# An approach to evaluating software process adaptation

Paul Clarke [1] and Rory V. O'Connor [2, 3]

[1] Lero Graduate School in Software Engineering, Dublin City University, Ireland
pclarke@computing.dcu.ie
[2] Dublin City University, Ireland
[3] Lero, the Irish Software Engineering Research Centre
roconnor@computing.dcu.ie

**Abstract. P**rocess maturity reference frameworks such as ISO/IEC 15504 and the Capability Maturity Model Integrated (CMMI) seek to assist software process improvement (SPI) efforts by prescribing a roadmap for improving the capability of the development process. However, such frameworks are not widely adopted in the practice [1], [2], especially in smaller software development organisations where the development process is often modified based on business events [3]. Such modification of the development process represents an attempt to harmonise the process with the changing needs of the business, which is a dynamic capability. Dynamic capabilities refer to the ability of businesses to adapt to changing circumstances and according to the *evolutionary* theory of the firm [4], organisations that possess greater dynamic capability are more successful. This paper introduces *dynamic SPI capability* - the ability to adapt the software process relative to changing situational circumstances – as a method for evaluating software process adaptation.

**Keywords:** SPI, process adaptation, dynamic SPI capability.

## 1 Introduction

The past two decades have witnessed significant growth in the software development business and in parallel there has been a sustained investment in research into the process of software development. One of the principal developments in the software process domain has been the emergence of process maturity frameworks, with ISO/IEC 15504 [5] and CMMI [6] considered the most dominant [7-9]. Although process maturity reference frameworks are sometimes criticised for being cumbersome and costly [10], ISO/IEC 15504 [5] and CMMI [6] have been shown to deliver significant benefits for software development [11-16].

While the benefits of process maturity frameworks and other software development models have been demonstrated, they are not widely adopted in practice [1-3], [17], [18]. Furthermore, some research suggests that temporal contextual factors are critical in identifying the most appropriate process [10], [19], [20], especially in SMEs [21]. It is therefore not surprising to discover that the software development process has been reported as being volatile and that process improvements are often initiated in response to business events [3]. The ability of a business to learn from business events

and to improve business processes to address changing needs is what economists describe as a dynamic capability – and according to the *evolutionary* theory of the firm [4], organisations that have greater dynamic capability are more likely to be successful. Therefore, a dynamic capability to improve the software development process with respect to changing circumstances, a characteristic which we have defined as *dynamic SPI capability*, is beneficial for business success in software development organisations.

Given that the dynamic SPI capability of an organisation is important for business success, it would be advantageous for organisations to be able to observe their dynamic SPI capability. However, to date SPI research efforts have been concerned largely with process capability rather than with dynamic process capability and as a result there is no existing mechanism for observing the dynamic SPI capability of a software development setting. Therefore, this paper identifies the key considerations when attempting to determine the dynamic SPI capability in a software development setting and proposes an approach to unifying these considerations so as to bring visibility to this important capability.

This paper is structured as follows: Section two provides additional background information on the relevance of dynamic capability to software development organisations. Sections three and four outline approaches to making determinations in relation to the two components of dynamic SPI capability: extent of SPI activity and extent of situational change. Section five identifies related future work that the authors intend to carry out and finally, section six presents a discussion and conclusion.

## 2 Dynamic Capability

In the field of economics, the evolutionary theory of the firm [4] is concerned with the concept of dynamic capability. Dynamic capability relates to the ability of an organisation to continually transform the business routines in response to changing environments and new understandings, and the evolutionary theory of the firm suggests that this ability gives rise to the dynamism that will ultimately propel the organisation to success [22]. The firm, therefore, is promoted as "a locus where competencies are continually built, managed, combined, transformed, tested and selected", where the vital consideration relates to how "new knowledge [is] materialised in new competencies", and where "a lock-in to inefficient routines" is perceived as a major threat to a company's prospects [23]. Consequently, a dynamic capability to transform routines is considered to provide a basis for competitive advantage [23], a point that has already been observed in relation to the software development routines by Poulin [24], who suggests that with respect to software process capability, establishing an organisation's ability to optimise the development process may provide a better approach than traditional audits. Therefore, rather than examining process capability and prescribing an improvement path, an alternative view suggests that one should focus on maximizing the capability to transform the process, and that this transformational capability will automatically render an improved process.

The initial stage on a process maturity roadmap generally represents a state of low process capability, with subsequent stages gradually enhancing the process implementation, finally culminating with the process optimisation stage, wherein the software development process is continually being optimised in order to best address the software development needs of the organisation. Therefore, in a sense, existing process maturity frameworks do consider dynamic SPI capability – but only at the highest level of maturity. However, where process maturity reference frameworks are adopted, the highest maturity level (optimising) is not often achieved [25]. Furthermore, the evolutionary theory of the firm establishes that it is dynamic capability, rather than process maturity, that is of paramount importance for optimum process adaptation. Perhaps this raises the case for process optimisation to be brought forward as a consideration in process maturity frameworks, into a position where process optimisation is an inherent consideration for every process decision at every level in a process maturity framework. This consideration is, however, one for the committees responsible for maintaining the various process maturity frameworks – the focal point of this paper is to outline the components of dynamic SPI capability along with some mechanisms for determining the dynamic SPI capability of a software development setting.

In order to determine the dynamic SPI capability of a software development setting, two principal components must be established: (1) the extent of SPI activity and (2) the degree of situational change. The following sections discuss these two key components.

## 3 Extent of SPI Activity

In order to examine the dynamic capability with respect to the software development process, it is first necessary to determine the extent of SPI activity in an organisation. For the purpose of this paper, SPI activity is defined as *"the set of SPI actions implemented by an organisation, which is manifested as a series of modifications to the software development process"*. A review of the literature in the process assessment and auditing domain reveals that there is no dedicated approach to expressly examining the amount of software development process change that has occurred over a period of time. By *expressly,* we mean (1) in a single engagement and (2) collecting only data that is related to the extent of process change (not process capability). Since no express method for determining the amount of SPI activity pre-existed, it was necessary to develop a dedicated SPI activity survey instrument. Such a survey instrument would need to be systematically derived from a comprehensive and recognized software development process reference framework. ISO/IEC 12207 [26], which has been developed and maintained by international consensus, offers an ideal reference framework for the construction of an SPI activity survey instrument.

The ISO/IEC 12207 [26] based SPI activity survey instrument developed and presented below can be used to examine the extent of SPI activity. This instrument is different from traditional process assessments in that it directly and explicitly examines the extent of SPI actions and is not concerned with making process capability determinations. With the software development process constituting an

important and complex component of the overall business process for software developing organisations, and acknowledging the importance of dynamic process capability as encapsulated in the evolutionary theory of the firm [4], software development and quality management practitioners, as well as auditing agents, can apply the SPI activity survey instrument in order to directly determine the extent to which the software development process is being evolved. Researchers can also use the SPI activity survey instrument, and the authors of this paper are presently applying the approach as part of a broader research project that is examining the influence of SPI on the evolution of small to medium sized (SME) software development companies.

It is possible to utilise the process assessment vehicles associated with process maturity reference frameworks in order to determine the amount of SPI activity. This would involve conducting two process assessments on two different dates, and thereafter performing a finite difference analysis on the assessment results. However, this twin assessment approach has a number of drawbacks. Firstly, it requires two engagements with the software development organisation, which is time consuming and which can be difficult to orchestrate from a practical researching perspective. Secondly, process assessments, such as those in ISO/IEC 15504 [5] and CMMI [6] collect data and generate a maturity ratings rather than just investigating the amount of SPI activity, and therefore represent a somewhat inefficient tool for evaluating SPI activity. Thirdly, adopting an ISO/IEC 15504 [5] or CMMI [6] process assessment vehicle to determine SPI activity might diminish the capacity to secure candidate participants in the SME sector, since prescribed process maturity reference frameworks have themselves already met with resistance to implementation in SMEs. For these three reasons, traditional process assessments would represent an inefficient use of resources for the purpose of making express SPI activity determinations.

Taking these drawbacks into account, and owing to the apparent absence of any established dedicated resource for determining the amount of SPI activity, we developed a new method for evaluating SPI activity, a method based around the application of a dedicated SPI activity survey instrument.

## 3.1 Evaluating SPI activity using a dedicated survey instrument

In the case of ISO/IEC 15504 [5], the ISO/IEC 12207 [26] process listing is used as the underlying process reference list. ISO/IEC 12207 [26] is an internationally developed and maintained listing for software processes and therefore represents a useful reference point when examining software processes in any setting.

It is the premise of this paper that in order to evaluate the amount of SPI activity in an organisation, ISO/IEC 12207 [26] can be used as a comprehensive point of reference. However, the creation of a survey instrument based on ISO/IEC 12207 [26] needs to be structured and systematic, and this paper presents an approach suited to converting an international standard into a survey instrument, followed by an explanation of how the method was applied in the case of transforming ISO/IEC 12207 [26] into an appropriate survey instrument for evaluating the extent of SPI activity in an organisation.

### 3.1.1 Method for converting an international standard to a survey instrument

Many international standards consist of verbose text that seeks to accurately and completely describe an item of technical matter. However, such comprehensive text-based descriptions are not easily fashioned into survey instruments, especially when practical considerations, such as the time required to conduct the survey, are taken into consideration. Therefore, this paper outlines a technique for resolving verbose text-based international standards back to comprehensive, yet practical, survey instruments. An overview of this technique is presented in figure 1.

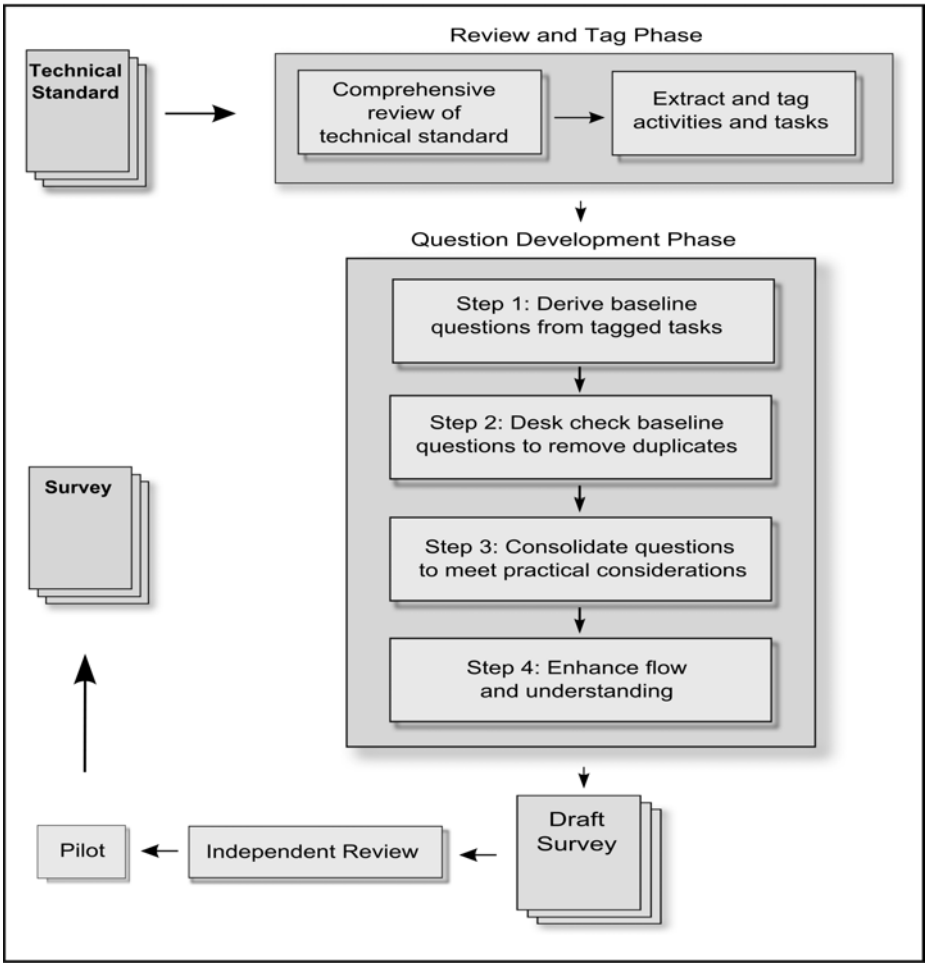


**Fig. 1.** Survey Instrument Development Technique

The initial phase, the *Review and Tag* phase, involves reviewing the international standard, so as to develop a thorough understanding of all the material comprising the standard. Thereafter, the various components of the international standard are tagged

– in order to identify the key activities. This requires that close attention is paid to all actions in the international standard, ensuring that no important detail is overlooked.

Following the tagging exercise, the *Question Development* phase is undertaken. This is a four-step activity that involves transforming the tagged details, as output from the initial phase, into a representative, accurate, comprehensive and readable survey instrument. Notes that explain any modifications, along with rationale for changes, must be maintained at each step in the question development phase – this allows for later examination of the survey construction exercise, including the possibility of auditing the artefacts so as to verify that appropriate decisions have been taken throughout the survey construction activity. Such artefacts can thereafter be published along with the survey findings if required.

The first step of the question development phase involves using the tagged details in order to derive a baseline set of questions. This results in a baseline suite of questions that preserve all of the essential details that are present in the international standard itself. In the second step of the question development phase, the baseline suite of questions is desk-checked so that any duplications or areas of overlap are resolved. This is necessary in order to efface cross-references that can exist in international standards.

The third step of the question development phase consolidates the list of questions with respect to practical considerations. The target survey duration is among the practical considerations, and the survey constructor must judge the appropriate type and number of questions for the survey. The consolidation of questions also requires a considerable deal of judgement, coupled with expertise, on the part of the survey constructor, but should nonetheless seek to preserve the original makeup and structure of the international standard, retaining all major components such that the resulting survey is clearly identifiable as a derivative of the original standard. Having consolidated the questions in an appropriate fashion, the fourth and final step of the question development phase involves reviewing the survey so as to enhance the clarity of individual questions and to optimise the flow of the survey so as to best achieve the survey objectives.

Having completed the question development phase, the survey constructor presents a draft version of the survey instrument to software process and process standards domain experts so as to elicit independent feedback on the content, accuracy, and likely effectiveness of the interview in obtaining the required information. Following completion of the independent review, the survey instrument should be revised so as incorporate the feedback from the expert reviewer. Once again, a copy of the changes applied should be maintained so as to allow for later examination of the technique.

### 3.1.2 Application of conversion method to ISO/IEC 12207

The technique identified in figure 1 was applied to ISO/IEC 12207 [26] and a systematically-derived draft SPI activity survey instrument was produced. This draft survey instrument was submitted to key ISO/IEC 12207 [26] editorial committee members for review, after which a final rendering of the SPI activity survey instrument was produced. Further details of the survey instrument creation can be found in [27]. In producing the SPI activity survey instrument, a detailed review of

ISO/IEC 12207 [26] was carried out. Following this review, a diagrammatical overview of the activities contained in ISO/IEC 12207 [26] was produced – this overview is reproduced here in figure 2.

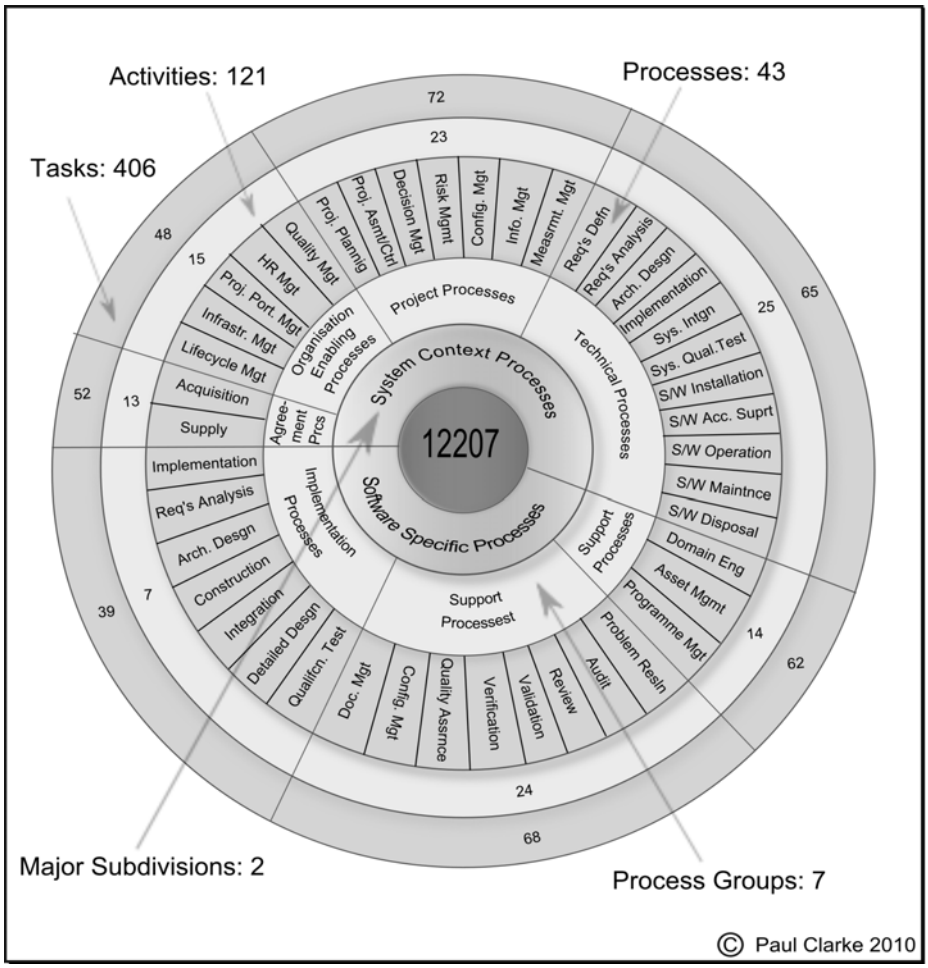


**Fig. 2.** ISO/IEC 12207 Topology

This section has outlined the approach adopted in constructing an express SPI activity survey instrument from ISO/ISE-12207 [26]. However, in order to determine the dynamic SPI capability of a software development setting, it is also necessary to determine the extent of situational change that has occurred.

## 4 Extent of Situational Change

In order to examine the extent of situational change in a software development setting, it is necessary to utilise a comprehensive reference framework of the factors that affect the software development process – rather than being concerned with changes to the general environment, in examining dynamic SPI capability, we are only interested in changes to the environment that are potential triggers for SPI. However, a literature review of the domain confirms that no single general and comprehensive reference framework of the situational factors that affect the software development process presently exists. Therefore, the authors have systematically developed such a reference framework of the factors that affect the software development process [28]. As with the development of the express SPI activity survey instrument outlined earlier, a systematic approach was adopted in the development of the reference framework of the situational factors that affect the software development process. This involved a synthesis of twenty-two of the most influential works from seven distinct domains: software development models and standards, risk factors for software development, software development cost estimation, software development environmental factors, software process tailoring, degree of required software process agility, and the software engineering book of knowledge [29]. The approach to developing the reference framework of the factors affecting the software process is outlined in the figure 3.

The framework construction exercise utilised numerous data sources, including Boehm's Cost Constructive Model (CoCoMo) [30], Putnam's SLIM model [31], Albrecht's Function Point Analysis (FPA) [32], CMMI [6], and ISO/IEC 12207 [26]. Following the *Review and Tag* phase, a baseline of three hundred and ninety-seven factors affecting the software development process was identified. Since the data sources are thematically related, this initial baseline contained some duplication – both literal and conceptual – and therefore it was necessary to distil a consolidated reference framework. This distillation exercise required a concentrated data analysis effort and therefore, data analysis techniques from Grounded Theory [33] were applied so as to ensure that a rigorous and systematic approach was adopted. Borrowing the *constant comparison* [34] and *memoing* [35] data analysis techniques from Grounded Theory, the baseline set of factors was systematically consolidated into a comprehensive set of forty-four individual situational factors that affect the software development process. These factors are classified under eight categories and have a total of one hundred and fifty-seven sub-factors, as outlined in figure 4.

### 4.1 Evaluating situational change using a dedicated survey instrument

The situational factors affecting the software development process indicated in figure 4 are used as a reference framework for the development of a comprehensive survey instrument that will determine the extent of situational change that has occurred over a period of time. In keeping with the approach adopted in the earlier SPI activity survey instrument, again a systematic technique is adopted. The *Question*

*Development*, *Independent Review* and *Pilot* phases identified in figure 1 are re-used – this time to develop a survey instrument from the reference framework of the factors affecting the software development process. The resulting survey instrument will facilitate a determination of the amount of situational change that has occurred by examining changes to the factors affecting the software development process over a period of time.

In section 3, we introduced an express approach to determining the amount of SPI activity in a software development setting over a period of time. This section has identified an approach to determining the extent of situational change in a software development setting. In order to determine the dynamic SPI capability in a software development setting, we must unify these two components.

## 5 Future Work - Dynamic SPI Capability

As outlined earlier, dynamic SPI capability relates to the ability to adapt the software development process in tune with changing situational circumstances. Furthermore, two key components are required in order to determine the dynamic SPI capability in a software development setting: (1) the extent of SPI activity, and (2) the extent of situational change. Earlier sections have identified approaches to determining both of these components. The next stage in our research involves the identification of an appropriate method for integrating these two components into a form that enables the visualisation of the dynamic SPI capability – this is a work in progress but at present there are two candidate approaches.

Firstly, the two distinct components could be unified into a single key performance indicator (KPI). This would involve expressing the one component in a ratio form relative to the other component. There are a number of benefits to having the dynamic SPI capability accessible in a single KPI numeric form; for example, it would be possible to easily compare the dynamic SPI capability over time in different settings and it would be permissible to offer a guidance scale of performance in relation to SPI. However, there are some challenges to integrating these two components into a single KPI: for example, the two components are quite different in nature and therefore, it may not be useful to attempt to integrate them into a single numeric form.

Secondly, the two distinct components could be visualized in a four-quadrant type graph, where the first dimension would depict the extent of the SPI activity and the second dimension would capture the extent of situational change. This particular approach would overcome the noted challenge in relation to unifying the two components into a single numeric form – however, it also has some drawbacks; for instance, it would be more difficult to compare the dynamic SPI capability in two different settings.

Over the coming months, the authors will be actively examining the options for unifying the two components of dynamic SPI capability that have been outlined in this section.

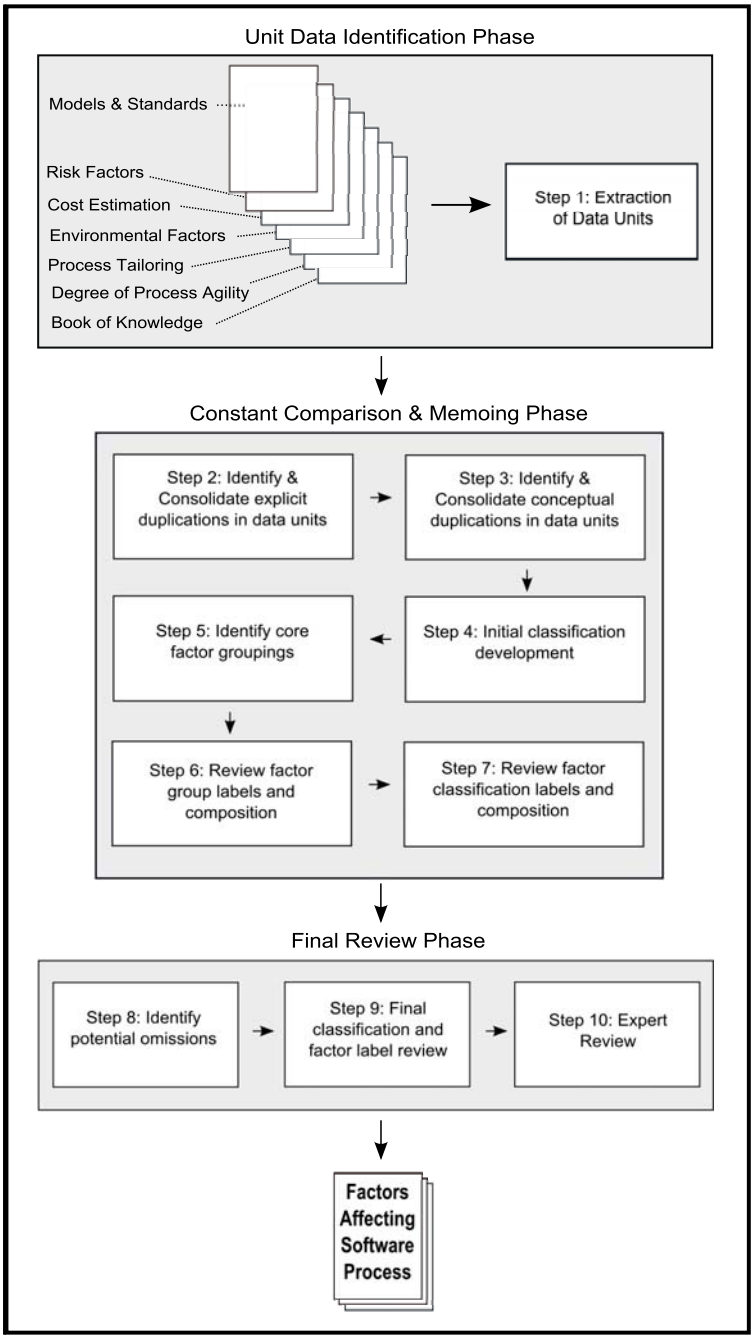


**Fig. 3.** Situational Factors Reference Framework Development Technique

## 6 Discussion and Conclusion

The quality of the software development process directly affects the quality of the software product, and since the technology, business environment and company circumstances are subject to continual change, there is an ongoing requirement for SPI. Existing approaches to SPI, such as ISO/IEC 15504 [5] and CMMI [6] assess the capability of processes in an organisation. These process maturity reference frameworks prescribe a phased process maturity roadmap, with the earlier stages characterised by minimum process implementation and the later stages gradually improving the process maturity, with the final stage being dedicated to continuous process optimisation.

The concept of process optimisation is related to the evolutionary theory of the firm [4], which suggests that the dynamic capability of an organisation to modify its business processes is an important driver for business success. If it is the case that dynamic capability is central to the formula for business success, then software development organisations would benefit from being dynamically capable with respect to the software development process. Process maturity frameworks such as ISO/IEC 15504 [5] and CMMI [6] do acknowledge process optimisation as an important attribute, but it is only evident at the most mature stage. Therefore, organisations that adopt such process maturity references frameworks, and who do not progress to the most mature stage, may fail to realise the benefits of dynamic capability as described by the evolutionary theory of the firm [4].

If dynamic capability is important, and we suggest that it is, then there should be a method for examining the dynamic process capability in an organisation. For software development organisations, this includes the ability to examine dynamic capability with respect to the software development process, or to use the term introduced in this paper: *dynamic SPI capability*. In order to observe dynamic SPI capability, it is necessary to determine the amount of SPI that has taken place in a software development environment setting over a period of time, and to concurrently examine the changes in the situational context that are important considerations for the software development process. This paper outlines two mechanisms: one for making express determinations in relation to the amount of SPI activity that has occurred and a second mechanism that enables the determination of the amount of change that has occurred in the situational context. Together, these two mechanisms can be combined to make an overall determination in relation to the dynamic SPI capability of a software development setting over a period of time.

It is not the intention of the approach outlined in this paper to devalue the very significant benefits and contributions that are already available in capability maturity frameworks such as ISO/IEC 15504 [5] and CMMI [6]. However, as a research community we must accept that despite being in existence for a long number of years, these approaches are not widely adopted across the broader spectrum of the software development industry. Equally, where process maturity frameworks are implemented, they are often tailored to individual settings and software development settings themselves are subject to changing circumstances on a regular basis. Consequently, an important characteristic of a software development process is its ability to continually adapt to meet the needs of the changing environment – and one could argue that the

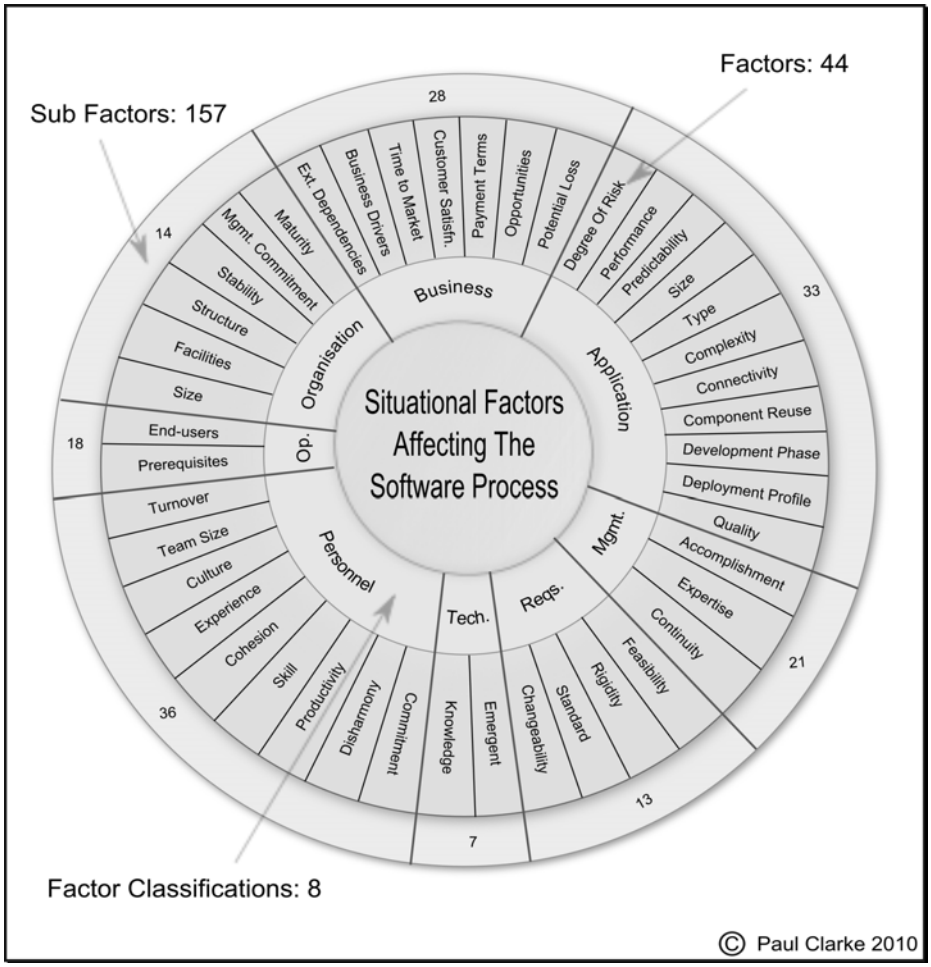


**Fig. 4.** Situational Factors Affecting the Software Development Process

amount of process adaptation required in a setting is a function of the degree of situational change in that setting. By developing an approach to examining situational change, and through the application of the dynamic capability concept to the software development process, this paper has presented a systematically-derived and novel approach to evaluating software process adaptation in a software development setting.

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

## References


1. McConnell, S.: Closing the Gap. IEEE Software, 19 (1), 3-5 (2002)
2. Staples, M., Niazi, M., Jeffery, R., Abrahams, A., Byatt, P., Murphy, R.: An Exploratory Study of Why Organizations do Not Adopt CMMI. Journal of Systems and Software, 80 (6), 883-895 (2007)
3. Coleman, G., O'Connor, R.: Investigating Software Process in Practice: A Grounded Theory Perspective. Journal of Systems and Software, 81 (5), 772-784 (2008)
4. Nelson, R.R., Winter, S.: An evolutionary theory of economic change. The Balknap Press of Harvard University Press, Cambridge, Massachusetts, USA (1982)
5. ISO/IEC: 15504-1 information technology - process assessment - part 1: Concepts and vocabulary. ISO / IEC, Geneva, Switzerland (2004)
6. CMMI Product Team: CMMI for development, version 1.2. Software Engineering Institute, CMU/SEI-2006-TR-008. Pittsburgh, PA, USA (2006)
7. Salviano, C.F., Figueiredo, A.: Unified Basic Concepts for Process Capability Models. In: Proceedings of The Twentieth International Conference on Software Engineering and Knowledge Engineering (SEKE'08), pp. 173-178. (2008)
8. McBride, T., Henderson-Sellers, B., Zowghi, D.: Project Management Capability Levels: an empirical study. In: Proceedings of the 11th Asia-Pacific Software Engineering Conference, pp. 56-63. (2004)
9. Haapio, T.: A Framework for Improving Effort Management in Software Projects. Software Process: Improvement and Practice, 12 549-558(2007)
10. Fayad, M.E., Laitnen, M.: Process Assessment Considered Wasteful. Communications of the ACM, 40 (11), 125-128 (1997)
11. Herbsleb, J., Carleton, A., Rozum, J., Siegel, J. and Zubrow, D.: Benefits of CMM-Based Software Process Improvement: Initial Results. Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA (1994)
12. 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)
13. Lawlis, P., Flowe, R., Thordahl, J.: A Correlational Study of the CMM and Software Development Performance. Crosstalk, The Journal of Defense Software Engineering, 8 (9), 21-25 (1995)
14. 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)
15. Wegelius, H., Johansson, M.: Practical Experiences on Using SPICE for SPI in an Insurance Company. In: EuroSPI 2007 Industrial Proceedings, pp. 2.27-2.34. ASQF, Potsdam, Germany (2007)
16. El Emam, K., Birk, A.: Validating the ISO/IEC 15504 Measures of Software Development Process Capability. Journal of Systems and Software, 51 (2), 119-149 (2000)
17. McAdam, R., Fulton, F.: The Impact of the ISO 9000:2000 Quality Standards in Small Software Firms. Managing Service Quality, 12 (5), 336-345 (2002)
18. Ludewig, J.: Software engineering in the year 2000 minus and plus ten. Springer-Verlag, IN: Informatics - 10 years back, 10 years ahead. LNCS 2000/2001. Berlin / Heidelberg, Germany (2001)
19. Benediktsson, O., Dalcher, D., Thorbergsson, H.: Comparison of Software Development Life Cycles: A Multiproject Experiment. IEE Proceedings - Software, 153 (3), 87-101 (2006)
20. 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)

21. Kautz, K.: Software Process Improvement in very Small Enterprises: Does it Pay Off? Software Process: Improvement and Practice, 4 (4), 209-226 (1998)
22. Chandler, A.D.: Organizational Capabilities and the Economic History of the Industrial Enterprise. The Journal of Economic Perspectives, 6 (3), 79-100 (1992)
23. Cohendet, P., Kern, F., Mehmanpazir, B., Munier, F.: Knowledge Coordination, Competence Creation and Integrated Networks in Globalised Firms. Cambridge Journal of Economics, 23 (2), 225-241 (1999)
24. Poulin, L.A.: Achieving the Right Balance between Process Maturity and Performance. IEEE Canadian Review, 56 (-), 23-26 (2007)
25. Davenport, T.H.: The Coming Commoditization of Processes. Harvard Business Review, 100-108 (2005 (June))
26. ISO/IEC: Amendment to ISO/IEC 12207-2008 - systems and software engineering – software life cycle processes. ISO, Geneva, Switzerland (2008)
27. Clarke, P., O'Connor, R.: Harnessing ISO/IEC 12207 to examine the extent of SPI activity in an organisation. In: Proceedings of the 17th Conference on European Systems & Software Process Improvement and Innovation (EuroSPI2010), Springer-Verlag, Heidelberg / Berlin, Germany (2010)
28. Clarke, P., O'Connor, R.V.: The Situational Factors that Affect the Software Development Process. IEEE Transactions on Software Engineering (Submitted)
29. IEEE: Guide to the software engineering book of knowledge (SWEBOK). IEEE Computer Society, Los Alamitos, CA, USA (2004)
30. Boehm, B., Clark, B., Horowitz, E. et al.: Software cost estimation with cocomo II. Prentice Hall PTR, Upper Saddle River, NJ, USA (2000)
31. Putnam, L.: A General Empirical Solution to the Macro Software Sizing and Estimating Problem. IEEE Transactions on Software Engineering, 4 (4), 345-361 (1978)
32. Albrecht, A.J.: Measuring application development productivity. In: Proceedings of the IBM Applications Development Symposium, pp. 83. GUIDE International and SHARE, Inc., IBM Corporation (1979)
33. Glaser, B., Strauss, A.: The discovery of grounded theory: Strategies for qualitative research. Aldine de Gruyter, Hawthorne, NY, USA (1976)
34. Corbin, J., Strauss, A.: Basics of qualitative research. Sage Publications Limited, Thousand Oaks, CA, USA (2008)
35. Bryant, A., Charmaz, K.: The SAGE handbook of grounded theory. SAGE, Thousand Oaks, CA, USA (2007)