2007

An Investigation of Barriers to the Adoption of Software Process Best Practice Models

Rory V. O'Connor
Dublin City University, roconnor@computing.dcu.ie

Gerry Coleman
Dundalk Institute of Computing, gerry.coleman@dkit.ie

An Investigation of Barriers to the Adoption of Software Process Best Practice Models

Rory V O'Connor
School of Computing
Dublin City University
Ireland
Email: roconnor@computing.dcu.ie

Gerry Coleman
Department of Computing
Dundalk Institute of Computing
Ireland
Email: gerry.coleman@dkit.ie

Abstract

This paper presents the results of a Grounded Theory study of how Software Process Improvement (SPI) is applied in the practice of software development, focussing on what is actually happening in practice in the software industry in relation to the adoption of SPI 'Best Practice' Models, such as ISO 9000 and CMMI. The results produce a picture of attitudes and perceptions in relation SPI best practice models, grounded in the field data, and reveals that many software managers reject SPI because of the associated implementation and maintenance costs and are reluctant to implement SPI models such as ISO 9000 and CMMI. This paper presents the findings in relation to Cost of Process and the factors affecting it, including Bureaucracy, Documentation, Communication, Tacit Knowledge and organisational Creativity and Flexibility and the associated impact on the adoption of SPI best practice models.

Keywords

Software process, Software Process Improvement, Best practice models, CMMI, ISO 9000, XP.

Introduction

A software process essentially describes the way an organisation develops its software products and supporting services, such as documentation. Processes define what steps the development organisations should take at each stage of production and provides assistance in making estimates, developing plans, and measuring quality. The process and associated activities are often documented as sets of procedures to be followed during development. However, the documentation is not the process but should clearly represent the process as it is implemented within an organisation. To simplify understanding and to create a generic framework which can be adapted by organisations, software processes are represented in an abstract form as software process models.

There is a widely held belief that a better software process results in a better software product, with authors such as Humphrey (1995) stating that "*to improve your product, you must improve your process quality*". In support of this Zahran, (1998) states that "*it is a widely accepted fact that the quality of a software product is largely determined by the quality of the process used to maintain and develop it*". These ideas have led to a focus on Software Process Improvement (SPI), which can be traced back to the 1970s and 1980s and the work of Crosby (1979) and Juran (1988), who demonstrated that, in the area of production management, product quality could be improved through a better production process.

In the Information Systems domain, SPI aims to understand the software process as it is used within an organisation and thus drive the implementation of changes to that process to achieve specific goals such as increasing development speed, achieving higher product quality or reducing costs. SPI models have been developed to assist companies in this regard and purport to represent beacons of 'best practice'. Contained within the scope of these models, according to their supporters, lies the road to budgetary and schedule adherence, better product quality and improved customer satisfaction.

In an attempt to ensure software project success, some large software organisations have used 'best practice' process improvement models, such as the Capability Maturity Model/Capability Maturity Model Integrated (CMM/CMMI) (Ahern et al, 2004) and the International Organisation for Standardisation (ISO) 9000 series (ISO, 1992). More recently, agile methodologies such as Extreme Programming (XP) (Beck, 2000) have been used in SPI programmes as a way of improving delivery time and increasing customer satisfaction and these

agile approaches have been widely embraced by software organisations. Although commercial SPI models have been highly publicised and marketed, they are not being widely adopted and their influence in the software industry therefore remains more at a theoretical than practical level.

In the case of CMMI, evidence for this lack of adoption can be seen by examining the SEI CMMI appraisal data for the years 2002 to 2006 (SEI, 2006), in which time just 1,581 CMMI appraisals were reported to the SEI. It is clear that this represents a very small proportion of the world's software companies and company in-house developers. In addition, there is evidence that the majority of small software organisations are not adopting standards such as CMMI. For example, an Australian study (Staples et al, 2007) found that small organisations considered than CMMI "*would be infeasible*". Further investigation of the SEI CMMI appraisal data reveals that in the case of Ireland – a country whose indigenous software industry is primarily made of small to medium sized organisations (SME) - fewer than 10 CMMI appraisals were conducted during the 2002 – 2006 period, from a population of more than 900 software companies. Therefore it is also clear that the Irish software industry is largely ignoring the most highly-publicised SPI models. In the case of CMMI (and its predecessor CMM), Staples and Niazi (2006) discovered, after systematically reviewing 600 papers, that there has been little published evidence about those organizations who have decided to not adopt CMMI.

Accordingly the motivation for our research originates in the premise that, in practice software companies are not following 'best practice' process improvement models. On this basis, we set out to answer the question: *Why are software companies not using 'best practice' SPI models*?

Study Overview

A context and scope for the study was set as follows: In order to ensure the participation of software development professionals who would be familiar with the considerations involved in using both software process and process improvement models, it was decided to limit the scope to software product companies. In addition, given the geographical location of the researchers, it was considered best to confine the study to indigenous Irish software product companies who naturally operate within the same economic and regulatory regime. Furthermore, restricting the study to indigenous Irish software product companies, significantly increased the prospects of obtaining the historical information required to understand process foundation and evolution which would not be the case with non-Irish multinationals operating in the country, as their process would likely have been initially developed and used within the parent company prior to being devolved to the Irish subsidiary.

The investigation of software process in practice relies heavily on eliciting and understanding the experience of those who use the software processes in situ and the interpretation of these experiences and the reality of the situation under study. The study therefore, naturally lends itself to the application of qualitative research methods, as they are orientated towards how individuals and groups view and understand the world and construct meaning out of their experiences. Accordingly, the methodological approach taken was that of Grounded Theory (Glaser and Strauss, 1967), whose aim is to develop a theory from data rather than to gather data in order to test a theory or hypothesis. This manifests itself in such a way that rather than beginning with a pre-conceived theory in mind, the theory evolves during the research process itself and is a product of the continuous interaction between data collection and data analysis (Goulding, 2002).

Grounded theory uses qualitative methods to obtain data about a phenomenon and a theory emerges from that data, where that theory is grounded in the reality as represented in the data. As the objective with the methodology is to uncover theory rather than have it pre-conceived, grounded theory incorporates a number of steps to ensure good theory development. The analytical process involves coding strategies: the process of breaking down interviews, observations, and other forms of appropriate data, into distinct units of meaning, which are labelled to generate concepts. These concepts are initially clustered into descriptive categories. The concepts are then re-evaluated for their interrelationships and, through a series of analytical steps, are gradually subsumed into higher-order categories, or one underlying core category, which suggests an emergent theory. For a fuller discussion on grounded theory, the rationale behind its selection, and how it was implemented in this study please refer to (Coleman and O'Connor, 2007).

This study was divided into three distinct phases: firstly a Preliminary Phase to assist with framing the study, testing the interview guide and approach; a Detailed Phase which developed the initial concepts and categories and enabled evaluation of the theoretical sampling process; and a Final Phase which further developed the categories and concepts to produce the grounded theory. In total, the three phases of the study involved 25 interviews across 21 companies. The participants in the Preliminary Stage were chosen from personal contacts of the researchers. For later stages, in parallel with making contact with individuals known second-hand to the researchers, 'cold' e-mailing was used to set up the next series of interviews.

Grounded Theory provides mechanisms to identify the categories into which the concepts discovered in the data can be placed to explain the relationships between these categories to provide the overall theoretical picture; and

to identify a key category or theme that can be used as the fulcrum of the study results. In this instance, the analysis showed that there was one central category *Cost of Process* to support the two theoretical themes *Process Formation* and *Process Evolution*. Each category and code can be linked to quotations within the interviews and these are used to provide support and rich explanation for the results.

Study Findings

The study has found that all of the companies were tailoring standard software processes to their own particular operating context such as the size of the company, the target market, and project and system type. In addition there was evidence from the data suggesting that managers instigate SPI as a reaction to business occurrences which the current process did not adequately cater for.

The research question addressed in the study, *why are software companies not using 'best practice' SPI models* produced the study's core category *Cost of Process*. Implementing and maintaining any SPI initiative incurs significant cost. Participant companies perceive *Documentation* as the greatest process-related cost-inducing element. There was also a clear link between the amount of *Documentation* carried out and the size and growth stage of the company; the smaller the company the greater the hostility towards *Documentation*. However, even in the larger organisations, *Documentation* was regarded as a 'necessary evil'. Many companies substituted verbal *Communication* for *Documentation*, and co-located their development teams in an effort to reduce process cost. A benefit of doing this was an increase in the sharing of *Tacit Knowledge*.

From the commercial SPI perspective, the study was dominated by two particular models CMM(I) and ISO 9000, and the development methodology XP. It is interesting to note that the respondents did not differentiate between processes and methodologies and categorised XP as a process. As a result, XP, albeit tailored to various degrees, was by far the most popular commercial 'process' model used by organisations across all size sectors. XP was perceived to have the least associated *Cost of Process* and its low level of *Documentation* was deemed to be attractive. Where managers were familiar with CMM(I) they were against introducing it to their new organisations arguing that, whilst it may have a role in a very large multinational, it had no role in a small software product company. ISO 9000 also received major criticism from the majority of the study companies many of whose managers, again, had used it previously. However, three companies in the study are ISO 9000 certified, all of those sought certification for business reasons and not process / quality reasons. Overall, respondents felt that the resources required to implement the commercial models far exceeded the benefits that may accrue.

In the course of the study interviews, few of the managers concerned expressed any enthusiasm about process or process improvement models. A far greater emphasis was placed on product, with process often believed to be a 'brake' on product development. The managers believed process to have a significant cost which, in their respective companies, they attempted to keep to a minimum. What the managers perceived as the Cost of Process centred on a number of factors and these are represented as a network diagram in Figure 1.

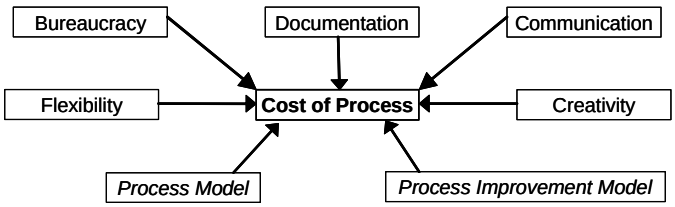

Figure 1: Cost of Process Network

The key *Cost of Process* factors *Bureaucracy*, *Documentation*, *Creativity* and *Flexibility* are presented in the following subsections, along with a discussion on the associated impact on both the Process Model and Process Improvement Model used. In keeping with the fundamental tenets of Grounded Theory, extracts of the interview transcripts will also be presented in support of the findings.

Bureaucracy

The category *Bureaucracy* covers items including the time and resources which the managers in the study believe are required to administer and apply the software process used in their organisations. In essence, managers divided process into two separate categories, 'essential' and 'non-essential'. 'Essential' process was that which was most closely linked to the product; requirements gathering, testing and design. 'Non-essential' process, which in the view of managers could often be omitted, included process/quality-related documents and plans, software measurement, and even many management activities such as planning, estimating, and staging meetings. The interviews capture this in a number of different ways. Three separate managers described some

process activities as a 'luxury' and not something essential to creating software products. The use of the word luxury is quite significant, as it is a synonym for 'extravagance', 'indulgence', or 'something inessential' (New Oxford Thesaurus of English, 2001).

The following comment from a company 2 illustrates this point: "*In the earlier stages when we would do a design document, we would have all the team members giving their feedback on how that would impact on the system. Now we have to bypass that because of time constraints. We just don't have the luxury of having everybody around a table*". Another manager, this time using code reviews, had a similar view of process: "*I've sat in development code reviews and seen a bunch of people discussing, not whether a particular block of code would work, but whether it was an example of good programming style and that discussion going on for 4 or 5 hours. It's wonderful to have the luxury to do it*". In addition, Company 3 considered process definition not worth putting resources into: "*The other key thing was resources. We didn't have the luxury of assigning someone, saying "look you go off and spend a couple of months designing and putting a process in place"… or even a day a week putting process in place*". All three examples show that where managers believe they have limited resources they do not wish to allocate them to activities which, in their opinion, do not contribute directly to meeting product development deadlines.

Another belief of the practitioners is that following a process as it is defined is 'overkill'. A widespread opinion prevailed that there was an easier or less time-consuming way to achieve their objective and many were happy to ignore their own processes to do so, as exemplified by this interviewee: "*If you're talking about 2- or 3-week projects which are sometimes what we're dealing with, it tends to be overkill to go through a full lifecycle and we have a number of ways of getting round that*". The overkill described demonstrates managers are complicit in bypassing their own process believing that by eliminating some process steps, you also eliminate some of the costs.

The interview extracts above demonstrate that many of the managers, far from being process converts, believe that many process activities are not essential and require too much time and resource. One of the process activities that managers consider can often delay, or hinder, product development, is *Documentation*.

Documentation

Forward and Lethbridge (2002) define software documentation as "*Any artifact whose purpose is to communicate information about the software system to which it belongs, to individuals involved in the production of that software*". The managers interviewed for this study believe *Documentation* is one of the single biggest contributors to the *Cost of Process*. *Documentation* incurs a cost through the actual time taken to record the chosen information, but also through opportunity cost in that, whilst staff are engaged in *Documentation*, they are not engaged in what management often see as more 'worthwhile' activities, such as coding. Reduced *Documentation* was associated with situations where managers had high levels of trust in their developers and their experience as explained in this interviewee comment: "*It comes down to experience, what are the key things to do. It's not about writing reams of documentation nor having huge heavyweight process*".

Across the interviewees, Process-related *Documentation* was seen as an overhead, which can delay development activity and whose merits, in many instances, can be difficult to convey to engineers, as described by this comment: "*So often, people were filling in time sheets and lists weeks after the project had finished in order that the quality process could be seen to pass its audit*". Smaller companies, especially, feared having to allocate people, either to write the *Documentation* in the first place, or to manage it on an ongoing basis. Despite this there was an acceptance that, with growth, more formality in *Documentation* would be required. This was a matter of real concern as one manager explained: "*With more people we would have to get involved in more administration, more recording and more documentation. And you could end up hiring administrators purely to document your processes and to ensure they are being followed*".

In accordance with the reticence to document, many managers linked improving the software process with the creation of additional *Documentation*. This was a commonly held view and is discussed later in this paper.

Communication

Because *Documentation* was seen by the managers as such a significant process cost, they believed that, if they could reduce their *Documentation* requirement, they could reduce the cost of their software process. Taking Forward and Lethbridge's (2002) definition of software *Documentation* - a way of communicating information about the software system to the individuals involved - many managers encourage verbal *Communication* as a way of sharing information and reducing the *Documentation* load. Within the study organisations, there is often conflict between explicit knowledge, represented by *Documentation*, and *Tacit Knowledge*, which is the undocumented, intuitive know-how of the individual or team. Recognising this, the companies in the study attempt to capitalise on exploiting *Tacit Knowledge*, and verbal *Communication*, and this is brought out in the practices they adopt. One company explained how they use simple *Documentation* and developer co-location to

achieve knowledge sharing: "*At that stage the product and project design was done on an A4 piece of paper and when something needed changing you could talk to the guy next to you because he knew what you were doing and you knew what he was doing*".

Informal *Communication*, and knowledge sharing, benefits from this form of team co-location. It can be achieved merely by having the entire team share a common office area. One manager described how his team profited from this arrangement: "*In my team, we are sitting close together so I can just pull out the chair and start chatting and get interaction going. You've got people talking over and back. You just wouldn't have the same spontaneity in email*". Even where the office layout doesn't support this sort of easy *Communication*, it can often occur spontaneously, a factor brought out by one of the start-up companies: "*We're efficient in the way we apply whatever process we have and communication is fast and it doesn't require a big meeting, just a bunch of people talking in corridors until they get a problem solved and things move quickly*".

There is a conviction, firmly-held in the larger companies, that *Documentation* alone will not ensure that all team members have a shared understanding of a project's or business's requirements and that deficiencies here can be overcome through informal *Communication*. By contrast, there is an acceptance in many of the smaller companies, that, though *Tacit Knowledge* and informal *Communication* is the norm, *Documentation* is necessary on occasion. This is exemplified by one of the companies who are engaging offshore third-parties to do some of their development work: "*Before that the actual production component was done in house. So from a documentation point of view you weren't as tight because the person you wanted was next to you or down the corridor. So they could ask you "what exactly did you mean by that?" But now the quality of the documentation has to be spot on*".

Despite this, all of the larger companies, or those with a higher level of *Documentation*, still report extensive informal *Communication* in their organisation: "*Even the way we write the requirements spec, we still find ourselves in a lot of verbal discussions with people who are just trying to understand the background and the issues. And a lot of the outputs from those discussions don't get documented, they just get agreed*".

A support approach to *Tacit Knowledge* that companies often undertook, in an attempt to reduce *Communication* overhead, was to keep team sizes small. Small teams allow companies more potential to co-locate them, enable more informal *Communication*, and obviate the need for greater *Documentation*. The experience of one company shows how they aim to achieve this: "*If you keep a team size small and the guys are all talking about what they are doing and describing it, discussing it, changing it around, there will be less need for them to refer to a document that they are all familiar with*".

Despite this, even the companies who use *Tacit Knowledge* extensively recognise that it has its limitations and may ultimately carry its own cost. This is especially true of those companies who are using XP and who worry about the emphasis on informal *Communication* at the expense of *Documentation*. In addition to carrying a *Documentation* load, process was also perceived by managers as having a negative impact on a development team's *Creativity* and *Flexibility*, as discussed in the next section.

Creativity and Flexibility

Software companies, especially start-ups, need to be flexible, creative, dynamic and capable of delivering products quickly in order to survive. Therefore, any deployed software process must support *Flexibility* and *Creativity*. From the interview data, though it is evident that Irish software companies value *Creativity* and *Flexibility*, many believe that process can stifle these desirable attributes, and its use should therefore be carefully considered. Some of the start-up companies see processes as primarily of benefit to established companies as Company 3 describe: "*If you want to be more sure of the results, the processes will give you more likelihood of being sure, but it's probably a bit like playing it safe. I think you won't get the same level of innovation or creativity*".

One company felt that they had too much process and felt that it impacted negatively on their *Creativity* and innovation: "*I think that product development is about being inventive and creative and new ideas coming forward and being developed quickly into something mainstream. And when you don't see that happening I think that too much is being stifled*".

Product companies focus on product development and fear that increased process will detract from that focus and that the price of additional process is a decrease in *Flexibility*, as illustrated by this interviewee comment: "*When we set up we had more supervisory and managerial roles in that group than we have now and we had to scale that back which has made things a lot more flexible. I do think you have to be nimble, quick and capable of being responsive in our position. That works well and I don't want to lose it*".

Others also reduced the amount of process they used because of they impact they felt it was having on *Flexibility*: "*We started following a formal process for a while, but the guy who was driving that left and we abandoned it. What we have now is quite flexible, and not very formal*". But fears of processes impacting

negatively on *Creativity* and *Flexibility* are not the preserve of smaller software companies, as one company explains: "*All of our competitors are several times our size, we are the smallest in the field and our only way of dealing with those guys is to make them look old and fat*". All of the extracts above show how innovation, *Creativity* and *Flexibility* are seen by the managers concerned as the lifeblood of their companies. Because SPI is sometimes seen as the enemy of *Creativity* and *Flexibility* it provides evidence why business events, which cannot be resolved using the existing process appear to be the main drivers of process change / SPI. Many managers believe the attributes of innovation, *Creativity* and *Flexibility* carry business advantages far in excess of the proposed benefits of repeatability, consistency, and quality which are associated with process and process improvement models.

Process Models

Not unlike the other aspects of process and process improvement, the choice of which model to use was also linked with its associated cost of adoption and implementation. As stated above, all of the companies interviewed are using a tailored Software Development Process, which in most cases is based on a standard industry model. The following subsections will discuss the *Cost of Process* impacts on the two most popular process models to emerge from the interviews, RUP (Rational Unified Process) and XP.

RUP

In some of the companies, their initial software process had been tailored from the RUP. Almost all of those who used elements of it have since dispensed with it completely, blaming *Documentation* as exemplified by this company: "*RUP has a lot of documentation associated with it and that didn't work for the sort of lead times we were striving for*". Others blamed the complexity and 'weight' of it: "*When I started we had an approximation of the RUP. It was over-engineered, over the top process kind of stuff. RUP is unimplementably complex. Even people in the past who have been into heavily engineered process found it impractical*".

Some companies, who had at one point considered using the RUP, subsequently rejected it because of the complexity of the associated support tool, Rational Rose: "*What I would think is the challenge is to deliver a set of tools that are easy to use, and in as far as possible form part of the design cycle. UML/Rational Rose have been attempts in that direction but are very bulky and heavy*". Whilst others merely felt that Rational Rose was too expensive: "*At one stage we started going down the Rational route. It was going to cost us 300 grand and it was money we didn't have so we backed away from it*".

Because of the costs, both resource-wise and in tool purchase, many companies moved away from processes based on RUP to ones based on XP.

Extreme Programming

Whereas the RUP was seen to have merit but to be too expensive to deploy widely, XP, as a development methodology, attracted far greater support among the interview sample. Used by companies at all size levels, the tailored versions of XP, which the companies deployed, were seen to be very cost effective. One manager argued that XP provides the fastest time to market capability of all the models available: "*There's now no way we could deliver faster with a different process than with this. XP gives you a lot of advantages in delivering quickly even on small projects*". Widespread gains were also reported from applying short iterations and test-first development, as explained by this interviewee comment: "*I think a lot of the attributes of XP, around test-first design and iterations, and rapid feedback to developers are hugely valuable*".

In cases where XP replaced an existing process, the predecessor had a much greater *Documentation* requirement. Therefore any successor with a reduced *Documentation* overhead had a real chance of succeeding. This clearly proved the case as higher levels of employee buy-in to XP were reported: "*The developers actually like doing it because it gives them a chance to get clear in their head what the task is before they start to write it, because they have started to use it even though the feature hasn't existed yet*".

The ability to reduce the 'process' elements in development was a key factor in the success of XP. Companies reported developer benefits and how easily they embraced the methodology. When introducing XP, companies believed they got good value for money with the methodology. The ability to implement the practices piecemeal, and the use of iterations with regular feedback meant, particularly in the case of the smaller organisations, that for the first time they had control over development activity. It's best summed up in the following excerpt from Company 16: "*XP was very cost effective because you didn't have to implement everything. You just had to implement those things that worked for you. And it did give us visibility into the software development process which was key*".

Though XP had clearly provided benefit for many organisations in the study group, some had reached what might be classified as the 'post-XP' stage, where, through using it, they had identified perceived limitations with the methodology. Ironically, despite many managers' reluctance to commit resources to *Documentation* tasks,

by far the most common complaint about XP related to the insufficiency of *Documentation* produced by the method: "*We tried XP and we found that the documentation trail was extremely weak or even non-existent*". Fears were also expressed that the absence of *Documentation* would make it more difficult for new team members to understand the system thus also highlighting a limitation of *Tacit Knowledge*: "*If we were hiring and bringing in a bunch of new grads, would XP work given that there is no documentation for people to look at?*".

Process Improvement Models

A key part of this research was to examine why software product companies do not appear to be following 'best practice' SPI models. There are a number of best practice models in existence but only the CMMI (and its predecessor the CMM) which are specifically geared for software, and ISO 9000 whose origins lie in manufacturing, resonated with the companies interviewed. Of the 21 study companies, 3 are ISO 9000 certified and one is embarking on the ISO 9000 certification process. None of the companies are using the CMMI.

CMM/CMMI

As the most widely publicised SPI models, it was important to this study to determine the attitudes of indigenous Irish software product companies towards them. Awareness of CMMI among the managers was far lower than was the case with ISO 9000. Though a number of the managers interviewed had experience of CMM(I) from previous employment, none had incorporated it into their present positions. However, as with ISO 9000, it was where managers had previous experience of using CMM(I) that greatest hostility to its introduction arose. An example of a greater body of opinion is the manager in company 5 who, when asked what working with CMM was like in his previous company, responded: "*It [CMM] was dire. It just got in people's way. It was almost designed to get in people's way. It wasn't designed to enhance the development process. It wasn't for me*".

Support for the opinions of the manager of Company 5 came from Company 10's software development manager who previously worked in a large multinational which used CMM: "*CMM is neither efficient nor would return huge benefits. Somebody with experience could go in and have much more effect in a lightweight way if they understood what they were doing*". Company 11 rejected it feeling it would hinder their ability to deliver quickly: "*If you look at CMM it was delivered for the likes of NASA. We might sell a piece of software that needs to be delivered in 3 months. So, the overhead of instigating a very rigorous CMM-style process is outweighed by the time it takes to deliver it*". The opinions of one manager, who having investigated CMMI and chosen not to introduce it, represents all companies who reached the same conclusion: "*We felt CMMI was overkill for the level of development that we were doing and so it wasn't really pursued*".

The belief that CMMI contain excessive levels of detail and require high levels of administration was expressed by a number of the participants. Notwithstanding the fact that they criticised ISO 9000 for not being suitable for software, CMMI did not generate increased support even though they are software specific. The criticisms levelled against it, indicate it is 'excessive', 'over the top', 'heavyweight', 'onerous', 'bureaucratic' and 'too detailed'. Managers were then asked under what circumstances might they use, with the manager from Company 9 comment being representative: "*It will depend on the companies with whom we will engage. Maybe where we get to the stage where we are dealing with government or defence and they are looking for certification, then we will go for it. That's because there is a business decision to tackle those customers and therefore the process has to evolve to get certified. You wouldn't do it the other way round. That would be crazy*".

ISO 9000

Where a manager has previously used a process or process model that they felt had a beneficial impact on development, that process was generally imported into their new environment. By contrast, where managers had prior experience of a model they felt didn't work then they rejected its use within their new companies. This was most significantly felt in the case of ISO 9000. In some cases, best exemplified by Company 8, opposition to the introduction of ISO 9000 centred on its perceived emphasis on procedure rather than product Quality: "*I worked in companies who were so hung up on ISO 9000. And it just didn't work. They made crap products but by God they had ISO in*".

ISO 9000 was seen to be closely associated with *Bureaucracy* as the participants variously describe ISO 9000 as 'way over the top', carrying 'a lot of baggage', being 'heavyweight', and having significant 'overhead'. The major opposition to it is because of what managers believe is its overemphasis on *Documentation*, which was best summarised by Company 5: "*But in one way ISO doesn't focus on the important bits at all, it's still a very paper driven thing. You can get away with having an ISO system that doesn't actually do any source code control at all and still get your ISO certification*". Small software companies and start-ups are especially wary of ISO and the amount of *Documentation* required by the standard. Company 16, who are preparing to enter a regulated market, attempted, unsuccessfully, to introduce ISO on start-up: "*We started off with trying to follow a*

kind of ISO model, and that was just crushing us in paperwork and we abandoned it because we have a small number of engineers and we needed to be producing output".

From the earlier interview extracts, and further analysis of the study data, there is a strong link between the reason for ISO 9000 rejection and the *Cost of Process* arguments, with companies reluctant to engage in SPI because of its association with increased *Documentation*, it is not surprising that they would be hostile to ISO 9000 if they perceive it as having a similar *Documentation* requirement.

The three companies in the study who have ISO 9000 certification pursued it primarily for business reasons. Of the remaining 18 companies, only one company is actively considering it, and this is because they are entering a regulated sector. For Company 1, the introduction of ISO 9000, was undertaken to gain a contract from a telecommunications Multi-National; Company 6 who sell to the pharmaceutical sector were facing market barriers (FDA approval) and need ISO 9000 to remove these.

Company 14 is the only other company in the study to have ISO 9000 certification and cited 'market advantage' as the reason they pursued it: "*It was felt by upper management that it would be very advantageous if we had it up front. It gives you more weight that you are serious about what you are doing. It gives you a good name and good reputation*".

The fact that ISO 9000 certification is being sought for business reasons rather than process improvement reasons, lends credence to the perspective that process change is reactive (to business events) and not proactive (for quality / process improvement). From the interviews, and detailed analysis of the managers' views, there is no evidence that in these companies certification was pursued in order to achieve improved quality or development capability. Company 12's CTO, who best represents the views of a number of the study managers, describes it in the following terms: "*If somebody said tomorrow, you won't sell into the financial services sector unless you are CMMI or ISO 9000 compliant or whatever, we would very quickly get certification. It's commercial reality that if someday you are forced to do something, you will do it quickly*".

Within the interviews, there were quotes from 15 of the 21 companies which are critical of ISO 9000 from a *Documentation*, *Bureaucracy* and administrative perspective. This leaves a situation where more than three quarters of the companies in this study firmly oppose the adoption of ISO 9000 in their software development.

Discussion

The research question addressed in the study, *why are software companies not using 'best practice' SPI models* produced the study's core concept *Cost of Process*. Implementing and maintaining any SPI initiative incurs significant cost, and the financial and time implications of introducing some of the commercial SPI and quality models was presented above. Significantly, the resources required to implement SPI are proportionately much greater in smaller companies, and those smaller companies intent on, firstly, survival and then stability, have many competing and higher priorities than SPI. As all of the study companies, at time of interview, fell into the EU-defined SME category, it is therefore perhaps not surprising that they would reflect greater hostility to SPI models that required them to divert resources from what they would perceive as more deserving activities. For many of the interviewees, SPI creates an additional burden or weight to their development efforts resulting in increased *Documentation* and *Bureaucracy*. Companies, to reduce their process overhead, substituted verbal *Communication* for *Documentation*. Development teams were co-located to ensure ease of verbal exchange and reduce the need for the written word. Even larger companies attempted to reduce *Documentation* cost by decomposing teams into smaller, more manageable, units. A benefit of doing this was an increase in *Tacit Knowledge* exchange, whereby the knowledge present in each team member was more easily shared. SPI was also resisted by the smaller companies who believed it would negatively impact their *Creativity* and *Flexibility*.

Overall, respondents felt that the resources required to implement the commercial models far exceeded the benefits that may accrue. In some cases however, managers saw no benefit at all to the commercial models and believed they would hamper business prospects.

Implications for Industry

The findings of this research contain useful lessons for software entrepreneurs who need to make decisions about process and process change within their organisations as they grow. The study has uncovered evidence that many companies are benefiting from informal *Communication*, particularly verbal *Communication*, and *Tacit Knowledge* at the expense of detailed *Documentation*. Any organisation that follows this route needs to be aware of the advantages and disadvantages associated with this approach. Companies who have gained from sharing *Tacit Knowledge* have generally had a workspace and supporting environment conducive to informal information exchange between employees. Organisations who have a more closed and rigid workspace will have to consider measures to overcome this if they are to implement a policy supporting informal *Communication*.

Implications for Researchers

Small software companies, in the first instance, focus exclusively on survival. This, in part, explains the success of agile methodologies whose 'light', non-bureaucratic techniques support companies in survival mode attempting to establish good, fundamental software development practices. Though CMMI is firmly anchored in the belief that better processes mean better products, many small Irish software product companies are merely concerned about getting a product released to the market as quickly as possible. Development models, such as those within the agile family, rather than CMMI or ISO 9000, are perceived as supporting this objective. This clearly poses questions for CMMI and ISO 9000 researchers. Despite the fact that researchers may classify methodologies as only one element within a software process, practitioners, as shown in this study, clearly do not make such distinctions between methods and process. SPI researchers must reflect on the fact that, as this study shows, small companies are significantly more interested in methods than process, and methods such as XP are far more attractive to practitioners in these situations than processes such as CMMI or ISO 9000.

The question of how CMMI can produce positive results in small settings has been explored by a number of researchers, however, the argument put forward within our study is that small software companies grudgingly commit resources to SPI only when absolutely necessary and even then operate off a minimum process. As a result, 'one-size fits all' models such as CMMI are always going to find it difficult to penetrate small software organisations. Such contextual realities must be considered by SPI researchers.

Therefore, the wider implications are that though significant research time has been spent on endeavouring to prove that CMMI can work in small settings, perhaps too little time has been spent investigating why software SMEs are not prepared to adopt or even experiment with these models. Thus, examining the reasons for the rejection of CMMI by small software companies is something that could be usefully addressed in future studies.

Study Limitations

Grounded Theory has some definite limitations (Norman, 2007). As qualitative research studies using semi-structured interviews, Grounded Theory investigations centre on respondents' opinions. The findings, and the resultant theory, depend on the data gathered in the field, that is directly from the participant interviews. However, this opinion is the respondent's view or perception of what is taking place, which of course may be at odds with reality. However, it is not the role of researchers to second-guess their interviewees. As such, researchers must accept the veracity of what respondents say during the study interviews (Hansen and Kautz, 2005). Notwithstanding the issues surrounding semi-structured interviews, the opinions of the participants are vital. In this research, even though the reality of the situation could be potentially different to that described, it is the managers' perception of what is happening, and it is on this perception that they base their decisions.

Another potential limitation of the research is the fact that interviews were only sought, and conducted, with senior managers. Whilst extensive efforts were made to ensure proper diversity in the field data, and that reports were gathered from different sized companies in different sectors, the interview pool consisted solely of a very senior person in each organisation. A study purely from the engineers' perspective might generate a different outcome, but it would lack the crucial 'big picture' view that senior managers can provide. Similarly, it is generally the senior managers who have decision-making responsibility for process model adopted. A study focussing exclusively on engineers would be deficient in depth and breadth of organisational approaches.

Conclusions

Though it is not new to claim that SPI has an associated cost, many companies are deterred from investigating SPI models because of a perceived cost. Managers' perceptions are that SPI means increased Documentation and *Bureaucracy*. Such a perception is widespread and is seen as a 'feature' of CMMI. Whether or not this is true is a moot point. The fact that managers associate CMMI with increased overhead results in most small company instances in the model not being considered as a solution or even worthy of investigation.

Supporters of CMMI claim that use of the models can lead to greater predictability and repeatability. Paradoxically, this works against CMMI from the perception of small, early-stage, software firms. Many small software companies, some of who may have only a single product in their marketing suite, would argue that each project and situation is new to them and that *Creativity* and *Flexibility* are far higher on their list of desired capabilities than predictability and repeatability. The companies in this study have shown that they see agile methodologies as supporting *Creativity* and *Flexibility*. Accordingly, it is easy to see how XP has achieved substantially higher usage in indigenous Irish software companies than CMMI.

Given the volume of material in the literature, it is perhaps surprising that there was no reference whatsoever, by any of the study respondents, to the ISO/IEC 15504 ('SPICE') software process assessment standard. Despite its relatively long existence, ISO/IEC 15504 has failed to pierce the consciousness of Irish software product managers and was not listed as a process option by them, this despite the fact that it is an ISO standard designed

specifically for SPI. The literature available on ISO/IEC 15504 suggests that it can be scaled for use by small and very small companies much more easily than CMMI. However, the complete absence of knowledge about the standard should give cause for concern amongst its founders and advocates.

As this study was limited to one geographical location, an expanded study to include indigenous software product companies in other countries would provide further validity for this research and indicate if the findings can be replicated elsewhere, or if they are peculiar to the Irish context. In addition, much software is developed outside the software product company domain, such as bespoke software solutions and in-house software departments of non-software companies. These developers also use software processes and a study of these, in this non-software product company environment, could be counter-balanced against this work.

References

Ahern, D.M., Clouse, A. & Turner, R., 2004, *CMMI Distilled: A Practical Introduction to Integrated Process Improvement*, 2nd Ed, Addison Wesley.

Beck, K, 2000, *Extreme Programming Explained: Embrace Change*, Addison Wesley.

Coleman G. and O'Connor R., Using grounded theory to understand software process improvement: A study of Irish software product companies, *Journal of Information and Software Technology*, Vol. 49, No. 6, pp. 531-694.

Crosby, P.B., 1979, *Quality is Free: The Art of Making Quality Certain*, McGraw-Hill, USA.

Forward, A. and Lethbridge T. C., 2002, The Relevance of Software Documentation, Tools and Technologies, in Proceedings of ACM Symposium on Document Engineering, Virginia, USA, pp. 26-33.

Glaser, B. and Strauss, A., 1967, *The Discovery of Grounded Theory: Strategies for Qualitative Research*, Chicago, Aldine.

Goulding, C., 2002, *Grounded Theory: A Practical Guide for Management, Business and Market Researchers*, Sage Publications.

Hansen, B. and Kautz, K, 2005, Grounded Theory Applied – Studying Information Systems Development Methodologies in Practice, in Proceedings of 38th Annual Hawaiian International Conference on Systems Sciences, Big Island, HI

Humphrey, W.S., 1995, *A Discipline for Software Engineering*, Addison Wesley, Boston, MA

International Organisation for Standardisation, 1992, Quality Management and Quality Assurance Standards, Part 3: Guidelines for the Application of ISO 9001 to the Development, Supply and Maintenance of Software, Geneva, Switzerland.

Juran, J.M., 1988, *Juran on Planning for Quality*, The Free Press, New York.

New Oxford Thesaurus of English, 2001, Oxford University Press.

Norman, G.. A Grounded Theory of Software Process Improvement Model Adoption. PhD thesis, West Virginia University, Morgantown, USA, 2007.

SEI, 2006, Process Maturity Profile - CMMI SCAMPI Class A Appraisal Results Mid-Year Update 2006, viewed May 2007, <www.sei.cmu.edu>

Staples, M., Niazi, M., 2006. Systematic Review of Organizational Motivations for Adopting CMM-based SPI. Technical Report PA005957, National ICT Australia.

Staples, M, Niazi M, Jeffery, R, Abrahams, A, Byatt, P and Murphy, R, 2007, An exploratory study of why organizations do not adopt CMMI, *The Journal of Systems and Software*, Vol. 80, No. 6, pp 883–895.

Zahran, S., 1998, *Software Process Improvement*, Addison Wesley, Boston, MA

Copyright