Clarke, P. and O'Connor, R. V., An empirical examination of the extent of software process improvement in software SMEs., Journal of Software: Evolution and Process, Vol. 25, No. 9, pp. 981–998, 2013.

An empirical examination of the extent of software process improvement in software SMEs

Paul Clarke and Rory V. O'Connor

Dublin City University, Ireland.
Lero, the Irish Software Engineering Research Centre.

SUMMARY

Although earlier studies revealed much about software process improvement (SPI) in Small- to Medium-sized Enterprises (software SMEs), no earlier research set out to determine the full extent of SPI being implemented in software SMEs. Therefore, this study was designed from the outset to elicit all instances of SPI, which we term *SPI events* – no matter how small or informal. We make the important new discovery that SMEs initiate a considerable amount of SPI, albeit in varying quantities in different organisations. No earlier study reported that the practice of SPI was so widespread in software SMEs, and this is perhaps related to the extensive scope of the enquiry adopted in this study. Our study also finds that the significant majority of SPI in software SMEs is minor or moderate in nature, sometimes leveraging the human capital via improvements in tacit knowledge. Software development is an intrinsically human intensive activity and it therefore follows that the maximisation of the human capital in an organisation is a source of competitive advantage. However, contemporary process maturity frameworks and quality management standards do not fully exploit the capacity of human capital and may therefore diminish rather than improve the competitive advantage of software SMEs.

Keywords: Software process, software process improvement, empirical study, software SMEs.

1. INTRODUCTION

While many software process researchers understand the importance of software process management via SPI, it has been reported that in practice, managers can lack a commitment to SPI initiatives, resulting in a low process priority [1]. Furthermore, it has been reported that software SMEs tend to only implement SPI in response to negative business events [2]. These earlier studies were focused on examining the motivations for and perceptions of SPI and the findings are not particularly encouraging from a software SME perspective. For software SMEs, and indeed for software development companies of all sizes, the software development process is a large and complex component of the overall business process. With the effective management of business processes being recognised as important for business success [3], [4], it follows that the effective management of the software development process is also important for business success.

Although the effective management of the software development process is important for business success, the findings from earlier studies suggest that software SMEs do not adopt a proactive or highly prioritised approach to SPI [1], [2]. This raises the problem that software SMEs may be limiting their competitive advantage by not adopting a stronger SPI priority. In order to better understand this problem domain, we developed a novel approach to examining SPI in practice, an approach that facilitates the identification of all instances of SPI (which we term *SPI events)* in an organisation. Our new approach to examining SPI encompasses the broadest possible range of SPI events, from the largest SPI initiative right through to the smallest, informal instance of SPI. The application of this new approach will help to extend our knowledge of SPI in practice in software SMEs, and will permit a more informed discussion regarding the actual practice of SPI in software SMEs.

Earlier SPI studies have provided valuable insights into the role of SPI in software SMEs. Some of these earlier studies have examined the benefits from implementing specific SPI actions, finding that positive outcomes can be achieved through focused SPI initiatives. For example, a number of software SMEs derived both short- and long-term benefits from SPI in areas such as requirements management, estimation and product quality [5-9]. Other studies focused on the benefits of SPI in individual software SMEs, reporting positive outcomes in terms of decreased production costs [10], and improvements in productivity and quality [11]. Earlier research has also focused on identifying the high priority process improvements for software SMEs [12-16], on the success factors for SPI in software SMEs [17], and on the impact of SPI on business growth in software SMEs [18].

In addition, earlier research has also highlighted that SMEs cannot afford the costs of implementing large software development frameworks [19], [20]. As a result, some related work has examined the larger software development frameworks, trying to identify the most important process components for software SMEs [21-24]. While much of the research identified above focused the research effort through the lens of existing process frameworks, other related research has highlighted the role of tacit knowledge and small informal process improvements in software SMEs [25-27]. It is therefore important that any mechanism for investigating SPI in software SMEs should provide visibility not just of formal or large SPI initiatives but it should also facilitate the elicitation of SPI events that are more informal or tacit in nature.

The related research identified above provides an interesting and valuable insight into the role of SPI in software SMEs. However, none of this earlier research was designed to capture the full extent of SPI, both formal and informal, across the full set of software development related processes. Therefore, this study was designed specifically to make determinations in relation to the extent of SPI across the complete spectrum of software specific and system context processes as identified in ISO/IEC 12207 [28]. This is the first published study to conduct such a broad examination of SPI in software SMEs, with the purpose of providing increased visibility of the extent of SPI in software SMEs, thus improving our understanding of SPI as practiced in software SMEs.

The remainder of this paper is structured as follows: in section 2 we provide and overview of the study, while in section 3 we present the data analysis. In section 4, we evaluate the data, comparing the findings to earlier studies so as to identify key areas of commonality and difference. In section 5, we outline some study limitations as well as suggestions for future research. Finally, in section 6, we present a conclusion.

2. STUDY OVERVIEW

The software development process is defined as the sequence of steps required to develop or maintain software [29]. Within this sequence of steps, there exist multiple activities, each of which, depending on the needs of the individual software development setting, can be implemented to varying degrees. SPI can be considered to be improvements to the sequence of steps required to develop or maintain software. These improvements relate not just to the sequencing of the steps or to the inclusion (or exclusion) of certain steps, but also to the specific implementation of the individual steps. Since this study is concerned with examining the broadest possible spectrum of modifications to the complete set of software development processes, the study must as a pre-requisite employ a robust and reliable software development process reference framework.

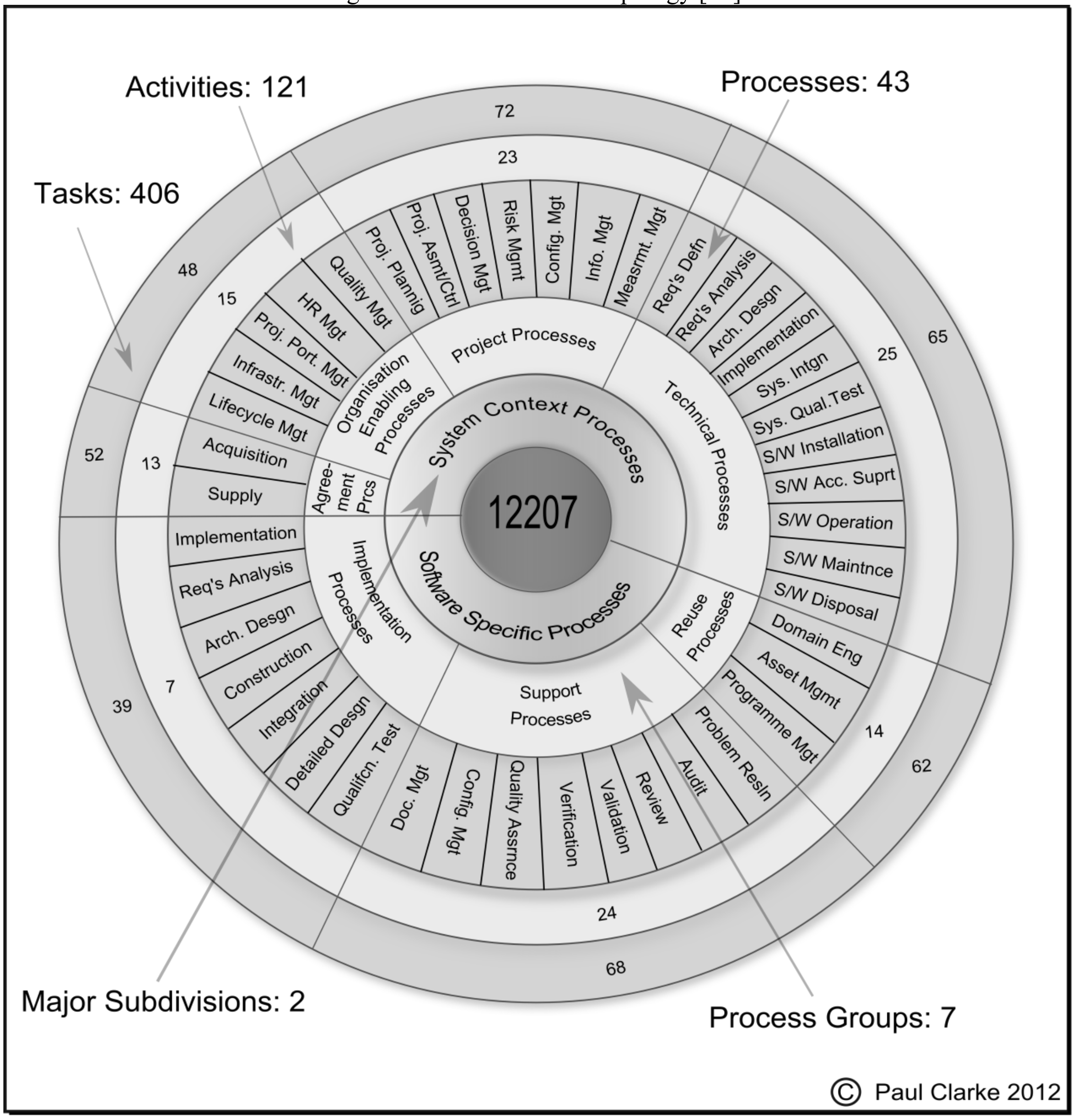
2.1 Software Development Process Reference Framework

When selecting the software development process reference framework for this study, a number of considerations had to be satisfied. Firstly, the chosen process reference framework had to be comprehensive in nature, identifying and describing the broadest possible range of software development related processes. Secondly, since a number of different software SMEs will participate in the study, it is important that the selected process reference framework should be independent of any specific software development approach – it should identify the process components rather than prescribe the process implementation. Thirdly, in order to maximise the credibility of the research, insofar as is possible the chosen process reference framework should be consensually agreed and generally accepted in the software development community. While a number of possible reference frameworks could be harnessed in order to conduct this study, for example the process listing contained within CMMI [30] or SEWBOK [31], no single framework addresses the three considerations more completely than ISO/IEC 12207 [28]. Consequently, ISO/IEC 12207 was selected as the process reference framework to underpin the SPI investigation.

According to ISO/IEC 12207, the steps involved in the lifecycle of software development belong to one of two classifications: software specific processes and system context processes. Software specific processes concern the activities directly related to the core software development effort, such as constructing detailed designs and writing code. System context processes relate to all the non software specific activities that are needed in the broader lifecycle of systems development, such as project planning and systems operation. ISO/IEC 12207 identifies 43 software specific and system context processes, with over 400 corresponding tasks, and is therefore considered to be comprehensive in nature. Thus, ISO/IEC 12207 satisfies the first consideration for the software process reference framework for this study.

ISO/IEC 12207 "describes [the] continuing responsibilities that must be achieved and maintained during the life of the process… the functions to be performed rather than organizations to execute them" [32], and as such it provides a "meta-model that defines common software engineering activities independently of a particular life-cycle model" [33]. Therefore, ISO/IEC 12207 satisfies the second consideration for the software process reference framework for this study. Regarding the third consideration for the process reference framework for this research, the approach of the International Organisation for Standardisation (ISO) to drafting and accepting standards involves a democratic voting system, wherein at least 75% of the participating national bodies must approve a standard prior to publication [28]. Furthermore, ISO/IEC 12207 was developed in collaboration with the Institute of Electrical and Electronics Engineers (IEEE) Computer Society. As a result, ISO/IEC 12207 is generally accepted in the software development community, and thus satisfies the third consideration for the software process reference framework for this study. An overview of ISO/IEC 12207 is presented in Figure 1.

Figure 1. ISO/IEC 12207 Topology [28].



In this study, we consider that SPI can take the form of a modification to any aspect of the software specific or system context processes of ISO/IEC 12207. For example, a change to the software construction process (one of the software specific processes) is considered to be an instance of SPI. Similarly, a change to the infrastructure management process (one of the organisation enabling processes) is also considered to be an instance of SPI. Furthermore, and as outlined in the introduction of this paper, this research will examine the full spectrum of possible software process modifications to each of the processes outlined in ISO/IEC 12207 – from small, informal changes, to large, formal SPI initiatives. By adopting the comprehensive

ISO/IEC 12207 process reference framework and through examining the broad spectrum of possible process changes, this research elicited a thorough representation of SPI in each of the participating software SMEs.

2.2 SPI Data Collection

In this study, we formulated a novel approach to examining SPI in the participating software SMEs. This new approach involved transforming ISO/IEC 12207 [28] into a survey instrument that could be discharged in a single engagement in order to determine the extent of SPI in software organisations. The new SPI survey instrument was systematically developed from ISO/IEC 12207 (refer to Figure 1) using the technique outlined in [34]. The survey instrument development technique first involved a thorough analysis of ISO/IEC 12207, carefully noting all of the process groups, activities and tasks. This resulted in the development of 173 questions that could be applied to task of eliciting all SPI events in an organisation. The participants were informed that the study was interested in SPI of any nature and the questions took the general form of: *Has there been any modification in the approach to…[some aspect of the software development process]?*

Subsequent steps in the survey instrument development technique involved the removal of duplications in the questions (as there is quite a significant degree of cross-referencing in ISO/IEC 12207) and also in the consolidation of the questions in order to meet the practical considerations (for example, the time taken to discharge the survey instrument). One of the final steps in the SPI survey instrument development technique involved external reviewing by experts in software processes, including the current and former editors of the ISO/IEC 12207 standard. Following the integration of the expert feedback, the SPI survey comprised of 63 individual questions that cover the broad spectrum of software process activities as presented in ISO/IEC 12207.

At the study outset, we had explored the possibility of adopting two process assessments in order to collect the study data. A first assessment could have been deployed at the start of the year under investigation and a second assessment at the end of the year under investigation. Following the completion of the second process assessment, a finite difference analysis could have been conducted on the two process assessment results – hence, rendering the difference in process implementation between the two dates. However, there were a number of strong motivations for not adopting this dual process assessment approach for the data collection.

Firstly, since process assessments are primarily designed to examine process maturity, they do not present an efficient mechanism for determining the process modification. Secondly, the relatively large amount of time required to conduct two process assessments could dissuade organisations from participating in the study. Thirdly, this study is concerned with software SMEs, a sector which has tended to view process maturity frameworks as being infeasible for their needs [35]. As a result, approaching software SMEs with the prospect of conducting multiple process assessments could appear unsuitable to some of the candidate organisations. Fourthly, process assessments might overlook some of the smaller and more informal process improvements implemented by software SMEs, thus rendering an incomplete view of the SPI implemented in the participating organisations. As outlined in the introduction, this study is concerned with developing a complete view of SPI in practice, and in this respect, it is desirable to include the informal SPI that earlier research highlighted as being important in software SMEs [25-27]. The new SPI survey instrument developed as part of this study ensures that all modifications to the software development process (no matter how small or informal) can be detected along with larger SPI initiatives.

2.3 Timeframe and Organisations

During the period of March to July 2011, the SPI survey instrument was deployed to fifteen participating companies that satisfied the European Commission definition of an SME [36]. The majority of the participating organisations retained their head office in the Republic of Ireland, though a number of the companies were based elsewhere, including locations such as the USA and Chile. Three of the participating companies had less than 10 staff, while 4 of the companies had between 10 and 19 staff. The remainder of the participating organisations had between 20 and 129 staff. Each interview required on average 2 hours to complete, giving a total of approximately 30 hours interviewing time, and the interviewee was generally the Software Development Manager. Following the completion of each interview, the data was carefully recorded in electronic format. The result was a set of 15 interview records that were in a form suited to data analysis.

Table 1. Software process modification rating scale.

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

Using the SPI survey instrument, each of the 63 survey instrument questions examined the degree of modification to aspects of the software development process over the past year. Each participant was asked to rate the amount of process modification according to the Likert scale identified in Table 1. Using this four point scale, it was possible to capture all manner of SPI events, from the most minor of informal process modifications right up to the largest process improvement initiatives. As identified in section 2.2, each of the survey instrument questions took the form of: *has there been any modification in the approach to [some aspect of the process]?* For each question, the study participant could first seek clarification on the meaning of the question. Having understood the question, the participant would then identify the process modification rating. When identifying the process modification rating, the participant might also comment on the rating. These additional comments were recorded and as we shall see in the data analysis and discussion sections of this paper, the commentary often afforded considerable insight into the views, realities and activities of practitioners regarding SPI.

3. DATA ANALYSIS

As identified in the earlier sections, the SPI survey instrument collected data regarding SPI events, with each SPI event being accorded a process modification rating. Therefore, we can analyse SPI from two distinct perspectives:

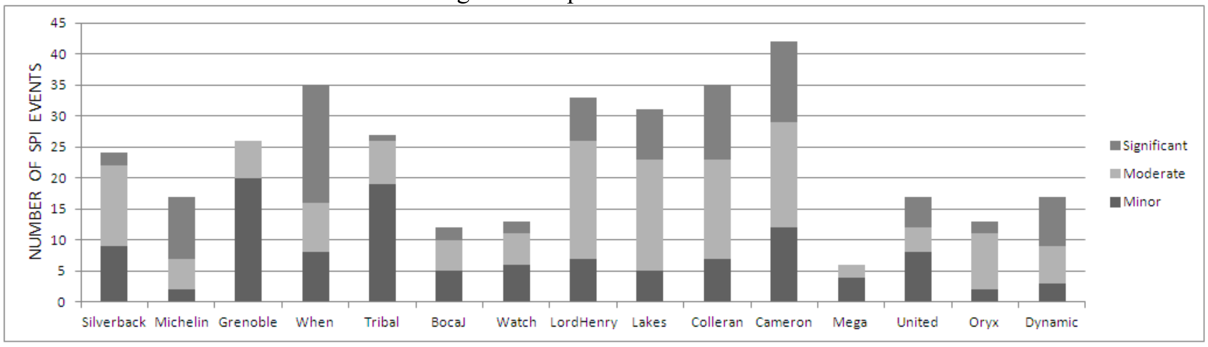
(1) The *number of individual SPI events* can be calculated for each software SMEs, hence permitting the researchers to investigate the breadth of SPI that is being implemented in software SMEs in practice;
(2) The *total amount of SPI activity* can be calculated for each of the software SMEs by summing the process modification values for each of the reported SPI events, as follows:

$$SPI\ activity = \sum_{j=1}^{M} ModificationValue(j)$$

Where j is each software process question in the SPI survey instrument and M is the total number process questions in the survey instrument (as outlined in Section 2.2, there are 63 of these SPI questions). The modification values are as outlined in Table 1.

This is the first published study to quantify the number of SPI events and the total amount of SPI activity in software SMEs, considering both the formal and the informal process modifications. In the following section, employ these measures so as to investigate the SPI reported in the study.

Figure 2. Reported SPI Events.



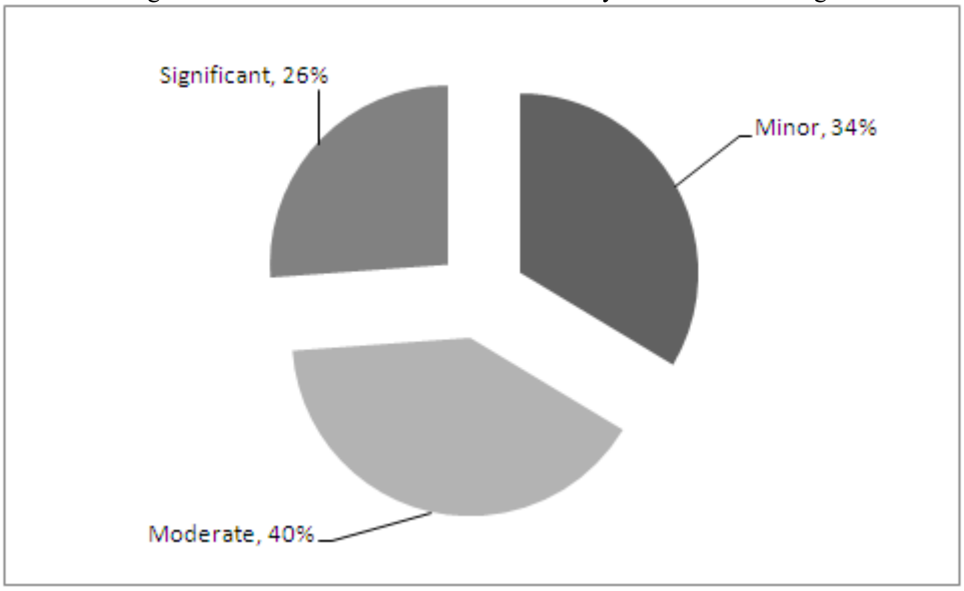
3.1 SPI Reported in the Study

An analysis of the data reveals that the lowest recorded number of SPI events is in the organisation *Mega*[1], which has just 6 SPI events over the year under examination (refer to Figure 2). The largest recorded number of SPI events is in the organisation *Cameron,* which reports 42 SPI events over the year. While there is quite a large difference in the highest and lowest recorded number of SPI events, the most striking characteristic of the data is that the

[1] Since the study collects data that is potentially of a sensitive nature, the identity of each of the participating organisations has been protected through the use of randomly allocated pseudonyms.

significant majority of SPI events (74%) are classified by the organisations as either minor or moderate in nature (refer to Figure 3). Most of the SPI that is taking place can therefore be considered to represent adjustments to the earlier process, with over one third of all SPI events being classified as minor adjustments.

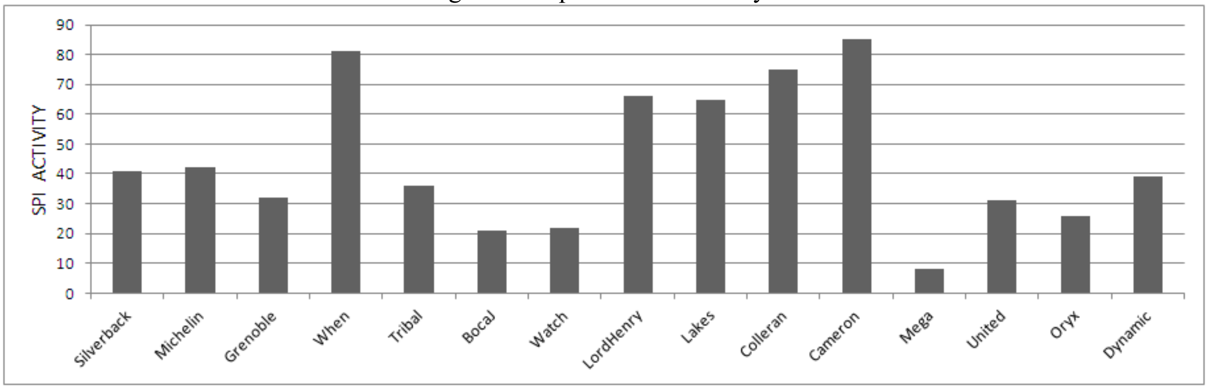
Figure 3. Overall breakdown of SPI events by modification rating.



In addition to analysing the number of SPI events, we are also interested in investigating the total amount of SPI activity in organisations. As with the SPI events, the organisation *Cameron* records the highest amount of SPI activity, while *Mega* records the lowest amount of SPI activity. As outlined above, the SPI activity figure is the sum of each SPI event multiplied by the reported modification value for that event (refer to Table 1). For example, the organisation *Silverback* reported 24 SPI events and the corresponding SPI activity for *Silverback* is 41 (minor [1*9] + moderate [2*13] + significant [3*2]).

The outline trend for SPI activity (refer to Figure 4) can be seen to be broadly similar to the outline trend for SPI events (refer to Figure 2) which suggests that the relationship between these two views is positively associated. We can conclude that organisations that are performing more individual SPI events are also conducting greater overall SPI activity.

Figure 4. Reported SPI Activity.



3.2 Type of SPI activity reported in the study

The data analysis reveals that there is a rich variety of SPI activity being conducted in software SMEs, with all but two of the process areas in the ISO/IEC 12207 being reported as having been a focus for SPI in a least one of the participating organisations. However, some processes are consistently presenting as undergoing improvement in the majority of software SMEs, while other processes are consistently presenting as receiving little improvement. Table 2 and Table 3 summarise the most common and least common areas for SPI in the software SMEs that participated in the study.

Table 2. Most commonly reported areas for SPI – with mapping to ISO/IEC 12207 [28].

Description	**ISO/IEC 12207 Reference**
Specifying requirements	6.4.2; 7.1.2
Developing and documenting software units and databases	7.1.5
Providing and maintaining an infrastructure	6.2.2
Identifying opportunities and tendering for business	6.1.2
Planning for installation in the target environment	6.4.7
Operations and maintenance requirements	6.4.9; 6.4.10
Test coverage	6.4.3-7; 7.1.5-7

Table 3. Least commonly reported areas for SPI – with mapping to ISO/IEC 12207 [28].

Description	**ISO/IEC 12207 Reference**
Collecting and maintaining quality cost data	6.2.1
Defining and executing a software disposal strategy	6.4.11
Transforming the requirements into an architecture	6.4.3; 7.1.3
Developing, maintaining and executing test procedures	7.1.5
Conducting reviews of software integration	6.4.5; 7.1.6
Developing designs for external interfaces and databases	6.4.3; 7.1.3
Reviewing process implementation and project management	7.2.6
Conducting independent audits of projects or deliverables	7.2.7
Definition of data and databases	7.1.2
Definition of test requirements at architectural definition stage	7.1.3

The approach to tendering, bidding and negotiating with clients is also reported as undergoing a considerable amount of improvement. For a number of the participating SMEs, this is related to the emergence of partnering arrangements, whereby multiple companies will partner in order to enhance their offering to potential clients. In this respect, one company reports that they are presenting themselves to the market as *"more of a solutions company"*, while another participating organisation reports that considering new reseller agreements, the sales process now includes a *"decision… as to whether we bring [a third party] into the sale"*. There is also a confirmation from two other participating organisations that they have improved their tendering and negotiating, stating that there is *"a little more conversion… over milestones and payments… as well as the detail"*, and reporting that they are *"more stringent about making sure that we're only going after profitable deals, or ones that are a strategic move – we're just trying to be more cautious"*. Another organisation reports that they have developed a *"different offer"* altogether for the market.

A number of the participating organisations also report minor or moderate modifications in the approach to software installing, and planning for installation, in the target environment. In the case of two organisations, they report increases in the degree of automation associated with installations. Another organisation has highlighted an increased need for frequent releases, stating that over the past year, there has been *"a lot more releases in stages"* and that they are now *"more structured for those types of projects"*. A further organisation highlights that there was a *"change in the installation strategy... due to downsizing [of the company itself]"*.

In addition to the common areas identified above for SPI activity, the participating organisations also report that there is an increased consideration for the feasibility of operations and maintenance. In order to reduce future support and maintenance costs, a number of organisations report that they have increased levels of refactoring. Another organisation reports that they *"are looking to cost more accurately the ongoing maintenance and support costs"*, with another company again stating that in order *"to reduce the maintenance burden... we're much more conscious, from the requirements point of view, of the support burden"*. As is discussed in more detail in the Discussion section, there is a difference between classical SPI in terms of actual improvement actions that can be observed in the software process and those improvements that are more of a tacit nature. While tacit improvement may not be immediately measurable in the process as an observable effect, improvements in the mental check-listing of process participants does represent an improvement to the method of work. The extent and importance of tacit knowledge-led SPI is discussed in more detail in the next section.

Although one of the organisations reports that owing to decreases in company headcount, there is a reduction in testing activity, the majority of the organisations report general improvements in testing. For some organisations, this is simply that the *"test suite is growing slowly over time"*. For other organisations this is a more explicit improvement to the test process, with one respondent stating that *"testing is more well defined and more rigorous than it had been"* and other respondent confirming that *"we're allocating more time at the development stage to performance testing than we would have done a year ago"*. Another organisation reports that they are *"measuring more the code coverage of our automated tests"*. Some of the participating companies also report improvements in the effort estimation process, stating that the estimates are *"more well defined than they would have been"* and that there is more team involvement in estimation exercises. Greater team involvement in estimation is reported by one respondent to *"give ownership [for estimates] back"* to the team, with the result that estimates are becoming *"more accurate"*.

The participating organisations also report improvements in the approach to performing change requests (CRs). Describing the company as previously having been *"very ad hoc"* in the handling of CRs, one organisation has enforced the recording of CRs in a tracking system. Another organisation reports that they are *"a lot more rigorous"* about change management for requirements, with resulting increases in the visibility and tracking of CRs. Some of the improvements in visibility and tracking of CRs have been realised through the adoption of new tooling support. As we shall discuss in more detail in the Discussion section, improvements in tooling are often reported by the participating SMEs as enablers of improvements in the approach to the full lifetime of activities associated with software development.

3.2.2 Least common areas for SPI activity

Not one of the participating organisations reported improvements in the approach to collecting and maintaining quality cost data, with one organisation providing the insight that they "*haven't a clue*" regarding the cost of quality. While other organisations weren't quite so explicit in relation to the approach to costing quality, the body language of participants and the tone of their responses suggest that this is an area which receives very little attention from software SMEs. Similarly, the accumulated evidence of this research suggests that software SMEs pay very little attention to the disposal of software. The participating organisations reported that they do not have to be concerned with the secure disposal of software or with decommissioning software systems. Where sensitive data was contained in systems, the organisations reported that the client was largely responsible for such concerns.

In general, the participating organisations reported a strong tendency to merge architectural and design considerations. Predominately, they reported that from their point of view, the design and the architecture were not treated as separate concerns, but rather that the design was the architecture and vice-versa. The software SMEs in the study also report little change to the process of reviewing process quality procedures or to reviewing process implementation in general. Again, the indifference evident in the responses implies that these are areas to which the participating SMEs pay little attention. This suggests that the software development process evolves in a non-structured fashion rather than in a focused or controlled manner in software SMEs.

The analysis of the study data also reveals that the participating organisations have made almost no improvement to the process of independently auditing either projects or software deliverables. Quite a large number of the participating organisations asserted that, in fact, they never perform independent audits of their projects or products. The participating companies also report very low levels of process improvement with respect to the development of designs for external interfaces and databases. There was also no evidence of any major initiatives to improve the approach to defining data and databases over the study period, and no reported improvement in the definition of preliminary test requirements at the architectural definition stage. Furthermore, the participating software SMEs report little change to the process for developing and maintaining test procedures or to the process for reviewing integration with other systems.

4. DISCUSSION

An in-depth examination of the data reveals that this study has collected valuable new information in relation to the nature of SPI in software SMEs. These new findings raise interesting challenges for established process maturity reference frameworks and quality management standards. Furthermore, the findings of this study have implications in terms of the application of ISO/IEC 12207 [28] to software SME. Prior to elaborating on these findings, we first present the parallels with earlier research.

4.1 Parallels with Earlier Research

Some of the findings from the data analysis confirm the feedback from earlier related studies. For instance, in this study we find that many of the participating organisations are making improvements to the general area of requirements management, including capture, sign-off, and change management. This is not surprising, as a number of earlier studies have highlighted the importance of improvements in the areas of requirements management for software SMEs. Earlier research suggested that it can be important to focus SPI on the estimation process and that improvements in the testing process have positive benefits for software SMEs [5], [37] – our study confirms both of these earlier findings. Other findings from earlier studies are also confirmed in our study: software SMEs can have a low process priority [1] resulting in little focus on the collection of quality cost data; tacit knowledge-led SPI has an important role to play in software SMEs [25-27].

In our study, such tacit knowledge-led SPI is particularly evident in the language that participants used, including: we're now *"more aware of"*; aspects of the process are *"discussed further"* (rather than being documented or formalised); we're *"certainly thinking about [it]... more than last year"*; that *"we're more conscious of [it]"*. Tacit knowledge-led SPI also takes the form of extensions to mental checklists. For example, one organisation reported that *"we're probably more concerned about the maintenance so, when we're in code, we will try and carve out all of the redundant stuff just because we want to reduce the maintenance burden"*. There is no formal change that could be recorded, just that there is an increased awareness of the need for refactoring. Since this increased awareness is giving rise to increased refactoring in practice, this represents an improvement in the process of software construction. However, since this is not formally implemented in practice, and there are no records or measurements in relation to the improvement, it is likely to be overlooked by conventional process assessments.

4.2 New Insights

While the present study has confirmed a number of findings from earlier research, the novel approach to examining SPI adopted in our study has yielded a number of important new insights regarding SPI in software SMEs. The study presented herein takes the broad range of software specific and system context processes into consideration, and investigates both the small informal process modifications and the larger, more formal instances of SPI. Taking this approach to examining SPI, we make the important new discovery that software SMEs tend to implement quite a large amount of SPI, albeit in varying quantities in different organisations. Earlier SPI studies in software SMEs did not report evidence of relatively high degrees of SPI, and this new finding is both intriguing and valuable as it suggests that the software development process is being continually evolved in software SMEs. Furthermore, the data from this study demonstrates that the software development process in software SMEs is largely evolving through a series of minor or moderate adaptations. This can be considered to be a type of tweaking of the process rather than fundamental reshaping of the process. Or as one organisation described some of their changes: *"there has been some very minor changes… but nothing fundamental"*.

The finding that many minor and informal process improvements are taking place in software SMEs is significant and it has implications for established process maturity frameworks. In such frameworks, the top level of maturity is concerned with optimising, an activity that is intended to ensure that the software development process is continually harmonised with emerging needs, knowledge and technologies. However, such frameworks are rarely implemented in software SMEs, and even if they were to be implemented in SMEs, experience from larger organisations suggests that very few organisations would progress to highest level *optimising* stage. The evidence of this study suggests that software SMEs are in practice optimising their software development process in a largely informal way, through a series of minor or moderate improvements to the development process. Perhaps this is the only way that software SMEs can afford to implement process change. Whatever the cause of such behaviour, it is interesting to note that software SMEs would appear to have embraced the optimising principle of the highest maturity level of established process maturity reference frameworks, while at the same time largely ignoring the staged process maturity concept in general. This finding prompts us to ask the question: Should the bodies responsible for developing process maturity frameworks re-evaluate the role of optimisation at all levels of process maturity?

Although much of the process change is minor or moderate in nature, the extent of the recorded change was surprising. At the outset of the study, the authors had not expected to record such extensive SPI in practice in software SMEs – and no earlier study had suggested that this might be the case. The findings reveal that the software development process landscape is constantly evolving in software SMEs. One of the participants asserts that "*it doesn't really matter what you have formally because tomorrow it is all going to change"*, with another company stating that *"everything that we've changed… will be changed again"*. In addition to this high degree of process change, we also find that software SMEs espouse an *a la carte* approach to process methodology adoption. One organisation states that *"we don't formally follow the methodologies… [that] as a business, we're not capable of fully following one process"*. As a result the company reports that in terms of implementing software development processes, *"we've kind of cherry picked a bit"*. While there are dangers to cherry picking in this manner, such as implementing an agile development process without any refactoring, this finding is also of interest as it provides evidence that when it comes to the software development process, there is no one size fits all [38]. Each organisation must implement and adapt their software development process in tandem with their needs and resources. Or as one software process researcher states, *"it is reasonable to assume that the optimal process is not static but is organization-dependent and time-dependent, and will have to be modified as the context in which the organization operates evolves"* [39]. The evidence collected in this study supports this assumption.

While the ability to mix and merge different development methodologies offers some benefits for software SMEs, this study also highlighted the danger of unbridled process methodology manipulation. For example, in relation to code refactoring, one of the participating organisations reported that *"we try to do more refactoring but getting management agreement for refactoring that doesn't produce any new features is a difficult challenge"*. This particular comment highlights the struggle with limited resources in smaller organisations. Software SMEs can be fighting for existence and as a result may try to maximise short term value at a cost to long term maintainability. In this sense, there can be a temptation to promote the attractive value-led features of agile software development approaches [40] while demoting the important supporting agile practices. Of course, in the long term, such decisions could have highly undesirable outcomes, with the possibility of ending up with a code base that is expensive to support and maintain.

The study also revealed another important new insight regarding the views of practitioners concerning SPI. Essentially, the study data indicates that among practitioners, there is some considerable degree of ambiguity as to what exactly constitutes SPI. In one instance, an organisation reports that an aspect of the process is unchanged, but that *"the change I would say is that we have gotten better"*. In this case, the respondent was referring to the initiation of whiteboard sessions to discuss the implications of new requirements. In another case, an organisation reports that *"we do a project post-mortem meeting, we always did, but not to the same level that we do now"*. While the post-mortem meetings and the whiteboard sessions may have happened to some minor extent in earlier periods, they are considered to have increased in frequency (and probably also in detail). Since the process is generally neither documented nor policed in software SMEs, an increased frequency of post mortem meetings or whiteboard sessions is difficult to classify. In a more formally defined environment, this could be classified as increased process adherence, however, in the absence of a clear definition of the development process, this increased tendency towards white board sessions and post mortem meetings is considered to be best classified as a process improvement in its own right – though very much informal in implementation. Discoveries such as these highlight that there is lack of a common understanding and raises an important new question: What constitutes SPI?

In this study, many of the participating organisations also reported that they had adopted new tools. Six of the organisations adopted completely new tooling for aspects for their software development. In some cases, these tools related to improved capability to visualise and track requirements and change requests, hence improving communication and awareness. In other cases, the new tools were adopted in order to improve the process for product deployment out to clients. In another company, a hosted cloud-based application was used to improve the ease of document sharing with clients. Such evidence suggests that software SMEs continually integrate new tooling solutions to aid the task of software development. In one concrete example of tool adoption improving the task of software development, one of the participating organisations started to use the same tool for both source code control and for live deployment for web based applications. Historically, no single tool could accomplish both of these tasks, and therefore, this tooling advancement allowed the organisation to improve the efficiency of its software development and deployment activities. Therefore, tooling advances can be important for improving the efficiency of work practices in software SMEs.

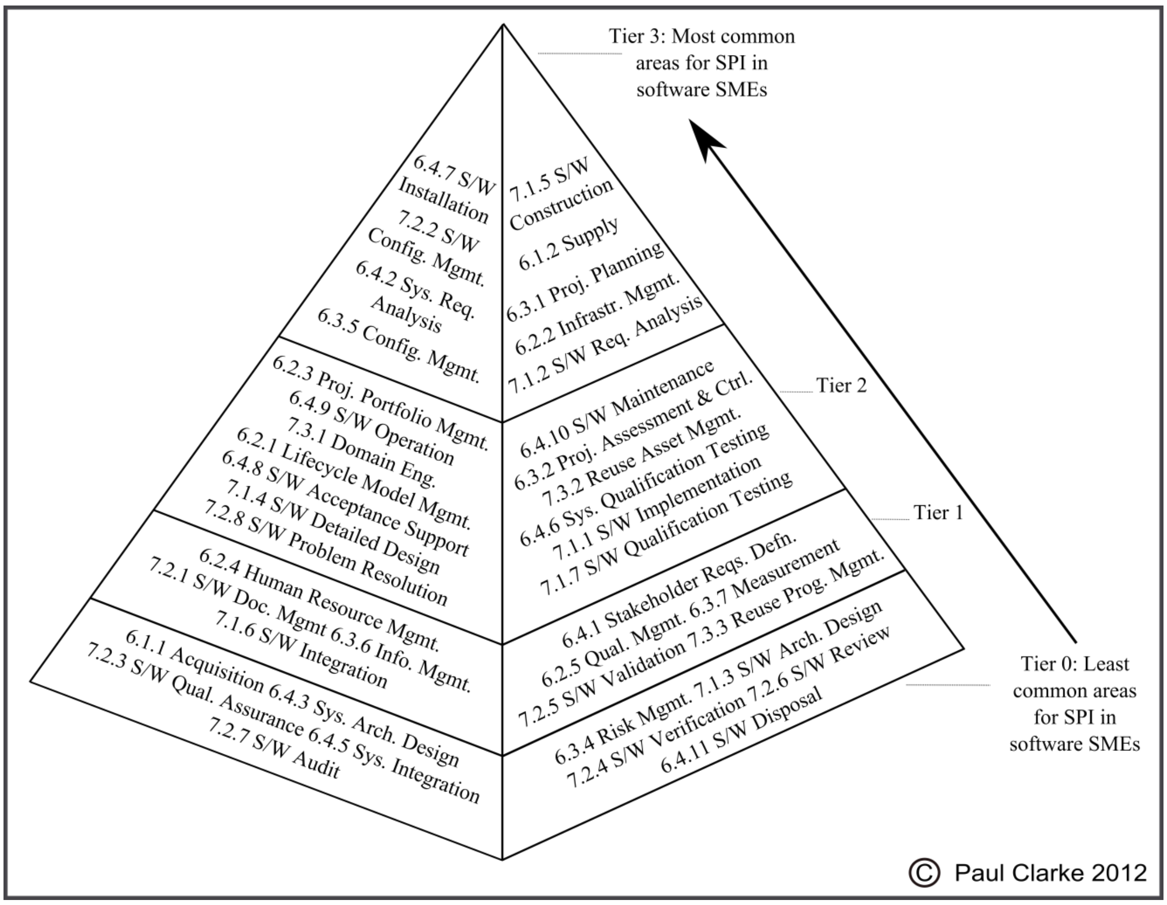
While new tooling solutions can offer improved efficiencies, the study demonstrated that there can be confusion regarding the role of tools in SPI. One organisation reports that *"the electronic tracking system that we're using is being used a lot more often now"*. The suggestion is that the increased usage of the tool is not representative of a process improvement. However, increased utility of a tracking system is representative of at least improved process adherence, and could possibly be considered to be an instance of process improvement. While changes to the utilisation of supporting tools (or to the tools themselves) can sometimes represent instances of SPI, the findings of this study suggests that software SMEs don't consistently distinguish between instances of SPI and instances of tooling improvements.

We also find that organisations report that they desire to become more independent from the demands of individual customers and that they seek to become more strategic in their actions. For instance, one company reported that the selection of requirements has become "*less market driven"* resulting in *"more breathing space"*. A second organisation comments that they're *"trying to move away from the question 'can we' and more towards 'should we'"*, and in so doing they consider that they are *"becoming more capability-led and less market-led"*. A third company commented that *"over the past year… the business got a clearer focus on what it was trying to build product-wise whereas before, there could be a new idea every week."* While the study was not focused on examining why such changes were occurring, the nature of the feedback suggests that the companies consider that a degree of liberation from the whims and demands of customers is generally viewed in a positive light. This is especially true when an increased focus can be devoted to independent, strategic product decision making. In their early years, software SMEs can struggle for survival and may be over-extended in terms of developing a market or retaining one or more critical business accounts. Perhaps after this period of passed, software SMEs become more stable and viable businesses, with the result that they are more capable of managing the sometimes unrealistic demands of customers.

4.3 Implications for ISO/IEC 12207

As outlined in section 2, the SPI survey instrument utilised in this study was based on ISO/IEC 12207 [28]. Therefore, it is possible to analyse the reported process improvements in terms of the actual process listing in the ISO/IEC 12207 standard. From analysing the data collected in this study, a number of key ISO/IEC 12207 process presented as undergoing SPI more frequently. Notably, the requirements analysis, project planning, software installation and configuration management processes were consistently reported as areas for process improvement. A full ISO/IEC 12207 based hierarchy of SPI for software SMEs is presented in Figure 5, with further details on the development of the hierarchy and broader discussion available in Clarke and O'Connor [41].

Figure 5. ISO/IEC 12207-based hierarchy of SPI activity for software SMEs.


Tier 3: Most common areas for SPI in software SMEs
6.4.7 S/W Installation
7.2.2 S/W Config. Mgmt.
6.4.2 Sys. Req. Analysis
6.3.5 Config. Mgmt.
7.1.5 S/W Construction
6.1.2 Supply
6.3.1 Proj. Planning
6.2.2 Infrastr. Mgmt.
7.1.2 S/W Req. Analysis
Tier 2
6.2.3 Proj. Portfolio Mgmt.
6.4.9 S/W Operation
7.3.1 Domain Eng.
6.2.1 Lifecycle Model Mgmt.
6.4.8 S/W Acceptance Support
7.1.4 S/W Detailed Design
7.2.8 S/W Problem Resolution
6.4.10 S/W Maintenance
6.3.2 Proj. Assessment & Ctrl.
7.3.2 Reuse Asset Mgmt.
6.4.6 Sys. Qualification Testing
7.1.1 S/W Implementation
7.1.7 S/W Qualification Testing
Tier 1
6.2.4 Human Resource Mgmt.
7.2.1 S/W Doc. Mgmt 6.3.6 Info. Mgmt.
7.1.6 S/W Integration
6.4.1 Stakeholder Reqs. Defn.
6.2.5 Qual. Mgmt. 6.3.7 Measurement
7.2.5 S/W Validation 7.3.3 Reuse Prog. Mgmt.
6.1.1 Acquisition 6.4.3 Sys. Arch. Design
7.2.3 S/W Qual. Assurance 6.4.5 Sys. Integration
7.2.7 S/W Audit
6.3.4 Risk Mgmt. 7.1.3 S/W Arch. Design
7.2.4 S/W Verification 7.2.6 S/W Review
6.4.11 S/W Disposal
Tier 0: Least common areas for SPI in software SMEs
© Paul Clarke 2012

The survey instrument that was developed by this research in order to examine SPI activity adopts the same language and process classification that is presented in ISO/IEC 12207. In order to investigate the scope of ISO/IEC 12207 in terms of processes for software SMEs, a closing question was incorporated into the SPI survey instrument so as to provide the participants with an opportunity to comment on the completeness of the survey. This question encouraged participants to identify any process improvements that they had adopted that were not addressed in the survey instrument. Overwhelmingly, the participating organisations asserted that they considered the survey to cover all of the areas related to their software development process. They stated that *"it's been pretty comprehensive"*, *"you've pretty much covered the entire software lifecycle"*, and *"I think you've gone through the full lifecycle"*. Since ISO/IEC 12207 is developed and maintained by an international panel of software development experts, is not surprising that the study participants find that the survey instrument is comprehensive in terms of addressing the software development process. However, the participants did identify some drawbacks associated with the application of ISO/IEC 12207 in the setting of software SMEs.

There are some indications that the language adopted in ISO/IEC 12207 is not easily accessible in software SME settings. One of the participants reported that he found the language to be *"almost awkward"*, with another participant stating that the questions relate to *"very formal mechanisms"*. It is true that ISO/IEC 12207 is a large, valuable and formally developed resource but perhaps the extent of the international standard and the associated formality render the standard unsuited to software SMEs. Just as earlier studies have indicated that software SMEs consider process maturity frameworks to be infeasible rather than non-beneficial [35], perhaps ISO/IEC 12207 is similarly infeasible for software SMEs. Whether or not ISO/IEC 12207 is infeasible for software SMEs, in practice ISO/IEC 12207 has very little exposure in this sector – with none of the participants having previously utilised the standard.

With ISO/IEC 12207 being such a comprehensive and carefully constructed software lifecycle process reference framework, it seems regrettable that it is not utilised in smaller software development organisations. Although ISO/IEC 12207 is designed to provide a comprehensive reference for the atomic processes required for software development, the evidence of this study indicates that there are some significant language and stylistic differences when compared with the software development process as practiced in software SMEs. With over 400 process tasks, ISO/IEC 12207 is likely to be considerably beyond the scope of individual software SMEs. However, some gaps exist in the language and concepts. Some of the terminology which forms part of the vernacular of software SMEs (and many larger software development organisations) doesn't feature anywhere in ISO/IEC 12207; for example, *refactoring* and *timeboxing*. This particular observation resonates with a broader issue – that there is a lack of a generally adopted dictionary of terms for software development. Although there has been considerable and valuable investment in the development of dictionaries of terms for software development, for example in both ISO/IEC 12207 and SWEBOK [31], in practice there does not appear to be a generally adopted dictionary. There would be significant benefits for the software development community as a whole if all participants used the terminology in a more consistent fashion.

4.4 The Role of Agile Software Development

Over the past decade, various agile software development approaches have been developed to broadly address the principles of the agile manifesto [40]. Since agile software development is widely practiced, it is possible that the type and extent of SPI evidenced in this study is related to the general agile philosophy. Agile software development encourages individual participants to be more involved in the delivery of value, to avoid investing time in the development of supporting documentation that may not be needed, and to increase the level of communication within the team via a series of regular and structured meetings. All of these characteristics may account for the type of tacit-knowledge led process improvement that has been observed in this study. Certainly, the increased empowerment of individuals in the agile software paradigm is likely to promote rather than inhibit the improvement of individual working practices.

While the adoption of agile software development may account for some of the observations in the study, it cannot singularly claim to be the catalyst for all of the SPI recorded in the study – across the spectrum from significant to minor SPI actions. Agile software development has been heavily oriented towards accommodating the need to quickly deliver valuable features and to accommodate changing requirements. However, the volatility of software requirements is just one of many factors affecting software process decisions [42] and therefore, agile software development alone is unlikely to be able to fully address the broader needs of software development organisations. Naturally, the creators of agile software development approaches can rightly assert that they were not necessarily designed as a panacea for all of the challenges associated with software development. Rather, these approaches offered a better fit than traditional approaches (for example, the Waterfall approach [43]), in terms of catering for fluctuating software requirements or for valuable features that should be rapidly incorporated into applications or products.

Considering the discussion in the previous paragraphs, it appears reasonable to conclude that the adoption of agile software development may account for some of the observations in this study – particularly those improvements that have been reported at a tacit knowledge level. However, agile software development alone does not account for the full extent and type of SPI reported in this study. It is the view of the authors that there is an inevitable need to evolve and adapt software development processes in order to address changing situational contexts and therefore, a conglomeration of many different approaches, a tailoring to individual contexts, and a capacity to adapt the broader process considerations to changing situations may offer the most practical solution to the software process needs of organisations. The evidence recorded in this study suggests that in practice, software SMEs are already actively engaged with such process adaptation.

5. LIMITATIONS AND FUTURE RESEARCH

While this study had taken every effort to incorporate all aspects of the software development process, it should be noted that the adoption of ISO/IEC 12207 [28] is a limiting factor in the study. Although ISO/IEC 12207 [28] is comprehensive in nature, no framework can claim to be absolutely complete. Therefore, we acknowledge that there is a possibility that some aspects of the software development process could have been overlooked. However, a more comprehensive process reference framework does not exist, plus we incorporated measures to check for potentially missing process activities.

The study is also limited in terms of the number of participating organisations. Although the study involved accessing a variety of busy personnel in fifteen participating SMEs – and such access is very difficult to realise – the sample size is not especially large from a numerical point of view. However, a considerable depth of information has been gathered and the sample size is large when compared to similar studies in this domain. Nonetheless, a similar study incorporating a larger number of participating companies could expect to make stronger claims in relation to the generalisability of findings – and perhaps this is an area worth considering for future research.

Other future research could focus on examining the nature of tacit software process knowledge. It would be beneficial to have an agreed method for investigating and quantifying tacit knowledge improvements vis-à-vis the software development process. Furthermore, existing process maturity frameworks could benefit from extending their scope to encompass tacit knowledge considerations. With software development being a human intensive activity, it is natural that many aspects of the improvement in the method of work occur in a social manner. This includes human-centric activities such as dialogue and learning through experience. Given the considerable dependence on people in software development, it would appear to be vitally important that human aspects of software development are adequately catered for in future software development process models and frameworks.

The findings presented in this paper discuss the extent of SPI in software SMEs, and it has been both interesting and beneficial to examine the extent of SPI in such settings using the novel SPI survey instrument. However, this study was not focussed on examining why certain changes were occurring and other changes were not being implemented. This represents an additional limitation in this research – since the organisations do not offer *sameness* other than in satisfying the European Commission definition of an SME [36] (which in itself is quite a broad classification). Therefore, the findings in relation to the ubiquity of mostly minor or moderate SPI events in the participating software SMEs may be the greater contribution of this research – rather than the classification of the most and least commonly reported areas for SPI (which may be the result of the situational context of the participating companies). Most likely, aspects of the changing situational context are informing SPI decisions. Consequently, future work to understand the interplay between the changing situational context and SPI decisions would represent an important contribution to our knowledge with respect to software development process evolution.

6. CONCLUSION

In this study, we examined the extent of SPI in 15 software SMEs over a 12 month period. An analysis of the data has yielded the important new finding that software SMEs continually implement SPI, both formally and informally, across a broad of range of processes areas. Certain process areas were reported to be undergoing SPI more regularly, including requirements management, change control, infrastructure provision, and tendering or negotiating with clients. Furthermore, the participating organisations reported an increased concern with respect to the full lifecycle costs of software development, such as operations and maintenance.

The findings of the present study also extend our body of knowledge regarding the very nature of SPI in software SMEs. Rather than occasionally undertaking a few large SPI initiatives, software SMEs instead prefer to make small adjustments to the software process on a regular basis. The degree of process change recorded in the study reinforces the view that process adaptation is an important activity for software SMEs. The study also finds that aspects of the reported SPI are of a tacit nature: ideas, discussions, awareness, accumulated learning experienced. While such tacit knowledge has an important role to play in software SMEs, it may not be easily detectable using traditional process assessment approaches. Furthermore, although the concept of process optimisation is evident in the highest tier of established process maturity frameworks, it may be inadequately addressed in the lower maturity tiers, with the result that such frameworks are deficient in terms of addressing the need to adapt the process to changing situational contexts. Therefore, we recommend that future research efforts focus on identifying approaches to supporting process optimisation and on visualising improvements in tacit knowledge in software SMEs.

Our study utilised ISO/IEC 12207 [28] as the process reference model, and the research findings confirm that software practitioners consider ISO/IEC 12207 to be a comprehensive process reference model. However, some of the participants reported that they found the language and structure to be overly formal and mechanistic – such that it doesn't harmonise well with their process language and implementation. This feedback points to a much broader issue in the software process field, in that there is lack of consistent use of language and terminology. In this regard, the authors recommend that beneficial future work could develop a unified and generally accepted dictionary of terms – one that is accepted and used by both researchers and practitioners, and in both large and small organisations.

Although SPI is commonplace in software SMEs, such organisations have largely neglected to implement established process maturity and quality management frameworks. Additionally, software SMEs report that they adopt a mix-and-match philosophy to their software development process, mixing aspects of different prescribed software development approaches in order to match their needs. This raises an interesting and important question: Given that software SMEs are highly personalised in their approach to software process implementation and SPI, how can future software process research best support this large and vital sector of the software development community?

Acknowledgements. This work was supported, in part, by Science Foundation Ireland grant 10/CE/I1855 to Lero - the Irish Software Engineering Research Centre (www.lero.ie).

REFERENCES

[1] Baddoo N, Hall T. De-motivators for software process improvement: an analysis of practitioners' views. Journal of Systems and Software 2003; 66 (1): 23-33. DOI: 10.1016/S0164-1212(02)00060-2.

[2] Coleman G, O'Connor R. Investigating software process in practice: A grounded theory perspective. Journal of Systems and Software 2008; 81 (5): 772-784. DOI: 10.1016/j.jss.2007.07.027.

[3] Kaplan RS, Norton DP. The Balanced Scorecard - Measures That Drive Performance. Harvard Business Review 1992; 70 (1): 71-79.

[4] Sureshchandar GS, Leisten R. Holistic Scorecard: strategic performance measurement and management in the software industry. Measuring Business Excellence 2005; 9 (2): 12-29.

[5] Sanders M (Ed.). The SPIRE Handbook. Better, Faster, Cheaper Software Development in Small Organisations. Centre for Software Engineering Limited: DCU, Dublin, Ireland, 1998.

[6] Sanders M, Richardson I. Research into long-term improvements in small- to medium-sized organisations using SPICE as a framework for standards. Software Process: Improvement and Practice 2007; 12 (4): 351-359. DOI: 10.1002/spip.319.

[7] Cater-Steel A, Rout T. "SPI long-term benefits: Case studies of five small firms," in Software Process Improvement for Small and Medium Enterprises - Techniques and Case Studies. Oktaba H, Ed. IGI Global: Hershey, PA, USA, 2008.

[8] Fleck D. A process for very small projects. Proceedings of the 22nd Annual Pacific Northwest Software Quality Conference. PNSQC/Pacific Agenda: Portland, Oregon, USA, 2004; 107-115.

[9] Montoni M, Rocha AR. A methodology for identifying critical success factors that influence software process improvement initiatives: An application in the brazilian software industry. Proceedings of the 14th European Conference on Software Process Improvement. Springer-Verlag: Heidelberg/Berlin, Germany, 2007; 175-186.

[10] Biro M, Ivanyos J, Messnarz R. Pioneering Process Improvement Experiment in Hungary. Software Process: Improvement and Practice 2000; 5 (4): 213-229. DOI: 10.1002/1099-1670(200012)5:4<213:AID-SPIP129>3.0.CO.

[11] Ferreira AIF, Santos G, Cerqueira R, Montoni M, Barreto A, Rocha AR, Soares Barreto AO, Silva Filho RC. ROI of software process improvement at BL informatica - SPI is really worth it. Industrial Proceedings of the 14th European Conference on Software Process Improvement. Publication Series about SW Quality, 2007; 12.27-12.34.

[12] Von Wangenheim CG, Anacleto A, Salviano CF. Helping small companies assess software processes. IEEE Software 2006; 23 (1): 91-98.

[13] Richardson I. Software process matrix: a small company SPI model. Software Process: Improvement and Practice 2001; 6 (3): 157-165. DOI: 10.1002/spip.144.

[14] Colla P, Montagna J. Framework to Evaluate Software Process Improvement in Small Organizations. Lecture Notes in Computer Science (Number: 5007) 2008; 36-50.

[15] Habra N, Alexandre S, Desharnais JM, Laporte CY, Renault A. Initiating software process improvement in very small enterprises: Experience with a light assessment tool. Information and Software Technology 2008; 50 (7-8): 763-771. DOI: 10.1016/j.infsof.2007.08.004.

[16] Oktaba H, Garcia F, Piattini M, Ruiz F, Pino FJ, Alquicira C. Software Process Improvement: The Competisoft Project. IEEE Computer 2007; 40 (10): 21-28.

[17] Dangle KC, Larsen P, Shaw M, Zelkowitz MV. Software Process Improvement in Small Organizations: A Case Study. IEEE Software 2005; 22 (6): 68-75. DOI: 10.1109/MS.2005.162.

[18] Nikitina N, Kajko-Mattsson M. Impact of growing business on software processes. Proceedings of the 17th European Conference on Systems, Software and Services Process Improvement (EuroSPI 2010). Springer: CCIS 99/2010. Berlin Heidelberg, 2010; 189-200.

[19] Huang W, Li R, Maple C, Yang H, Foskett D, Cleaver V. A novel lifecycle model for Web-based application development in small and medium enterprises. International Journal of Automation and Computing 2010; 7 (3): 389-398.

[20] O'Connor R, Coleman G. Ignoring 'Best Practice': Why Irish Software SMEs are rejecting CMMI and ISO 9000. Australasian Journal of Information Systems 2009; 16 (1): 7-30.

[21] Niazi M, Babar MA, Ibrahim S. An empirical study identifying high perceived value practices of CMMI level 2. Proceedings of the 9th International Conference on Product Focused Software Process Improvement. Springer-Verlag: LNCS 5089/2008. Berlin / Heidelberg, Germany, 2008; 427-441.

[22] Niazi M, Babar MA. Identifying high perceived value practices of CMMI level 2: An empirical study. Information and Software Technology 2009; 51 (8): 1231-1243. DOI: 10.1016/j.infsof.2009.03.001.

[23] Wilkie FG, McFall D, McCaffery F. An evaluation of CMMI process areas for small- to medium-sized software development organisations. Software Process: Improvement and Practice 2005; 10 (2): 189-201. DOI: 10.1002/spip.223.

[24] McCaffery F, Taylor PS, Coleman G. Adept: A Unified Assessment Method for Small Software Companies. IEEE Software 2007; 24 (1): 24-31.

[25] Ryan S, O'Connor RV. Development of a team measure for tacit knowledge in software development teams. Journal of Systems and Software 2009; 82 (2): 229-240. DOI: 10.1016/j.jss.2008.05.037.

[26] Basri S, O'Connor RV. A study of software development team dynamics in SPI. Proceedings of the 17th European Conference on Systems, Software and Service Process Improvement (EuroSPI 2010). Springer Verlag: CCIS 172/2010. Berlin / Heidelberg, Germany, 2011; 143-154.

[27] Fuller A. Factors that Enable Or Inhibit Australian SME Software Development Organisations in Developing and Implementing Improved Processes. Faculty of Communication and Information Technology, Griffith University, Brisbane, Australia: Brisbane, Australia, 2001.

[28] ISO/IEC. ISO/IEC 12207-2008 - Systems and Software Engineering – Software Life Cycle Processes. ISO: Geneva, Switzerland, 2008.

[29] Humphrey WS. A Discipline for Software Engineering. Addison-Wesley: Reading, Massachusetts, USA, 1995.

[30] SEI. CMMI for Development, Version 1.2. Software Engineering Institute: CMU/SEI-2006-TR-008. Pittsburgh, PA, USA, 2006.

[31] IEEE. Guide to the Software Engineering Book of Knowledge (SWEBOK). IEEE Computer Society: Los Alamitos, CA, USA, 2004.

[32] Moore JW. IEEE/EIA 12207 as the foundation for enterprise software processes. The Joint 1998 Proceedings of the Pacific Northwest Software Quality Conference and the 8th International Conference on Software Quality. PNSQC/Pacific Agenda: Portland, Oregon, USA, 1998; 326-333.

[33] Tilley T, Cole R, Becker P, Eklund P. A Survey of Formal Concept Analysis Support for Software Engineering Activities. Springer-Verlag: IN: Formal Concept Analysis. LNCS 3626/2005. Berlin / Heidelberg, Germany, 2005.

[34] Clarke P, O'Connor R. Harnessing ISO/IEC 12207 to examine the extent of SPI activity in an organisation. Proceedings of the 17th Conference on European Systems & Software Process Improvement and Innovation (EuroSPI2010). Springer-Verlag: CCIS 99/2010. Heidelberg / Berlin, Germany, 2010; 25-36.

[35] 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 2007; 80 (6): 883-895. DOI: 10.1016/j.jss.2006.09.008.

[36] European Commission. Commission Recommendation of 6 May 2003 concerning the definition of micro, small and medium-sized enterprises. 2003/361/EC. Official Journal of the European Union 2003; L (124): 36-41.

[37] Keane B, Richardson I. Quality: Attitudes and Experience within the Irish Software Industry. Lecture Notes in Computer Science: Software Process Improvement (Volume 3792/2005) 2005; 49-58. DOI: 10.1007/11586012.

[38] Boehm B, Turner R. Balancing Agility and Discipline - A Guide for the Perplexed. Pearson Education Limited: Boston, Massachusetts, USA, 2003.

[39] Poulin LA. Achieving the Right Balance Between Process Maturity and Performance. IEEE Canadian Review 2007; 56 (-): 23-26.

[40] Fowler M, Highsmith J. The Agile Manifesto. Software Development 2001; August issue: 28-32.

[41] Clarke P, O'Connor RV, Yilmaz M. A hierarchy of SPI activities for software SMEs: Results from ISO/IEC 12207-based SPI assessments. To appear in: Proceedings of the 12th International Conference on Software Process Improvement and Capability dEtermination (SPICE 2012). Springer-Verlag: Berlin Heidelberg, Germany, 2012; 62-74.

[42] Clarke P, O'Connor RV. The situational factors that affect the software development process: Towards a comprehensive reference framework. Journal of Information and Software Technology 2012; 54 (5): 433-447.

[43] Royce W. Managing the development of large software systems: Concepts and techniques. Western Electric show and Convention Technical Papers. IEEE Computer Society: Los Alamitos, California, USA, 1970.