Changing Situational Contexts Present a Constant Challenge to Software Developers

Paul Clarke[1,2] and Rory V. O'Connor[1,2(✉)]

[1] Dublin City University, Dublin, Ireland
{pclarke,roconnor}@computing.dcu.ie
[2] Lero, the Irish Software Engineering Research Centre, Dublin, Ireland

Abstract. A software process can take many forms and its optimality demands that it should be harmonised with the needs of the given software development situational context. This theoretical proposition is reasonably clear. However, the finer details of the interaction between the software process and the factors of the situational context are much less obvious. In previously published research, the authors have elaborated a reference framework that identifies the factors of a situational context that affect the software process. In this paper, we report on the application of our reference framework in an examination of the changing nature of software development situational contexts. Our corresponding study of fifteen software development companies indicates that certain factors appear more subject to change than others. This finding is a potentially important insight that can help us with the recurring challenge of adapting the software process to changing circumstances.

Keywords: Software engineering · Situational context · Software development process · Software process adaptation

1 Introduction

Software development is a complex activity that is dependent on the performance of many individuals, in a multitude of different settings, and using a variety of development approaches. Recent decades have witnessed the emergence of many different software development approaches, some of which have met with widespread acceptance, including various agile methodologies [1-3] based upon the Agile Manifesto [4], CMMI [5], ISO/IEC 15504 [6] and ISO 9001 [7]. Despite this widespread acceptance of certain approaches, it is still believed that a degree of process adaptation, sometimes referred to as process tailoring, is required in order to address the needs of individual projects [8, 9]. Indeed, the impact of individual project characteristics has long been noted as a key consideration when designing a software process, leading to the claim that the most fundamental requirement of a software process is that it should "fit the needs of the project" [10].

The needs of software projects are dependent on the situational context wherein the project must operate and therefore, the most suitable process can be considered to be "contingent on the context" [11]. For this reason, software developers must "evaluate a

R.V. O'Connor et al. (Eds.): EuroSPI 2015, CCIS 543, pp. 100–111, 2015.
DOI: 10.1007/978-3-319-24647-5_9

wide range of contextual factors before deciding on the most appropriate process to adopt for any given project" [12], thus ensuring that the development approach should "best fit the conditions, product, talent, and goals of the markets and organisations" [13]. It is doubtful that experts in the software development field would argue about the importance of situational context, however, there is only limited published research into the morphology of software development contexts. Grounded in the accumulated knowledge of previous research into risk management, project cost estimation, software development standards, and software process tailoring (among other underlying research themes), the Situational Factors reference framework [14] is considered by the authors to be the most comprehensive source of information on software development contexts presently available. This framework organizes 44 factors of the context known to affect the software process under 8 classifications: Personnel, Technology, Requirements, Management, Application, Business, Organisation and Operations (refer to Figure 1).

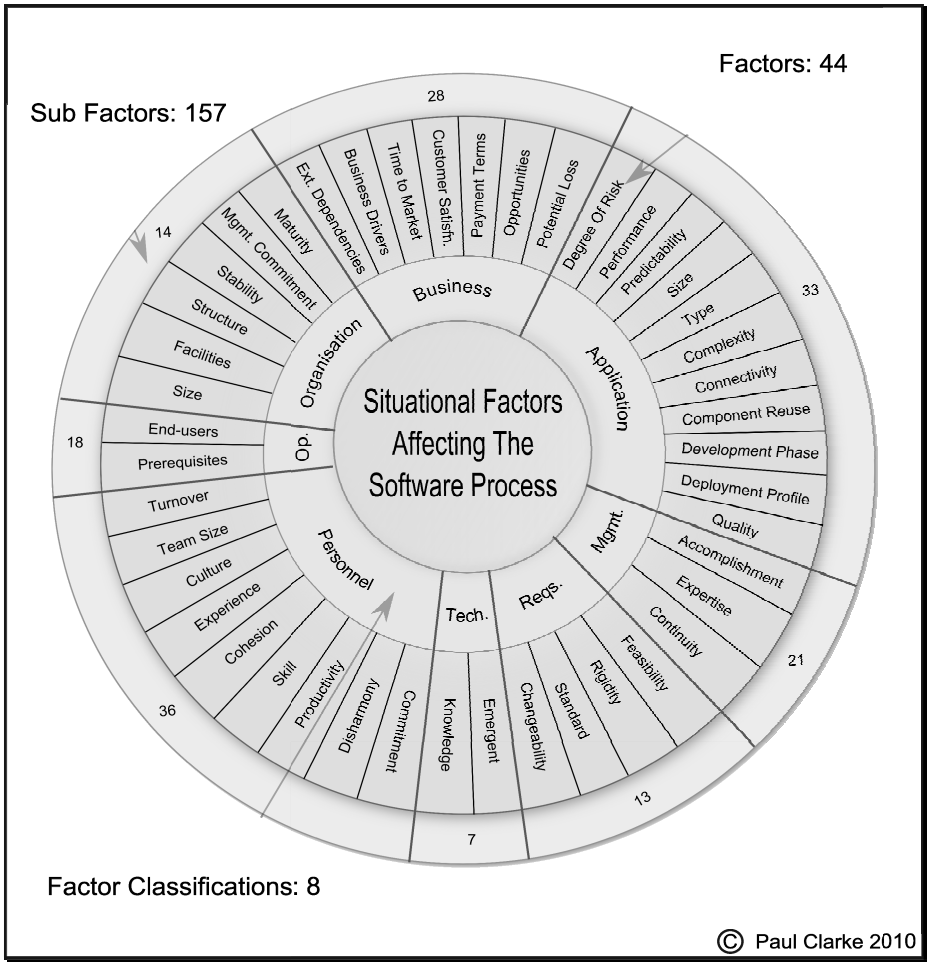

Fig. 1. Situational Factors Affecting the Software Development Process

If we look to the natural order of things in a general sense, we soon discover that change is a recurring challenge. Software developers are not immune to this challenge as they too must perceive changes in their environments and adapt to best address new realities. Of significance, it is recognised that this ability to adapt may be a critical capability for all types of businesses, that it is a key enabler of competitive advantage [15]. In the study presented in this paper, we have examined one aspect of this intriguing yet elusive capability: the nature of change in software development settings. Perhaps unsurprisingly, through the application of the Situational Factors reference framework [14] we have found that contextual change is ubiquitous in software development settings. However, we have also formed some insights into the characteristics of change, which we believe to be an important discovery that should be considered when developing and evolving software development approaches.

The remainder of this paper is structured as follows: Section 2 presents an overview of the study details. In Section 3, we provide an analysis of the data collected, while Section 4 offers a broader discussion on the implications of the data analysis. Section 5 contains the conclusion.

2 Study Details

This section outlines details of the research method, the survey instrument employed and the study timeframe and participants.

2.1 Research Method

This study adopted a mixed method research methodology, an approach that combines quantitative and qualitative techniques to collect, analyse and present both types of data [16]. Mixed method research is pragmatic in nature, it is concerned with designing the method to best suit the study context. Our study context is concerned with the situational factors affecting the software process, and examining this phenomenon ideally requires the collection of both quantitative data (for example, in the case of the reported magnitude of change to situational factors) and qualitative data (for example, in relation to enhanced explanations from participants). With both quantitative and qualitative data required in order to fully explore the research subject, a mixed method approach is therefore desirable, and for which a situational factors survey instrument was designed and discharged.

2.2 Situational Factors Survey Instrument

Since no pre-existing technique was available for examining situational change in software development settings, this study formulated a novel approach which involved transforming the situational factors reference framework [14] into a survey instrument. A key focus of the survey instrument was to provide a profile of the type of situational change that had occurred over the preceding 12 months to situational factors that are known to affect the software development process. The guiding principle was that all of the 44

individual factors in the reference framework should be addressed in individual questions in the survey, and where appropriate multiple questions should be developed for an individual factor (for example, where a large number of sub-factors exist).

Gradually, a series of questions were developed, taking the basic form of: *Have there been any modifications to [an aspect of the situation that can affect the software development process]?* By structuring the questions in this way, it was possible to get information on all changes – no matter how large or how small. This approach permits the elicitation of a comprehensive view of the extent and type of situational changes that have manifested in an organisation. In constructing the survey instrument, the basic classifications and factors of the situational factors framework were preserved. Therefore, the main body of the survey instrument has eight separate sections, one for each of the classifications in the situational factors framework. This step permitted the researchers to more easily guide participants through the survey and to provide updates on progress as the survey instrument was discharged.

Some of the sub-factors from the underlying situational factors framework were also included in the survey instrument for examining situational change. For example, in relation to the *Prerequisites* factor, the following question was developed: *Over the past year, has there been any modification to the operational prerequisites, including applicable standards and laws?* Through using the examples associated with the questions (for example: *applicable standards)*, the finer detail regarding the sub-factors can also incorporated into a question. Using this technique, the fidelity of the underlying situational factors reference framework is significantly retained in the resulting survey instrument. This step was considered important as it ensured that the full scope of the situational factors framework was reflected in the survey instrument.

The survey instrument was subjected to a pilot with an industry partner. The purpose of the pilot was to check that the survey instrument was fit for purpose and that it could be discharged in a practical fashion. Moreover, the pilot was used to check that the participant could relate to and understand the various questions contained in the survey instrument. At the commencement of the pilot, the industry partner was informed that it was a pilot-run and they were encouraged to provide feedback on the content, flow and understandability of the survey. The primary item of feedback was a suggestion to reiterate throughout the survey discharge that the preceding year was the focus of the study, a recommendation which was adopted.

Regarding the content, flow and purpose of the survey instrument, the industry partner was positive concerning the general experience, and felt that the survey instrument provided an interesting mechanism for examining situational changes that affect the software development process. The pilot was the final phase in the survey instrument creation, which ultimately contained a total of 49 individual questions.

2.3 Study Timeframe and Participants

During the period of March to July 2011, the survey instrument was deployed to fifteen organisations, each of which satisfied the European Commission definition of an SME [17]. The majority of the participating organisations were primarily based in the Republic of Ireland, though a number of the companies were principally located elsewhere,

including locations such as the USA and Chile. Three of the participating companies had fewer than 10 staff, with a further 4 companies having between 10 and 19 staff. The remainder of the participating organisations had between 20 and 129 staff. Each interview required approximately 1.25 hours to complete, giving a total of 18.75 hours interviewing time. The interviewee titles included Chief Technology Officer (CTO), Chief Executive Officer (CEO), Engineering Manager (EM), Managing Director (MD), Development Manager (DM), Director of Finance (DF), Director of Engineering (DE), and Chief Operating Officer (COO), with the scope of roles varying from company to company. The primary objective was the elicitation of a complete and accurate information set, and it was therefore sometimes necessary to interview more than one person in a single company.

A listing of the study participants (by role and company pseudonym) is provided in Table 1.[1]

Table 1. Participating organisations and interviewee job title

Company Pseudonym	Interviewee Job Title
Silverback	CTO
Grenoble	CEO, EM
Mega	MD
Cameron	MD, DM
Colleran	CEO
Lakes	MD, CTO
United	MD
Watch	DF, DE
BocaJ	MD
Tribal	DE
Dynamic	DE
Michelin	DE, DM
LordHenry	DE
When	COO
Oryx	COO, DM

Table 2. Modification rating scale for situational change

Modification Value	Modification Interpretation
0	No modification
1	Minor modification
2	Moderate modification
3	Significant modification

When eliciting the responses from interviewees, a modification rating for each reported change was agreed with the participant according to the details provided in

[1] In order to ensure the anonymity of the participating companies, pseudonyms (as opposed to actual company names) are utilised herein.

Table 2. This enabled the elicitation not just of the factors that were subject to change, but also of the extent of change to individual factors. Thus, a richer and more qualified data set was obtained.

3 Data Analysis

A basic analysis of the study data reveals that some aspects of the situational context are routinely reporting relatively large degrees of change while other aspects of the situation are subject to only minor change (or in some cases, no change at all). In this section, we present details of the most and least common areas for situational change. An overview of the hierarchy of situational change is presented in Figure 2.[2]

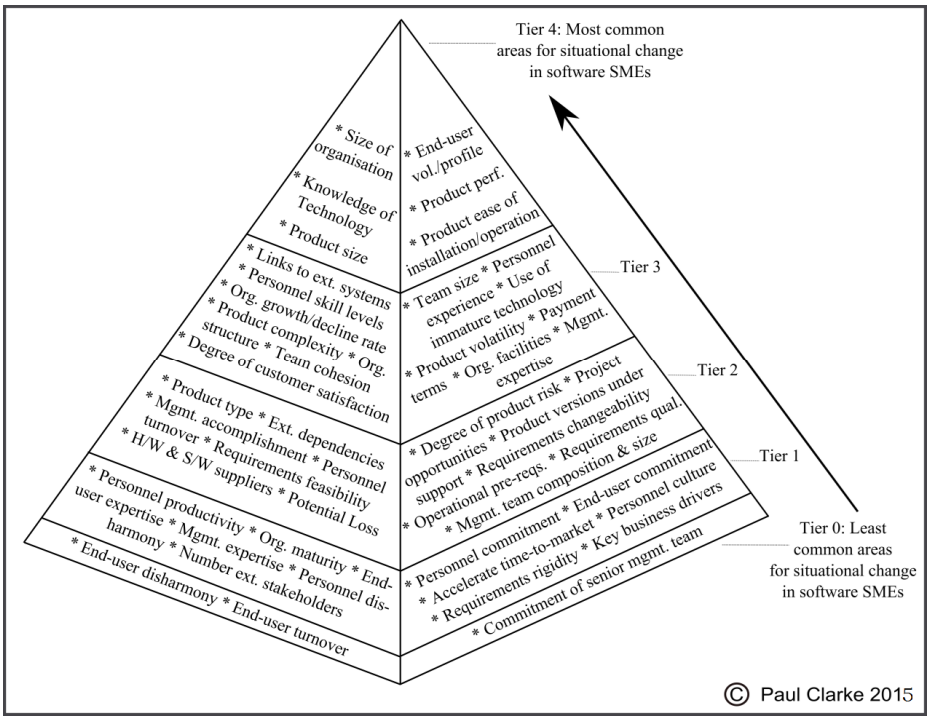

Fig. 2. Hierarchy of situational change for software SMEs

[2] The hierarchy presented in Figure 2 was constructed by analysing the responses from all participants to each question in the situational change survey instrument. Both the frequency and the amount of reported change in each situational aspect were jointly considered, with more frequent and/or more significant situational changes being placed higher on the pyramid.

3.1 Situational Factors Reporting Change

Staff headcount presented as the most common area for situational change across the study group. All fifteen participating companies reported changes to their headcount, with 11 of the companies reporting increases to headcount levels during the study timeframe. Nine of the participating companies witnessed headcount increases of 25% and greater over the year under investigation, while two of the organisations experienced headcount reductions of 40% or more during the same period. These reported changes to headcount figures represent significant fluctuations, indicating that headcount volatility is a major challenge in small and medium sized software companies. Such volatility is a major catalyst for process change, and may even suggest that small and medium sized software companies are more in need of regular process management than larger, more stable organisations.

The participating companies also reported considerable change in the volume and profile of end users of their software products, with 11 companies reporting increases to the net volume of transactions processed or to the volume of end users. In some of these cases the organisations reported a significant increase in the number of end users or volume of transactions that their products must support. Of note, not a single company reported a reduction in the number of end users or volume of transactions in their products. Furthermore, two of the participating companies reported changes to the profile of end users, resulting in a need for their products to cater for different types of end users.

Of the participating companies, 13 reported increases in the knowledge of technology. In some cases, this involved supporting new operating systems, such as Linux and emerging mobile device operating systems. In other cases, these changes were focused on the software development infrastructure, including changes to integrated development environments (IDEs) and changes to compilers. While the extent of the reported change in knowledge of technology varies across the study group, almost all of organisations reported an increase in their knowledge of technology mostly through the adoption and utilisation of emerging technologies.

The majority of the participating companies, twelve in total, also reported an increase in the required performance of their products. Those companies that did not report increases in performance requirements did not report decreases either, but rather that the performance requirements remain unchanged. In terms of the size of the products, 10 of the participating companies reported increases in one form or another. For some organisations, this increase took the form of increased data storage requirements while for other organisations, the reported change relates to the size of the code base. One of the companies reported a slight decrease in the code base as a result of an intensified refactoring effort.

A total of nine of the participating organisations reported that an increase in the required ease of installation and operation of their software products – emphasising the need to continually improve the installation procedures and to constantly strive to improve the end user experience of their product(s).

3.2 Situational Factors Reporting Little or No Change

Of all the situational factors examined in the study, just a single factor was reported as unchanged in all of the participating companies: *senior management team commitment to projects*. With respect to this finding, it should be noted that the personnel participating in the study were senior managers – who might be unlikely to report a decrease in their commitment to their project(s). Just two of the participating organisations reported a change to the number of external stakeholders over the period of investigation. In one case, this was the result of engaging external systems integrators on a more regular basis. In the case of the second organisation, the opposite effect was reported: systems integrators were no longer being used to deploy systems but rather the company had started to work more directly with its clients.

There was little reported change regarding the turnover of product end-users. For some companies, this was accounted for by the nature of their product(s). For example, several organisations developed middleware applications with little or no direct end-user interaction (but rather just a few specialised users would configure or interact with the product). The participating organisations also reported little or no change in the personnel culture, with disharmony levels (including interpersonal conflicts) remaining largely the same as in early periods. This finding is perhaps surprising when we consider the reported headcount volatility in the participating group – as the introduction of new people can be accompanied by friction within teams.

4 Discussion

There are a number of features of the data analysis that serve to highlight the challenges imposed on small and medium sized software companies as a result of changing situational contexts. Perhaps the most striking of these challenges is the rate of growth or decline in the *Organisational Size* situational factor (i.e. headcount). While it is to be expected that each of the companies might report some change to headcount, it was not anticipated that the rate of change would be so significant. Two of the participating companies witnessed a reduction in headcount of 40% or greater - losing 15 out of 35 employees in one case, and 8 out of 20 employees in the second case. A further 9 of the 15 participating companies experienced headcount growth of 25% or higher, with 6 of these organisations growing their headcount by 50% or more. In some cases, these percentage increases are partly accounted for by the fact that the organisations were very small at the start of the study. However, in other cases, a relatively large number of new personnel were introduced to the participating company. Large changes in headcount can have a significant effect on the process of work, and the data collected in this study suggests that this is not an uncommon phenomenon in small and medium sized software companies.

A further significant challenge potentially originates from changes reported in the *Operational End-users* situational factor, for example through increases in the volume of transactions that software products must process. Ten of the participating companies reported increases in the volume of transactions. Some of the organisations note that their "*traffic continues to grow*", with others reporting the increase to be *"very*

significant". Further evidence of this challenge is evident in related responses that *"storage requirements are growing"* and that an increasing database size *"is one of the biggest challenges that we have now"*. Increased throughput and storage demands can place a heavy burden on software products, and increasing throughput and storage capacity in a system is an area that may require specialised and costly attention. However, small and medium sized software companies don't necessarily have a great deal of bandwidth or expertise in terms of addressing such challenges. Therefore, along with headcount challenges, increased demand for product throughput and storage may necessitate a change to the software process.

The issue of ever-increasing transaction volumes and storage issues may be further exacerbated by the reported demand for increases in the *Application Performance* situational factor and in this respect, it is possible that in fact, increases in *Operational* factors (such as the volume of transactions or end-users) are one of the primary drivers for change in the *Application Performance* situational factor. Twelve of the participating companies reported an increased required performance in their product(s) over the year under investigation. One of the organisations reported that the performance requirement had *"gone up by 20%",* while a second company reported that performance had *"increased probably by 30%"*. Another organisation again reported that their product *"has to run twice as fast"*. Other companies reported that the increased performance requirement is *"significant"*, that it is *"always increasing",* and that *"the customer is always demanding more fast and more reliable [products]"*. This increased demand for higher performance may present a significant challenge to the limited resources at the disposal of small and medium sized companies - as performance-related activities may detract from the development and evolution of product features designed to attract customers. Such activities may also affect the software process, for example, testing mechanisms and hardware selection processes may undergo change.

In addition to the challenges already noted, indications from the data are that the *Technology Knowledge* situational factor is also subject to significant change, and this suggests that the supporting technologies are themselves continually changing. Eight of the participating companies reported that they adopted new technologies over the year under investigation. Some of the organisations report changes to the programming related environment, including languages, compilers and associated tooling. Other companies report that they had to support additional operating systems, including traditional desktop operating systems and newer mobile device operating systems. The adoption of new technology requires effort, with one respondent stating: *"we've started using several different technologies over the past year and people have had to skill up on them and share their knowledge among the team"*. And such upskilling and knowledge sharing may require a change to the underlying software process. For example, it may be necessary to introduce a process – if only for a period of time – that enables knowledge sharing across the relevant parties.

5 Conclusion

In the broader business domain, it is acknowledged that the ability of an organisation to adapt their business processes in response to changing situational contexts may be a key source of competitive advantage [15]. For software development companies, the software process is a large and complex component of the overall business process, and therefore an area, which is a potential source of competitive advantage. However, there are very significant challenges when aligning changes in situational context with software process changes – as has been highlighted [18]: the essential observation being that the complexity in the relationship between the process and the corresponding context is too large to fully qualify. Our best route forward with this problem may therefore start with gaining a greater appreciation of software development situational contexts, and in particular in understanding which aspects of such contexts are witnessing more frequent or greater degrees of change. If the key concerns of context can be identified, then the problem may be reduced – and those tasked with developing and evolving software process approaches can consider incorporating mechanisms for process adaptation that are aligned with situational factors that witness more frequent or greater change – this is the essential importance of this work.

The research presented herein exhibits a number of limitations and weaknesses. It is limited to just fifteen companies, and these companies are either small or medium in size. Hence, the generalisability of findings requires further investigation. The data is collected from a small number of individuals within the participating companies and thus it exhibits the weakness that it may not be a complete view of the actuality of the context changes in the participating companies. Although difficult to realise from a practical perspective, broader focus group discussions may have helped to reduce the bias introduced by working with just one or two individuals from each of the companies. Nonetheless, the inquiries conducted required approximately 19 hours of interviewing time with participants, and considerable detail was obtained. Furthermore, a comprehensive situational factors reference framework [14] (itself a product of related research) was adopted, and this was carefully crafted into a survey instrument that was subject to an industrial pilot as part of its validation.

On the subject of the general nature of situational change in the participating companies, a number of our key findings from this exploratory research suggest that changing situational contexts present a constant and significant challenge to software developers. High levels of headcount volatility are consistently reported across the study group, which inevitably means that regular process adaptation is required. This may account for the finding in related published material that software process adaptation is a regular occurrence [19], and that greater levels of process adaptation are positively associated with increased business success [20]. It may also legitimise efforts to model the relationship between the software process and its corresponding situational context [21-23].

Of interest also is the revelation that there is some potential discordance with previously reported key areas for process improvement in smaller companies. For example, a number of previous studies highlight the importance of improvement to requirements capture in smaller companies [24, 25], whereas the findings from our

inquiry suggest that requirements changeability and requirements quality are not witnessing relatively high levels of change. However, it should be stressed that these earlier studies were not looking to the broader business challenges but focused just on the core software development process (i.e. situational factors related to *Organisation* and *Operation* fell outside the scope of these earlier software process improvement studies). With these organisational and operational factors reporting relatively high levels of change within our study, it is perhaps the case that the role of such factors and their influence on the software process is not sufficiently well elaborated upon at the present time. If we take any of the presently popular software development approaches (e.g. agile methodologies [1-3], CMMI [5], ISO/IEC 15504 [6] and ISO 9001 [7]), there exists a software lifecycle centric focus that does not appear to offer direct provisions for dealing with significant fluctuations in headcount. This gap appears worthy of further investigation – as on the basis of the evidence collected in our study, there is constant and sometimes significant change to headcount.

Since software development contexts are continually changing, it would therefore be advantageous for researchers and practitioners to focus some of their energies on the objective of enhancing our understanding of the relationship between software contexts and software processes - and studies such as the one reported herein represent an important initial step along the journey to realising this objective.

Acknowledgments. This work is supported, in part, by Science Foundation Ireland grant 03/CE2/I303_1 to Lero, the Irish Software Research Centre (www.lero.ie).

References

1. Schwaber, K., Beedle, M.: Agile Software Development with SCRUM. Prentice Hall, Upper Saddle River (2002)
2. Beck, K.: Extreme Programming Explained: Embrace Change. Addison-Wesley, Reading (1999)
3. Poppendieck, M., Poppendieck, T.: Lean Software Development: An Agile Toolkit. Addison-Wesley, Upper Saddle River (2003)
4. Fowler, M., Highsmith, J.: The Agile Manifesto. Software Development, pp. 28–32 (2001)
5. SEI: CMMI for Development, Version 1.3. CMU/SEI-2006-TR-008. Software Engineering Institute, Pittsburgh, PA, USA (2010)
6. ISO/IEC: IS0/IEC 15504: Information Technology - Process Assessment, Part 1 to Part 5. International Organisation for Standardization, Geneva, Switzerland (2005)
7. ISO: ISO 9001:2000 - Quality Management Systems - Requirements. International Organisation for Standardization, Geneva, Switzerland (2000)
8. Coleman, G., O'Connor, R.V.: Investigating software process in practice: A grounded theory perspective. Journal of Systems and Software **81**(5), 772–784 (2008)
9. Kautz, K.: Software process improvement in very small enterprises: does it pay off? Software Process: Improvement and Practice **4**(4), 209–226 (1998)
10. Feiler, P., Humphrey, W.: Software Process Development and Enactment: Concepts and Definitions. Software Engineering Institute, Carnegie Mellon University, CMU/SEI-92-TR-004. Pittsburgh, Pennsylvania, USA, (1992)

11. Benediktsson, O., Dalcher, D., Thorbergsson, H.: Comparison of software development life cycles: a multiproject experiment. IEE Proceedings - Software **153**(3), 87–101 (2006)
12. MacCormack, A., Verganti, R.: Managing the Sources of Uncertainty: Matching Process and Context in Software Development. Journal of Product Innovation Management **20**(3), 217–232 (2003)
13. Subramanian, G.H., Klein, G., Jiang, J.J., Chan, C.: Balancing four factors in system development projects. Communications of the ACM **52**(10), 118–121 (2009)
14. 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)
15. Teece, D.J.: Explicating dynamic capabilities: the nature and microfoundations of (sustainable) enter-prise performance. Strategic Management Journal **28**(11), 1319–1350 (2007)
16. Petter, S.C., Gallivan, M.J.: Toward a framework for classifying and guiding mixed method research in information systems. In: Proceedings of the 37th Hawaii International Conference on System Sciences, January 5–8. IEEE Computer Society, Los Alamitos (2004)
17. Commission, European: Commission Recommendation of 6 May 2003 concerning the definition of micro, small and medium-sized enterprises. Official Journal of the European Union, L. 2003/361/EC **124**, 36–41 (2003)
18. Dyba, T.: Contextualizing empirical evidence. IEEE Software **30**(1), 81–83 (2013)
19. Clarke, P., O'Connor, R.V.: An empirical examination of the extent of software process improvement in software SMEs. Journal of Software: Evolution and Process **25**(9), 981–998 (2013)
20. Clarke, P., O'Connor, R.V.: The influence of SPI on business success in software SMEs: An empirical study. Journal of Systems and Software **85**(10), 2356–2367 (2012)
21. Jeners, S., O'Connor, R.V., Clarke, P., Lichter, H., Lepmets, M., Buglione, L.: Harnessing software development contexts to inform software process selection decisions. Software Quality Professional **16**(1), 35–36 (2013)
22. Jeners, Simona, Clarke, Paul, O'Connor, Rory V., Buglione, Luigi, Lepmets, Marion: Harmonizing software development processes with software development settings – a systematic approach. In: McCaffery, Fergal, O'Connor, Rory V., Messnarz, Richard (eds.) EuroSPI 2013. CCIS, vol. 364, pp. 167–178. Springer, Heidelberg (2013)
23. Clarke, Paul, O'Connor, Rory V.: An Approach to Evaluating Software Process Adaptation. In: O'Connor, Rory V., Rout, Terry, McCaffery, Fergal, Dorling, Alec (eds.) SPICE 2011. CCIS, vol. 155, pp. 28–41. Springer, Heidelberg (2011)
24. Keane, Brendan, Richardson, Ita: Quality: attitudes and experience within the irish software industry. In: Richardson, Ita, Abrahamsson, Pekka, Messnarz, Richard (eds.) EuroSPI 2005. LNCS, vol. 3792, pp. 49–58. Springer, Heidelberg (2005)
25. Sanders, M.: The SPIRE Handbook. Better, Faster, Cheaper Software Development in Small Organisations. Centre for Software Engineering Limited, DCU, Dublin, Ireland (1998)