

Using grounded theory to understand software process improvement: A study of Irish software product companies

Gerry Coleman ^a, Rory O'Connor ^{b,*}

^a *Department of Computing, Dundalk Institute of Technology, Dundalk, Co. Louth, Ireland*

^b *School of Computing, Dublin City University, Glasnevin, Dublin 9, Ireland*

Available online 13 February 2007

Abstract

Software process improvement (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. Accordingly, SPI researchers must be equipped with the methodologies and tools to enable them to look within organisations and understand the state of practice with respect to software process and process improvement initiatives, in addition to investigating the relevant literature. Having examined a number of potentially suitable research methodologies, we have chosen Grounded Theory as a suitable approach to determine what was happening in actual practice in relation to software process and SPI, using the indigenous Irish software product industry as a test-bed. The outcome of this study is a theory, grounded in the field data, that explains when and why SPI is undertaken by the software industry. The objective of this paper is to describe both the selection and usage of grounded theory in this study and evaluate its effectiveness as a research methodology for software process researchers. Accordingly, this paper will focus on the selection and usage of grounded theory, rather than results of the SPI study itself.

© 2007 Elsevier B.V. All rights reserved.

Keywords: Software engineering; Software process improvement; Qualitative research methods; Grounded theory

1. 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 provide assistance in making estimates, developing plans, and measuring quality. There is a widely held belief that a better software process results in a better software product, which has led to a focus on SPI to help companies realise the potential benefit. SPI models [12,16], developed to assist companies in this regard, 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. Translating these benefits into practice has, however, proved challenging. Opponents believe that these models operate primarily at a theoretical level, are too prescriptive and bureaucratic to implement in practice, and require a subscribing company to adapt to the models rather than having the models easily adapt to them. 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.

The motivation for our research originates in the premise that software companies are not following ‘best practice’ process improvement models. On this basis, we initially set out to explore two primary questions: *Why are software companies not using ‘best practice’ SPI models?*, and *What software processes are software companies using?* to create a rich, explanatory theory of software process in practice. This then produced the following research questions:

* Corresponding author. Tel.: +353 1 700 5643; fax: +353 1 700 5990.

E-mail addresses: gerry.coleman@dkit.ie (G. Coleman), roconnor@computing.dcu.ie (R. O'Connor).

- RQ1 How are software processes initially established in a software company?
- RQ2 How do these software processes change?
- RQ3 What causes these software processes to change?
- RQ4 How do the operational and contextual factors, present in organisations, influence the content of, and adherence to, software processes?

In order to answer these questions it was first necessary to define both a context and scope for the study. 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.

This paper is organized as follows: Section 2 describes the background for the choice of research methodology and explains the reasoning behind the choice of grounded theory. Section 3 provides an overview of grounded theory and Section 4 describes how it was applied in this study, along with the study's main findings. In Section 5 an evaluation of grounded theory as a research methodology for SPI researchers is discussed. Finally, Section 6 presents some concluding remarks.

2. Research methodology

Philosophical assumptions underpin the research process which dispose researchers towards different paradigms and methodologies [8]. The two research paradigms that have received most attention in the literature can be broadly labelled as positivist and phenomenological [37] or positivist and interpretivist [5]. The most commonly used terms to differentiate these paradigms with respect to their associated methods and techniques, are *quantitative* and *qualitative*, respectively, with *quantitative methods* being

based on the positivist paradigm while *qualitative methods* are built on a phenomenological worldview [13,17].

Creswell [13] suggests that the choice of paradigm adopted by researchers will depend on the 'worldview' that exists within their discipline. The paradigm chosen will also largely depend on the way in which previous research has addressed similar problems, existing theories in the area, known variables, the research questions and the extent to which measures have been developed and validated. In addition, pragmatic reasons such as the time, resources and access available are also necessary conditions. To determine the research design of the present study, the degree of fit between research questions and methodological choices available to researchers needs to be considered.

Scientific enquiry, which employs quantitative research methods, is used to establish general laws or principles [7] and its approach can provide answers which have a provable base. However, if one wants to study human behaviour and the social and cultural contexts in which it functions, then the limitations of quantitative research become apparent [31] and direct researchers towards qualitative techniques. Advocates of qualitative methods in software engineering research propose that a principal advantage of their usage is that they force researchers to delve into the complexity of the problem rather than abstract away from it thus making the results richer and more informative [42].

2.1. Qualitative research

Qualitative research is directed primarily at collecting and analysing non-numeric data with the aim of achieving information depth rather than breadth [4]. Where quantitative research is concerned with questions such as, how much?, how many?, how often?, qualitative research is linked with questions such as why?, how?, and in what way? In addition, where quantitative research frequently operates in a *deductive* way, qualitative research frequently operates in an *inductive* way. 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. Saunders et al. [39] describe it thus, 'Where you commence your research project from a deductive position, you will seek to use existing theory to shape the approach which you adopt to the qualitative research process and to aspects of data analysis. On the other hand, where you commence your research project from an inductive position, you will seek to build up a theory which is adequately grounded in a number of relevant cases'. Similarly, inductive-based research can also play an important role in the generation of hypotheses [18].

2.2. Qualitative research in software development

The use of qualitative research in software development studies has been more widely embraced within Information

Systems (IS) than within Software Engineering (SE). The focus on technological issues in SE studies, such as [6,14,15,26,27], and the associated extensive use of quantitative methods, has been criticised by Bertelsen [3] who argues for the use of qualitative research in SE. He contends that as SE is a ‘*socio-culturally, not a technically, constituted phenomenon*’ any research conducted “cannot be based exclusively on natural science approaches but must include a way to understand psychological, social, and cultural phenomena”. We agree with Bertelsen in believing that, to get an accurate picture of SPI in practice, one must investigate beyond purely technological factors. However, much of the published work, which uses qualitative research methods and which explores issues beyond technology, resides in the area of IS. Therefore, to see what lessons can be learned for qualitative studies in software development, which address social and cultural issues, in addition to technological factors, it is necessary to draw on experiences from IS research.

Hevner and March [25] believe the goal of IS research is to support the application of information technology for managerial and organisational purposes. Lee and Liebenau [28] believe that qualitative research is required in IS because, ‘*while there has been great success in applying natural science and engineering models to research into computer technology, they have been inadequate and inappropriate in explaining the human, group, organisational and societal matters which surround the use of information systems*’. Myers [31] notes that there has been a move away from technological to managerial and organizational issues, and this, he feels, is responsible for an increased interest in the use of qualitative research.

As the objectives of this research relate to generating theory, which is built on the ‘voices’ and ‘experience’ of software practitioners, a qualitative approach was chosen as the appropriate methodological vehicle for the study. Also, the study setting is indigenous Irish software companies and a particular strength of qualitative research is its ability to explain what is going on in organisations [1]. Of the qualitative methodologies available, grounded theory offered the best mechanism for achieving the research objectives. The reasons grounded theory was chosen are as follows:

- Given the lack of an integrated theory in the literature as to why software companies are avoiding SPI models, an inductive approach, which allowed theory to emerge based on the experiential accounts of software development managers themselves, offered the greatest potential.
- It has a set of established guidelines for conducting inductive, theory-generating research.
- It is renowned for its application to human behaviour. Software development is a labour intensive activity and software process relies heavily on human compliance for its deployment.
- It is an established and credible methodology in sociological and health disciplines (e.g. nursing studies, psy-

chology), and a burgeoning one in the IT arena. This study provided an opportunity to apply a legitimate and suitable methodology to the software field.

The founders of grounded theory have not only been concerned with the processes associated with social psychology but also with the conditions that give rise to these processes. Furthermore, like other grounded theorists [2,24,34], this study attempts to understand a dimension of software development in practice. From a software process perspective the role of individual actors, and their environmental surroundings and conditions, weighs heavily on how the process is practiced. Facilitating the gathering and analysis of those human experiences and the associated interrelationships with other human actors, coupled with situational and contextual factors, are particular strengths of the methodology.

3. Grounded theory

Grounded Theory was first established by Glaser and Strauss [20]. The theoretical foundations of grounded theory stem from Symbolic Interactionism, which sees humans as key participants and ‘shapers’ of the world they inhabit. Grounded theory was created from the ‘constant comparative’ method, developed by Glaser and Strauss, which alternated theory building with comparison of theory to the reality unveiled through data collection and analysis. The emphasis in grounded theory is on new theory generation. A theory, according to Strauss and Corbin [44], is ‘*a set of well-developed categories (e.g. themes, concepts) that are systematically interrelated through statements of relationship to form a theoretical framework that explains some relevant social, psychological, educational, nursing or other phenomenon*’. 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 continuous interplay between data collection and analysis of that data [23]. According to Strauss and Corbin, the theory that is derived from the data is more likely to resemble what is actually going on than if it were assembled from putting together a series of concepts based on experience or through speculation [44].

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. They 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.

3.1. Components of grounded theory

3.1.1. Theoretical sampling

Theoretical sampling refers to the process of collecting, coding and analysing data whilst simultaneously generating theory. Interviews, both formal and informal, are at the core of the data collection process. Because the grounded theorist does not know in advance where the theory is going to lead them, only the initial sampling can be planned. Based on the emerging theory, researchers may change the list of questions asked to reflect more closely the emergent categories.¹ Based on category development, the researchers might then choose to interview certain types of individual or seek out other sources of data. As the concepts and categories continue to emerge, theoretical sampling becomes an ever-changing process. Grounded theory researchers engage in ‘constant comparison’ between the analysed data and the emerging theory. This process continues until ‘theoretical saturation’ has been reached, ie. whereby additional data being collected is providing no new knowledge about the categories.

3.1.2. Open coding and analysis

From the interview transcripts the researchers analyse the data line-by-line and allocates codes to the text. The analytical process involves coding strategies: the process of breaking down interviews and observations into distinct units of meaning which are labelled to generate concepts. The codes represent concepts that will later become part of the theory. The codes themselves provide meaning to the text and may be created by the researchers, or may be taken from the text itself. A code allocated in this way is known as an *in vivo* code. *In vivo* codes are especially important in that they come directly from the interviewees, do not require interpretation by the researcher, and provide additional ontological clarification or context-description. From the initial interviews, a list of codes emerges and this list is then used to code subsequent interviews. At the end of the sampling process a large number of codes should have emerged.

3.1.3. Axial coding

Axial coding is the process of relating categories to their subcategories (and) termed axial because coding occurs around the axis of a category linking categories to subcategories at the level of properties and dimensions. This involves documenting category properties and dimensions from the open coding phase; identifying the conditions, actions and interactions associated with a phenomenon and Relating categories to subcategories.

3.1.4. Selective coding

Selective coding is the process of integrating and refining the theory. Because categories are merely descriptions of

the data they must be further developed to form the theory, the first step is to identify the central, or ‘core’ category around which the theory will be built. As the core category acts as the hub for all other identified categories, it must be central in that all other categories must relate to it and it must appear frequently in the data.

3.1.5. Memoing

Memoing is ‘the ongoing process of making notes and ideas and questions that occur to the analyst during the process of data collection and analysis’ [40]. Typically, ideas which are recorded during the coding process, memos assist in fleshing out the theory as it emerges and are written constantly during the grounded theory process. Memos may take the form of statements, hypotheses or questions. In the latter part of the study, following extensive coding and analysis, memos become increasingly theoretical and act as the building blocks for the final report.

3.2. Which version of grounded theory

Since the initial launch of grounded theory, the Glaser and Strauss alliance gradually separated until each was developing a different version of the methodology. First in 1990 [45], and in a follow-up [44], Strauss, now in conjunction with Corbin, created an updated version of grounded theory with extended coding systems. This new implementation of the methodology drew criticism from Glaser [19] for being formulaic and thereby forcing a theory from the data rather than letting the theory naturally emerge as suggested in the original incarnation. Some of Glaser’s criticisms were acknowledged by Strauss and Corbin and were incorporated into [44]. Glaser continues to argue in favour of being true to the original belief that the theory should ‘emerge’ from the data and claims that Strauss and Corbin’s approach means, not a grounded theory but a ‘forced’ description. Strauss and Corbin reject this saying the data ‘are not being forced; they are being allowed to speak’.

Glaser, and Strauss and Corbin also differ on other fundamentals. Glaser believes that the research problem and question are only discovered when coding begins whilst Strauss and Corbin believe a question should be pre-set as it sets the boundaries around the study area. Similarly, Strauss and Corbin adopt a more pragmatic approach than Glaser by assuming the researcher enters the field with some knowledge of the phenomenon to be studied. Also, the role of the literature separates the authors, with Glaser believing that the literature should be largely avoided before study commencement for fear of creating prior assumptions, whilst Strauss and Corbin recognise that there should be some pre-exposure to the literature which should be referred to as the need arises.

As a results of these divergences, it is incumbent on every researcher using grounded theory to indicate which implementation of the methodology they are using. Though acknowledging and recognising the spirit of Glaser’s original version, this study employed the Strauss and Corbin

¹ A category is a ‘phenomenon, that is, a problem, issue or event that is defined as being significant to the respondents’ [44].

approach [44] for the following reasons. Strauss and Corbin argue that the researcher's prior 'experiential data', basically their personal or professional experience, is supportive of theory building and contributes to 'theoretical sensitivity', the ability to understand the data's important elements and how they contribute to theory. The experience factor is also highlighted in [18], who describes the concept of the 'cultural insider', as one who has prior expertise or practitioner knowledge of the domain. Having operated as software process consultants and professional software engineers for a number of years, the researchers' 'insider knowledge' offered potential benefits to theory building, and strongly supported the use of Strauss and Corbin's version of the methodology in this study. The researchers' professional experience also provided a familiarity with the literature surrounding the study area thus supporting theoretical sensitivity.

The motivation for our research, as described in Section 1 to this paper, also encourages the use of the Strauss and Corbin version of grounded theory as they favour setting the research question in advance of commencing a grounded theory study, rather than it being allowed to 'emerge' at the coding phase as advocated by Glaser.

3.3. Grounded theory in information systems development

Because of its interpretivist emphasis, and its ability to explain socio-cultural phenomena, grounded theory has been primarily used in the fields of sociology, nursing and psychology from the time of its establishment in the late 1960s. Since then, however, it has widened its reach into the business sector and latterly into the IS field, where it has been used to explain intentions, actions, and opinions regarding management, change and professional interactions. Silva and Backhouse [43] support its use arguing that, '*qualitative research in information systems should be led by theories grounded in interpretive and phenomenological premises to make sense and to be consistent*'. Myers [31] believes that grounded theory has gained growing acceptance in IS research because it is a very effective way of developing context-based, process-oriented explanations of the phenomena being studied.

Probably the best example of the use of grounded theory in the IS field is [32]. This study showed how grounded theory could be used to explain the impact on two organisations that implemented CASE tools to support their software development activity. The use of grounded theory in Orlikowski's study enabled a focus on the contextual issues surrounding the introduction of CASE tools as well as the role of the key actors instigating, and at the receiving end of, their adoption.

A number of researchers have used grounded theory to look at a diverse range of socio-cultural activities in IS. [2] used a novel combination of action research and grounded theory to produce a grounded action research methodology for studying how IT is practiced. Others have used the methodology to examine, the use of 'systems thinking'

practices [21], software inspections [10,41], process modelling [9], requirements documentation [34] and virtual team development [38,35, and 24] used grounded theory to study the use of development practices in a Danish software company and concluded that it was a methodology well suited for use in the IS sector.

4. The SPI study

This section describes how the theory was developed through the different stages of the study. First, we describe how the research questions were expanded following the initial and Stage 1 interviews and how the grounded theory contained therein emerged from the coding, memoing and categorising activities central to the methodology, during Stage 2. The emergent grounded theory is summarised and shown as a network diagram which identifies the relationships between the major themes, core category, linked categories, and associated attributes.

4.1. Preliminary Study Stage

Despite the research questions being clearly defined, the theoretical sampling approach of grounded theory means it is unclear in advance the number and types of practitioners that need to be interviewed to meet the research objectives. Because of this, a Preliminary Study Stage was embarked upon to generate more detailed information on how the sampling process should progress. In all, a total of 21 companies participated in the entire study, as illustrated in Table 1 with companies 1–3 participating in the Preliminary Study.

Company 1 was chosen as, within it, one of the researchers had several contacts including the CEO. The company is small (three software developers) and has been in business for over 10 years. This was a good company to commence with as their business history helped address a number of the research questions directly. The company has been in operation sufficiently long to have considered the issues around software process, has both expanded and contracted rapidly primarily due to the economic boom and subsequent downturn associated with the period 1997–2002, has been selected as a subcontractor by a major telecommunications multinational, and has been awarded ISO 9000 certification. The initial interview with the CEO lasted for over an hour. Because of the ease of access, and the diversity of his experience, a second person from company 1 was then interviewed. Interview 3 was also conducted in a company in which one of the researchers had a personal contact at senior level. This company has a larger number of software developers than Company 1 and has a presence outside of Ireland. Interview 4 was undertaken with a very small company where the CEO is personally known to one of the researchers.

The objectives of the research indicated that the grounded theory created from this study would be based on the views and opinions of software practitioners and,

Table 1
Company and individual participation breakdown

Co.	Market sector	Total no. of employees	No. of employees in s/w development	Interviewee	Study Stage
1	Telecommunications	6	3	Development manager	P
2	Company secretarial	50	20	Product manager	P
3	Telecommunications	10	3	CEO	P
4	Telecommunications	70	30	CTO	1
5	Telecommunications	12	6	Development manager	1
6	Compliance management	100	40	Quality manager	1
7	Enterprise	150	100	Product manager	1 & 2
8	E-Learning	120	70	Development manager	1
9	Information quality	27	9	Development manager	1
10	Telecommunications	15	12	Development manager	1
11	Telecommunications	160	110	CTO	1 & 2
12	Financial services	35	23	CTO	1
13	Financial services	130	90	Product manager	1
14	Interactive TV	60	40	Product manager	1 & 2
15	Public sector	150	90	Product manager	2
16	Medical devices	19	9	CTO	2
17	Telecommunications	70	35	CTO	2
18	Public sector	3	3	CEO	2
19	HR solutions	30	15	General manager	2
20	Games infrastructure	40	20	Product manager	2
21	Personalisation	50	40	Technical director	2

throughout the study, semi-structured interviews were the method used to capture these. This approach is favoured by many as it has the potential to generate rich and detailed accounts of the individual's experience [23]. To support the semi-structured interviewing process, an interview guide, based on the researchers experience as 'cultural insiders' and their prior familiarity with the literature, was created for use with the first two interviews. There were 53 questions divided over four categories: Company Background, Company Development, People Issues and Software Development Strategy.

The first interview was taped and then transcribed and printed. The interview was then coded, by hand, in accordance with the open coding procedure of grounded theory. Following this, a pair of scissors was used to cut the coded pieces of text into individual strips. Then separate bundles of paper corresponding to identically coded sections were created. Memos were written as and when they occurred to the researchers during the coding. The second interview was coded in the same way as the first one, with the second being compared to the first and coded where possible according to the list of codes generated from the first interview. On completion there was an additional slew of memos but also an increasing amount of paper strips. As managing this increasing, and diverse, paper volume throughout the study was going to be impractical, a word processor was then used to manage the coding process, the links between the codes and quotations from the data, and any linking memos.

The initial interviews highlighted several drawbacks with the interview guide. These centred around its length, as capturing all of the information took an inordinate amount of time, stretching the goodwill of the interviewees, which has the secondary effect of leaving some potential fruitful lines of enquiry unexplored. These limitation drove the develop-

ment of a second interview guide which contained 34 across three categories: Company Background, People Issues and Software Development Strategy. It also contained a list of memos, and guidance for questioning, which had been generated from analysis the initial interviews, and was in accordance with the grounded theory constant comparative approach. This interview guide was then used on interview 3 and in each successive instance, the interviews and the line of questioning concentrated more on the memos and codes from the prior interview coding and analysis than on the formalised question set.

The conclusion of interview 4 heralded the end of the Preliminary Study Stage, which was primarily used to drive the theoretical sampling process. The stage highlighted two issues in particular which would steer the immediately subsequent sampling activity. First, analysis of the software companies' target market indicated that the intended list of companies, in the full study, should incorporate as many sectors as possible. Second, it was obvious that a word processor was not going to be a practical way of managing grounded theory analysis; a specialist qualitative analysis tool, which supported coding and categorising, was essential.

In addition, the finding from the Preliminary Study Stage that target product market also potentially had an influence on the software process used meant that the intended list of study companies should incorporate as many sectors as possible. A number of reference sources were used to compile the list including the Internet, trade magazines and yearbooks and professional/industry associations. In conjunction with this, the identity of the individual with responsibility for software process, within the identified companies, was sought. This resulted in a further 21 interviews across 18 companies and was conducted over two stages, Stage 1 and Stage 2.

4.2. Software support for grounded theory

Having evaluated the range of tools which are used for data management in qualitative research, we were attracted to the ease of use and comprehensive feature set of Atlas TI [30], a tool designed specifically for use with grounded theory was selected. Atlas allows for the linking, searching and sorting of data. It enables the researchers to keep track of interview transcripts, manage a list of codes and related memos, generate families of related codes and create graphical support for codes, concepts and categories. It also supports the axial and selective coding process as proposed by Strauss and Corbin [44], which is used in this study. Having installed the software, the interview transcripts from the Preliminary Study Stage were entered into the Atlas database. Having the ability to assign and allocate codes with quotations from multiple interviews speeded up the process dramatically and eased data management significantly. It also created an easier ‘visual plane’, which enabled clearer reflection and energised proposition development. A sample list of codes from this stage is contained in Table 2.

4.3. Conducting the full study – Stages 1 and 2

4.3.1. Study stage 1

In parallel with making contact with individuals known second-hand to the researcher, ‘cold’ e-mailing was used to set up the next series of interviews. The cold ‘e-mailshot’ proved surprisingly successful and generated a positive overall response rate of around 30%, which was much higher than anticipated. Study Stage 1 involved interviews with an additional 11 companies. Each interview lasted between one and one-and-a-half hours and the initial propositions emanating from the data analysis were used as general topics for investigation. Closely following the tenets of grounded theory meant that, following the initial open coding, the interviews were then re-analysed and coded axially across the higher-level categories that had emerged from earlier interviews. Any memos, or propositions, that emerged through the coding process were recorded for further analysis and inclusion as questions in subsequent interviews. A consequence of this was that the interview guide was constantly updated.

In conjunction with the theoretical sampling process, the constant comparative method was also used. This involved comparing interview-to-interview and searching for any themes or patterns in the data. Constant comparison assists in identifying concepts which go beyond description to explanations of the relationships within the data. By comparing the emerging facts for similarities or differences,

Table 2
Sample codes as assigned using Atlas TI

Absence of process	Automated documentation	Background of CEO
Acceptance test process	Automated testing	Beginnings of formality
Actual process Vs ‘official’ process	Background drives SPI	Benchmarking

broad categories, which have dimensional properties, emerge. Though a number of theoretical concepts emerged during the early fieldwork, the researchers decided to re-evaluate the study progress following the interview with Company 14. Despite the fact that similar occurrences were appearing within the data, straightforward analysis of the companies interviewed up to this point indicated a significant emphasis had been placed on one market sector. This was not a deliberate intention of the researchers but merely reflected the companies who agreed to be interviewed and the sequence in which they occurred. Though there were no significant differences in the data emanating from these software companies compared to other market sectors, in order to have real confidence in the emerging theory it was important to broaden the target company market. This approach is in accordance with both Strauss and Corbin [44] and Goulding [23], who advocate diversity in the data gathering and ‘staying in the field’ until no new evidence emerges. The researchers believed that to conclude the sampling process at this point would constitute premature closure, a mistake often associated with grounded theory [19].

4.3.2. Study Stage 2

Stage 2 involved the participation of seven new companies and comprised 10 further interviews. Three of the Stage 2 interviews involved re-interviewing Stage 1 participants. Companies 15–21 were the new subjects used for Stage 2 interviewing, whilst companies 7, 11 and 14 were the focus of re-interviews. Re-interviewing some of the original contributors is a technique available to grounded theory studies and is supported by [22] who states that during theory development, ‘the interpretation should be presented to the original informants to ensure that it is an honest representation of participant accounts’. Building on the need for diversity within the data, the companies in Stage 2 came from different business sectors than those in Stage 1.

During the Stage 2 fieldwork, there was still some time devoted to capturing company demographic data but there was now a clear focus based on the Stage 1 outputs and maximum effort was made to ensure the categories and sub-categories were fully ‘saturated’. The combination of theoretical sampling and constant comparison ensures that an appropriately wide range of individuals and companies are interviewed which culminates in the core categories being saturated. Throughout the Stage 2 process, however, coding centred on the emerging categories and axial coding progressed to selective coding whereby the conceptual themes and the categories were developed further to the point of saturation. During Stage 2, full category saturation was reached after an additional 9 interviews as, in line with Goulding’s [23] assertion, similar incidences within the data were now occurring repeatedly.

4.4. The emergent categories

Where axial coding’s role is to identify the categories into which the discovered codes and concepts can be placed,

Table 3
Themes, core category and main categories

Theme	Category
<i>Process formation^a</i>	<i>Background of software development manager</i> <i>Background of founder</i> <i>Management style</i> <i>Process tailoring</i> <i>Market requirements</i>
<i>Process evolution</i>	<i>Process erosion</i> <i>Minimum process</i> <i>Business event</i> <i>SPI trigger</i> <i>Employee buy-in to process</i> <i>Hiring expertise</i> <i>Process inertia</i>
<i>Core category</i>	<i>Bureaucracy</i> <i>Documentation</i> <i>Communication</i> <i>Tacit knowledge</i> <i>Creativity</i> <i>Flexibility</i>
<i>Cost of process</i>	

^a From hereon, the themes, categories and core category produced by the study are denoted in italics.

selective coding is used to explain the relationships between the categories to provide the overall theoretical picture. The objective of selective coding is to identify a key category or theme that can be used as the fulcrum of the study results [44]. In this instance, the analysis showed that there was one central category to support and link the two theoretical

themes. Furthermore, as the relationships were developed and populated, new categories emerged that were not explicitly covered by the outcomes generated in Stage 1. The final list of themes, the core category and the main categories identified by the study are shown in Table 3.

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. The ‘saturated’ categories and the various relationships were then combined to form the theoretical framework. The network feature, contained within the Atlas TI suite, allows the researchers to present their findings in graphical or pictorial fashion, thus, for this study, creating a clear image of how the research themes, categories and subcategories are interrelated Fig. 1.

The root node of the framework, *Process Formation*, is a conceptual theme and is a predecessor of its two categories, *Background of Software Development Manager* and *Market Requirements*. The *Background of Software Development Manager* determines the *Process Model* used as the basis for the company’s software development activity and this *Process Model* is then subject to *Process Tailoring*. The *Background of Software Development Manager* coupled with the *Background of Founder* of the company creates an associated *Management Style* and this, in conjunction with the tailored process model, creates the company’s initial *Software Development Process*.

Software Process Evolution occurs as follows. Over time, the *Software Development Process* experiences *Process Erosion*. The key causes of *Process Erosion* are the *Cost of*

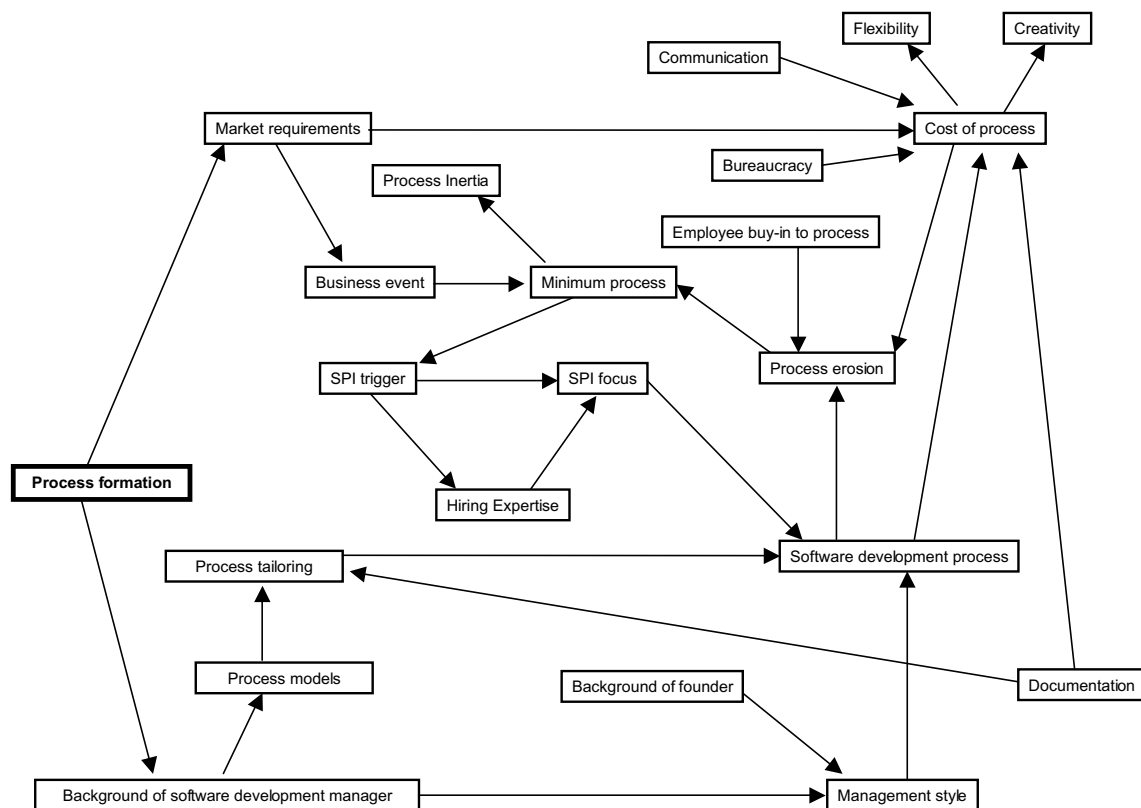


Fig. 1. The theoretical framework.

Process and Employee Buy-in to Process. *Process Erosion* eventually leads to a *Minimum Process*, which is the de facto operational *Software Development Process* until a *Business Event* renders it no longer sufficient. The *Business Event* causes an *SPI Trigger* and where the *SPI* activity is needed is the subject of *SPI Focus*.

Following the *SPI* initiative, a new *Software Development Process* emerges. Soon after *Process Erosion* begins to recur and, as development activities begin to drift back to a *Minimum Process*, some of the gains made during the *SPI* initiative are lost. The organisation then moves into a state of *Process Inertia*, whereby it is apathetic towards any further process change. This continues until another *Business Event* causes the *SPI* cycle to repeat as described above.

4.5. Study findings

On the primary question of what software processes are software companies using, the study has found that all of the companies are *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. These findings on operational context also address RQ4.

One of the key theoretical themes addressed by the research was *Process Formation*, which related to RQ1. The findings show that this depends on several factors including the *Background of the Software Development Manager*, essentially the expertise that manager has accumulated over their working and educational lives, the demands of the market in which the company operates, the founder's *Management Style*, and the organisational culture.

The second key theoretical theme of the study, *Process Evolution*, addresses RQ2 and RQ3. There, evidence from the study data suggests that managers instigate *SPI* as a reaction to trigger events, essentially business occurrences which the current process does not adequately cater for. The *Triggers* for process change can be either positive or negative. The field data shows that many of the companies feel they do not have the capability to deal with the change from within their own resources and, therefore, hire an individual externally who has the necessary expertise to deal with the *Business Event*. However, companies experience difficulty in institutionalising any *SPI* gains and subsequent retrenchment reflects a clear *Erosion* from the process in place immediately following the *SPI* initiative. This *Erosion* eventually resolves to a *Minimum Process* which is 'barely sufficient' to satisfy the organisation's business objectives. The periods between *SPI* initiatives witness *Process Inertia*, wherein the existing process is capable of satisfying all of the business demands that arise. The *SPI* cycle only restarts when the appropriate *Business Event* triggers the necessity for change.

The other primary 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 CMMI and ISO 9001, and the development methodology XP. 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 CMMI or ISO 9000 they were against introducing it to their new organisations. Overall, respondents felt that the resources required to implement the commercial models far exceeded the benefits that may accrue.

4.5.1. Generalisability

Strauss and Corbin contend that the use of a theory-building methodology is to build theory and, therefore, in grounded theory studies, the researcher is talking more about explanatory power than generalisability [44]. In this case context is always relevant to any grounded theory study whereas generalisability describes a situation that is essentially context-free. The findings from this research are context-dependent and this is reflected in the categories and therefore it is not proposed that the findings are generalisable beyond the defined study boundaries.

Within this study, the research covered concepts and their relationships and explored the conditions under which events, happenings, actions and interactions could occur and the ensuing consequences. The study also examined dimensional variation and provided explanation. Therefore, it can be stated that if the developed concepts are sufficiently abstract, they are likely to occur in similar or slightly different form in other software product companies. Yin [46] describes this approach as 'analytic generalisation' where the generalisation is of theoretical concepts and patterns. This is distinguished from the more typical 'statistical generalisation' whereby an inference is made about a population based on data collected from a sample. In this research the outcome of the 'analytic generalisation' process has resulted in a general conceptualisation of the technological, human and organisational factors linked with implementing software process and process improvement programmes. This outcome has implications for both practice and research and contributes to our knowledge of *SPI*.

4.5.2. Limitation of the study

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. Unlike quantitative studies, where independent laboratory conditions may prevail, grounded theory relies on opinion. However, this opinion is the respondent's view or perception of what is taking place, which of course may be at odds with reality. In many instances there may be no supporting evidence to verify the opinion expressed. In addition, it is possible, that the participants may report what they believe the researchers wishes to hear. This may be particularly true of smaller companies who are reluctant to admit that they are not following received best practice, as this is not something they wish to make public. Like companies who may not wish to publish negative results, for fear that it presents the organisation in a bad light, participants may be tempted to do likewise in qualitative interview-based studies, in order to be seen in a favourable light by the interviewer, or to boost the status of the company. The outcome of all of this is that researchers must accept the veracity of what respondents say during the study interviews [24].

Notwithstanding the issues surrounding semi-structured interviews, the opinions of the participants are vital. In this study, 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. It is these actions and interactions, arising from the participants opinions, beliefs, and perceptions, which are essential to a grounded theory study.

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. In most cases the managers interviewed are one or more steps removed from those who are carrying out many of the process steps promoted or defined by them or the organisation. However, whilst a study gathering data purely from the engineers' perspective might generate a different outcome, it would lack the crucial, over-arching 'big picture' view that senior managers can provide. Similarly, it is generally the senior managers who have decision-making responsibility for such as, process model adopted, hiring, product road maps and target market. This knowledge of corporate events would typically be far beyond what engineers could provide from their lower position in the organisational hierarchy and, therefore, a study of process in practice which focused exclusively on engineers would be seriously deficient in depth and breadth of organisational approaches.

5. Evaluation

How a grounded theory is presented offers a number of challenges to the researcher in terms of structure, level of detail included, and how the data are portrayed to display evidence for the emergent categories. Strauss and Corbin's criteria for evaluating a grounded theory include assessing the theory itself, assessing the adequacy of the research process, and determining if the theory is sufficiently well grounded [45]. The next sections will look at each of these in turn.

5.1. Assessing the theory

The following four factors are suggested by Strauss and Corbin [45] to assess a grounded theory:

- Fit – The theory must fit the substantive area and correspond to the data.
- Understanding – The theory makes sense to practitioners in the study area.
- Generality – The theory must be sufficiently abstract to be a general guide without losing its relevance.
- Control – The theory acts as a general guide and enables the person to fully understand the situation.

In terms of *fitness*, there is always a danger that researchers develop a theory, of the studied phenomena, that embodies their own ideals and perceptions as well as popular views and common myths. When these theories subsequently do not fit the developed categories very well, the consequences are often a forcing of the data to do so, and rejection of the data that do not fit or cannot be forced to do so. Therefore it is imperative that, for a grounded theory to fit, it is induced from the diverse set of collected data. In this way it is closely related to the actual realities of the substantive areas and applicable to dealing with them. The theory developed in this study has faithfully adhered to the inductive methods contained in the grounded theory methodology. Though the researchers are "cultural insiders", their professional expertise was used merely to assist theoretical sensitivity rather than drive the theoretical conclusions. The constant comparative method, overturning of some early categories as new data came to light, generation and testing of interim hypotheses, and constant re-evaluation of the interview transcripts ensured that researcher bias was minimised and theoretical fit maintained. Furthermore, towards the end of Study Stage 2, less time was spent exploring issues which did not directly relate to the hypotheses and greater effort was made to ensure the categories and subcategories were fully 'saturated', i.e. no new information about that category was revealed through further coding from additional interviews.

A theory that closely represents the realities of an area will make sense and be *understandable* to practitioners in that area. This understanding is important in that it encourages the theory's usage, increases awareness of the issues faced,

and provides a mechanism for instigating change. In developing the grounded theory in this study, the concepts and categories were carefully developed to support understanding by software development personnel. Where appropriate, in-vivo codes were used. In vivo codes have an important role to play as they are the actual words or phrases used by the practitioners and thus reflect their reality as they perceive it. Using in vivo codes ensured that the developed theory closely corresponded to the realities of software process in practice. Also, in Study Stage 2, some of the Stage 1 participants were re-interviewed in light of the Stage 1 findings. The developing theory was presented to the re-interviewed participants and the new interviewees who had not been included in Stage 1. To prevent potential response bias, the theory was only presented to the interviewees after the interviews had been conducted. The reactions of the interviewees to the presented theory was very positive and one which they believed represented their reality as they perceived it.

From a *generality* perspective the researcher must ensure that the categories contained within the theory should not be so abstract as to lose their sensitising characteristics, but yet should be sufficiently abstract to make the theory a general guide to constantly changing situations. The issue of generalisation in relation to this study will be discussed in detail in Section 4.5.1.

Glaser and Strauss [20] argue that, ‘*a theory with controllable concepts of sufficient generality, that fits and is understandable, gives anyone who wishes to apply these concepts to bring about change a controllable theoretical foothold in diverse situations*’. In summary, the theory should ensure the person who uses it has enough control in the situations they encounter to make the application of the theory worth considering. The theory should allow the person to be able to understand and analyse situations, be able to predict change and its consequences, and be capable of revising his actions, or the theory itself, if appropriate. To enable this, the theory must provide a sufficient number of categories and concepts and explain the relationships between them. The theory in this study has achieved this by providing a comprehensive set of categories with detailed interrelationships to explain how process is formed and the reasons for change. Using both the methodological tools, and those provided by the supporting software, each category and the strength of relationships between them has been fully explored and tested. Hypotheses, derived from, and related to, the controllable situations which face software practitioners, have also been tested. Deviant cases have been sought to ensure theory robustness and applicability. Through these approaches, and the investigation of the SPI literature, a comprehensive theory was developed which can be considered by practitioners faced with situations demanding an SPI solution.

5.2. Adequacy of the research process

In judging the quality of any research study designed to generate theory, the reviewer should be able to make judge-

ments about the research process [44]. As readers are not actually present during the research activity, they must be provided with information to allow them to assess its adequacy. This information relates to how the original sample was selected, how the categories and core category emerged and subsequently drove the sampling process and how were any hypotheses were treated during the analysis activity.

The selection of the original sample was covered in detail in Section 4.1 and showed how a Preliminary Study provided a base on which the research could proceed. Category development continued throughout the research. The incidents and actions, that pointed to the categories, emerged during the interview analysis. Strauss and Corbin suggest that to support the identification of categories the researcher should look for phrases such as ‘because’ or ‘since’. Then, to find the consequences, you follow up on such terms as ‘as a result of’ and ‘because of’. There are numerous examples of these phrases in the field data, particularly so in the case of the *Business Event* and *SPI Trigger* categories.

It was clear from the very early interviews, particularly in the start-up companies, that *Background of Software Development Manager* was central to the initial process that a software company used. This drove an early line of questioning as the manager’s background was clearly being used to set-up the initial development process. Later on in the study however, when larger companies were interviewed, it also emerged that *Hiring Expertise* was a key solution to process difficulties, with many of the study participants themselves being exemplars of this category.

The selection of the core category, *Cost of Process*, was made during Stage 2 of the study, though attributes of it had been apparent in Stage 1. In selecting the core category, the researchers closely followed the steps recommended by Strauss and Corbin [44], including the fact that all other categories must relate to it and that it appears frequently in the data. Analysis of the Stage 1 data showed that companies were concerned with *Documentation*, and the cost of generating and maintaining it, a factor also highlighted by [29]. In addition managers were anxious about issues around the ‘weight’ of process, the need to retain *Creativity/Flexibility*, and the basic lack of resources to implement SPI. Whilst many of these were contenders as core categories in their own right, it was the additional analysis from Stage 2 that demonstrated that SPI was being avoided for reasons of ‘Cost’ and that the companies’ concerns were expressions, or dimensions, of Cost. Strauss and Corbin suggest that considerable manipulation of the data is required before a core category emerges. The fact, therefore, that it did not crystallise until Stage 2 provided reassurance to the researchers that the correct category had been identified.

5.3. Grounding the findings

Strauss and Corbin also provide a list of criteria to assist in determining how well the findings are grounded. These are [44]:

- Are concepts generated and are the concepts systematically related?
- Are there many conceptual linkages and are the categories well developed?
- Is variation built into the theory and are the conditions under which variation can be found built into the study and explained?
- Has [the theory generation] process been taken into account?
- Do the theoretical findings seem significant and does the theory stand the test of time?

The foundations of any theory are a set of concepts grounded in the data. Table 4 shows an example of some of the codes produced from the coding processes and includes both terms used by the practitioners, and conceptual codes assigned by the researchers, where many of the researchers-assigned codes denote concepts generated from the analysis of the data. Through the use of network diagrams (for high-level example see Fig. 1) we established the linkages and relationships between concepts, which categories act as predecessors and successors within the theory, and how the categories link to the core category and research themes.

Strauss and Corbin suggest that variation is important because it signifies that a concept has been examined under a range of different conditions and dimensions. Though this research is concerned with indigenous Irish software product companies, we have endeavoured to incorporate the views of as wide a range of practitioners as possible. Furthermore, Stage 2 of the study expanded the range of interview participants to achieve coverage of a greater range of markets, and thus reduced the prospects of phenomena relating only to specific market domains, or company size.

The process of data collection and analysis, is important because it enables theory users to explain action under changing conditions. Whilst the interviews represent a snapshot of the period in time in which they were conducted, much of the focus of questioning related to the conditions prevalent in the company, how these changed, and what circumstances or events gave rise to these changes. Much was made of how things used to be in the organisation concerned, how things were at the time of the interview, and the evolutionary path that was followed to arrive at that juncture.

Strauss and Corbin argue that a researcher could merely go through the motions and arrive at findings which are mundane and insignificant. It is the researchers belief that the study findings are significant and add to the literature on SPI. The significance of the findings and their implications for practice and theory will be discussed in the next section.

6. Conclusions

This section will address the conclusion from the SPI study itself, some comments on the usage of grounded theory and explore some future research directions.

6.1. The SPI study

The SPI study makes several key contributions. By careful and comprehensive comparison, analysis, and abstraction of interviews with 21 software product companies, we provide a grounded understanding of the practice of software process; explain the factors that influence the way process is established and evolves in software companies; and describe the reasoning behind why software companies largely ignore commercial best practice software process and process improvement models.

There is an absence of published material describing how process is initially formed in software product companies. This research provides a new contribution in this area, using evidence from practice, a theory has been generated which explains the factors which influence the first software process a company will use. This study also contributes to knowledge and understanding of the domain of process change and process improvement. Unlike much of the literature, which discusses how to implement SPI, this study demonstrates why SPI is undertaken. Understanding the reasons for SPI, and the interrelationships between the key associated variables, provides vital knowledge and information to the field.

By deployed a qualitative methodology, more associated with the social sciences, in a primarily scientific field, the use of grounded theory in this way has culminated in empirically valid theory and has the capacity to provide encouragement to other researchers to bring alternative methodologies to bear on aspects of software development.

6.2. Using grounded theory in software process research

Software engineering is a highly social activity. In attempting understand this activity, the researchers may use many different methods in order to describe, explore and understand such social activity. These methods can be subdivided into two broad categories: Quantitative methods, which are concerned with quantifying social phenomena through the collection and analysis of numerical data; and Qualitative methods, which emphasise personal experiences and interpretation, and are more concerned with understanding the meaning of social phenomena and focus on links among a larger number of attributes across relatively few cases.

In attempting to study human behaviour and the social contexts in which it functions, researchers are directed towards qualitative techniques. Evidence for this has been noted by [36] ‘*Social research involved the interaction between ideas and evidence, where ideas help researchers make sense of evidence, and researchers use evidence to extend and test ideas*’. Accordingly social research thus attempts to create or validate theories through data collection and data analysis, and its goal is exploration, description and explanation. Similarly, qualitative researchers have an advantage over their quantitative counterparts in that they can continually add to, and interpret, the research puzzle, whilst still gathering data [11].

In seeking an appropriate methodology to investigate the software process aspects of software engineering we have selected grounded theory as being a suitable candidate and describe the successful implementation of grounded theory in a study of SPI in indigenous Irish software product companies. The grounded theory approach is inductive, pragmatic and highly concrete methodology [33]. Using grounded theory in the software engineering context, the researchers task is to generate theory from holistic data gathered through naturalistic inquiry, to understand the interaction between software engineers and their environment and the impacts, consequences and outcomes of these interactions.

However, a note of caution should be recorded on the use of grounded theory in research work in SPI. We would argue that the methodology has much to offer such research, the nature of this type of study means that the prior experience of the researcher can significantly influence methodological success. Bringing grounded theory's 'unconventional' approach to the area of SPI has the potential to provide major challenges to the novice researcher. We believe that, to succeed, a grounded theory researcher should be both experienced in conducting detailed research studies and a 'cultural insider'. The absence of these credentials could prove fatal for novice researchers, who may wish to use grounded theory in software process studies, and suggests that an alternative methodology should be considered.

6.3. Future work

One of the major contributions of this work is a grounded theory explaining how software process is initially established in a software start-up. The literature lacks a comprehensive investigation of software process initiation and usage in beginning software product companies. The opportunity arises therefore for other researchers to explore this area to determine support for, or a challenge to, the generated theory.

This research is concerned with how software process is practiced in indigenous Irish software product companies. A study which concentrated on the practices used by indigenous software product companies in other countries in Europe and beyond, 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. However, much software is developed outside the software product company domain. There is a wide spectrum of organisations whose business ranges from bespoke software solutions to the in-house software departments of non-software companies. These developers also use software processes and a study of how these are formed, evolve and improve, in this non-software product company environment, could be counter-balanced against this work.

References

- [1] D. Avison, F. Lau, M. Myers, P. Nielsen, Action research, in: Communications of the ACM, January, vol. 42, No. 1, 1999, pp. 94–97.
- [2] R. Baskerville, J. Pries-Heje, Grounded action research: a method for understanding IT in practice, in: Accounting, Management and Information Technologies, No. 9, 1999, pp. 1–23.
- [3] O.W. Bertelsen, Towards a unified field of SE research and practice, in: IEEE Software, November/December, 1997, pp. 87–88.
- [4] L. Blaxter, C. Hughes, M. Tight, How to Research, second ed., Open University Press, 2001.
- [5] A. Bryman, Social Research Methods, Oxford University Press, Oxford, 2001.
- [6] C. Buchman, Software process improvement at alliedsignal aerospace, in: Proceedings of the 29th Annual Hawaiian International Conference on System Sciences, vol.1, Software Technology and Architecture, 1996, pp. 673–680.
- [7] R.B. Burns, Introduction to Research Methods, fourth ed., Sage Publications, Beverly Hills, CA, 2000.
- [8] G. Burrell, G. Morgan, Sociological Paradigms and Organisational Analysis, Heinemann, London, 1979.
- [9] L. Carvalho, L. Scott, R. Jeffery, An exploratory study into the use of qualitative research methods in descriptive process modelling, in: Information and Software Technology, No. 47, 2005, pp. 113–127.
- [10] J. Carver, V. Basili, Identifying implicit process variables to support future empirical work, in: Journal of the Brazilian Computer Society, October–December, 2003.
- [11] K. Charmaz, Constructing Grounded Theory: A Practical Guide Through Qualitative Analysis, Sage Publications, Beverly Hills, CA, 2006.
- [12] M.B. Chrissis, M. Konrad, S. Shrum, CMMI: Guidelines for Process Integration and Product Improvement, Addison Wesley, Reading, MA, 2003.
- [13] J.W. Creswell, Research Design: Qualitative, Quantitative, and Mixed Methods Approaches, second ed., Sage, Beverly Hills, CA, 2003.
- [14] M.K. Daskalantonakis, Achieving higher SEI levels, in: IEEE Software, July, 1994, pp. 17–24.
- [15] R. Dion, Process improvement and the corporate balance sheet, in: IEEE Software, July, 1993, pp. 28–35.
- [16] K. ElEmam, J.N. Drouin, W. Melo, SPICE: The Theory and Practice of Software Process Improvement and Capability Determination, Wiley, London, 1998.
- [17] W.A. Firestone, Meaning in method: the rhetoric of quantitative and qualitative research, Educational Researcher 16 (7) (1987) 16–21.
- [18] B. Fitzgerald, 1998, An empirical investigation into the adoption of systems development methodologies, in: Information and Management, vol. 34, 1998, pp. 317–328.
- [19] B. Glaser, Basics of Grounded Theory Analysis: Emergence Vs Forcing, Sociology Press, Mill Valley, CA, 1992.
- [20] B. Glaser, A. Strauss, The Discovery of Grounded Theory: Strategies for Qualitative Research, Chicago, Aldine, 1967.
- [21] R. Goede, C. De Villiers, The applicability of grounded theory as research methodology in studies on the use of methodologies in IS practices, in: Proceedings of SAICSIT, 2003, pp. 208–217.
- [22] C. Goulding, Grounded theory: some reflections on paradigm, procedures and misconceptions, Technical Working Paper, University of Wolverhampton, UK, 1999.
- [23] C. Goulding, Grounded Theory: A Practical Guide for Management, Business and Market Researchers, Springer, Berlin, 2002.
- [24] B. Hansen, K. Kautz, Grounded theory applied – studying information systems development methodologies in practice, in: Proceedings of 38th Annual Hawaiian International Conference on Systems Sciences, Big Island, HI, 2005.
- [25] A. Hevner, S. March, The information systems research cycle, in: IEEE Computer, November, 2003, pp. 111–113.
- [26] J. Herbsleb, D. Zubrow, D. Goldenson, W. Hayes, M. Paulk, Software quality and the capability maturity model, in: Communications of the ACM, vol. 40, No. 6, 1997, pp. 30–40.
- [27] W.S. Humphrey, T., Snyder, R. Willis, Software process improvement at Hughes aircraft, in: IEEE Software, July, 1991, pp. 11–23.
- [28] A.S. Lee, J. Liebenau, Information systems and qualitative research, in: A. Lee, J. Liebenau, J.I. DeGross (Eds.), Proceedings of Informa-

- tion Systems and Qualitative Research, Kluwer Academic, Boston, MA, 1997.
- [29] T.C. Lethbridge, J. Singer, A. Forward, How software engineers use documentation: the state of the practice, in: *IEEE Software*, November/December, 2003, pp. 35–39.
- [30] T. Muhr, *Atlas TI User's Manual*, Scientific Software Development, Berlin, 1997.
- [31] M.D. Myers, 1997, Qualitative research in information systems, in: *Management Information Systems Quarterly*, vol. 21, No. 2, June, 1997, pp. 241–242.
- [32] W. Orlikowski, CASE tools as organizational change: investigating incremental and radical changes in systems development, in: *Management Information Systems Quarterly*, vol. 17, No. 3, 1993, pp. 309–340.
- [33] M. Patton Quinn, 'How to use Qualitative Methods in Evaluation', Sage, Beverly Hills, CA, 1987.
- [34] N. Power, A grounded theory of requirements documentation in the practice of software development, PhD Thesis, Dublin City University, Ireland, 2002.
- [35] S. Qureshi, M. Liu, and D. Vogel, A grounded theory analysis of e-collaboration effects for distributed project management, in: *Proceedings of 38th Annual Hawaiian International Conference on Systems Sciences*, Big Island, HI, 2005.
- [36] C. Ragin, *Constructing Social Research*, Sage, Beverly Hills, CA, 1994.
- [37] C.S. Reichardt, T.D. Cook, Beyond qualitative versus quantitative methods, in: T.D. Cook, C.S. Reichardt (Eds.), *Qualitative and Quantitative Methods in Evaluation Research*, Sage, Beverly Hills, 1979, pp. 7–32.
- [38] S. Sarker, F. Lau, S. Sahay, Using an adapted grounded theory approach for inductive theory building about virtual team development, in: *The Data Base for Advances in Information Systems*, vol. 32, No. 1, 2001, pp. 38–56.
- [39] M.N.K. Saunders, P. Lewis, A. Thornhill, *Research Methods for Business Students*, Pitman, London, 1996.
- [40] R.S. Schreiber, The 'How To' of grounded theory: avoiding the pitfalls, in: R.S. Schreiber, P. Noerager Stern (Eds.), *Using Grounded Theory in Nursing*, Springer, Berlin, 2001.
- [41] C. Seaman, V. Basili, An empirical study of communication in code inspections, in: *Proceedings of the 19th International Conference on Software Engineering*, May, Boston, MA, 1997, pp. 17–23.
- [42] C. Seaman, Qualitative methods in empirical studies of software engineering, *IEEE Transactions on Software Engineering*, vol. 25, No. 4, 1999.
- [43] L. Silva, J. Backhouse, Becoming part of the furniture: the institutionalisation of information systems, in: A. Lee, J. Liebenau, J.I. DeGross (Eds.), *Proceedings of Information Systems and Qualitative Research*, Chapman and Hall, London, 1997.
- [44] A. Strauss, J.M. Corbin, *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*, second ed., Springer, Berlin, 1998.
- [45] A. Strauss, J.M. Corbin, *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*, first ed., Springer, Berlin, 1990.
- [46] R.K. Yin, *Case Study Research: Design and Methods*, second ed., Sage Publications, Beverly Hills, CA, 1994.