha, E.W. Mayr, E.V. Vorozhtsov (Eds.), Computer Algebra in Scientific Computing. XII, 502 pages. 2005.

Vol. 3717: B. Gramlich (Ed.), Frontiers of Combining Systems. X, 321 pages. 2005. (Subseries LNAI).

Vol. 3716: L. Delcambre, C. Kop, H.C. Mayr, J. Mylopoulos, Ó. Pastor (Eds.), Conceptual Modeling – ER 2005. XVI, 498 pages. 2005.

Vol. 3715: E. Dawson, S. Vaudenay (Eds.), Progress in Cryptology – Mycrypt 2005. XI, 329 pages. 2005.

Vol. 3714: H. Obbink, K. Pohl (Eds.), Software Product Lines. XIII, 235 pages. 2005.

Vol. 3713: L.C. Briand, C. Williams (Eds.), Model Driven Engineering Languages and Systems. XV, 722 pages. 2005.

Vol. 3712: R. Reussner, J. Mayer, J.A. Stafford, S. Overhage, S. Becker, P.J. Schroeder (Eds.), Quality of Software Architectures and Software Quality. XIII, 289 pages. 2005.

Vol. 3711: F. Kishino, Y. Kitamura, H. Kato, N. Nagata (Eds.), Entertainment Computing - ICEC 2005. XXIV, 540 pages. 2005.

Vol. 3710: M. Barni, I. Cox, T. Kalker, H.J. Kim (Eds.), Digital Watermarking. XII, 485 pages. 2005.

Vol. 3709: P. van Beek (Ed.), Principles and Practice of Constraint Programming - CP 2005. XX, 887 pages. 2005.

Vol. 3708: J. Blanc-Talon, W. Philips, D.C. Popescu, P. Scheunders (Eds.), Advanced Concepts for Intelligent Vision Systems. XXII, 725 pages. 2005.

Vol. 3707: D.A. Peled, Y.-K. Tsay (Eds.), Automated Technology for Verification and Analysis. XII, 506 pages. 2005.

Vol. 3706: H. Fuks, S. Lukosch, A.C. Salgado (Eds.), Groupware: Design, Implementation, and Use. XII, 378 pages. 2005.

Vol. 3704: M. De Gregorio, V. Di Maio, M. Frucci, C. Musio (Eds.), Brain, Vision, and Artificial Intelligence. XV, 556 pages. 2005.

Vol. 3703: F. Fages, S. Soliman (Eds.), Principles and Practice of Semantic Web Reasoning. VIII, 163 pages. 2005.

Vol. 3702: B. Beckert (Ed.), Automated Reasoning with Analytic Tableaux and Related Methods. XIII, 343 pages. 2005. (Subseries LNAI).

Vol. 3701: M. Coppo, E. Lodi, G. M. Pinna (Eds.), Theoretical Computer Science. XI, 411 pages. 2005.

Vol. 3700: J.F. Peters, A. Skowron (Eds.), Transactions on Rough Sets IV. X, 375 pages. 2005.

Vol. 3699: C.S. Calude, M.J. Dinneen, G. Păun, M. J. Pérez-Jiménez, G. Rozenberg (Eds.), Unconventional Computation. XI, 267 pages. 2005.

Vol. 3698: U. Furbach (Ed.), KI 2005: Advances in Artificial Intelligence. XIII, 409 pages. 2005. (Subseries LNAI).

Vol. 3697: W. Duch, J. Kacprzyk, E. Oja, S. Zadrożny (Eds.), Artificial Neural Networks: Formal Models and Their Applications – ICANN 2005, Part II. XXXII, 1045 pages. 2005.

Vol. 3696: W. Duch, J. Kacprzyk, E. Oja, S. Zadrożny (Eds.), Artificial Neural Networks: Biological Inspirations – ICANN 2005, Part I. XXXI, 703 pages. 2005.

Vol. 3695: M.R. Berthold, R.C. Glen, K. Diederichs, O. Kohlbacher, I. Fischer (Eds.), Computational Life Sciences. XI, 277 pages. 2005. (Subseries LNBI).

Vol. 3694: M. Malek, E. Nett, N. Suri (Eds.), Service Availability. VIII, 213 pages. 2005.

Vol. 3693: A.G. Cohn, D.M. Mark (Eds.), Spatial Information Theory. XII, 493 pages. 2005.

Vol. 3692: R. Casadio, G. Myers (Eds.), Algorithms in Bioinformatics. X, 436 pages. 2005. (Subseries LNBI).

Table of Contents

Introduction

Agile Methods and Software Issues

SPI Studies

Improvement Methods

Quality and Knowledge Management

Engineering and Development

Software Process Improvement – EuroSPI 2005 Conference

R. Messnarz, P. Abrahamsson, and I. Richardson

EuroSPI , c/o ISCN LTD, Bray, Co. Wicklow, Ireland
http://www.eurospi.net

Abstract. This book constitutes the refereed research proceeding of the 12th European Software Process Improvement Conference, EuroSPI 2005, held in Budapest, Hungary in November 2005. The 18 revised full papers presented were carefully reviewed and selected from 40 submissions. The papers are organized in topical sections on agile methods, SPI studies, improvement methods, engineering and development, and quality and knowledge concepts.

1 EuroSPI

EuroSPI is a partnership of large Scandinavian research companies and experience networks (SINTEF, DELTA,STTF), the ASQF as a large German quality association, the American Society for Quality, and ISCN as the co-coordinating partner. EuroSPI collaborates with a large number of SPINs (Software Process Improvement Network) in Europe.

EuroSPI conferences present and discuss results from improvement projects in industry and research, focusing on the benefits gained and the criteria for success. Leading European universities, research centers, and industry are contributing to and participating in this event. This year's event is the 12th of a series of conferences to which international researchers contribute their lessons learned and share their knowledge as they work towards the next higher level of software management professionalism.

The biggest value of EuroSPI lies in its function as a European knowledge and experience exchange mechanism where researchers, industrial managers and professionals meet to exchange experiences and ideas and fertilize the grounds for new developments and improvements.

1.1 Board Members

ASQ, http://www.asq.org
ASQF, http://www.asqf.de
DELTA, http://www.delta.dk
ISCN, http://www.iscn.com
SINTEF, http://www.sintef.no
STTF, http://www.sttf.fi

I. Richardson et al. (Eds.): EuroSPI 2005, LNCS 3792, pp. 1 – 3, 2005.

1.2 EuroSPI Scientific Programme Committee

Pekka Abrahamsson, VTT Electronics, Finland
Vincenzo Ambriola, University of Pisa , Italy
Aybuke Aurum, University of New South Wales, Australia
Alexander Beriozko, Russian Academy of Sciences, RU
Stefan Biffl, TU Wien , Austria
Miklos Biro, University of Budapest, Hungary
Christian Bunse, Fraunhofer IESE, Germany
Marcus Ciolkowski, University of Kaiserslautern, Germany
Karl Cox, NICTA, Australia
Kevin C. Desouza, University of Illinois, USA
Howard Duncan, Dublin City University, Ireland
Taz Daughtrey, James MADISON University, USA
Tore Dybå, SINTEF, Norway
Jan Pries-Heje, ITU, Denmark
Sian Hope, University of Wales, Bangor,UK
Karl Heinz Kautz, Copenhagen Business School, Denmark
Dieter Landes, Coburg University of Applied Sciences, Germany
Mikael Lindvall, Fraunhofer Center, USA
Patricia McQuaid, California Polytechnic University, USA
Matthias Mueller, University of Karlsruhe, Germany
Juergen Muench, Fraunhofer IESE, Germany
Joerg Niere, University of Siegen, Germany
Markku Oivo, University of Oulu, Finland
Elixabete Ostolaza, European Software Institute, Spain
Kjetil Molokken-Ostvold , Simula Research Laboratory, Norway
Vincent Ribaud, University of Brest, France
Ita Richardson, University of Limerick, Ireland
Gunther Ruhe, University of Calgary, Canada
Per Runeson, University of Lund, Sweden
Martin Shepperd, Bornemouth University, England
Kerstin Siakas, Technological Education Institute of Thessaloniki, Greece
Kurt Schneider, University of Hannover, Germany
Tor Stålhane, Norwegian University of Science and Technology, Norway
Colin Tully, University of Middlesex, UK
Timo Varkoi, Tampere University of Technology, Finland
Claes Wohlin, Blekinge Institute of Technology, Sweden

1.3 EuroSPI Scientific Chairs

Dr Richard Messnarz
General Chair of EuroSPI
ISCN, Ireland and Austria
rmess@iscn.com

Dr Ita Richardson
EuroSPI Scientific Programme Committee Chair
University of Limerick , Ireland
Ita.Richardson@ul.ie

Dr Pekka Abrahamsson
EuroSPI Scientific Programme Committee Chair
VTT Technical Research Centre of Finland
Pekka.Abrahamsson@vtt.fi

2 How to Read the Proceedings

Since its beginning in 1994 in Dublin, the EuroSPI initiative outlines that there is not a single silver bullet to solve SPI issues but you need to understand a combination of different SPI methods and approaches to achieve real benefits. Therefore each proceeding covers a variety of different topics and at the conference we discuss potential synergies and combined use of such methods and approaches. This proceeding contains selected research papers for 5 topics:

SPI and Agile Methods & Soft Issues (3 papers)
SPI and Improvement Methods (3 papers)
SPI Studies (3 papers)
SPI and Quality & Knowledge Concepts (3 papers)
SPI in Engineering and Development (6 papers)

2.1 Recommended Further Reading

In [1] we integrated the proceedings of 3 EuroSPI conferences into one book which was edited by 30 experts in Europe. In [2] you find the most recent EuroSPI research proceeding published by Springer andbased on EuroSPI 2004.

References

1. Messnarz R., Tully C. (eds.), Better Software Practice for Business Benefit - Principles and Experience, IEEE Computer Society Press, ISBN: 0-7695-0049-8, paperback, 409 pages, Wiley-IEEE Computer Society Press, September 1999
2. Dingsøyr, T. (Ed.) , Software Process Improvement 11th European Conference, EuroSPI 2004, Trondheim, Norway, November 10-12, 2004. Proceedings, 2004, X, 207 p., Softcover, ISBN: 3-540-23725-9, in: Lecture Notes in Computer Science, Vol. 3281 , Springer Verlag, November 2004

Framework of Agile Patterns

Teodora Bozheva and Maria Elisa Gallo

European Software Institute, Parque Tecnológico Edif. 204,
48170 Zamudio (Bizkaia), Spain
{Teodora.Bozheva, MariaElisa.Gallo}@esi.es
http://www.esi.es

Abstract. The variety of agile methods and their similarity could be a problem for software engineers to select a single or a number of methods and to properly execute them in a project. A pattern describes a problem, which typically occurs under certain circumstances and a basic approach to solve it providing opportunities to adapt the solution to the problem. The agile patterns, described herein, are based on the principles and practices of the best known agile methodologies. While individual practices included in any of these methods vary, they all have particular objectives and related to them activities. Therefore, every pattern is described as to show the core solution to a particular problem. Special attention is paid to the rationale for applying the agile patterns: what are the business drivers to adopting them; in what cases do they bring benefits; how could they be introduced in an organization.

1 Introduction

Nowadays lots of organizations face the need to adapt quickly to modifications requested by their customers, changes on the market or challenges from competitors. This happens in small as well as in large organizations, in ones following standard (ISO 9001:2000, CMMI) or their own processes. These business needs force the companies evaluate how the agile methods could address their necessities.

Agile methods recognize that any project, team and organization has its unique peculiarities and respond to the specific needs via business value based prioritization, short feedback cycles and quality-focused development. When appropriately applied the agile practices bring a number of business benefits as better project adaptability and reaction to changes, reduced production costs, improved product quality and increased user satisfaction with the final solution.

The agile methods differ in the approaches to software development and management they propose. Some focus more heavily on project management and collaboration practices. These include Adaptive Software Development (ASD) [6], Scrum [7], Lean Development (LD) [10] and DSDM [8]. Others, such as eXtreme Programming (XP) [3], Feature-driven Development (FDD) [9] and Agile Modeling (AM) [5], focus more extensively on software implementation practices. Nevertheless, all the methods stick to the principles of maintaining good understanding of the project

I. Richardson et al. (Eds.): EuroSPI 2005, LNCS 3792, pp. 4 – 15, 2005.

objectives, scope and constraints, developing software in short, feature-drive iterations, receiving constant feedback from the customer and the developers, and focusing on the delivery of business value.

An important issue in defining an organizational process is that all the elements it consists of reflect properly the specifics of the environment, in which the process will be implemented. When selecting an agile method, the business and organizational context, in which it will be applied, determines the benefits that could be achieved for a project and for an organization. In their book [1] B. Boehm and R. Turner have defined the "home grounds" in which agile and disciplined methods are most successful. Additionally, they define five factors, which help the organization determine whether they are in either the agile or disciplined area, or somewhere in between. These are size, criticality, personnel, dynamism, and culture.

Our work on applying agile practices in different organizational contexts inspired the development of the framework of agile patterns, presented in this paper. The idea is instead of providing a complete method, which might not be fully applicable in any situation, to provide a set of patterns addressing different aspects of the software development process, which could be combined in such a way as to fit to the peculiarity of a project. The patterns are derived from the most widely known lightweight methods XP, Scrum, FDD, AM, LD, and ASD.

Implementing a software development process based on patterns has several advantages:

- The patterns address activities performed by software engineers and project managers who are accustomed to using well structured information like pattern definitions.
- Patterns describe individual practices in a general enough way to be applied in different situations. Therefore they can be easily tried out and included in definitions of new processes. Adopting a small set of new practices gives a more profound understanding of the practices themselves and of the benefits from applying them together, which facilitates the continuous process improvement.
- As each pattern is selected and adapted as to best fit a project and organizational context, the whole process will be more suitable for that context than any general one.

We describe the framework of agile patterns in section 2 and in section 3 we present lessons learnt from applying the patterns in the industry.

2 Framework of Agile Patterns

A pattern describes a problem, which typically occurs under certain circumstances. It also describes a basic approach to solve the problem providing opportunities to adapt the solution to the particular problem context. In general, a pattern has three essential elements: problem, solution and consequences. Each solution consists of activities that, when collectively applied, resolve the problem. The solution is abstract enough to make it possible to apply it in different situations. The consequences are results and trade-offs of applying the pattern.

Three key terms take part in the agile methods: *practices*, *concepts* and *principles*. *Practices* describe specific actions that are performed in the whole process of software development, e.g. create product backlog (SCRUM). *Concepts* describe the attributes of an item, e.g. a project plan. *Principles* are fundamental guidelines concerning software development activities, e.g. empower the team (LD).

To be coherent with the agile methodologies the framework of agile patterns (FAP) includes definitions of three types of patterns: practice patterns, concepts and principles.

2.1 Practice Patterns

In the FAP each agile pattern is described by means of the following attributes:

- **Intent:** a short description of what the objective is;
- **Origin:** methodologies, from which the pattern originates;
- **Category** to which the pattern belongs. With respect to the type of issues addressed, the patterns are grouped in the following categories: *Project and Requirements Management, Design, Implementation and Testing, Resource Management, Contract Management* and *Software Process Improvement.*
- **Application scenario:** context, in which the pattern is to be applied;
- **Roles:** people involved in carrying out the pattern and their responsibilities;
- Main and alternative **Activities** that constitute the pattern. Activities can invoke other patterns;
- **Tools** that support the pattern execution;
- **Guidelines** for performing the activities including suggestions for making a decision about which alternative solution to choose when.

This structure is closest to the one proposed by E. Gamma in [2]. Compared to the classic pattern definition (problem-solution-consequences), Intent and Application scenario correspond to the problem attribute. Activities matches to solution. Some patterns provide alternative solutions to the same problem. This typically happens when the problem is addressed by more than one agile method and different solutions to it are proposed. Guidelines include hints for performing the activities and the consequences from them. An example of a pattern is:

CodeIntegrator

Intent: To have working code at the end of every day. In a software development environment with collective code ownership, the idea is to build the system every day, after a very small batch of work has been done by each of the developers.

Origin: LD: Synch and Stabilise (Daily build and smoke test); XP: Integrate often

***Category*:** Implementation & Testing

***Application scenario*:** After implementing a piece of code

***Roles*:** Developers

Activities

- Check out source code from the configuration management system.
- Put together the newly implemented and the existing code.

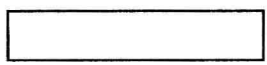
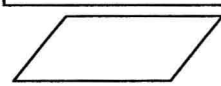
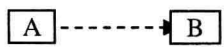
- Check to see, if anyone else has made changes to the same code, and if so, resolve the conflicts by applying CodeImplementer.
- Apply AcceptanceTester.
- Check in the new code.

***Tools*:** Version control tools support this activity.
Guidelines:

- Continuous integration avoids or detects compatibility problems early. If changes are integrated in small batches, it will be infinitely easier to detect and fix problems.
- A single integration point (computer) has to be defined.
- Every developer is responsible for integrating his/her own code always when there is a reasonable break. This could be when all the unit tests run at 100% or some smaller portion of the planned functionality is finished. Only one developer integrates at a given moment and after only a few hours of coding.
- All the tests have to pass successfully after integrating the system. Each integration results in a running system. Integration happens every 1-5 hours, at least once a day.

Apart from the natural language description of the patterns, every category is graphically illustrated showing which patterns, concepts and principles it includes, and the relationships between them.

On the graphics the following symbols are used:

Symbol	Meaning
	Principle
	Pattern
	Concept
	*"invokes"*
	Pattern A supports B, but it is optional to use A when implementing B

As an example, let's consider the *Implementation and Testing* category (Fig.1).

The patterns and their intents, which belong to this category, are the following ones.

Code Implementer: Implement code
FDDCoder: Implement defect-free code following FDD
XPCoder: Implement defect-free code following XP
SoftwareInspector; Find out defects in project work products