Jeners, S., Clarke, P., O'Connor, R. V., Buglione, L., and Lepmets, M. Harmonizing Software Development Processes with Software Development Settings – A Systematic Approach, 20th European Conference on Systems, Software and Services Process Improvement (EuroSPI 2013), CCIS Vol. 364, Springer-Verlag, June 2013

Harmonizing Software Development Processes with Software Development Settings – A Systematic Approach

Simona Jeners[1], Paul Clarke[2], Rory V. O'Connor[3], Luigi Buglione[4], Marion Lepmets[2]

[1]RWTH Aachen University, Research Group Software Construction, Germany
simona.jeners@swc.rwth-aachen.de
[2] Regulated Software Research Group, Dundalk Institute of Technology, Ireland
{Paul.Clarke, Marion.Lepmets}@dkit.ie
[3] Dublin City University, Ireland
roconnor@computing.dcu.ie
[4] ETS Montréal / Engineering.IT SpA, Italy
luigi.buglione@eng.it

Abstract. The software process landscape is rich in complexity and many alternative software development approaches have emerged over the past 40 years. However, no single software development approach is universally implemented and it seems likely that no single approach can be universally useful. One of the primary reasons that no single approach is universally useful is that no two software development settings are identical. We have assembled a team of recognized academics, who together with industrial collaborators, plan to map the complex world of software processes with the context of software development projects. The results of our initial mapping efforts, reported in this paper, demonstrate that although there are challenges in an undertaking such as this, the outcomes are potentially of considerable value to both software researchers and practitioners.

Keywords: Software Process, Situational Factors, Process Improvement, Mappings, Systematic Approach.

1 Introduction

When compared with some of the more established engineering disciplines, it has been claimed that the profession of software engineering can be considered to be in its youth [1]. However, arguments to the contrary also exist: that the practice of software development may already be quite mature [2], and that software engineering may not be a true engineering discipline at all [3]. Whether software development is or is not a true engineering discipline may for many practitioners represent an academic debate. In practice, software development is beset with many challenges and constraints. The variety of problems to which software is proposed as a solution is very broad, and the tooling and materials employed in software development are constantly evolving. Nonetheless, many general models and frameworks for software development have been published, and some of these approaches have proven to be beneficial.

Owing to the rich variety of software development settings (for example: the nature of the application being developed, team size, requirements volatility), the implementation of a set of practices for software development may be quite different from one setting to another. Process capability and maturity frameworks (CMFs), such as CMMI-DEV [4] and ISO/IEC 15504 [5], recognize that different implementations of software processes are possible and provide mechanisms for assessing any given implementation. Furthermore, CMFs also provide a roadmap for process improvement. However, evidence of the benefits of CMFs is predominately restricted to larger organisations [6], [7]. Limited evidence of the benefits of CMFs for smaller software development settings also exists [8-10]. However, it has been suggested that such approaches may not be suited to the needs of small software development organizations – and it would appear that in practice, smaller organizations tend not to adopt CMFs.

Together with other so-called *traditional* approaches, such as Quality Management Standards (e.g. ISO-9001), CMFs have been criticized for being overly restrictive (or *heavy)* in terms of their ability to support the innovative and speculative nature of software development [11]. As a result, the Agile Manifesto [12] was devised as an alternative philosophy to developing software, addressing some of the limitations of traditional approaches. In particular, the agile manifesto emphasizes the need for working software over extensive documentation, while also promoting the frequent delivery of smaller usable features rather than waiting a long time to deliver a single large system. A number of agile software development approaches, generally termed agile methodologies, have been developed [13], [14]. Furthermore, published studies have demonstrated the benefits of adopting an agile software development approach, including increased productivity, improved time to market [15] and reduced code defect densities [16]. While the advent of agile methodologies has delivered benefits to software development initiatives, it has also been noted that the general philosophy may suffer from a number of limitations. For example, it has been argued that agile development methodologies may require a very skilled software developer, a *premium* developer [17], and that some approaches place an impractical demand on customer collaboration [18].

The preceding paragraphs describe just a small subset of the approaches to software development (herein termed *Improvement Reference Models* (IRMs)) that have been proposed over the past few decades. And despite the benefits of each individual approach, no single approach has been universally adopted. Rather, software development projects and organizations appear to choose a base model that works for them, thereafter adapting and changing their specific processes to address their own specific needs [19]. Therefore, the basic requirement of a software development process is that it "*should fit the needs of the project*" [20]. Although it is relatively straightforward to understand that a software development process should ideally be harmonized with the context within which the software must be developed and delivered, no earlier published research has focused on identifying the relationship between aspects of software development settings (which we term the *situational context*) and the broad dimensions of software development processes. Therefore, this research is motivated to address this gap, and in order to do so, the authors have secured the participation of both industrial and academic collaborators. Together, and over an extended period of time, these collaborators will develop a systematic approach to identify the relationships between factors of situational context and various aspects of the software development process. Our approach could also support the IT projects or IT departments that use frameworks such as ITIL [21] and CMMI-SVC [22]. However, we have chosen to focus first on the software development area.

Our primary goal is to support projects to efficiently achieve their objectives by a systematic improvement of their internal processes. A high Return on Investment (ROI) is a prerequisite for this improvement, i.e. perform improvement initiatives that bring the most benefit and can be managed by the project without risking the project goals and constraints (time, cost and quality).

To support projects, we aim to develop a systematic approach that identifies best practices from different improvement reference models (IRMs) that are best suited for an IT project. Our approach considers the following aspects:

- **Value/Benefit**: The context of the project must be considered to identify the best practices that bring the most benefit.
- **Cost:** The adoption cost of the best practice should not jeopardize the achievement of the project goals.

The remainder of this paper is structured as follows: Section 2 introduces a systematic approach wherein different contributors iteratively map factors of situational context to software development processes. In Section 3 we report on the initial application of this approach and in Section 4, we reflect on the challenges and efficacy of the approach. Section 5 presents a conclusion as well as outlining future work plans.

2 Approach

This section outlines a systematic approach adopted in order to map situational factors to IRMs practices. The approach has two main phases (fig. 1): **(1) Trial Approach** – experts perform a subjective mapping between a subset of situational factors and IRM practices; **(2) Broader Mapping Program** – more experts and IT project members evaluate the mappings between a larger set of situational factors and IRMs that will support the improvement of the systematic mapping approach. The **Trial Approach** consists of the following steps:

T1. Secure the participation of experts for trial. Our goal is to involve many experts to perform or evaluate mappings between situational factors and various IRM practices.

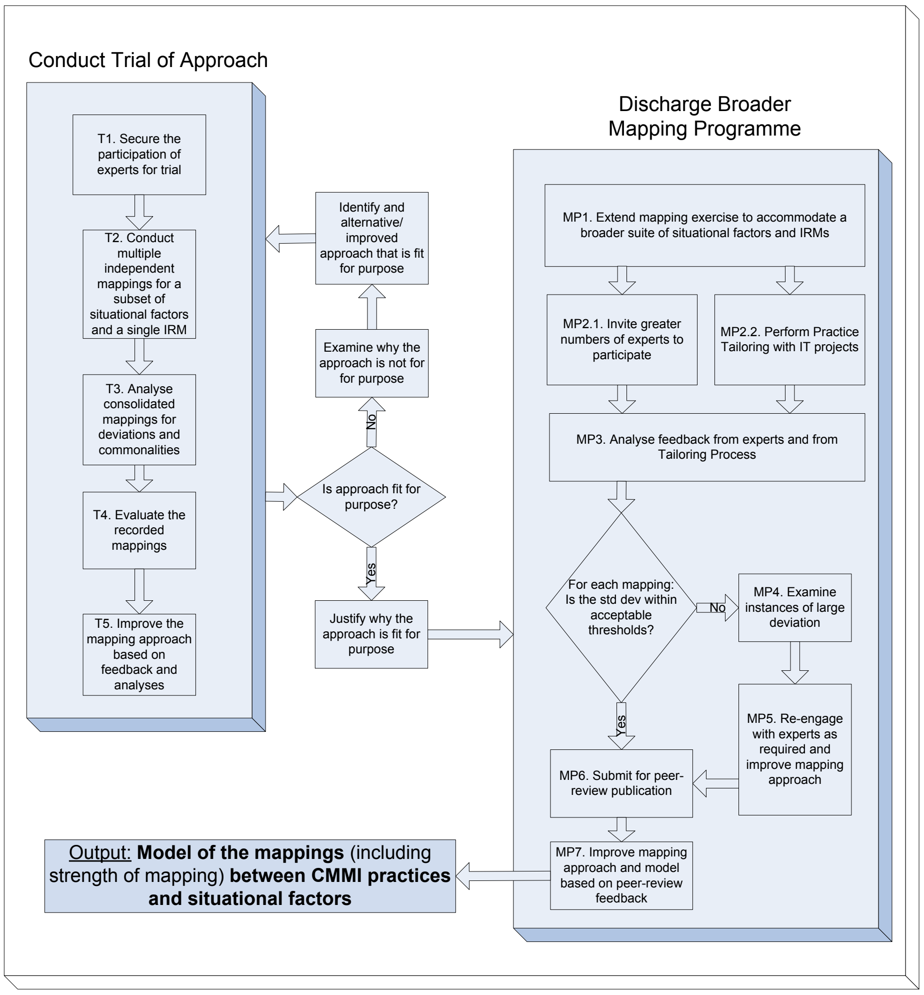
Conduct Trial of Approach
T1. Secure the participation of experts for trial
T2. Conduct multiple independent mappings for a subset of situational factors and a single IRM
T3. Analyse consolidated mappings for deviations and commonalities
T4. Evaluate the recorded mappings
T5. Improve the mapping approach based on feedback and analyses
Identify and alternative/ improved approach that is fit for purpose
Examine why the approach is not for for purpose
No
Is approach fit for purpose?
Yes
Justify why the approach is fit for purpose
Discharge Broader Mapping Programme
MP1. Extend mapping exercise to accommodate a broader suite of situational factors and IRMs
MP2.1. Invite greater numbers of experts to participate
MP2.2. Perform Practice Tailoring with IT projects
MP3. Analyse feedback from experts and from Tailoring Process
For each mapping: Is the std dev within acceptable thresholds?
No
MP4. Examine instances of large deviation
MP5. Re-engage with experts as required and improve mapping approach
Yes
MP6. Submit for peer-review publication
MP7. Improve mapping approach and model based on peer-review feedback
Output: Model of the mappings (including strength of mapping) between CMMI practices and situational factors

Fig.1. Overview of the proposed mapping approach

T2. Conduct multiple independent mappings for a subset of situational factors and a single IRM. Our goal is to perform a series of independent mappings on a subset of the factors and practices. A *performer* conducts subjective mappings of the perceived strength of the relationship between a practice and a situational factor, according to the following four-point ordinal scale:

- **3** – the practice highly supports the project in managing the situation described by the situational factor
- **2** – the practice supports the project in managing the situation described by the situational factor
- **1** – the practice weakly supports the project in managing the situation described by the situational factor

- **0** – the practice does NOT support the project in managing the situation described by the situational factor

The independent results of each performer are consolidated to obtain an overview of the mapping strength for each factor and IRM practice.

T3. Analyse consolidated mappings for deviations and commonalities.

Examine the contributions from the various performers and seek to confirm common understandings for the different factors and practices. Note commonalities and address instances of deviation as deemed appropriate. This may result in a revised set of consolidated mappings.

T4. Evaluate the recorded mappings.

Conduct an independent evaluation of the consolidated mappings through the use of an *evaluator*. The evaluator performs a review considering not just their subjective opinion but also the previously consolidated performers input - leading to better results.

T5. Improve the mapping approach based on feedback and analyses. Mark-up the previous mappings based on the combined feedback. (Note that at once the research advances to this stage, it is envisaged that a number of practitioners will be engaged in the further improvement of the mapping approach prior to discharging the broader mapping programme).

The **Broader Mapping Programme** comprises of the following steps:

MP1. Extend mapping exercise to accommodate a broader suite of situational factors and IRMs. Based on our systematic approach, identify the mapping strength for more situational factors and more IRMs (Note that our approach can be extended to address IT projects from other domains (Services, Functional Safety)).

MP2.1. Invite greater number of experts to participate. Involve further experts to participate in the evaluation and improvement of the mapping approach. An online survey may help to get feedback on the method and on the results for additional domains (e.g. Services, Functional Safety).

MP2.2. Perform Practice Tailoring with IT projects. Evaluate the mapping results by using these results in practice. An industrial partner will choose development projects with different characteristics aiming to identify IRM practices that are best suited to the given project situation(s). We aim to conduct a Tailoring-Workshop with the members of these projects: first, we identify the situational factors that are most relevant for the project; secondly, we provide the project with our mappings, as a recommendation for practice adoption. Based on the benefit and on the cost for the adoption, the project makes a decision which practices should be adopted. During the practice adoption, we aim to collect feedback from the project: did the adoption of the practices bring the desired benefit, i.e. helped managing a certain critical situation in the project?

MP3. Analyse feedback from experts and from project members. Consolidate feedback from practitioners and the impact on the mapping framework.

MP4. Examine instances of large deviation. Identify the mappings where there is a large deviation between our results and the feedback from the experts and project members.

MP5. Re-engage with experts as required and improve mapping approach. In a Retrospective-Workshop with selected experts and project members, examine the large deviations and identify improvements to our systematic mapping approach.

MP6. Submit for peer-review publication. The mapping approach along with implementation outcomes is submitted for academic peer-review.

MP7. Improve mapping approach and model based on peer-review feedback. The findings collected during the peer-review are used to make final improvements to the mapping approach.

The outputs from the two phases outlined above are (1) a systematic approach to objectively map situational factors and IRM practices; (2) a matrix with the relationships between software development settings and software development processes.

3 Application and Results

This section outlines the steps performed so far (T1 to T4) and the results achieved:

T1. Secure the participation of experts for trial. Inviting and motivating different experts to participate in the trial. The authors of this paper were all involved in the trial.

T2. Conduct multiple independent mappings for a subset of situational factors and a single IRM. First, a set of situational factors and IRMs was defined. To identify the relationships between situational contexts and software development processes, it is important that comprehensive and reliable reference frameworks are employed. For the software development processes, any software process model could potentially be employed. However, of all the process models published to date, the two most comprehensive are ISO/IEC 12207 [23] and the CMMI-DEV. Both of these two resources are comprehensive and have been widely applied in practice; therefore, either was suited to our mapping task. Since our industrial collaborators expressed a strong preference for CMMI-DEV (this was their area of expertise), it was decided that the CMMI-DEV would be employed as the process reference for the mapping exercise.

Regarding the situational context for software development, again a number of possible reference frameworks existed. The work of Xu and Ramesh [24] identifies twenty distinct situational factors, while later works include even greater numbers of factors – for example, Petersen and Wohlin [25] identify twenty-one factors, and Bekkers et al. [26] list thirty distinct factors. However, it is the situational factors reference framework developed by Clarke and O'Connor [27] that is both the most recent and the most comprehensive contribution to date regarding situational context. Clarke and O'Connor [27] have systematically included the earlier identified works in the development of their framework. Furthermore, their situational factors reference framework also incorporates important seminal contributions from a range of related domains, including risk factors for software development (e.g. [28]), software cost estimation (e.g. [29]), and software process tailoring (e.g. [30]). For the initial mapping exercise, we randomly selected two different situational factors: "*performance of application(s)/product(s)*", and "*changeability of requirements*".

As per the process outlined in Section 2, four performers attempted an initial mapping, with a fifth academic performing the evaluation of the mappings. A template was created to document the subjective mappings of each performer (Fig. 2 provides a snapshot). This template contains all the practices of CMMI-DEV ML2 and ML3 (since these two processes are widely used by organizations [31]) categorized by their process areas and maturity levels. For each practice, the performer could specify the mapping strength 0-3 by marking the corresponding cell with "x". The number of "x" indicates the number of performers that agreed to a certain mapping strength. As the mappings are subjective, we introduced a justification column to document the reasoning of the experts for the chosen strength. After the performers finished specifying their mappings independently, their respective inputs were consolidated.

IRM practices / Situational factor		**Required performance of application(s)/product(s)**				
		Concerned with the performance demands that are placed product(s)/application(s) under development. For example, product(s)/application(s) may be required to process a high number of transaction peRSecond.				
		Consolidated Independent Mappings				**Justification**
CMMI Process Area		3	2	1	0	
Measurement and Analysis	**MA**					
Establish and maintain measurement objectives that are derived from identified information needs and objectives.	SP 1.1	x	xxxx			If there are specific performance requirements, then it may be necessary to set objectives and measures in relation to the performance of application(s)/product(s).
Specify measures to address the measurement objectives.	SP 1.2		xxxxx			
Specify how measurement data will be obtained and stored.	SP 1.3			xxxx	x	Although the collection of measurement data may be importnant where performance is an important consideration, this does not imply that it is necessary to specifiy how the measurements will be obtained or analysed.
Specify how measurement data will be analyzed and reported.	SP 1.4			xxx	xx	
Obtain specified measurement data.	SP 2.1		xxxx	x		If ther are specific performance criteria to satisfy, then the collection and analysis of the measurement data is going to be necessary.
Analyze and interpret measurement data.	SP 2.2		xxxxx			
Manage and store measurement data, measurement specifications, and analysis results.	SP 2.3		x	xxxx		If there are specific performance criteria to satisfy, then the measurement data/results may need to be stored and communicated to stakeholders.
Report results of measurement and analysis activities to all relevant stakeholders.	SP 2.4		x	xxx	x	

Fig.2. A Fragment of the mapping

T3. Analyse consolidated mappings for deviations and commonalities. Based on the mapping consolidation, we conducted a discussion based on three principles.

Principle 1: Instances of significant disparity would be prioritized for discussion. For example, if each of the 4 participants had a different mapping strength for a situational factor to a CMMI-DEV practice, then clearly there was considerable disagreement on the strength of the relationship and hence, a discussion was warranted to establish if there was a lack of common understanding.

Principle 2: Instances where one (or more than one) of the participants had considered that there was no relationship between a situational factor and a CMMI-DEV practice (and others disagreed) were also prioritized for discussion. This was considered important as the decision to rule out any relationship between a factor and a practice could have important implications for the overall work.

Principle 3: As a general rule, if the reported mapping strengths were clustered in just two or three adjacent cells, such instances could be de-prioritized (with the exception of rule number 2 above – i.e. one of the cells was a 0 [or no relationship] mapping).

In the discussion, we use the idea of the "poker planning"-method for cost estimation [32], asking the contributors with the minimum and maximum strength to justify their selection. This often led to an adjustment to the initial inputs.

T4. Evaluate the recorded mappings. The consolidated and analysed mappings were independently evaluated by an experienced academic evaluator (who was not involved in the mapping process up to this point). The evaluator identified the frequencies of provided mapping strengths as a mechanism for taking all views into account and for assisting in calculating the overall mapping between a situational factor and the CMMI-DEV procedure. This led to a series of evidences that as follows:

Situational Factor 1 – Required Performance of Application(s)/Product(s): Process Areas (PA) with the strongest mapping to the performance factor were MA (Measurement & Analysis), PMC (Project Monitoring & Control), SAM (Supplier & Agreement Management), RD (Req. Development) and REQM (Requirement Management), and VAL (Validation). Adding process categories of these PAs, we see that the Support and the Project Management categories (two PAs for each category) were related to ML2; and the Engineering process category (with two PAs) to ML3. When we look at the staged representation of the CMMI-DEV, those four ML2 process areas are effectively requested to have good performance as REQM defines guidelines to manage the project requirements, SAM leads to a good relationship with (sub)providers and assures the fulfillment of requirements for the supplier deliveries, PMC requests monitoring the project results to fulfill its requirements and MA is the basis for this monitoring using and analyzing different metrics. On ML3, RD is the main input for any software lifecycle (SLC) activity.

Situational Factor 2 – Requirements Changeability: The PAs with the strongest mapping to requirements changeability were CM (Configuration Management), PMC (Project Monitoring & Control), PP (Project Planning),

REQM (Requirement Management), IPM (Integrated Project Management), RD (Requirement Development) and RSKM (Risk Management). In other words, the process categories on ML2 were Support (one PA) and Project Management (three PAs); and on ML3 Project Management (two PAs) and Engineering (three PAs) categories.
In summary, both RD and REQM are grouping practices that aim to collect, define, analyze and manage the requirements (incl. their changes), while PP and PMC are their counter-side in terms of planning and controlling that variability, often expressed in the so-called '*scope creep*' phenomenon, as well described in the IFPUG Function Point Analysis CPM (Counting Practice Manual) [33]. At ML3, IPM defines practices to track and resolve critical dependencies caused by requirements changes with the different stakeholders, while RSKM helps identifying and analyzing the risks that can be caused by the changes.

4 Retrospective

In this section we will briefly outline some of the key challenges encountered while executing this study, the actions taken to address them and open challenges for the continued evolution of the research.

An important early task to address was the selection of suitable expert participants and the associated administrative and coordination issues for project execution. The lead researcher used a network of personal contacts, which were initially established at European and international software process conferences. From a starting point of 2 experts, a further 3 were recruited. All correspondence was conducted via email and teleconference facilities (Skype), which was hindered by scheduling/availability of experts, time differences, etc. However, the geographical co-location (Ireland) of 3 of the experts alleviated some of these difficulties.

When conducting the initial mapping exercise, it became apparent that the various experts had applied subjective interpretations regarding certain situational factors and CMMI-DEV practices, which led to inconsistent initial mappings. For example, the situational factor *Commitment of Personnel* was interpreted differently requiring discussions during teleconferences. This led to a description of each situational factor being added to the template to ensure a more consistent interpretation of the factors (refer to Fig. 2, rightmost column, second row). Despite this addition, the situational factor regarding human-centric activities still proved extremely difficult to reconcile among experts regarding different interpretations, resulting in the decision to not include such factors in the initial phase and to more carefully consider these issues at a later stage.

In addition, during this initial exercise there was substantial discussion on the usage of a four-point ordinal scale, with suggestions that a 5 or even 8-point scale could be more appropriate as it could lead to a richer understanding of the relationships. However, to date the decision is to maintain a 4-point scale.

A final point worthy of comment relates to the time and logistical issues surrounding the consolidation of results. This required between 1.5 and 2 hours of intensive discussion per situational factor for 3 experts to analyze and agree. Potential logistical issues would arise here if a larger number of experts were used. In addition the usage of a relatively simple Excel-based spreadsheet made progress with altering and consolidating mappings slow. This could be aided by the creation of an enhanced spreadsheet harnessing macros or possibly a database system. A final remaining challenge to be addressed relates to the selection of an appropriate form of evaluation for both the research approach and outputs. As this work progresses, this will become a more critical consideration. However, at this early stage in the research, this remains an open challenge.

5 Conclusion and Future Work

The software process landscape is rich in complexity and many alternative software development approaches have been developed over recent decades. However, no single software development approach is universally implemented or useful. One of the primary reasons for this is the significant variation that is witnessed in software development endeavors. Just one software developer completes some software projects, while other projects require a large team. There is a broad range in the value of software projects, and a wide spectrum to be satisfied in terms of the criticality of operational domain. Some software development efforts are highly innovative with emerging requirements, while other efforts may offer greater requirements certainty earlier in the implementation cycle.

Given such variation in software development settings, it is not surprising to discover a wide variety of approaches to software development. However, although a variety of approaches exist, the authors of this paper contend that insufficient guidance is offered on the activity of tailoring software processes and process improvement efforts to individual settings. Therefore, it is important that further research be dedicated to examining the relationship between software development settings and software development processes. In this respect, we have assembled a team of recognized academics, who together with industrial collaborators, plan to map the complex world of software processes with the context of software development projects.

In this paper, we have outlined an approach to identify mappings between processes and project settings. We have reported on our initial experiences from the application of the process. These initial findings highlight some of the significant challenges that our mapping project has to overcome. For example, we have had to expand the previously available descriptions of situational factors with concise definitions that permit a more consistent interpretation of the role of individual factors. We have also discovered that the role of human-centric factors, such as the commitment of employees, is difficult to agree upon. Hence, the mapping of human-centric factors has been postponed to a later phase. Since the broader mapping program represents a very large undertaking, we plan to complete the work in an iterative fashion over a broad period of time. Therefore, the essential purpose of this paper is to highlight the need for this research, identify an approach to ground the mapping exercise, and to report on the initial mapping of two situational factors to all of the practices of CMMI-DEV. In the future, we aim to decrease the subjectivity of such mappings by proposing an approach to systematically map situational factors to processes and by the involvement of more experts from research and industry. Therefore, we envisage that later reports of this research activity will contain mapping tables that will serve as valuable new resources for both practitioners and researchers. Such mapping tables will identify, for the first time, the combined view of researchers and practitioners on the relationship between aspects of situational contexts and software development processes.

6 References

1. Jacobson, I., Ng, P., McMahon, P., Spence, I., Lidman, S.: The Essence of Software Engineering: The SEMAT Kernel. Queue, 10 (10), 40-51 (2012)
2. Schaefer, R.: Software Maturity: Design as Dark Art. SIGSOFT Software Engineering Notes, 34 (1), 1-36 (2009)
3. Denning, P.J., Riehle, R.D.: The Profession of IT. is Software Engineering Engineering? Communications of the ACM, 52 (3), 24-26 (2009)
4. SEI: CMMI for development, version 1.3. Software Engineering Institute, CMU/SEI-2006-TR-008. Pittsburgh, PA, USA (2010)
5. ISO/IEC: IS0/IEC 15504: Information technology - process assessment, part 1 to part 5. International Organisation for Standardization, Geneva, Switzerland (2005)
6. Herbsleb, J., Goldenson, D.: A systematic survey of CMM experience and results. In: Proceedings of the 18th International Conference on Software Engineering (ICSE 1996), pp. 323-330. IEEE Computer Society, Los Alamitos, California, USA (1996)
7. Gibson, D., Goldenson, D. and Kost, K.: Performance results of CMMI-Based Process Improvement. Software Engineering Institute, Carnegie Mellon University, CMU/SEI-2006-TR-004. Pittsburgh, Pennsylvania, USA (2006)
8. Cepeda, S., Garcia, S., : Is CMMI Useful and Usable in Small Settings? CrossTalk, The Journal of Defense Software Engineering, 21 (2), 14–18 (2008)
9. Cater-Steel, A., & Rout, T.: SPI long-term benefits: Case studies of five small firms. In: H. Oktaba (ed.): Software Process Improvement for Small and Medium Enterprises - Techniques and Case Studies. IGI Global, Hershey, PA, USA (2008)
10. Laporte, C.Y., Desharnais, J.M., Abouelfattah, M., Bamba, J.C., Renault, A., Habra, N.: Initiating Software Process Improvement in Small Enterprises: Experiments with Micro-Evaluation Framework. In: Proceedings of the International Conference on Software Development, pp. 153-163. (2005)
11. Dyba, T., Dingsoyr, T.: Empirical Studies of Agile Software Development: A Systematic Review. Information and Software Technology, 50 (9-10), 833-859 (2008)
12. Fowler, M., & Highsmith, J.: The Agile Manifesto. Software Development, 28-32(2001)

13. Beck, K.: Extreme programming explained: Embrace change. Addison-Wesley, Reading, Massachusetts, USA (1999)
14. Schwaber, K., Beedle, M.: Agile software development with SCRUM. Prentice Hall, Upper Saddle River, New Jersey, USA (2002)
15. Reifer, D.J.: How Good are Agile Methods? IEEE Software, 19 (4), 16-18 (2002)
16. Fitzgerald, B., Hartnett, G., Conboy, K.: Customising Agile Methods to Software Practices at Intel Shannon. European Journal of Information Systems, 15 (2), 200-213 (2006)
17. L. Constantine, Methodological Agility, http://www.ddj.com/architect/184414743
18. Greer, D., Conradi, R.: Software Project Initiation and Planning - an Empirical Study. IET Software, 3 (5), 356-368 (2009)
19. Coleman, G., O'Connor, R.: Investigating Software Process in Practice: A Grounded Theory Perspective. Journal of Systems and Software, 81 (5), 772-784 (2008)
20. Feiler, P., Humphrey, W.: Software process development and enactment: Concepts and definitions. SEI, Carnegie Mellon University, CMU/SEI-92-TR-004. Pittsburgh, Pennsylvania, USA (1992)
21. Taylor, S., Cannon, D. and Wheeldon, D.: ITIL The Cabinet Office, (2011)
22. SEI: CMMI for Services, Version 1.3, CMU/SEI-2012-TR-034. Software Engineering Institute, Pittsburgh, PA, USA (2010)
23. ISO/IEC: ISO/IEC 12207-2008 - systems and software engineering – software life cycle processes. ISO, Geneva, Switzerland (2008)
24. Xu, P., Ramesh, B.: Software Process Tailoring: An Empirical Investigation. Journal of Management Information Systems, 24 (2), 293-328 (2007)
25. Petersen, K., Wohlin, C.: Context in industrial software engineering research. In: Proceedings of the 3rd International Symposium on Empirical Software Engineering and Measurement, pp. 401-404. IEEE Computer Society, Washington (2009)
26. Bekkers, W., van de Weerd, I., Brinkkemper, S., Mahieu, A.: The Influence of Situational Factors in Software Product Management: An Empirical Study. In: Proceedings of the Second International Workshop on Software Product Management (IWSPM '08), pp. 41-48. IEEE Computer Society, Los Alamitos, CA, USA (2008)
27. Clarke, P., O'Connor, R.V.: The Situational Factors that Affect the Software Development Process: Towards a Comprehensive Reference Framework. Journal of Information and Software Technology, 54 (5), 433-447 (2012)
28. Benaroch, M., Appari, A.: Financial Pricing of Software Development Risk Factors. IEEE Software, 27 (5), 65-73 (2010)
29. Boehm, B., Clark, B., Horowitz, E. et al.: Software cost estimation with cocomo II. Prentice Hall PTR, Upper Saddle River, NJ, USA (2000)
30. Cameron, J.: Configurable Development Processes. Communications of the ACM, 45 (3), 72-77 (2002)
31. SEI: CMMI for SCAMPI SM Class A Appraisal Results 2012 Mid-Year Update, . SEI, CMU (2012)
32. Grenning, J.: Planning Poker. Renaissance Consulting – April 2012, (2002)
33. IFPUG: Counting Practices Manual (Version 4.3). International Function Points User Group, http://www.ifpug.org/?p=83 (October 2009)