An Empirical Study of Software Process in Practice

Gerry Coleman,

Department of Computing & Mathematics,
Dundalk Institute of Technology, Dundalk, Ireland.
Tel.: +353 42 935 2828
e-mail: gerry.coleman@dkit.ie
Member of the Irish Software Engineering Research Consortium - ISERC
(http://www.iserc.ie/)
Funded by Science Foundation Ireland (http://www.sfi.ie/)

Abstract

In adopting a software process model, many small software companies are ignoring standard process models and models for process improvement. This study uses an empirical approach to investigate what processes software companies are using on a day-to-day basis and examines why these companies are rejecting "best practice" approaches.

1. Introduction

The success of a software project is generally judged on its ability to meet users' expectations, be delivered on time and adhere to original budget. In an attempt to ensure software project success, some large software organisations have used software process improvement models, such as the Capability Maturity Model Integrated for Software (CMMI-SW) and the International Organisation for Standardisation (ISO) 9001 series [1][2]. Today, however, a significant proportion of software development work is carried out by small to medium enterprises (SMEs). Because of their size, software SMEs face particular challenges when developing software, and in choosing an appropriate software process model. Evidence collected in this study shows that Irish software SMEs are not using standard software process improvement models, opting, rather, for proprietary or heavily-tailored approaches which are "good enough" for their requirements.

This paper reports on the "good enough" processes Irish software product companies are using in practice, what factors influence the composition of good enough process in software companies and examines why companies are choosing to reject standard process models in favour of a tailored minimum.

2. Background

From the data reported from the Software Engineering Institute (SEI) and International Organisation for Standardisation (ISO) it is clear that, despite a number of years of promotion and marketing, use of their process improvement models is relatively low. Software Capability Maturity Model (SW-CMM) -based assessments show that from the end of 1997 to the end of 2002, 1,345 organisations had been appraised worldwide [3]. It has been calculated that ISO figures up to the end of 2000 show that, worldwide, between 12,000 and 20,000 software companies have been assessed or certified [4]. Furthermore, the total number of TickIT (the UK standard) holders, as of July 2003, stands at 1,157 [5]. However, when you take into account that the most recent figures show that Ireland's software industry alone has in excess of 900 companies [6], it is obvious that managers in small software companies are deciding that, within the context in which they operate in practice, these "best practice" models are not most appropriate mechanisms on which to base their software development effort. In recent times, some newer process models such as the Personal Software Process (PSP) and the Team Software Process (TSP) have emerged which have been tailored towards development in the small [7][8]. These approaches, though aimed at small teams, have really only generated major improvement in large companies and have faced the charge from small software companies of being overly prescriptive and bureaucratic [9].

As many of these small software companies are not only surviving, but thriving, the processes they are using are clearly "good-enough" for their needs.

This leads us to the following questions. What process models are small software companies using in practice? What factors determine the make-up of these process models? And why, despite the evidence offered by proponents, are small companies deciding to reject standard process improvement models such as CMMI?

These questions have yet to be properly addressed in the literature and the consequences are of major importance to a range of interested parties including, CTOs/software managers in software SMEs, process model designers, software support agencies and especially software standards bodies.

3. Research approach and methodology

In addressing the research questions, I have focused my attention on indigenous Irish software companies. The research I conducted examined how the current processes used in practice have evolved within the constituent companies. In many software SMEs, the founder, or the person who introduced the software process, is still in place. As such, with indigenous companies, tracing the process's evolution is possible. Such is not the case in Multi-national software companies, as attempting to trace how the process arrived at its current state would be practically impossible.

3.1 Grounded Theory

To ascertain what is going on in a given situation and to construct a theory around it demands the use of qualitative research techniques. What I used in this study was an inductive rather than a deductive process. A deductive process begins with existing theory, uses this to draw some hypotheses and through testing these hypotheses tests the theory itself. By contrast, inductive research attempts to gather explanation and meaning through the collection and analysis of empirical data.

The methodology I chose for the study was Grounded Theory [10]. Originally developed for use in the Social Sciences, grounded theory is now being applied to other domains to provide rich explanation of practice and to develop associated theory. The procedures of grounded theory are designed to develop a well-integrated set of concepts that provide a thorough theoretical explanation of social phenomena under study. The aim is to discover categories and concepts within empirically collected data, using these to generate emergent theories which are grounded in the data. Theoretical sampling ensures constant comparison of existing categories and drives the search for contrary ideas.

In my study, the prime method of data collection was taped structured interviews with additional documentation and artifacts used to extend and complete the data collection process.

The software tool "Atlas TI", designed for use with grounded theory, was used to support the data analysis and category and concept generation activities. Existing theory is used as and when it becomes relevant to the study. It can be used to support and challenge the emerging theory whilst the grounded theory approach can also be used to enhance existing theory.

During the study, I conducted interviews with 15 CTOs or software development managers across a range of software product companies with development teams ranging in size from 2 to 100. In the search for emergent theory, I deliberately selected a range of software companies across the application spectrum from pharmaceutical and telecommunications software providers to real-time control systems and small business applications developers.

Following each interview, using the Atlas TI toolset, I transcribed and *open coded* each interview and linked any associated data or documented artifacts from the participating organisation. Each subsequent interview was then conducted based on the ideas and concepts emanating from the previous interviews. After several interviews, concepts began to emerge. After a number of subsequent interviews, I used *axial coding*, whereby the relationships between concepts are examined, to determine the core categories and help explain the discovered phenomena. At the final stage of the coding process I used *selective coding* whereby I chose a core category and related it to its various sub-categories and associated attributes.

Throughout the data collection, coding and analysis I also engaged in *memoing*. Memos are essentially notes to yourself about some hypothesis you have about a category or property, and particularly about relationships between categories. Memos help drive the research and offer explanation for the unfolding theory.

At all stages of coding and analysis I referred to the literature, as and when appropriate, in an attempt to support or refute what was emerging.

4. Study Findings

4.1 Process models

The 15 companies interviewed in the study were using a range of software process models. Interestingly, none of the companies were using a process model in a "text-book" fashion, choosing instead either to, drop elements of their chosen model or, develop something proprietary instead.

Of the companies interviewed, 3 were using eXtreme Programming (XP) as their base process model [11]. Two of those were using it "quite aggressively" but none of the

3 were using all 12 elements of XP. Of the remaining 13 companies 7 had examined XP and several of those were considering piloting it on an upcoming project.

7 of the 15 companies had used the Rational Unified Process (RUP) or "an approximation" of it [12]. None of the 7 had deployed it as is, but had tailored it, or included tenets of it, within a proprietary model. 2 of the 7 companies who used RUP had subsequently shelved it.

The remainder of the companies were using either versions of the Waterfall model or some form of iterative development approach.

4.2 Factors influencing the process model used

The reasons for companies deciding whether they were going to use a standard model, or develop their own, proved particularly interesting. Outputs from the study showed that the key influencers on a process model decision related to the situational factors inherent to each company. Primary amongst these were:

➢ *Background of CTO/development manager*

The background of the development manager or CTO was a major influencer on the software process deployed. In many cases the software process used was that brought by the CTO to the company when they joined. If, for example, they used Rational in a previous employment, then generally this provided the basis for the software process in their current organisation. Some of the CEOs (who in the very small companies also acted as CTOs) came from non-software backgrounds. These individuals were the most hostile process critics and their organisations showed, what might be described as, least process maturity or cognisance. Conversely, the biggest process supporters came from those who had spent a number of years in software development prior to founding a software company or becoming CTO. The organisations containing these individuals showed the highest level of process awareness.

➢ *Customer/application type*

The Customer/application type also had a major influence on the degree, or type, of process used. The companies interviewed, who are operating in the telecommunications market, are selling products which will be used 24/7. In these cases, their customers placed a premium on high-quality and high-reliability. These demands have driven the introduction of process in these organisations resulting in a greater emphasis on testing and quality assurance activities. Similarly, another company interviewed is selling software products to multi-national corporations in the pharmaceutical sector. This business, governed by high-regulation, and deploying products worldwide, also demands high-quality, reliability, and importantly, certification. The software process in the company involved reflected this and they had been certified to ISO 9000/2000 standard. Without a strong process and associated certification they would not have generated much of their pharmaceutical business.

By contrast, other companies interviewed, selling non-business-critical applications, place less emphasis on process and had no desire to pursue any type of certification.

➢ *Situation pre-process*

In many of the organisations the process developed haphazardly and eventually encountered significant problems. The nature of those problems often drove the type of process that was subsequently implemented and the elements that took priority. For example, one organisation reported that, in the early stages, analysts worked with customers without agreeing a documented specification. Subsequently no design documents were created. As a result, this company decided it need an automated document management system and this became a process improvement priority. Another example, applied to a company who were unsuccessfully trying to manage multiple code bases across an ever-widening number of platforms. To resolve this, the organisation concerned recruited a raft of senior management personnel to get the problem under control and then establish a coherent process.

Another product company recruited a development manager who was greeted with a situation "where it was absolutely chaos". According to the development manager, the company "hadn't gone through the normal phases of growth in terms of processes and procedures. They worked 24/7; forever. They never made dates. It always went 5 times over budget. No one ever knew the status of the projects, it was just chaos." She tackled the situation by initially implementing basic project management practices and attempting to get some control on development. The organisation was also developing a number of customised versions of its basic product and attempting to support them. She phased out these customisations and the next phase of process involved scaling back the number of products developed and supported and the creation of a strong configuration management programme.

➢ *Project/team size*

As might be expected, the scale of the work, and the number of people required to do it has influenced the process in the companies interviewed. However, what is key here is the level of process used rather than the process itself. In most cases, where the project is small, and ergo team size, some process steps and associated documents are omitted during development – "what is suitable in a 2-person team is not suitable in a 20-person team". Most of the companies involved have attempted to keep their team size to maximum of 6 feeling that, beyond that, management becomes "too difficult". The largest company interviewed impose a 25-person maximum on team size.

They feel that their existing process can cope up to this level. Beyond this they feel their process will not scale and would need substantial revisiting and redefinition.

➢ *Product/Services model*

All of the companies interviewed were software product companies and the stage of a product's evolution can influence the process used. In all cases, the number of process steps used to develop and release a new product, are reported as greater than those for a new release of a product which has been on the market for several years. Where an upgrade like this is performed, a number of process elements, standard during new development, are omitted. Furthermore where software patches are required, the process deployed is often a skeleton of that used for a full product development.

➢ *Influence of key staff*

In the very small companies interviewed, key staff have a pivotal role in the adoption and application of process, the process often being the application of their historic practices. Equally, resistance to SPI by key staff can create major difficulties in attempting to introduce new ways of working. As the companies interviewed became larger, a more standardised process, which placed less emphasis on individuals, emerged.

Though a company's software process is fluid and dynamic, and changes as the company changes, the prioritisation of improvement strategies, in this way, shapes how a process evolves within an organisation.

4.3 Perceptions of Process

Some of the most interesting aspects of the study emerged however, when the managers were asked about what software process meant to them and their organisation. They were very strict in adhering to the good enough or just enough principle and separated process in their minds as having two component parts; the activities required to carry out software development (requirements capture, design, coding, testing etc.) and the associated documentation, recording and paperwork. It was this second element which worried the managers most.

In many cases, amongst the interviewees, process improvement was perceived as "more process", spawning the related fears of additional administration, recording and overhead.

One CTO put it thus – "*we knew we had too much [process] when there was more administration being done than development. 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.*"

Another, highlighting the fear of a significant administrative burden, put it more bluntly, *"from a making money perspective you want every engineer to be working on billable work every time"*.

Also, the "code wins" approach peppered the interviews as the following quotes illustrate:

"I think a lot of commercial products out there are vastly over-engineered. I have learned that the hard way through Yourdon and drew diagrams for 2 years and didn't produce any code."

"One of the things I don't like with software companies I have worked for is the amount of programmers who end up doing admin work that they don't particularly want to do. And they tend to be the most senior guys who will deliver the most bang for buck in terms of coding."

"I'm an engineer. I've got to write this software and it has to be delivered in 3 weeks time and there is the pressure of delivering that. And if you add process in on top of that, unless people get into the habit of doing it on a day-to-day basis, where you really instil it as it will take you 10 minutes a day or 8 hours at the end of the project, and at the end of the project you won't remember what happened if you did. But so often, people were filling in time sheets and lists 6 weeks after the project had finished in order that the quality process could be seen to pass its audit."

From the examples above and the numerous others contained within the interviews it was clear that ***cost*** was a primary reason that companies were opting for a "good enough process" approach and that cost was manifest through additional administrative activity and recording. Given that excessive process was perceived as having an unacceptable cost, I decided to examine if this would explain why software SMEs are deciding not to adopt some of the standard process improvement models such as CMMI-SW and ISO 9001.

4.4 Cost of Process and improvement models

One of the difficulties with CMMI-SW, and ISO 9001, is that they are essentially top-down and require implementation at a corporate level. For smaller companies and teams this poses a particular difficulty.

In the case of the SW-CMM there is very little evidence of the processes scaling down successfully to small companies. Recent data shows that of the more than 1300 organisations (reporting size data), that have been appraised, roughly 148 (11%) have 25 or fewer software development personnel and c.224 (17.1%) between 25 and 50 software developers. [3]. More instructively, however, of the 148 companies, with 25 or fewer personnel, only

2.8% or 4 companies have been ranked above SW-CMM level 3. This is the lowest percentage for any of the reported size categories and is hard evidence of the difficulties of scaling down the SW-CMM.

Another interesting fact is the time required for an organisation to progress to the next higher level in the SW-CMM scale. The average time for organisations to move from level 1 to level 2 was 22 months and from level 2 to level 3, 21 months. Given that almost 82% of 1 – 25 software development employee companies, and almost 74% of 26 – 50 software development employee companies are in these sectors, the figures require a significant commitment on the part of small companies to make the necessary progressions. Furthermore, for a small company to get from level 1 to level 3 will take an average of 43 months or in excess of 3.5 years. Devoting the resources necessary to achieve this level of process improvement over this lengthy time span is beyond the capability of the vast majority of small companies.

Further evidence of the potential cost of pursuing CMMI-SW is provided in [1] as the following passage shows:

"A considerable investment of resources is required for a full SCAMPI (Standard CMMI Appraisal Method for Process Improvement) appraisal. Data on actual resources used to date is not easy to obtain. Clearly the time needed for the on-site appraisal includes that for the activities and efforts of a SCAMPI lead appraiser, the sponsor, the co-ordinator, each appraisal team member, and each appraisal participant. There are many variables in reaching an estimate, such as number of process areas, size of the team, number of disciplines to be investigated, and so on. This might end up being in the range of 100 to 200 days of effort. However, for a full appraisal this is not an accurate measure of the total effort required. In order for the team to operate in "verification" mode, rather than in "discovery" mode, a much larger effort must precede the on-site visit to collect and organise all the individual pieces of evidence required by the team. The total cost could easily reach into the hundreds of thousands of dollars."

The authors go on to say, *"The ARC (Appraisal Requirements for CMMI) allows 3 classes of appraisals. The level of effort for each class varies based on the scope of the appraisal. In general, however, a Class C appraisal takes much less time than a Class B or A appraisal. The Class B appraisal provides a more in-depth look at the organisation [than Class C]. In most cases, it does not go into as much depth or detail as a Class A appraisal.* ***As a result it can generally be completed within a week (****my emphasis***)."*

The figures above suggest that the cost, in terms of effort and resources, makes it prohibitive for SMEs to pursue CMMI-SW certification. But how does this evidence compare with what has emerged from the study? What do the practitioners themselves feel about CMMI or ISO?

Of the companies interviewed, 2 have adopted ISO 9001. Both organisation were required to do so as their primary customers, a telecommunications multi-national in one case, and the pharmaceutical sector on the other, demanded it. The software companies estimated the cost, of achieving certification, to be €90,000 ($100,000+).

However, where there was no customer requirement for such certification, other companies have studiously avoided it, particularly where the managers concerned have worked with ISO or CMM prior to taking up their current roles.

One manager commented as follows, *"It comes back to the speed at which you are developing your products. If you look at CMM, it was delivered for the likes of NASA programming and you are given 2 – 3 years for your launch date. 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."*

Another small company were wary of CMM and its implications, *"It will depend on the companies with whom we will engage. It hasn't been a big deal. But 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".*

A manager who previously worked in a company rated at CMM level 3 stated, *"There is some sense in that stuff. But I certainly wouldn't go looking for CMM level 3 certification. It's way over the top. I'd be keener to hire somebody who has been through the mill and who understands a set of key tenets across similar industries. It (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."*

However, managers who previously worked in ISO 9001-certified organisations were even more hostile to its adoption.

As one put it *"And so I'm nervous especially of things like ISO, that I think carries a lot of baggage that doesn't necessarily give you benefit. Today as a business benefit it's not relevant to us. We talk of "we have equivalent processes" but it's for the future I think."*

Another was more critical of ISO, *"A conscious management decision was not to go completely ISO. If we had gone ISO in the early stages we do believe that that would have sunk the company because we would be really only conforming to these standards for the sake of it and to get a badge which you don't need in reality. We are winning over a lot of our competitor's clients and that's without ISO. I believe it would take a big investment of time and money and think it's way over the top."*

One manager reasoned that ISO was not suitable for software thus, *"But in one way ISO doesn't focus on the important bits at all, it's still a very paper driven thing. In other words, you can get away with having an ISO system that doesn't actually do any source code control at all and still get your 9001 certification. I've seen it done."*

Surprisingly, one company who had achieved ISO 9001 certification were quite sanguine about it, *"But what you find now is the opposite when you get to the bigger customers, the very big ones, they sometimes say just because you are ISO 9001 certified doesn't mean you produce a good product"*.

Finally one manager described her experience of working in a company with ISO certification, *"I found it absolutely ridiculous. I couldn't work there. I felt it killed creativity. If I hear the word ISO I break out in a rash! Any company I've ever gone into who lives like those, they tend to be really dull places."*

Clearly, in the case of process improvement models for software development, there is a major gap between theory and practice. Except in the cases where customers are demanding it, managers in software SMEs are rejecting CMMI-SW and ISO because of the perceived cost and bureaucratic overhead associated with their adoption. Similar phrases continually cropped up in relation to CMMI and ISO during the interviews; "rigid", "baggage", "bureaucracy", "buried in paper", "forced into filling out lots of forms", "bulky", "heavy", "luxury", "major drag factor", "over the top" "overkill", "we don't have time" and "wouldn't have the patience".

Whether these perceptions are actually true in relation to CMMI and ISO, it is this mindset that the proponents of these standards have to address.

5. Conclusions and further work

This study examined software process as it is practiced in Irish small to medium-sized software enterprises. A grounded theory approach was used to determine what processes were used by the participant companies, what factors created these processes and how they evolved and investigated why Irish SMEs are rejecting software process improvement models such as CMMI-SW and ISO 9001. The clearest finding from the study is that process, though necessary in terms of its development-related activities, has a cost in terms of administration and bureaucracy. The practitioners concerned wish to minimise this cost and see process improvement models as adding "more process" which translates as increased bureaucracy.

In the next phase, I plan to return to some of the participants to investigate what adjustments could be made to the CMMI-SW to make it more attractive to small software companies. I also intend to examine how XP is functioning as a process model and how it has been rolled out amongst the companies who were experimenting with it during the study period. It will also be instructive to see if there are any lessons from using XP in practice which can lead to it being accommodated within CMMI. From the above I intend to construct a contingency model which can guide small software companies in the selection and application of software process as they evolve thereby ensuring that their active process remains "good enough" to generate continued success.

6. References

[1] Ahern, D.M., Clouse, A. & Turner, R., *CMMI Distilled 2^{nd}Ed*, Addison Wesley, 2004.

[2] ISO 9001:2000, www.iso.org.

[3] Process Maturity Profile Year end 2002, http://www.sei.cmu.edu/sema/pdf/SW-SW-CMM/2003apr.pdf

[4] The ISO survey of ISO 9000 and ISO 14000 certificates, http://www.tantara.ab.ca/b_board.htm

[5] International Register of TickIT-Certified Organisations, http://www.tickit.org/cert-org.htm.

[6] Irish Software Industry Statistics, 1991-2002, http://www.nsd.ie/htm/ssii/stat.htm

[7] Humphrey, Watts S. *A Discipline for Software Engineering*, Addison Wesley, 1995.

[8] Humphrey, Watts S. *Introduction to the Team Software Process*, Addison Wesley, 2000.

[9] Coleman, G. and O'Connor, R. "Power to the Programmer: Using Measurement to Optimise the Software Process at the Individual Level", Proc. 11th ESCOM Conference, Munich, 2000.

[10] Corbin, J.M. & Strauss A. *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*, Sage Publications, 1998.

[11] Beck, K. *Extreme Programming Explained: Embracing Change*, Addison Wesley, 1999.

[12] Kruchten, P. *The Rational Unified Process: An Introduction*, Addison Wesley, 2004.