Recent Developments in Product-Focused Software Process Improvement - PROFES 2013 Conference Report

Daniel Graziotin
Free University of Bozen-Bolzano
Piazza Domenicani 3, Bolzano/Bozen, Italy
+39 0471 016257
daniel.graziotin@unibz.it

Andreas Jedlitschka
Fraunhofer Institute for Experimental Software Engineering
Fraunhofer-Platz 1, Kaiserslautern, Germany
+49 631 6800 2260
andreas.jedlitschka@iese.fraunhofer.de

ABSTRACT

This report summarizes the presentations and discussions that happened at PROFES 2013, the 14th International Conference on Product-Focused Software Process Improvement, which was held June 12-14, 2013 in Paphos, Cyprus. The main theme of PROFES is software process improvement (SPI) motivated by product, process, and service quality needs. PROFES 2013 addressed both quality engineering and management topics, divided into the areas of Decision Support in Software Engineering, Empirical Software Engineering, Managing Software Processes, Safety-Critical Software Engineering, Software Measurement, Software Process Improvement, and Software Maintenance.

Categories and Subject Descriptors

D.2.1 [**Software Engineering**]: Requirements/Specification; D.2.4 [**Software Engineering**]: Software/Program Verification; D.2.8 [**Software Engineering**]: Metrics; D.2.9 [**Software Engineering**]: Management; H.1.2 [**Models and Principles**]: User/Machine Systems; H.4.2 [**Information Systems Applications**]: Types of Systems;

Keywords

Management, Measurement, Performance, Experimentation, Security, Human Factors, Standardization, Simulation, Modeling.

1. INTRODUCTION

The 14th International Conference on Product-Focused Software Process Improvement (PROFES) [17] was held June 13-14, 2013 in Paphos, Cyprus. PROFES is a forum for practitioners, researchers, and educators to present and discuss experiences, ideas, innovations, as well as concerns related to software process improvement (SPI) motivated by product, process and service quality needs. The pre-conference program on June 12, 2013 consisted of two tutorials, one given by Hermann Kaindl from Vienna University of Technology on “Model-based Transition from Requirements to High-level Software Design” [23] and the other one given by Jens Heidrich from Fraunhofer IESE on “Software Effort Estimation and Risk Management” [16]. The main conference PROFES 2013 addressed both quality engineering and management topics, presented in three keynotes, 22 full papers, and 10 short papers divided into the areas of Decision Support in Software Engineering, Empirical Software Engineering, Managing Software Processes, Safety-Critical Software Engineering, Software Measurement, Software Process Improvement, and Software Maintenance. The keynotes are summarized in Section 2. The regular papers are presented in Section 3 of this report. The short papers follow in Section 4.

2. KEYNOTES

The 2013 research-oriented keynote was delivered by Stefan Wagner from the University of Stuttgart, Germany [36]. He discussed current techniques and tools that support making software quality visible to the developers while they are working on their tasks. Quick and instant feedback enables developers to better control the quality of their products. To be able to interpret the given information in a comprehensible and repeatable way, Wagner emphasized the use of quality models and presented examples from the QUAMOCO research project [7]. For future research, the keynoter proposed integrating different techniques and results and incorporating knowledge obtained from data mining in repositories. The role of the developer becomes more crucial when it comes to creating high-quality, dependable software on the one hand and mastering complexity on the other hand.

The first industry keynote was given by Alexis Ocampo [32], who presented the ECO-MAPS approach designed by ECOPETROL. He emphasized the importance of identifying “information that adds value to the business processes and mitigates relevant risks.” In his talk, the keynoter presented the approach taken by ECOPETROL to move from “technology first, information last” to “high-quality information first, technology last”, the measurement of information quality, and the results.

The second industry keynote was given by Christos Xenis [38]. Based on experience from his work in an IT company, he outlined how he socializes and empathizes with developers, and how he tries to solve possible conflicts in the team. According to him, the more human is the connection with the developers, the more noticeable is the boost in the team performance. One reason could be that developers are less distracted by human-related issues. He pointed out that he admires research studies but also realizes that they often contribute only very little that is useful in terms of a company’s interests. He concluded that all research in Software Engineering should incorporate a phase where practitioners' interests are incorporated into the research design. Coincidentally, several studies presented at PROFES address the concerns of human aspects in software development and the incorporation of practitioners’ interest into research designs.

3. REGULAR PAPERS

The PROFES 2013 presentations of regular research papers were divided into seven main topics and a short paper track. In this section, the presentations of the regular papers and the discussions are summarized.

3.1 Empirical Software Engineering

The Empirical Software Engineering area comprised six studies. The opening paper by Kasurinen, Laine, and Smolander [24] discusses the possibility of applying ISO/IEC 29110, a standard related to the organization of small development teams, to the game industry. The game industry is a particular case of software development as creative processes and artistic visions need to be supported. The authors employed the grounded theory method to analyze seven very small game development companies. Among the main findings, the study suggests that very small game developing organizations mostly work with ad-hoc methods. They conclude that the ISO/IEC 29110 standard is employable in the game industry. However, it should incorporate a more iterative and flexible approach, promoting Agile approaches.

Hyysalo et al. [19] presented a model to support cognitive work in software development workflows. The model addresses the complexity of software development processes and supports the management of knowledge when undertaking a software development project. The model was built during a pre-study of the literature. Subsequently, the authors empirically validated the model with the help of an action research initiative in a large Information and Communication technology (ICT) company. The study resulted in two main contributions. The first is an understanding of what cognitive support is needed in workflows: cognitive support is needed for work context and situation awareness, knowledge management aspects, the use of experience and skills, decision-making, and problem solving. The second contribution is an examination of how cognitive support can be provided for software development workflows. The study proposes three levels of support. The first level is concerned with the definition of purposes and high-level objectives on what the work system provides and what is its role. The second level defines the criteria and the functions to measure how a system is achieving its purposes. The third level is concerned with how activities and tasks are implemented and what resources are needed is order to accomplish the tasks.

Wiklund et al. [37] presented a case study on the development impediments when adopting Agile methods in the development department of a large organization. Although the literature review suggested that the impediments are mostly related to software testing, the study found that the largest source of impediments at the department level during the early phase of agile transformation are related to project management, coordination, and communication. However, if the team is considered as the unit of analysis, test activities are still the largest source of impediments and testing-related competencies are often missing in teams.

Graziotin, Wang, and Abrahamsson [14] offered a cognitive study on the affective states (emotions, moods, and feelings) of software developers and how they correlate with their self-assessed productivity. The researchers observed eight developers (four students, four professionals) working on their individual software projects. Their affective states and their self-assessed productivity were measured in intervals of ten minutes each. The analysis of the correlation employed a linear mixed-effects model, which is rarely employed in Software Engineering research. Evidence exists that the affective states of valence (perceived attractiveness of the development task) and dominance (the perception of possessing adequate skills) are positively correlated with the self-assessed productivity of developers. Additionally, the investigation produces evidence of the value of psychological measurements in empirical Software Engineering research and opens up alternative ways of researching human aspects in software development.

Freire et al. [11] presented a customizable model-driven environment for supporting and conducting controlled experiments in Software Engineering. The approach is composed of a domain-specific language for modeling the domain of controlled experiments, model-driven transformations that map abstractions from the domain-specific language to workflow models for each participant in the experiment, and a workflow execution environment that guides experiment execution and monitoring. The authors conducted an exploratory study in order to assess the expressivity and feasibility of the approach and its elements. The exploratory study produced evidence on how the approach allows adequate modeling of a selected study and generation of the experiment workflow that supports its execution.

Kläs, Bauer, and Tiberi [26] focused on the challenges that arise when performing large-scale research projects with case studies on technology evaluation. The inquiry first identified these challenges in a focus group of eleven researchers, who are experts in the field. The analysis of the data gathered in the focus group produced 17 challenges when performing quantitative technology evaluation. The challenges are grouped into the categories: organizational issues, collecting the right data, providing combinable data, and defining a baseline. The authors propose an approach to addressing these challenges. A simplified version of GQM+Strategies should consolidate general project goals and align them with case study-specific goals. The researchers address confidentiality issues by communicating and aggregating only relative measurement results on the project level in combination with their concept of internal baselines. Finally, typical baseline approaches are extended by inter-release baselines and presented together with an evaluation of their advantages and disadvantages using expert-based evaluation. The approach is currently being employed in a large research project with more than 20 case studies.

3.2 Software Process Improvement

The Software Process Improvement area comprised three studies. Pernstål et al. [30] reported on a software process improvement (SPI) initiative for two automotive companies. The study focuses on the inter-departmental development of software-intensive automotive systems at two Swedish automotive companies. The research employed the improvement framework utilizing lightweight assessment and planning (iFLAP), a lightweight SPI framework. In particular, the study was about the improvement planning of nine issues previously identified by the authors regarding the process assessment steps in iFLAP. In total, 41 participants of the two companies assigned priorities to the issues in both a survey and a workshop. The results indicate that there is a shared view among staff and managers in an organization on what the most critical issues are: requirements engineering, early manufacturing involvement, and the establishment of roles and responsibilities.

Méndez Fernández and Wieringa [9] presented a problem-driven approach for requirements engineering improvement using artifact orientation. The iterative approach has four main phases. The first two phases are meant to empirically investigate the problem: understand the stakeholders and the goals and investigate the requirements engineering artifacts. The last two phases are related to the design of the treatment, its validation and dissemination. The artifact-based requirements engineering improvement is defined and implemented in a series of action research workshops with the stakeholders. Between the action research workshops, the approach is to build the requirements engineering model by conceptualizing the artifacts, roles, and milestones in isolation without direct stakeholder involvement. The stakeholders validate the model during the workshops, and the participants implement the concepts in a tool-supported manner. Finally, in the dissemination phase, the developed model is validated through pilot projects and is propagated. The paper provides encouraging preliminary results of the approach, which is being employed in two research projects.

Aaramaa et al. [1] address the issue of managing a constant flow of requirements in very large-scale projects. The case study they performed analyzed the challenges that product management faces in feature-oriented release-based software development in a large ICT company by means of semi-structured interviews. The results of the study enrich our understanding of the practice of requirements screening. Among other intriguing results, the key challenges found in requirements screening rely on the different, informal sources for requirements communication, the ways the initial screening is performed and optimized, the analysis of the impact of the requirements' implementation and their value, the prioritization and negotiation of the requirements, the length of the planning and implementation phases, and on how the tools effectively support the process.

3.3 Managing Software Processes

The Managing Software Processes area hosted three studies. Kuhrmann, Méndez Fernández, and Knapp [27] presented an investigation of the perceived value of two strategies when designing a software process: the activity-oriented strategy and the artifact-oriented strategy. The reported experiment studied the selection of software process design strategies in the context of the process life cycle from two different points of view: process engineers and process consumers. Two groups of students (one for each software process design strategy) attended workshops for analyzing, conceptualizing, and implementing a process in a tool-supported manner. The study opted for the Eclipse Process Framework as a representative of the activity-oriented paradigm and the V-Modell XT framework as a representative of the artifact-oriented paradigm. The evaluation consisted of two steps. In the first step, the participants acted as process engineers and self-assessed the process documentation. In the second step, the two groups exchanged the documentation and assessed the artifacts from the perspective of process consumers. The artifact-oriented strategy, implemented with the V-Modell XT framework, was perceived to be of high value in terms of serving process engineers. However, the same did not hold from the perspective of process consumers, where the activity-oriented framework was perceived to be of higher value. The findings imply that, from the viewpoint of a process consumer, the assessment of a process design strategy does not necessarily need to coincide with the perceived value of the resulting process.

Aleixo, Kulesza, and Oliveira Junior [2] presented a quantitative comparative study of software process lines, which are a set of software processes with a common base. The comparative study was used to evaluate two approaches to modeling software process lines: the compositional and the annotative approaches. The researchers chose the tools EPF Composer and GenArch-P to represent these two approaches for modeling large software process lines. The study compares the two approaches in terms of three metric attributes: modularity, size, and complexity. Among the discoveries, the GenArch-P annotative approach obtained better results in terms of size and complexity. The compositional approach had better support for modularity than the annotative approach. The study concludes that preliminary evidence exists that the GenArch-P annotative approach is better than the EPF Composer compositional approach.

Oliveira Junior et al. [22] presented an extension of the Software & Systems Process Engineering Metamodel (SPEM), which is a UML-based metamodel for modeling, representing, and managing software processes. The study enhances SPEM in order to support the identification and representation of variability in software process lines. This is achieved by extending SPEM with the UML Stereotype-based Management of Variability in product line (Smarty). The outcome of the study is Stereotype-based Variability Management for SPEM (SMartySPEM). SMartySPEM introduces a variability representation and a set of guidelines that improve the representation and configuration of software process lines based on SPEM. The study provides an application example of SMartySPEM.

3.4 Software Measurement

Basili, Lampasona, and Ocampo Ramírez [4] reported that Information Technology (IT) plays a significant role in business strategies. However, limited knowledge exists on how to effectively create the link between business and IT. The authors describe how a global player in the oil and gas industry is aligning their IT activities with their business goals by employing GQM+Strategies. The study found evidence that the grid generation process of GQM+Strategies seamlessly aligned the business at several levels of the organization.

Da Silva, Pitangueira Maciel, and Ramalho [35] presented a GQM-based conceptual framework and a suite of metrics on the maintainability – i.e., on the ease of understanding, analyzing, and managing changes - of Model-Driven Architecture (MDA) software process models. The study describes three case studies where the metrics suite was applied to five MDA process models. The researchers compared the metrics outcomes with the opinion of software engineers. The comparison yielded highly similar results, confirming that the application of the framework is feasible.

Amasaki and Lokan [3] reported a study on the effects of weighted moving windows when predicting software effort with model-based approaches. They note that it makes sense for a software company to employ past projects to predict the effort needed for future projects. Previous research proposed the employment of recent projects only for estimation, as such projects are arguably more representative of an organization’s current practice. That is, a window can be employed to limit the size of the training data. As new projects are finished, old projects are dropped from the training dataset. However, this approach does not assign any weight to the projects employed for the training data. The presented study evaluates the possibility of adding a weight to such moving windows to give recent projects higher importance in the window than older projects. The paper shows that weighted moving windows provide inferior estimation accuracy when small windows are used, compared with non-weighted windows. However, weighted windows provide superior accuracy when larger windows are used.

3.5 Decision Support in Software Engineering

The Decision Support area hosted three studies. Oza et al. [29] note that although a cloud-based infrastructure is being increasingly adopted by industry in distributed software development (DSD) industries, little is known about the risks and benefits of this adoption. The study reports on a cross-case qualitative analysis based on focus groups in the context of a DSD project between three European sites. Regarding the benefits of cloud computing in DSD, five benefits emerged from the study: rapid development, continuous integration, cost savings, code sharing, and faster ramp-up. The identified risks are operational and technical dependencies of the teams, unavailability of access to the cloud for non-administrative teams, and, when software crashes, code commitment and integration issues, as well as technical debt.

Christoforou and Andreou [6] reported on a model supporting the cloud adoption decision process based on fuzzy cognitive maps (FCM). FCM provide a graphical representation of a given real-world model in the form of an acyclic graph comprising cognitive states, i.e., concepts. The authors conducted a literature review of the factors influencing the cloud adoption process. The factors were turned into concepts of the FCM - e.g., legal, scalability - and were validated or extended by experts in the field. The experts completed a questionnaire about the relationships between the concepts, thus defining the connections between the FCM concepts. The model was then tested with two idealized extreme scenarios studies and four real-world cases. The model succeeded in matching its estimation with the real decision in three out of four cases.

Stylianos Yasemis and Andreou [39] presented a decision-making approach for mobile software development with influence diagrams. The study aims to approximate an answer to two crucial questions when developing mobile software: Should the development of an app proceed? Will the user acceptance level be high for a particular mobile app? The authors developed two influence diagrams in order to answer these questions. The diagrams are composed of interrelated factors, identified through a literature review and input from experts in the field. The factors for the first diagram comprise concerns like development cost and time, quality factors, and complexity. The factors for the second diagram comprise concerns such as pricing, usability, and aesthetics. The study tested the diagrams in four scenarios, two of which were fictional and extreme cases. The results suggest that the proposed model is rather successful and can be trusted for decision making on mobile app development.

3.6 Safety-Critical Software Engineering

Two studies were presented in this area. Porres et al. [31] reported on how the IEC 61508 generic, process-oriented standard when employed during the development of safety-critical systems demands rigorous documentation of the development process. However, a survey they conducted showed that the documentation of the process is often informal, to the point that it may be preferable to employ IEC 61508 as a process itself. The study investigates this opportunity. The authors employed SPEM to model parts of IEC 61508. By employing SPEM and the EPF Composer tool, and by documenting the method, the study aims to support the process authoring of companies aiming to develop IEC 61508 compliant products.

Pedersen Notander, Höst, and Runeson [28] conducted an industrial qualitative study on the industrial needs and challenges when developing safety-critical but flexible and agile systems. The authors interviewed five experienced safety and software engineers in the Aerospace, Automation, Robotics, and Transportation industries. The authors conclude that human factors and the quality of requirements are central for safety-critical systems. Overall, the study produces evidence that agile development and flexibility can co-exist with safety-critical software development.

3.7 Software Maintenance

This area was supported by two studies. Imazato et al. [20] presented an investigation of how cloned code – i.e., code fragments, which are identical or highly similar to other fragments – is modified and evolves. The motivation is that when repeated code is employed in a software project, the modification to such code is a time-consuming activity and may introduce more bugs. The authors analyzed the repositories of three established open-source software systems (Ant, ArgoUML, and jEdit). Among the results, the study discovered that between 73% and 89% of repeated code was modified at least once in the lifecycle of the projects and that between 31% and 58% of these modifications were simultaneous modifications for all the elements of the code snippets. Programming instructions such as variable declarations, "try" and "while" blocks were more likely to be modified than any other type of instruction.

Fujiwara et al. [12] proposed an approach for project managers to assess refactoring instances from version archives in a configuration management system. The study employs three metrics to study refactoring instances: refactoring frequency, defect density, and fix frequency, all over a given period. The approach employs the UMLDiff algorithm to detect refactoring instances. Then, the method detects frequencies of refactoring, defect introduction, and defect fixes. Finally, the computed frequencies are compared and analyzed over time. The authors conducted a preliminary study of the method, which was employed within the context of the Columba open-source project. The results show that the frequency of defect introduction tends to decrease in the period following frequent refactoring.

4. SHORT PAPERS

We consider it beneficial for the short papers to also be presented to SEN readers. Ten short papers were presented at PROFES 2013. Rana et al. [33] presented an evaluation of eight standard reliability growth models on defect data from a real software system employed in a large automotive industry. The analysis showed that models with simply two parameters provide a good fit with unrealistic asymptotes. The models with the best fit were the Logistic and the InflectionS models. The authors determined that modeling the change of testing effort over time will be critical in applying standard reliability growth models in the automotive sector.

Shrestha et al. [34] reported a lack of transparency in the existing guidelines for IT service management (ITSM). Therefore, they started a project for implementing a tool for process assessment to enable transparent ITSM. They followed the Software-Mediated Process Assessment (SMPA) approach, which is a semi-automatic approach where a software tool can facilitate data collection to determine process capabilities and provide improvement recommendations. The project employs Design Science Research Methodology to develop the tool. The tool will help organizations to self-assess ITSM processes, thus avoiding costly assessment from consulting firms.

Graziotin and Abrahamsson [13] launched a call for action towards a practitioner-friendly comparative analysis of JavaScript Frameworks (JSF) suitable for researchers and practitioners. The authors warned how research interests and output often deviate from those of practitioners. The researchers noticed that a recent proposal for a comparison framework employs software metrics to be employed on the JSF source code. However, a pilot study identified that practitioners have very different concerns, like the documentation of the JSF, community participation, and the pragmatics of the frameworks. Therefore, the proposed research design also employs the previously proposed metrics on the same software product produced using the different JSF. In addition, empirical data from practitioners will be collected to understand and validate their needs when choosing a JSF. The resulting comparison framework will be a step forward in conciliating software engineering research and software development practitioners. It will allow quick selection of a JSF, thus helping Web development companies to save time and resources.

Campos and Oliveira [5] presented an analysis of the feasibility of the Business Process Model and Notation (BPMN) model to represent a software development process. The evaluation employed the Open Unified Process modeled with BPMN. The authors administered a questionnaire with nine Likert items related to the effectiveness of BPMN to represent the Open Unified Process and its effectiveness of representation in general. The study used three case studies, with 24 participants. The results showed that BPMN is suitable for modeling software processes.

Honda, Washizaki, and Fukazawa [18] proposed a generalized, non-linear stochastic software reliability model. The model aims to simulate the high uncertainty of development environment by taking into account software development dynamics such as the skills of the team and changing requirements. The authors performed a series of simulations and compared their encouraging results with those of other reliability models.

Kettunen [25] addressed the issue of defining and measuring the performance of software development teams. The study proposes a performance orientation framework with provisional instrumentation support for high-performing software teams. Performance is seen as a multidimensional space, consisting of business excellence, operational excellence, and growth. The performance framework aims to guide software development organizations in positioning and – consequently - improving their teams in the performance dimensions, with the goal being comprehensive excellence.

Hanakawa and Obana [15] aim to address the complexity of large-scale projects and proposed the concept of process independency between infrastructure construction and software construction. This independency is defined as the degree by which interaction happens between infrastructure construction and software construction. That is, high process independency occurs when the two processes smoothly execute without disturbing each other.

Diebold [8] argues that in Software Engineering, the complexity-related issue exists that not a single technology exists to support all required elements of individual software processes. However, it is also true that some technologies just fit better than others. How then to select the best technologies for a software project? To answer this question, the author presented a framework for project-specific process configuration, namely the Process Configuration Framework. The approach is divided into four stages, namely technology categorization, ranking, combination, and process configuration. In a case study, the opinions of four domain experts were compared with the framework results. The comparison yielded similar results.

Jäntti and Rout [21] performed a case study research with three cases. The investigated research problem relates to how service operation processes are performed in IT service provider companies. More specifically, they investigated the challenges regarding service operation. Among the results, they found differences in the use of Service Level Agreements and problem management. Regarding continual improvement methods, the analysis revealed the usefulness of methods such as collecting feedback on service desk case resolutions, using Lean Six Sigma to remove bottlenecks from IT service management processes, and benchmarking service operation processes. The key challenges discovered were related to measurement and to the reporting of service operation processes, the classification of incidents, third-party service providers, and the management of feedback and interfaces between IT service operation processes.

Finnegan, McCaffery, and Coleman [10] warned how the low level of security in networked medical devices is an issue, entailing various types of risks, including risks for patients' health. Therefore, they propose a framework for the security assurance of medical devices. The framework is a combination of international standards and documents in order to positively impact device manufacturer in their design decisions during the development of networked medical devices. The objective outcome of the study is the development of a process reference model, a process assessment model, and a measurement framework for the assurance of medical device manufacturers' development processes. The current status of the research is a tailored process assessment model for system life cycle processes (ISO/IEC 15504-6 using ISO/IEC 15288 as the reference model) with a focused risk management process (IEC/TR 80001-2-2) and additional processes for security assurance (ISO/IEC 15026-4).

PROFES 2014 will take place in Helsinki, Finland, from 10 to 12 December 2014 (http://www.profes-conferences.org).

5. ACKNOWLEDGMENTS

The authors would like to thank all PROFES 2013 organizers and participants for their passionate participation and their constructive feedback.

6. REFERENCES

[1] Aaramaa, S., Kinnunen, T., Lehto, J. and Tausan, N. 2013. Managing Constant Flow of Requirements: Screening Challenges in Very Large-Scale Requirements Engineering. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 123–137. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_12.

[2] Aleixo, F.A., Kulesza, U. and Junior, E.A. de O. 2013. Modeling Variabilities from Software Process Lines with Compositional and Annotative Techniques: A Quantitative Study. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 153–168. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_14.

[3] Amasaki, S. and Lokan, C. 2013. The Evaluation of Weighted Moving Windows for Software Effort Estimation. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 214–228. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_18.

[4] Basili, V.R., Lampasona, C. and Ramírez, A.E.O. 2013. Aligning Corporate and IT Goals and Strategies in the Oil and Gas Industry. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 184–198. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_16.

[5] Campos, A.L.N. and Oliveira, T.C. de 2013. Software Processes with BPMN: An Empirical Analysis. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 338–341. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_29.

[6] Christoforou, A. and Andreou, A.S. 2013. A Cloud Adoption Decision Support Model Based on Fuzzy Cognitive Maps. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 240–252. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_20.

[7] Deissenboeck, F., Heinemann, L., Herrmannsdoerfer, M., Lochmann, K. and Wagner, S. 2011. The quamoco tool chain for quality modeling and assessment. *Proceeding of the 33rd international conference on Software engineering - ICSE '11* (New York, New York, USA, 2011), 1133–1142. DOI=http://dx.doi.org/10.1145/1985793.1985977.

[8] Diebold, P. 2013. How to Configure SE Development Processes Context-Specifically? *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 355–358. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_33.

[9] Fernández, D.M. and Wieringa, R. 2013. Improving Requirements Engineering by Artefact Orientation. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 108–122. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_11.

[10] Finnegan, A., McCaffery, F. and Coleman, G. 2013. A Security Assurance Framework for Networked Medical Devices. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 363–366. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_35.

[11] Freire, M.A., Accioly, P.R.G., Sizílio, G., Neto, E.C., Kulesza, U., Aranha, E. and Borba, P. 2013. A Model-Driven Approach to Specifying and Monitoring Controlled Experiments in Software Engineering. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 65–79. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_8.

[12] Fujiwara, K., Fushida, K., Yoshida, N. and Iida, H. 2013. Assessing Refactoring Instances and the Maintainability Benefits of Them from Version Archives. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 313–323. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_25.

[13] Graziotin, D. and Abrahamsson, P. 2013. Making Sense Out of a Jungle of JavaScript Frameworks - Towards a Practitioner-Friendly Comparative Analysis. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 334–337. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_28.

[14] Graziotin, D., Wang, X. and Abrahamsson, P. 2013. Are Happy Developers More Productive? *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (Paphos, Cyprus, 2013), 50–64. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_7.

[15] Hanakawa, N. and Obana, M. 2013. An Experience Report: Trial Measurement of Process Independency between Infrastructure Construction and Software Development. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 351–354. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_32.

[16] Heidrich, J. 2013. Software Effort Estimation and Risk Management. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 370–371. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_37.

[17] Heidrich, J., Oivo, M., Jedlitschka, A. and Baldassarre, M.T. 2013. Product-Focused Software Process Improvement. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (Berlin, Heidelberg, 2013). DOI=http://dx.doi.org/10.1007/978-3-642-39259-7.

[18] Honda, K., Washizaki, H. and Fukazawa, Y. 2013. A Generalized Software Reliability Model Considering Uncertainty and Dynamics in Development. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)*

(2013), 342–346. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_30.

[19] Hyysalo, J., Lehto, J., Aaramaa, S. and Kelanti, M. 2013. Supporting Cognitive Work in Software Development Workflows. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 20–34. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_5.

[20] Imazato, A., Sasaki, Y., Higo, Y. and Kusumoto, S. 2013. Improving Process of Source Code Modification Focusing on Repeated Code. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 298–312. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_24.

[21] Jäntti, M. and Rout, T. 2013. Improving IT Service Operation Processes. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 359–362. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_34.

[22] Junior, E.A. de O., Pazin, M.G., Gimenes, I.M. de S., Kulesza, U. and Aleixo, F.A. 2013. SMartySPEM: A SPEM-Based Approach for Variability Management in Software Process Lines. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 169–183. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_15.

[23] Kaindl, H. 2013. Model-Based Transition from Requirements to High-Level Software Design. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 367–369. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_36.

[24] Kasurinen, J., Laine, R. and Smolander, K. 2013. How Applicable Is ISO/IEC 29110 in Game Software Development? *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 5–19. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_4.

[25] Kettunen, P. 2013. Orienting High Software Team Performance: Dimensions for Aligned Excellence. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 347–350. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_31.

[26] Kläs, M., Bauer, T. and Tiberi, U. 2013. Beyond Herding Cats: Aligning Quantitative Technology Evaluation in Large-Scale Research Projects. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 80–92. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_9.

[27] Kuhrmann, M., Fernández, D.M. and Knapp, A. 2013. Who Cares About Software Process Modelling? A First Investigation About the Perceived Value of Process Engineering and Process Consumption. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 138–152. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_13.

[28] Notander, J.P., Höst, M. and Runeson, P. 2013. Challenges in Flexible Safety-Critical Software Development - An Industrial Qualitative Survey. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 283–297. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_23.

[29] Oza, N., Münch, J., Garbajosa, J., Yagüe, A. and Ortega, E.G. 2013. Identifying Potential Risks and Benefits of Using Cloud in Distributed Software Development. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 229–239. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_19.

[30] Pernstål, J., Gorschek, T., Feldt, R. and Florén, D. 2013. Software Process Improvement in Inter-departmental Development of Software-Intensive Automotive Systems - A Case Study. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 93–107. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_10.

[31] Porres, I., Heidenberg, J., Weijola, M., Nordman, K. and Truscan, D. 2013. Authoring IEC 61508 Based Software Development Process Models. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 268–282. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_22.

[32] Ramírez, A.E.O. 2013. ECO-MAPS: Information Quality-Driven Enterprise Modeling. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 3. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_2.

[33] Rana, R., Staron, M., Mellegård, N., Berger, C., Hansson, J., Nilsson, M. and Törner, F. 2013. Evaluation of Standard Reliability Growth Models in the Context of Automotive Software Systems. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 324–329. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_26a.

[34] Shrestha, A., Cater-Steel, A., Tan, W.-G., Toleman, M. and Rout, T. 2013. A Tool for IT Service Management Process Assessment for Process Improvement. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 330–333. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_27.

[35] Silva, B.C. da, Maciel, R.S.P. and Ramalho, F. 2013. Evaluating Maintainability of MDA Software Process Models. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 199–213. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_17.

[36] Wagner, S. 2013. Making Software Quality Visible. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 1–2. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_1.

[37] Wiklund, K., Sundmark, D., Eldh, S. and Lundqvist, K. 2013. Impediments in Agile Software Development: An Empirical Investigation. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 35–49. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_6.

[38] Xenis, C. 2013. Implementation of an Online Multi-level and Device-Independent Time and Attendance System. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 4. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_3.

[39] Yiasemis, P.S. and Andreou, A.S. 2013. Modeling and Decision Support of the Mobile Software Development Process Using Influence Diagrams. *14th International Conference on Product-Focused Software Process Improvement (PROFES 2013)* (2013), 253–267. DOI=http://dx.doi.org/10.1007/978-3-642-39259-7_21.