Qing Wang
Dietmar Pfahl
David M. Raffo (Eds.)

LNCS 4470

Software Process Dynamics and Agility

International Conference on Software Process, ICSP 2007
Minneapolis, MN, USA, May 2007
Proceedings

Springer

Qing Wang Dietmar Pfahl
David M. Raffo (Eds.)

Software Process Dynamics and Agility

International Conference on Software Process, ICSP 2007
Minneapolis, MN, USA, May 19-20, 2007
Proceedings

Springer

E2007003217

Volume Editors

Qing Wang
Chinese Academy of Science, Institute of Software
No. 4 South Fourth Street, Zhong Guan Cun, Beijing 100080,China
E-mail: wq@itechs.iscas.ac.cn

Dietmar Pfahl
University of Calgary, Schulich School of Engineering
Department of Computer Science & Electrical Engineering
2500 University Drive N.W., Calgary, Alberta T2N 1N4, Canada
E-mail: dpfahl@ucalgary.ca

David M. Raffo
Portland State University, School of Business Administration
P.O. Box 8491, Portland, OR 97207, USA
E-mail: raffod@pdx.edu

Library of Congress Control Number: 2007925873

CR Subject Classification (1998): D.2, K.6.3, K.6, K.4.3, J.1

LNCS Sublibrary: SL 2 – Programming and Software Engineering

ISSN 0302-9743
ISBN-10 3-540-72425-7 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-72425-4 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12062041 06/3180 5 4 3 2 1 0

Preface

This volume contains papers presented at the International Conference on Software Process (ICSP 2007) held in Minneapolis, USA, May 19-20, 2007. ICSP 2007 comprised two successful series of process-related workshops, the International Workshop on Software Process Simulation and Modeling (ProSim) and the Software Process Workshop (SPW).

The theme of ICSP 2007 was "Coping with Software Process Dynamics and Agility." Software developers work in a dynamic context of frequently changing technologies and limited resources. Globally distributed development teams are under ever-increasing pressure to deliver their products more quickly and with higher levels of quality. At the same time, global competition is forcing software development organizations to cut costs by rationalizing processes, outsourcing part or all of their activities, reusing existing software in new or modified applications and evolving existing systems to meet new needs, while still minimizing the risk of projects failing to deliver. To address these difficulties, new or modified processes are emerging, including agile methods and plan-based product line development. Open source, COTS and community-developed software are becoming more popular. Outsourcing coupled with 24/7 development demand well-defined processes to support the coordination of organizationally and geographically separated teams.

The increasing challenges faced by the software industry combine to increase demands on software processes.

ICSP 2007 was a continuation of two successful series of process-related workshops, ProSim (Software Process Simulation and Modeling Workshop) and SPW (Software Process Workshop). SPW and ProSim were conducted jointly for the first time in 2006 as a co-located event to ICSE 2006. ICSP 2007 continued a long tradition of software process research, positioning itself as the new leading-edge event for systems and software process research.

In response to the call for papers, 98 submissions were received from 14 different countries and regions: Australia, Brazil, Canada, China, France, Germany, Japan, Korea, The Netherlands, Pakistan, Spain, UK, USA, and Turkey. Every paper was rigorously reviewed and held to very high quality standards, and finally 28 papers were accepted as regular papers for presentation at the conference.

The papers were clustered around topics and presented in five regular sessions, each consisting of two threads. Topics included *Process Content, Process Tools and Metrics, Process Management, Process Representation, Analysis and Modeling, Experience Report,* and *Simulation Modeling.*

Highlights of the ICSP2007 program were two keynote speeches, delivered by Larry E. Druffel (President and CEO, SCRA, USA) and Merwan Mehta (Department of Technology Systems, East Carolina University, USA).

A conference such as this can only succeed as a team effort. All of this work would not have been possible without the dedication and professional work of many colleagues. We wish to express our gratitude to all contributors for submitting papers. Their work formed the basis for the success of the conference. We would also like to

thank the Program Committee members and reviewers because their work guaranteed the high quality of the workshop. Particular thanks also go to the keynote speakers for giving their excellent presentations at the conference. Finally, we would also like to thank the members of the Steering Committee, Barry Boehm, Mingshu Li, Leon Osterweil and Wilhelm Schäfer, for their advice, encouragement and support.

We wish to express our thanks to the organizers for their hard work. The conference was sponsored by the International Software Process Association (ISPA) and the Institute of Software, the Chinese Academy of Sciences (ISCAS) and the ISCAS Laboratory for Internet Software Technologies. We also wish to thank the 29th International Conference on Software Engineering (ICSE 2007) for sponsoring this meeting as an ICSE Co-located Event. Finally, we acknowledge the editorial support from Springer for the publication of this volume.

For further information, please visit our Web site at http://www.icsp-conferences.org/icsp2007.

March 2007

David M. Raffo
Qing Wang
Dietmar Pfahl

International Conference on Software Process 2007

Minneapolis, USA
May 19–20, 2007

General Chair

David M. Raffo, Portland State University, USA

Steering Committee

Barry Boehm, University of Southern California, USA
Mingshu Li, Institute of Software, Chinese Academy of Sciences, China
Leon J. Osterweil, University of Massachusetts, USA
Wihelm Schäfer, University of Paderborn, Germany

Program Co-chairs

Dietmar Pfahl, University of Calgary, Canada
Qing Wang, Institute of Software, Chinese Academy of Sciences, China

Program Committee Members

Stefan Biffl	Technische Universität Wien, Austria
Thomas Birkhölzer	University of Applied Science, Konstanz, Germany
Keith Chan	Hong Kong Polytechnic University, Hong Kong, China
Sorana Cimpan	University of Savoie at Annecy, France
Jacky Estublier	French National Research Center in Grenoble, France
Anthony Finkelstein	University College London, UK
Dennis Goldenson	Carnegie Mellon University, USA
Volker Gruhn	University of Leipzig, Germany
Paul Grünbacher	Johannes Kepler University Linz, Austria
Dan Houston	Honeywell, USA
LiGuo Huang	University of Southern California, USA
Hajimu Iida	Nara Institute of Science and Technology, Japan
Katsuro Inoue	Osaka University, Japan
Ross Jeffery	University of New South Wales, Australia
Natalia Juristo	Universidad Politécnica de Madrid, Spain
Rick Kazman	University of Hawaii, USA
Jyrki Kontio	Helsinki University of Technology, Finland

Jian Lv	Nanjing University, China
Ray Madachy	University of Southern California, USA
Frank Maurer	University of Calgary, Canada
Hong Mei	Peking University, China
Jürgen Münch	University of Kaiserslautern, Germany
Flavio Oquendo	University of South Brittany, France
Dewayne E. Perry	University of Texas at Austin, USA
Dietmar Pfahl	University of Calgary, Canada
Dan Port	University of Hawaii, USA
Antony Powell	Science Applications International Corporation, USA
David M. Raffo	Portland State University, USA
Juan F. Ramil	The Open University, UK
Andreas Rausch	Technische Universität Kaiserslautern, Germany
Günther Ruhe	University of Calgary, Canada
Mercedes Ruiz	University of Cádiz, Spain
Ioana Rus	Fraunhofer Center, USA
Kevin Ryan	University of Limerick, Ireland
Walt Scacchi	University of California, Irvine, USA
Barbara Staudt Lerner	Mt. Holyoke College, USA
Stan Sutton	IBM T. J. Watson Research Center, USA
Colin Tully	Middlesex University, UK
Qing Wang	Chinese Academy of Sciences, China
Yongji Wang	Chinese Academy of Sciences, China
Brian Warboys	University of Manchester, UK
Paul Wernick	University of Hertfordshire, UK
Laurie Williams	North Carolina State University, USA
Ye Yang	University of Southern California, USA
Yun Yang	Swinburne University of Technology, Australia

External Reviewers

Ahmed Al-Emran	University of Calgary, Canada
Marta Lopez	Universidad Complutense de Madrid, Spain
Alicia Mon	Universidad Nacional de la Matanza, Argentina
Ricardo Imbert	Universidad Politecnica de Madrid, Spain
Anna Cecilia Griman	Universidad Simón Bolivar, Venezuela
Oscar Dieste	Universidad Politecnica de Madrid, Spain
Carmen Zannier	University of Calgary, Canada

Lecture Notes in Computer Science 4470

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

Lecture Notes in Computer Science

For information about Vols. 1–4362

please contact your bookseller or Springer

Vol. 4483: C. Baral, G. Brewka, J. Schlipf (Eds.), Logic Programming and Nonmonotonic Reasoning. IX, 327 pages. 2007. (Sublibrary LNAI).

Vol. 4480: A. LaMarca, M. Langheinrich, K.N. Truong (Eds.), Pervasive Computing. XIII, 369 pages. 2007.

Vol. 4470: Q. Wang, D. Pfahl, D.M. Raffo (Eds.), Software Process Dynamics and Agility. XI, 346 pages. 2007.

Vol. 4464: E. Dawson, D.S. Wong (Eds.), Information Security Practice and Experience. XIII, 361 pages. 2007.

Vol. 4463: I. Măndoiu, A. Zelikovsky (Eds.), Bioinformatics Research and Applications. XV, 653 pages. 2007. (Sublibrary LNBI).

Vol. 4462: D. Sauveron, K. Markantonakis, A. Bilas, J.-J. Quisquater (Eds.), Information Security Theory and Practices. XII, 255 pages. 2007.

Vol. 4459: C. Cérin, K.-C. Li (Eds.), Advances in Grid and Pervasive Computing. XVI, 759 pages. 2007.

Vol. 4453: T. Speed, H. Huang (Eds.), Research in Computational Molecular Biology. XVI, 550 pages. 2007. (Sublibrary LNBI).

Vol. 4450: T. Okamoto, X. Wang (Eds.), Public Key Cryptography – PKC 2007. XIII, 491 pages. 2007.

Vol. 4448: M. Giacobini et al. (Ed.), Applications of Evolutionary Computing. XXIII, 755 pages. 2007.

Vol. 4447: E. Marchiori, J.H. Moore, J.C. Rajapakse (Eds.), Evolutionary Computation,Machine Learning and Data Mining in Bioinformatics. XI, 302 pages. 2007.

Vol. 4446: C. Cotta, J. van Hemert (Eds.), Evolutionary Computation in Combinatorial Optimization. XII, 241 pages. 2007.

Vol. 4445: M. Ebner, M. O'Neill, A. Ekárt, L. Vanneschi, A.I. Esparcia-Alcázar (Eds.), Genetic Programming. XI, 382 pages. 2007.

Vol. 4444: T. Reps, M. Sagiv, J. Bauer (Eds.), Program Analysis and Compilation, Theory and Practice. X, 361 pages. 2007.

Vol. 4443: R. Kotagiri, P.R. Krishna, M. Mohania, E. Nantajeewarawat (Eds.), Advances in Databases: Concepts, Systems and Applications. XXI, 1126 pages. 2007.

Vol. 4440: B. Liblit, Cooperative Bug Isolation. XV, 101 pages. 2007.

Vol. 4439: W. Abramowicz (Ed.), Business Information Systems. XV, 654 pages. 2007.

Vol. 4438: L. Maicher, A. Sigel, L.M. Garshol (Eds.), Leveraging the Semantics of Topic Maps. X, 257 pages. 2007. (Sublibrary LNAI).

Vol. 4433: E. Şahin, W.M. Spears, A.F.T. Winfield (Eds.), Swarm Robotics. XII, 221 pages. 2007.

Vol. 4432: B. Beliczynski, A. Dzielinski, M. Iwanowski, B. Ribeiro (Eds.), Adaptive and Natural Computing Algorithms, Part II. XXVI, 761 pages. 2007.

Vol. 4431: B. Beliczynski, A. Dzielinski, M. Iwanowski, B. Ribeiro (Eds.), Adaptive and Natural Computing Algorithms, Part I. XXV, 851 pages. 2007.

Vol. 4430: C.C. Yang, D. Zeng, M. Chau, K. Chang, Q. Yang, X. Cheng, J. Wang, F.-Y. Wang, H. Chen (Eds.), Intelligence and Security Informatics. XII, 330 pages. 2007.

Vol. 4429: R. Lu, J.H. Siekmann, C. Ullrich (Eds.), Cognitive Systems. X, 161 pages. 2007. (Sublibrary LNAI).

Vol. 4427: S. Uhlig, K. Papagiannaki, O. Bonaventure (Eds.), Passive and Active Network Measurement. XI, 274 pages. 2007.

Vol. 4426: Z.-H. Zhou, H. Li, Q. Yang (Eds.), Advances in Knowledge Discovery and Data Mining. XXV, 1161 pages. 2007. (Sublibrary LNAI).

Vol. 4425: G. Amati, C. Carpineto, G. Romano (Eds.), Advances in Information Retrieval. XIX, 759 pages. 2007.

Vol. 4424: O. Grumberg, M. Huth (Eds.), Tools and Algorithms for the Construction and Analysis of Systems. XX, 738 pages. 2007.

Vol. 4423: H. Seidl (Ed.), Foundations of Software Science and Computational Structures. XVI, 379 pages. 2007.

Vol. 4422: M.B. Dwyer, A. Lopes (Eds.), Fundamental Approaches to Software Engineering. XV, 440 pages. 2007.

Vol. 4421: R. De Nicola (Ed.), Programming Languages and Systems. XVII, 538 pages. 2007.

Vol. 4420: S. Krishnamurthi, M. Odersky (Eds.), Compiler Construction. XIV, 233 pages. 2007.

Vol. 4419: P.C. Diniz, E. Marques, K. Bertels, M.M. Fernandes, J.M.P. Cardoso (Eds.), Reconfigurable Computing: Architectures, Tools and Applications. XIV, 391 pages. 2007.

Vol. 4418: A. Gagalowicz, W. Philips (Eds.), Computer Vision/Computer Graphics Collaboration Techniques. XV, 620 pages. 2007.

Vol. 4416: A. Bemporad, A. Bicchi, G. Buttazzo (Eds.), Hybrid Systems: Computation and Control. XVII, 797 pages. 2007.

Vol. 4415: P. Lukowicz, L. Thiele, G. Tröster (Eds.), Architecture of Computing Systems - ARCS 2007. X, 297 pages. 2007.

Vol. 4414: S. Hochreiter, R. Wagner (Eds.), Bioinformatics Research and Development. XVI, 482 pages. 2007. (Sublibrary LNBI).

Vol. 4412: F. Stajano, H.J. Kim, J.-S. Chae, S.-D. Kim (Eds.), Ubiquitous Convergence Technology. XI, 302 pages. 2007.

Vol. 4411: R.H. Bordini, M. Dastani, J. Dix, A.E.F. Seghrouchni (Eds.), Programming Multi-Agent Systems. XIV, 249 pages. 2007. (Sublibrary LNAI).

Vol. 4410: A. Branco (Ed.), Anaphora: Analysis, Algorithms and Applications. X, 191 pages. 2007. (Sublibrary LNAI).

Vol. 4409: J.L. Fiadeiro, P.-Y. Schobbens (Eds.), Recent Trends in Algebraic Development Techniques. VII, 171 pages. 2007.

Vol. 4407: G. Puebla (Ed.), Logic-Based Program Synthesis and Transformation. VIII, 237 pages. 2007.

Vol. 4406: W. De Meuter (Ed.), Advances in Smalltalk. VII, 157 pages. 2007.

Vol. 4405: L. Padgham, F. Zambonelli (Eds.), Agent-Oriented Software Engineering VII. XII, 225 pages. 2007.

Vol. 4403: S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, T. Murata (Eds.), Evolutionary Multi-Criterion Optimization. XIX, 954 pages. 2007.

Vol. 4401: N. Guelfi, D. Buchs (Eds.), Rapid Integration of Software Engineering Techniques. IX, 177 pages. 2007.

Vol. 4400: J.F. Peters, A. Skowron, V.W. Marek, E. Orłowska, R. Słowiński, W. Ziarko (Eds.), Transactions on Rough Sets VII, Part II. X, 381 pages. 2007.

Vol. 4399: T. Kovacs, X. Llorà, K. Takadama, P.L. Lanzi, W. Stolzmann, S.W. Wilson (Eds.), Learning Classifier Systems. XII, 345 pages. 2007. (Sublibrary LNAI).

Vol. 4398: S. Marchand-Maillet, E. Bruno, A. Nürnberger, M. Detyniecki (Eds.), Adaptive Multimedia Retrieval: User, Context, and Feedback. XI, 269 pages. 2007.

Vol. 4397: C. Stephanidis, M. Pieper (Eds.), Universal Access in Ambient Intelligence Environments. XV, 467 pages. 2007.

Vol. 4396: J. García-Vidal, L. Cerdà-Alabern (Eds.), Wireless Systems and Mobility in Next Generation Internet. IX, 271 pages. 2007.

Vol. 4395: M. Daydé, J.M.L.M. Palma, Á.L.G.A. Coutinho, E. Pacitti, J.C. Lopes (Eds.), High Performance Computing for Computational Science - VECPAR 2006. XXIV, 721 pages. 2007.

Vol. 4394: A. Gelbukh (Ed.), Computational Linguistics and Intelligent Text Processing. XVI, 648 pages. 2007.

Vol. 4393: W. Thomas, P. Weil (Eds.), STACS 2007. XVIII, 708 pages. 2007.

Vol. 4392: S.P. Vadhan (Ed.), Theory of Cryptography. XI, 595 pages. 2007.

Vol. 4391: Y. Stylianou, M. Faundez-Zanuy, A. Esposito (Eds.), Progress in Nonlinear Speech Processing. XII, 269 pages. 2007.

Vol. 4390: S.O. Kuznetsov, S. Schmidt (Eds.), Formal Concept Analysis. X, 329 pages. 2007. (Sublibrary LNAI).

Vol. 4389: D. Weyns, H.V.D. Parunak, F. Michel (Eds.), Environments for Multi-Agent Systems III. X, 273 pages. 2007. (Sublibrary LNAI).

Vol. 4385: K. Coninx, K. Luyten, K.A. Schneider (Eds.), Task Models and Diagrams for Users Interface Design. XI, 355 pages. 2007.

Vol. 4384: T. Washio, K. Satoh, H. Takeda, A. Inokuchi (Eds.), New Frontiers in Artificial Intelligence. IX, 401 pages. 2007. (Sublibrary LNAI).

Vol. 4383: E. Bin, A. Ziv, S. Ur (Eds.), Hardware and Software, Verification and Testing. XII, 235 pages. 2007.

Vol. 4381: J. Akiyama, W.Y.C. Chen, M. Kano, X. Li, Q. Yu (Eds.), Discrete Geometry, Combinatorics and Graph Theory. XI, 289 pages. 2007.

Vol. 4380: S. Spaccapietra, P. Atzeni, F. Fages, M.-S. Hacid, M. Kifer, J. Mylopoulos, B. Pernici, P. Shvaiko, J. Trujillo, I. Zaihrayeu (Eds.), Journal on Data Semantics VIII. XV, 219 pages. 2007.

Vol. 4379: M. Südholt, C. Consel (Eds.), Object-Oriented Technology. VIII, 157 pages. 2007.

Vol. 4378: I. Virbitskaite, A. Voronkov (Eds.), Perspectives of Systems Informatics. XIV, 496 pages. 2007.

Vol. 4377: M. Abe (Ed.), Topics in Cryptology – CT-RSA 2007. XI, 403 pages. 2006.

Vol. 4376: E. Frachtenberg, U. Schwiegelshohn (Eds.), Job Scheduling Strategies for Parallel Processing. VII, 257 pages. 2007.

Vol. 4374: J.F. Peters, A. Skowron, I. Düntsch, J. Grzymała-Busse, E. Orłowska, L. Polkowski (Eds.), Transactions on Rough Sets VI, Part I. XII, 499 pages. 2007.

Vol. 4373: K. Langendoen, T. Voigt (Eds.), Wireless Sensor Networks. XIII, 358 pages. 2007.

Vol. 4372: M. Kaufmann, D. Wagner (Eds.), Graph Drawing. XIV, 454 pages. 2007.

Vol. 4371: K. Inoue, K. Satoh, F. Toni (Eds.), Computational Logic in Multi-Agent Systems. X, 315 pages. 2007. (Sublibrary LNAI).

Vol. 4370: P.P Lévy, B. Le Grand, F. Poulet, M. Soto, L. Darago, L. Toubiana, J.-F. Vibert (Eds.), Pixelization Paradigm. XV, 279 pages. 2007.

Vol. 4369: M. Umeda, A. Wolf, O. Bartenstein, U. Geske, D. Seipel, O. Takata (Eds.), Declarative Programming for Knowledge Management. X, 229 pages. 2006. (Sublibrary LNAI).

Vol. 4368: T. Erlebach, C. Kaklamanis (Eds.), Approximation and Online Algorithms. X, 345 pages. 2007.

Vol. 4367: K. De Bosschere, D. Kaeli, P. Stenström, D. Whalley, T. Ungerer (Eds.), High Performance Embedded Architectures and Compilers. XI, 307 pages. 2007.

Vol. 4366: K. Tuyls, R. Westra, Y. Saeys, A. Nowé (Eds.), Knowledge Discovery and Emergent Complexity in Bioinformatics. IX, 183 pages. 2007. (Sublibrary LNBI).

Vol. 4364: T. Kühne (Ed.), Models in Software Engineering. XI, 332 pages. 2007.

¥562.00元

Table of Contents

Process Content

Process Tools and Metrics

Process Management

Process Representation, Analysis and Modeling

Experience Report

Simulation Modeling

Extending Microsoft Team Foundation Server Architecture to Support Collaborative Product Patterns

Fuensanta Medina-Domínguez, Maria-Isabel Sanchez-Segura, Antonio Amescua, and Javier García

Computer Science Department, Carlos III Technical University of Madrid
Avda. Universidad, 30, Leganes 28911, Madrid, Spain
{fmedina, misanche, amescua, jgarciag}@inf.uc3m.es

Abstract. This paper provides a practical solution, based on process reuse and knowledge management techniques, to make software engineering theories more accessible, easier, and cheaper for software development organizations to implement. It shows how the PIBOK-PB architecture (Process improvement based on knowledge-pattern based) and the extensions of a commercial product, Microsoft solution Visual Studio Team System, are used to achieve this.

Keywords: Software Engineering, Process Management, Reuse, Patterns.

1 Introduction

Software engineering provides enough formalisms to guarantee the execution of a software project. However, if we look at the data on software projects, we will observe that, in 1995, on average, only around 20% of software projects were completed on time and within the budget [1]. What happened to the remaining 80%? These percentages have changed little since, and many projects still fail to comply with the triple constraints of scope, time and cost [2]. Poor project management and insufficient use of software engineering techniques are some of the reasons for this non-compliance.

This data can help us to understand that although the theories in the software engineering field are sufficiently matured and are widely known, it is the implementation of software engineering best practices that helps organizations improve their productivity, software quality, and reduce costs [3]. But it is very difficult to implement them because these organizations must first know the theory, and how to implement them successfully [4] [5] [6], which is not gathered in the literature. Evidence also reveals that the implementation of software processes in software development organizations is a complex and expensive process, mainly for small and medium enterprises [7]. We have focused on bridging the gap between theory and practice in software development processes and best practices to make them accessible, and thus beneficial, for small and medium enterprises.

The experience of the American Federal Aviation Administration (FAA) indicates that knowledge management combines positively with process improvement, benefits the organization and the process improvement programmes [8]. Some papers, like the one published in [9], suggest that technology, structure, culture, knowledge process

Q. Wang, D. Pfahl, and D.M. Raffo (Eds.): ICSP 2007, LNCS 4470, pp. 1–11, 2007.

architecture of acquisition, conversion, application, and protection are essential organizational capabilities or "preconditions" for effective knowledge management.

We believe that knowledge management can be applied to software engineering to transform software engineering data and information, described as process models, standards, methodologies, etc., to knowledge and innovation. This is possible once the knowledge of experts on process model, standards, methodologies, etc., is elicited and translated into a computable model. Therefore, a software system can make use of this knowledge in order to reduce the cost of process definitions and hasten the maturity of the processes.

Data and information must be encapsulated to allow their subsequent recovery and reuse, and their evolution into knowledge. The artefact to encapsulate this knowledge is the pattern concept. Patterns are an established and well-known format to capture engineering knowledge [10], but the real power of patterns to enable knowledge transfer for practical use is still under development due to the limited number of approaches that endow patterns with practical implementation. The authors proposed the *product pattern* concept as well as a model called PIBOK-PB to support process improvement based on patterns [11]. In this paper, we describe the architecture that supports the PIBOK-PB model as well as the Microsoft VSTS extensions to provide a collaborative practical solution based on process reuse and knowledge management techniques.

We have no evidence of research groups working on combining software engineering, patterns, and knowledge management techniques supported by collaborative working environments. The US Federal Aviation Administration (FAA) and the SINTEF group (Foundation for Scientific and Industrial Research at the Norwegian Institute of Technology) are currently working on process improvement based on knowledge management, but their solutions are not supported by collaborative working environments.

The remainder of this paper is structured as follows: section 2 describes PIBOK-PB model architecture; section 3 explains the extensions done to Microsoft VSTS to satisfy the PIBOK-PB architecture requirements; section 4 summarises related work and finally section 5 presents the conclusions and future trends.

2 PIBOK-PB Architecture Description

The PIBOK-PB model (Process Improvement Based On Knowledge – Pattern Based Model) [11] is a knowledge-based software process improvement model that would allow the use of *product patterns* as the artefact to encapsulate the knowledge for use in the development of activities and tasks of the process model chosen.

PIBOK-PB model [11] architecture is made up of four layers, see Fig. 1. These layers are described below:

- Organization layer: this layer processes the data and characteristics of the organization and their business processes. It is responsible for obtaining the data of the organization, for example, the kind and size of the organization, field of their business processes. In summary, this layer is responsible for gathering data, provided by the project manager, related to the forces of the organization.

This layer supports the appropriate selection of process models, methodologies, etc, for each organization and project, depending on their features.

- Project instantiation layer: this layer is responsible for obtaining information related to the project to be developed, for example, the kind of the project, the paradigm chosen to implement the project, the number of employees and their roles in this specific project. These data establish the context and forces of the project to be developed.

  This layer allows the instantiation of process models, methodologies, etc., in a specific project, so it can be customized.
- Patterns instantiation layer: this layer is responsible for the instantiation of the product patterns that best fit the activities included in the process model template selected. The product pattern is an artefact which contains the expert's knowledge to obtain a specific software product. The product pattern concept is described in more detail later.

  This layer allows the recovery of knowledge to be reused in the specific project under development.
- Collaborative layer: this layer provides a collaborative platform which contributes to the functionalities and advantages of collaborative environments. It also provides collaborative functionalities when they are demanded by the roles involved in the execution of a product pattern or when collaboration among patterns is required.

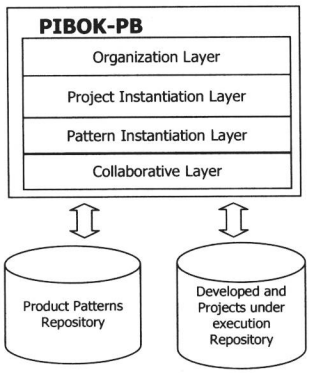

Fig. 1. PIBOK-PB model architecture

The four layers interact with two repositories:

- A repository of product patterns.
- A repository with the information of previously developed projects and projects in progress.

The interactions among layers are described below:

- The Organization and Project instantiation layers provide the context and forces of the organization and project the patterns instantiation layer. Both

layers store these data in the repository that contains the information on developed and current projects. In this way, the patterns instantiation layer will have enough information to execute the next rule, and find the potential product patterns to be applied in the activity and/or tasks. Only the problem field is necessary and this is obtained from the task or activity to be developed in the process model. When all the data are ready, the patterns instantiation layer can execute a rule similar to the following:

If you find yourself in this **context**
(and) with this **problem**
(and) entailing these **forces**
then
map a product pattern in your project
(and) look for more product patterns

- The Collaborative layer interacts with the patterns instantiation layer to provide the collaborative functionalities in order to develop the tasks and/or activities through product patterns. The collaborative layer also provides the patterns instantiation layer with the existing collaboration among product patterns.

In order to understand the above rule, the product pattern concept and its fields are described in detail. Product patterns come from the Alexandrian patterns [12] and are artefacts that gather the knowledge of software engineering experts to obtain a specific software product. Product is defined as anything produced during the whole software development process (for example, Effort and Duration Estimation Using COCOMO II. Available at: http://sel.inf.uc3m.es/C3/Journals/ProductPattern/ProductPattern.pdf. This concept was presented in [11]. The fields of product pattern have since been extended as follows:

- **Name:** the name can be a word or short phrase related to the product pattern.
- **Related Pattern:** static or dynamic relationships between this pattern and others.
- **Initial Context:** the pre-conditions under which the pattern is applicable - a description of the initial state before the pattern is applied.
- **Result Context:** state or configuration of the system after the pattern has been applied. Describing the positive and negative effects.
- **Problem:** description of the problem to be solved.
- **Forces:** restrictions that are classified as follows:
  - Kind of organization
  - Kind of system to be developed
  - Kind of client
  - Rationale
- **Solution:** static relationships and dynamic rules to describe how to achieve the objective. The solution consists of:
  - Process
  - Development time
  - Activities diagram
  - Adjustment degree (very low, low, normal, high very high)