Understanding the gap between software process practices and actual practice in very small companies

Mary-Luz Sánchez-Gordón, **Rory V. O'Connor**

Abstract

This paper reports on a grounded theory to study into software developers' use of software development process in actual practice in the specific context of very small companies. This study was conducted in three very small software product companies located in Ecuador. The data collection was based on semi-structured qualitative interviews with software project managers, focus group with software developers and was supplemented by literature and documents studies. We interviewed two types of participants (managers and developers), so as to ensure that we elicited a holistic perspective of how they approached the software development process in actual practice. The goal was to study what practices are actually used and their opinion and attitude towards the potential adopting of an international standard (ISO/IEC 29110) specifically designed for very small companies. With the collected data we performed an analysis utilizing Grounded Theory coding techniques, as this methodology promotes the focus on uncovering the real concerns of the participants. This study highlighted three areas of concern: customer, software product and development tasks coordination and tracking. The findings in this study give an insight towards the work products as they relate to software development process practices in very small companies and the important factors that must be considered to assist project success.

Keywords Grounded Theory • Software Process • Software Process Improvement • Very Small Entity • SPI • VSE

1   Introduction

At a time when technology advances almost daily, software development companies are under increasing pressure to improve productivity while maintaining quality and keeping costs to a minimum. The United Nations Information Economy Report (UNCTAD 2012) noted that to facilitate structural transformation and technological advancement, it is necessary for countries to build domestic capabilities to allow firms and organizations to engage in understanding and improving their processes.

All software companies are not the same and vary according to factors including size, market sector, time in business, management style, product range and geographical location. For example, a software company operating in India may have a completely different set of operational problems to contend with to a software company in Israel or Ireland. Even within a single geographical area such as Ireland, the range of operational issues faced by a small local Irish-owned firm can be radically different to those affecting a multinational subsidiary. The fact that all companies are not the same, raises important questions for those who develop software process and process improvement models. For example, as noted by (Laporte et al. 2008), the software industry is dominated by very small enterprises. Although an international classification exists for computer software and services, little international official data is available outside Europe and North America. In Europe, Eurostat uses the General Industrial Classification of Economic Activities within the European Communities (NACE Rev.2) that identifies

M. Sánchez-Gordón
Universidad Carlos III de Madrid
Madrid, España

R. V. O'Connor ()
Dublin City University,
Dublin, Ireland
e-mail: roconnor@computing.dcu.ie

computer software and related computer services as a subcategory (division 62: computer programming, consultancy and related activities; division 63: information service activities). In 2010, according to Eurostat ("Eurostat" 2014) 98.8% of enterprises in this sector were Small enterprises (< 50 employees). Micro enterprises (<20 employees) make up at least 97%, they employed more than 30% of people and made up 24% of turnover. The term "very small entity" had been defined by the ISO/IEC JTC1/SC7 Working Group (WG) 24 and subsequently adopted for use in the new ISO/IEC 29110 software process lifecycle standard as being "an entity (enterprise, organization, department or project) having up to 25 people" (ISO/IEC 2011).

It has been noted that there are significant differences between very small companies with 1 to 9 employees and small companies with 10 to 50 employees (Hofer 2002). Accordingly (McFall et al. 2003), a study of companies based in Northern Ireland showed that the priorities and concerns for organizations with fewer than 20 employees are different from those of larger organizations. This viewpoint is further supported by studies in Australia (Staples et al. 2007) and a worldwide study of VSEs on behalf of the ISO (Laporte et al. 2008).

In spite of their importance, it has been observed that one of the first challenges for small organizations is that their primary business objective is to survive (Paulk 1998; Coleman and O'Connor 2008b) because their resources are scarce. It has also been recognized that many VSEs can't afford the resources - in number of employees, expertise, cost, and time - or see a net benefit in establishing software life-cycle processes. There is sometimes a disconnect between the short-term vision of the organization, looking at what will keep it in business for another six months or so, and the long-term benefits of gradually improving the ways the company can manage its software development and maintenance (Johnson 2006; Laporte et al. 2008). Accordingly, the implementation of controls and structures to properly manage their software development activity is necessary and constitutes a major challenge (McCaffery and Coleman 2009). A software process improvement (SPI) initiative, properly managed and administered, can assist software companies in this regard. SPI aims to understand the software process as it is used within an organization and thus drive the implementation of changes to that process to achieve specific goals such as achieving higher product quality or reducing costs (Coleman and O'Connor 2008a) but the explicit definition of processes plays a key role in the major SPI initiatives (Ruiz-Rube et al. 2015). Prior to starting process improvement programs, it is imperative to have an understanding of current software development practices throughout the industry. Therefore, it is essential to understand the actual process in use in this kind of organization to put SPI in a proper perspective.

Considering the large percentage of small and very small software organizations across the world, relatively very few studies have focused on actual process that they are using in everyday software development work, and few of them has been specifically focused on very small companies. Therefore, little is known about this field (Moreno-Campos et al. 2014). In this paper, we address this gap based on viewpoints of the software project manager and software developer, in very small companies, as defined by ISO/IEC29110, which is organizations or companies with less than 25 people. Consequently, the results of this work provide an interesting insight into the state of the practice of software process and SPI in small software organizations.

This paper is organized as follows: Section 2 introduced the research background and related work. Section 3 provides an overview of research methodology. Section 4 describes how it was applied in this study, and section 5 presents the study's main findings. Finally, section 6 beside with summarizes the key findings and contributions from the research.

2   Background and related work

A software process has been defined as "*A set of activities, methods, practices and transformations that people use to develop and maintain software and the associated products (e.g. project plans, design documents, code, test cases and user manuals)*" (Zahran 1998). However, a process is not a rigid prescription for how to build computer

software. Rather, it is an adaptable approach that enables the people doing the work (the software team) to pick and choose the appropriate set of work actions and tasks (Pressman 2009). Accordingly the intent is always to deliver software in a timely manner and with sufficient quality to satisfy those who have sponsored its creation and those who will use it. Therefore, all companies follow a software process, either explicitly or implicitly.

To simplify understanding and to create a generic framework, which can be adapted by organizations, software processes are represented in an abstract form as software process models. Indeed, over the past decades, a number of different process models have been designed to help companies manage their software development activity (Mora et al. 2009) and a number of different approaches to software process assessment and improvement have been proposed. Likewise, disruptive factors such as gamification have been studied (Herranz et al. 2014). However, no single approach has achieved a generalized acceptance, which is not surprising, as there are a multitude of other contextual and situational factors that influence the choice of process and process management decisions (Clarke and O'Connor 2012a). Furthermore there have been attempts to develop a mechanism for relating process decisions and industrial contexts (Jeners et al. 2013). Furthermore the unique factors affecting VSEs distinguish them, from SMEs (Laporte and O'Connor 2014a, 2014b).

2.1   Software Process in small organizations

Despite the growing interest in this subject, no clear agreement has been achieved on how to distinguish between very small, small and medium-sized organizations. Moreover, as outlined in (Boehm and Turner 2003) the approach to software development can be view on a spectrum ranging from plan-drive approaches such as CMMI, to agile methods which as Scrum and XP.

Agile methods are based on iterative and incremental development using short development cycles (Boehm and Turner 2003). The most important priority of agile methods is to make the customer satisfied with early and continuous delivery of software. Although agile software development methods have caught the attention of software engineers and researchers worldwide, scientific research still remains quite scarce (Abrahamsson et al. 2010). On the other hand, the traditional software development world, characterized by the engineering and process improvement advocates, has plan-driven methods focus on the quality of the software artifacts and the predictability of their processes (Boehm and Turner 2003). The ISO/IEC 29110 standard (O'Connor and Laporte 2014) specifically address the software lifecycle needs of VSEs. Nevertheless, the companies were using a process model in a "text-book" fashion (Coleman and O'Connor 2008a; Von Wangenheim et al. 2006), choosing instead either to drop elements of their chosen model or, develop something proprietary instead. Likewise, software engineering work practices are chosen opportunistically, adapted and configured to provide value (Yilmaz et al. 2010) under the constrains imposed by the startup context (Paternoster et al. 2014).

Additionally, a number of in-vivo studies have been made on diverse aspects of the software process in small organizations such as requirements engineering problems and practice (Khankaew and Riddle 2014), the benefits of using Scrum (Caballero et al. 2011), software inspection (Misra et al. 2014), the practices of self-organizing Agile teams (Hoda et al. 2012), knowledge management in small teams (O'Connor and Basri 2012), factors that motivate software engineering teams (Verner et al. 2014), factors that may influence the adoption of best practice (Cater-Steel 2000), program of process improvement software (Schoeffel and Benitti 2012) and the influence of SPI on business success in software SMEs (Clarke and O'Connor 2012b). These studies use live observation or surveys as their research methodology, so that the findings are based on actual process but they do not address the entire software process.

2.2   Software Process Improvement in small organizations

There is evidence (O'Connor and Coleman 2009) (Staples et al. 2007) that the majority of small and very small software organizations are not adopting existing standards / proven best practice models because they perceive the

standards as being developed by large organizations and orientated towards large organizations, thus provoking the debate the in terms of number of employees, size does actually matter. Studies have shown that small firms' negative perceptions of process model standards are primarily driven by negative views of cost, documentation and bureaucracy. In addition, it has been reported that SMEs find it difficult to relate standards to their business needs and to justify the application of the international standards in their operations. Most SMEs cannot afford the resources for, or see a net benefit in, establishing software processes as defined by current standards (e.g. ISO/IEC 12207) and maturity models (e.g. CMMI).

The low adoption of best practices, as indicated from the previous surveys (Cater-Steel 2000; Laporte et al. 2008; McFall et al. 2003), suggests that process improvement should be a high priority for many development firms. The surveys also established that many firms are interested in improving their processes. Accordingly (Pino et al. 2008), many researchers are focusing their attention on adapting and using SPI models/methods and how to guide and prioritize the SPI efforts in SMEs. This means that often researchers consider small organizations together with medium enterprises, not differentiating their specific characteristics (Richardson and von Wangenheim 2007). Therefore, this can affect research results and their approach.

Furthermore the group (ISO/IEC JCT1/SC7 WG24) responsible for the development of the ISO/IEC 29110 (Laporte et al. 2015) standard as part of its work conducted a worldwide survey to find out more about the small enterprise's needs. It is interesting to note that in terms of certification and recognition, only 18% of very small enterprises (VSEs) are certified and over 74% indicated that it was important to be either recognized or certified. Of that group, ISO certification was requested by 40%; market recognition requested by 28% and only 4% were interested in a national certification (Laporte et al. 2008).

In relation to the most prominent models, new assessment methods tailored to the context of SMEs have been developed, such as, for example, in conformance with ISO/IEC 15504: RAPID (Rapid Assessment for Process Improvement for Software Development), SPINI (Software Process Improvement Initiation), MARES (Método de Avaliação de Processo de Software), and SPIRE (Software Process Improvement in Regions of Europe) among others. With regard to the CMMI, EPA is an example of an ARC class-C compliant method and ADEPT is its expanding. There also are guidelines to harmonize quality frameworks (Baldassarre et al. 2012). Finally, the approach presented in (Taylor et al. 2006) and the Agile Framework for Small Projects (AFSP) (Lee and Yong 2013) are derived from Boehm and Turner's Agility/Discipline assessment.

2.3   Related work

There is much prescriptive advice written about what software engineers should be doing but rarely is any description provided about what they are actually doing (Glass 2003). Although many authors refer to software developers using dominant, prevalent, or common practices, there has been little research to date to document actual use.

The closest related work is about the software process in practice (Coleman and O'Connor 2008a) but it is focused on Irish SMEs, other works deal with how to understand (Basri and O'Connor 2010) and evaluate (O'Connor 2012a) the issues that affect the adoption of software process standards by very small software companies, and knowledge management process practices in these type of companies (Basri and O'Connor 2011a). While this remains important, this study had been carried out in other geographical location considering the whole process to gain more insight about actual software process and the adoption of lightweight software process standards in very small companies. The idea is building a holistic picture of the research situation. It also provides further validity for our proposed research and indicates if the findings are peculiar to their context geographical.

3 Research Methodology

This section describes the research design carried out to address the gaps in the literature discussed in the previous section. It details the specific research questions that were used to guide the research, the theoretical perspective taken and the methodology applied to obtain answers to the questions posed.

The objective of the study is to investigate **what software process practices do VSEs actually use in practice in everyday software development work?** And **what is the opinion /attitude / sentiment / feeling towards the potential benefits of adopting a VSE specific standard such as ISO/IEC 29110?**

The motivation for this research originates in the premise that software industry is dominated by an overwhelming number of very small software development organizations which face a major challenge to achieve targets relating to delivering quality products on time, and within budget. It is therefore necessary to full understand the current status of use of practices and techniques in VSEs, prior to determining actions to improve the situation.

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.

3.1 Research approach

This study involves an empirical investigation that attempts to understand the software development process through the participants' interpretation of their context. The central role of the people performing software development tasks has the behavior of people as its integral part (Yilmaz et al. 2015). Therefore, the need for a deep understanding of the software development practices, problems, and needs in VSEs calls for a qualitative research approach (Basri and O'Connor 2010). The objective of the present study is more focused on creating a detailed description rather than creating a theory, accordingly a pure Grounded Theory (GT) method is not applicable but only GT coding process will be used in order to assist researcher in analyzing present study data (Hoda et al. 2012; O'Connor 2012b).

As depicted in Fig. 1, this study has two phases. In the first phase, all of the data collection processes are completed where three main methods in data collection process that have been used; individual and focus group interviews and document analysis. Document analysis was based on the artifacts delivered by each organization. For the interview and focus group purposes, researchers have been guide by an interview guide and focus group question guide. The guides are inspired on related work mentioned in section 2.3. In the second phase, GT coding process was used in analysis data. In the following sections, we describe the main components of the research approach as illustrated in Fig. 1.
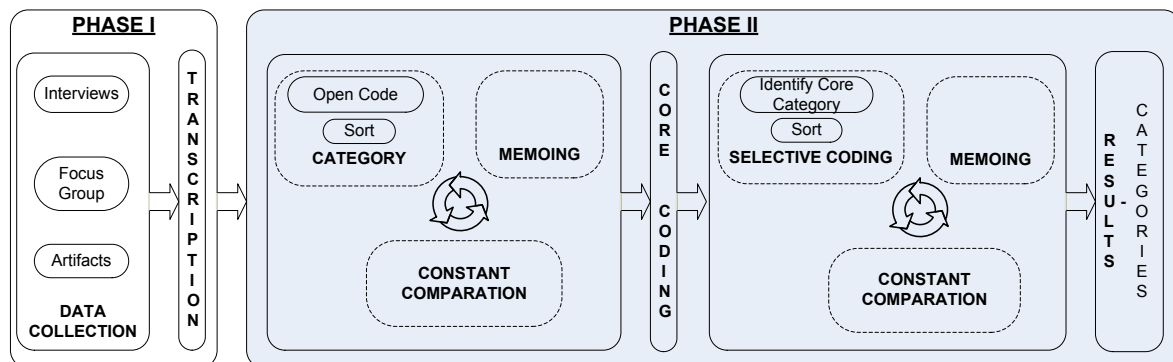
Fig. 1 Overview of the Research approach, data collection and analysis using Grounded Theory

3.1.1 Interviews and Focus Group

Interviews and Focus Groups are a resource-demanding data collection method; activities such as planning, conducting and analyzing are time-consuming by nature. However, interviewing people provides insight into their world; their opinions, thoughts and feelings (Hove and Anda 2005) and therefore we propose these as suitable data collections mechanisms.

In particular focus groups explicitly use dynamic group interaction as part of the method to achieve enhanced data gathering, as it can for example activate details of forgotten experiences and also generate better data through wide range of responses (O'Connor 2012b). This means that instead of the researcher asking each person to respond to a question in turn, people are encouraged to talk to one another: asking questions, exchanging anecdotes and commenting on each other's experiences and points of view (Kontio et al. 2004, 2008; Langford and McDonaugh 2003; O'Connor 2012b). Focus groups are thus carefully planned discussions, designed to obtain personal perceptions of the group members on a defined area of research interest. There are typically between 3 and 12 participants and the discussion is guided and facilitated by a moderator-researcher, who follows a predefined questioning structure so that the discussion stays focused (Kontio et al. 2008). In this study, there were between 2 and 4 interviewees due to the limited number of software developers in each participant organization.

Both methods should be used properly and the sessions should be planned and executed well and with appropriate rigor in order to avoid potential sources for unwanted bias. Interviews and focus group were also recorded in order to prevent loss of information. Thus, the interviewer was able to pay more attention to the subject.

3.1.2 Grounded Theory

This study has essentially employed the (Strauss and Corbin 1998) approach because the researchers have personal and professional experience on software development. It 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 (Coleman and O'Connor 2007). According to (Strauss and Corbin 1998), 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.

It is worth noting that GT is being increasingly used in the software engineering domain and in the study of Software Process in particular (Basri and O'Connor 2011b; Coleman and O'Connor 2008a; Hansen and Kautz 2005; Hoda et al. 2012; Kroeger et al. 2014).

3.2 Data collection

During the fieldwork the semi-structured interview and focus group were performed. In total the study involved 3 interviews and 3 focus groups across the 3 very small companies profiled in Table 1. The following subsections will present these findings in more detail. In keeping with the fundamental tenets of grounded theory, extracts of the interview transcripts will also be presented in support of the findings.

Table 1 Company Profile

Company	Business Sector	No. of Active Projects	No. of Developers	No. of Employees	Years of Operation	International Customers	No. of Participants PM	No. of Participants Developers
1	Enterprise	10	6	17	18	Yes	2	4
2	Financial Services	12	7	10	9	Yes	1	4
3	Enterprise	4	5	7	4.5	Yes	1	2

3.2.1 Recruiting Participants

Recruiting participants is a significant challenge for any research project as they have to spend time on what are often seen as a "non-productive" activity. As one of the authors has firsthand knowledge of the Ecuadorian software industry, potential candidates from this industry were been identified through prior working relationships. In addition, we developed and distributed an e-mail invitation for 30 enterprises of Ecuadorian Association of Software (AESOFT). Despite this effort, the organizations were selected based on availability therefore it was a crucial factor for their selection, which is a common practice (Benbasat et al. 1987).

To ensure participants were fully informed about the implications of their involvement in the research and to comply with the issue ethics, each potential VSE was provided with a research profile. In addition, each person who agreed to participate in the research project as an interviewee was asked to notify via e-mail that confirmed that they had understood the implications of their involvement and that they were willing to participate.

There were three organizations located in Ecuador which had the advantage of restricting to within the same geographic, economic and regulatory regime and thus facilitating direct comparison. The participant organizations are software product companies whose primary business is software development. They have been on the market from 4.5 to 18 years. All of the organizations were classified as VSE (ie. up to 25 people) with the number of software developers varying from 5 to 7 people including two CEO, who are also owners and founders. The organizational sizes varied from 7 to 17 employees.

In order to respect their confidentiality, we refer to our participants by numbers 1 to 3, rather than by name. Table 1 shows participant and their features. Overall, the data collection process took 3 months, which included identifying suitable companies, contacting and confirming potential respondents' process, conducting individual and focus group interviews process.

3.2.2 Artifacts and Document Review

Before beginning the meetings, the artifacts made available by each organization were collected in order to perform a prior analysis about their content and useful, the artifacts are documents. This approach was helpful to acquire first hand knowledge of the organization. This gave us the opportunity to learn about these issues before starting the interviews and allowed us to examine whether descriptions of software development methods existed and to determine which type of documentation was available to the staff. The document studies enabled us to how much support they could get from these materials. It made possible a fluent communication and understanding of respondents. These documents also inspired us during the interview.

Only one of the companies has some processes documented as they had previously used it to get a EFQM Certification (European Foundation for Quality Management), which required such documentation. The studies made it easier to evaluate to which extent such advice and directions were followed and which effect the recommendations had on the final implementation of methods in the projects.

3.2.3 Interviews and Focus Group

Online meetings were selected because the geographical location of the researchers (at the time of the study was conducted) was not in Ecuador. Although these online forms provide many advantages over traditionally conducted meetings (e.g., savings in travelling and venue costs, participants feel more comfortable giving negative or controversial feedback), they also have distinct drawbacks, too, such as the task of the moderator can be much more demanding in online than in face-to-face settings (Kontio et al. 2008).

The data was collected by conducting semi-structured interviews with Software Project Managers. The interviews were between two hours and a half and three hours in duration. We focused on the participants' experiences of working, in particular the challenges faced in their everyday work and the strategies used to overcome them.

Conducting Semi-structured interviews instead of completely structured ones help with emergence of the real concerns of participants rather than forcing a topic that may be viewed as trivial by the participants. As the nature of the interviews had been open, when the conversation moved towards new and interesting areas relevant to the subject, the interviewer pursued and explored the new directions. Keeping this in mind, the focus group was performed with software developers. The sessions lasted one hour and a half and one two hours. Again, this approach was helpful to understanding of respondents based on data collected previously.

All the data was personally collected and analyzed by the same individual in order to preserve consistency in the application of the method. Furthermore, she used the guides developed for the interviews and focus group. As an initial step, two experts revise these guides in order to validate their consistency and relevancy.

Moreover, every meeting was first voice recorded and then transcribed. A complete transcription is very time consuming, but it avoid the loss of data. In this way, it was easier to recall the content clearly and to gain a thorough insight into the all the data material. The transcriptions were used for the coding of data in the subsequent analysis phase.

4   Analysis

Data analysis may begin informally during interviews and continue during transcription, when recurring themes, patterns, and categories become evident (O'Connor 2012b). Coding is the key process in GT (Strauss and Corbin 1998). It is the first step of data analysis and begins in the early stages after the first interviews for data collection. They assert that the coding procedures in GT are neither automatic nor algorithmic - "*we do not at all wish to imply rigid adherence to them*". Therefore, flexibility may be necessary in certain circumstances. There are two types of codes produced as a result of data analysis or coding. This process involves the development of the codes, code-categories and inter-relationship of categories which is based on the GT process and coding strategy (Strauss and Corbin 1998). In order to help us code the interview transcript and linking these codes, the Atlas.ti software tool was used. It also produced the semantic network diagrams, which serve to represent information by intuitively accessible graphic means.

Three coding techniques proposed by GT methodology: open coding, axial coding and selective coding (Strauss and Corbin 1998) have been applied in order to assist researchers in analyzing qualitative data and are explained below in the context of this study.

4.1   Open Coding

Open Coding is the analytic process through which concepts are identified and their properties and dimensions are discovered in data (Strauss and Corbin 1998). The analytical process involves coding strategies which is the process of breaking down interviews and observations into distinct units of meaning which are assigned labels to generate concepts. We used open coding to analyze the interview transcripts in detail, this means the researchers analyze the data line-by-line and allocate codes to the text. The idea is to generate a list of concepts by examining the data. All these open codes were then linked and grouped based on similar issues on the broad categories that represent the unit of analysis. During this first phase of the coding process the researcher is comparing data and continually asking questions about what is not understood. Through the process of constantly questioning that the identification of categories occurs during open coding and the process moving beyond description to conceptualization begins. It is imperative that the researcher combines the open coding process with theoretical memo writings that record emerging categories.

While reading the interview transcripts, we made codes/comments about the statements that were important for the respondent's understanding of their software process development. It was also important to be able to trace

conclusions back to the three companies in order to see at the same time whether the results derived from a single company's data or might apply to several companies. Therefore, the codes were grouped in this way, and this allowed us to relate the individual results to a specific company or connect to software development processes in general.

The Atlas.ti tool allowed a visual representation of codes, which supported us to identify the source of the statements and simultaneously to view all the statements in one place to find relationships between the different respondents' statements.

4.2   Axial Coding

Axial coding is the process of relating categories to their subcategories (Strauss and Corbin 1998) to create code families. The term "axial" derives from the fact that coding occurs around the axis of a category, linking categories at the level of properties and dimensions. This involves documenting category properties and dimensions from the initial coding phase; identifying the conditions, actions and interactions associated with a phenomenon and relating categories to subcategories by the continuation of asking question and making comparisons, the inductive and deductive thinking process of relating subcategories to a category is the main emphasis of the axial coding.

Based on the codes, an iterative process was carried out in order to create a range of categories. The transfer of codes was applied when the codes placed were related to each other or belonged to primary or subordinate categories in such a way that the relationship was indicated by how they were placed in relation to each other. As a result, a visual picture of the many codes grouped was achieved.

4.3   Selective Coding

Selective coding is the process of selecting the core category, systematically relating it to other categories, validating those relationships and filling in categories that need further refinement and development. In this process, the first step is to identify the main or 'core' category related to the collected data. The core category acts as the hub for all other identified categories. Selective coding is not very different from axial coding, but takes place at a higher, more abstract level of analysis. Again, an iterative process was carried out in order to place the list of results into significant themes, which were used to organize the description of the study's results.

5   Results and Discussion

Based on the analysis process, we have identified 3 main related categories that shape the software process in VSE participants companies in the present study. The last category *Quality Standard* is concerned with understanding the issues that affect the adoption of software process standards. Table 2 shows all the main categories and variables that gave an influence to the software development process in very small companies. The details of the main categories are presented below, which grouped and listed out in detail the important variable that was gathered from the analysis of understanding the actual software process development in the study companies.

Table 2 Themes, core categories and categories

Theme	Category
Customer	
	Ensure customer satisfaction
	Understand the customer needs
	Align customer expectations
Product Software	
	Understand Requirements

	Design and Select Architecture
	Build the Software
	Test the Software
	Deploy the Software
	Operate the Software
Development Tasks Coordination and Tracking	
	Coordinate Activities
	Track Progress
	Stop Work
Software Quality Standard	
	Level of Acceptance
	Standard Benefits Awareness
	Barriers towards its adoption

5.1   Customer

To survive, very small companies need to be customer focused. All interviewees declared that their customers are their top priority as is evident in this example quotation from the software developers in Company 3 *"The customer should be happy so there will be more work"*. From there, to *Ensure Customer satisfaction*[1] is the most important concern because this means future opportunities of work with the same customer and/or recommendations with others. Likewise, to *understand the customer needs* is primordial, before beginning to work they hold meetings with them in order to create a proposal.

In addition, the CEO in Company 1 claims: *"We have defined formal documents for medium and large projects but they do not work for small projects. We have a statement of work and initial schedule because we felt that we had many problems at the end due to the expectations that were set very high. In order to reduce expectations and ease acceptance, we state what they want, what they are going to get and what we will not do"*. She is aiming for customer satisfaction by *aligning customer expectations* which can be achieved after understanding customer's expectation and then delivering as per the expectation. The expectation is made into a written agreement between the company and its customer.

5.2   Product

In order to appropriately and accurately address customer need it is important that the development team *understand requirements* of the end software product. This encompasses two activities: capturing and agreeing on what the software product will do based on customer's needs. From there, its priorities and impact are defined. When software developers are involved they feel more confident and relaxed throughout the software process. Software developers in Company 1 provide a good example: *"Being in the requirements elicitation give us knowledge ... when we are not involved we have to review the requirements specification if it is incomplete then we need feedback from the customers."*

As a result, a requirement specification document that meets the customer's needs is created. All of the participating companies learned about its importance when they had difficulties with the customer. In fact, if this document does not exist then it is very difficult to reach the customer satisfaction and finish the project if it is too hard. Therefore, specifications are written in such a way that they can easily be read and reviewed by customer and to allow the customer to provide feedback. This comment, from Company 3, shows why they feel they need more documentation. *""It has been fixing and improving requirements [document]. It has been a complex process of*

---

[1] From hereon, the themes, categories and core category are denoted in italics

having nothing to having a document to support us. It avoids problems as spending time without extra pay when the customers want you to do things that they never said because otherwise it [the product] will not work."

They just *design and select architecture* when they have to add components. They argue that they do not need to do it because they work with a product which is well known. The following, from software developers in Company 3, represents an example: "*We do not have database models because we develop on a product already done. We simply include tables*". Nevertheless, technical documents about interfaces and data migration are written in order to minimize misunderstandings and misconceptions.

To *build the software*, some of the software developers initially learn about aspects of the companies' software process such as common coding standards and by working on other product artifacts such as product manuals, etc. The Project manager in Company 3 put it more concisely: *"Initially, they work on small things like manuals and so on while they understand how it works, we go step by step"*. In addition, they test and verify their units of code based on their background. Likewise, the bug are reported and assigned to developers that can fix it – usually the ones who wrote the piece of code. After the problem is resolved, fixes are re-tested in order to check that fixes did not create problems elsewhere. However, the software developers in Company 2 pointed out: *"I believe that the [software] development is not a really problem for us, we have difficulty with managing our time because the same customers demand such thing [change]"*. They admit that it is not enough to be an expert developer or have support from them. Project manager in Company 3 said: "*Everyone does what he/she needs to do in order to build the functionality*".

To *test the software* is a major issue of concern because they have insufficient time. However, when a project is not small, they carried out two types of test, internal – with their own staff – and external – with the customer –. If there is not enough time for thorough testing they use their experience, along with discussion with project customer, to determine where testing should be focused. The project manager in Company 2 typified this thus *"when the customer insists that they require something urgent, we do not have a chance to do all the tests we had thought to do"*. Therefore, test reports are just written when they think that it is necessary. The two oldest companies (1 and 2) elaborate a test approval by their customers. Additionally, the company 2 with customers in the financial services industry writes a request to install a test version.

The *deploy the software* stage means the software product is available for use. Nevertheless, Company 2 points out: *"I have access to only development environment"* therefore technical staff in the customer site are responsible for installing. On the other hand, some training activities are carried out by software developers. The three companies provide user manuals and the two longest established companies (1 and 2) also provide training materials.

Finally, the *operate the software product* stage requires customer to use the software product in a live. The support is in situ and if it is necessary that software developers do it, too. After this period, the two oldest companies have a help desk. As the project manager in Company 1 reports: *"We managed to build our support department such that we can give a customer service by with a standard [own procedure] that led us to have recommendations from our customers due to our after-sales service"*. They have a software product to take control of the help desk. The purpose of a help desk is usually to troubleshoot incident or provide guidance about products. An incident could be a bug, faults, request new functionality and so on. It sometimes implies a sudden overwhelming and unforeseen event. Although they have a procedure there are customers who claim immediate attention - no matter how minor it may be. Lastly, software developers have a direct and informal communication with customers. The two longest established companies have a change control that includes a change request document. However, this situation consistently exceeded their capacity available to respond. Software developers in Company 2 outlines: *"we have a helpdesk when there is time it [incident] comes around if not, arrives by mail, phone or WhatsApp. It much depends on the customer. Sometimes, they are calling immediately to ask if it was already finished"*.

The last three categories *test, deploy and operate the software* are done if the customer require them.

5.3   Development Tasks Coordination and Tracking

In terms of the ongoing coordinating and tracking of tasks it was found that VSE employ a variety of techniques to manage coordination and tracking of tasks. The software developers in Company 1 explained that: *"As a developer the process that I have used it has always been the same: requirements elicitation, development [we do], testing phase, training phase [we sometimes do], documentation phase and final delivery"*. In consequence, the need to coordinate and control the work requires a complex and difficult degree of formalization of procedures that they do not have. Therefore, they have many problems to complete their projects under time and cost constrains. Software developers in Company 3, the most young company, stated: *"There a lot of projects that have been not yet closed, because when a new one comes we focus all the effort on that one while the other is still there waiting"*.

The *coordinate activities* stage includes initial plans in order to address their work and create a project repository. These plans are not always written – that is especially true for "small projects" - but the deadline is always known by developers. Project manager in Company 3 declared: *"...now I use an Excel [spreadsheet] but not all the time, I know the due date and that's it"*. In fact, developers ask for help in order to meet the deadline when they need it. As mentioned above, the project manager cannot plan for every possibility so he/she sets a soft deadline which will give him/her a crucial grace period to deal with unforeseen problems. Thus, he/she could even have time to recover from unexpected event that happen in others active projects. He/she always negotiate due dates with the customers. Moreover, they only keep a written a meeting record if it will be useful.

The *Track progress* stage is based on the work done. Only the older company has timesheet software to assign tasks and track time, however they point out that the quality of captured data is not particularly good. In the younger company, the project manager actively reviews the status of the developed code in the source control software system in order to assess the current status of the development work at any particular point in time. The last company is the only one directly asking each person their progress every one or two weeks. Software developers in Company 2 stated: *"We lack tracking; we need to know where we are, how we are and so on. For instance, it occurs to me that I need a meeting in order to know I am not going to meet the deadline. I will not deliver the customer something. Nevertheless, I have to know when it is ok to tell the customers that their deadlines are not possible, how to negotiate with them according the progress. This kind of things we are not managing ..."*.

Finally, the *stop work* stage is achieved with the acceptance record that means that the work is completed. After that, the oldest company gathers lessons learned, however their project manager claims: *"it is usually one of the most complicated parts of the project"*. Software Developers in Company 3 outlines: *"Closing processes include making changes, reports and its delivery, the customers are happy when they get what they want and then sign the acceptance record"*.

5.4   Software Quality Standard

The *Level of Acceptance* is low because none of the companies are accredited to or have plans to adopt any particular lightweight software quality standard because they would like to have to take control over their current development process. They argue that the software quality standards are not tailored with the current development process so it is a big challenge. The following three interview extracts describe this situation: *"There is still a lot to do, to document"*; *"Many companies do not adopt the standard because they think that quality issues are cumbersome and will not have a return on investment"*; and *"I think the first step is to have our well-defined process, we probably need to have our own product, and I think the next step is to address the quality issues"*

Standard Benefits Awareness

The analysis has also shown that there is an indicator that small companies are interested and are aware about software process and quality standards. The interviewed companies believe that the potential benefits from having a quality standard, and in particular ISO/IEC 29110, could be a quality product, improving company image, improving work process, creating consistency in development work, making the business more profitable because less time is spent on non-productive work. As one interview subject explained *"I think it [standard] is necessary, let us not beat around the bush, but you have to adapt it to the reality of the company. As I told you, each reality is different ... so I have assumed few things and implemented few things because it is necessary. You cannot live without it."*. This concept was backed up by another who explained *"If you could achieve the standard ... eventually you could decrease the costs because you would have a defined process"*. Finally a further participant remarked that *"The great benefit is a more controlled software development process so take less time to finish ..."*

Barriers towards its adoption

They even believe that they do not need it because they are small and have limited resources in the company. They were not interested in adopting any quality standard due to the cost, time and effort involved. In addition, there are perceived difficulties in implementing a new process that everyone can understand and follow clearly. One company in relation to CMMI explained such barriers as *"These methodologies are still very large for our size. There are still a gap between our human resource and our financial resources".* Another remarked on the effect of people related perceptions as barriers by stating that *"I just tell you the people. People should be involved ... there is always resistance to change"*. Another company explained that *"We made up our own methodology, it was adapted to our reality and it works, we need agility, unfortunately we also need to have formal documentation otherwise the customer relationships are complicated but I cannot overburden"*.

Finally, in order to encourage among small companies the adoption of a software quality standard such as ISO/IEC 29110 the respondents indicated that it requires:

- Minimum overhead of resources (time, people and financial)
- More information about the standards such as guidelines, deployment packages and certification process scheme.
- Papers about case studies of its adoption in terms of time required, workload and lessons learned.
- Expert Assistance.

Although only one project manager is knowledgeable in software quality standards. All of them agreed that ISO/IEC 29110 could be helpful. As a project manager in Company 1, the EFQM certified company, said: *"I think and I am increasingly convinced that many past years with adequate knowledge could be compressed into a tablet ... we have done things differently"*.

6   Conclusions

This paper reports on a grounded theory study of how software developers and software project managers use software development methodologies in practice in the context of very small companies. It has revealed that the VSEs use their own software process to work, which they have adapted to their own particular environment and have achieved a sustainable base. Furthermore, VSE management believes that they can improve using internal informal process alterations, rather than formal SPI programmes. Therefore, very small companies with more than four years on the market that carry out more than one project can identify with what is being stated in this study and with the described constrains of resources, personnel and time.

In this study, the practitioners indicated how they do their work to achieve a software product that ensures customer satisfaction therefore three areas of concern were clearly identified: *customer*, *software product* and their *work*. Regarding to the *customer*, *ensuring customer satisfaction* implies *understanding their needs* and going beyond to *align to their expectations*. In order to satisfy the customer, *understanding requirements* is crucial because it addresses the remaining part of the process: *design and select architecture*, *build the software*, *test the software*, *deploy the software* and *operate the software*. The current development process is brief, informal and very light, only one company has claimed to be based on agile methodologies while the others have an iterative process that sometimes is incremental. Furthermore, software developers carry out the process explained above by the project manager who is responsible to *coordinate activities*, *track progress* and finally *stop work*.

The interviewees admitted that almost all documentation is omitted in small projects. Conversely, only the issues related to customer communications are being documented in the others projects, particularly in the oldest companies. Nevertheless, these findings could be influenced by their national culture. As a result, there is a search for balance between protecting themselves from any disagreements with the customer and required workload. On one hand, the short list of work products include: Requirements Specification, Meeting Record and Acceptance Record. On the other hand the long list includes: Project Plan, Change Request, Progress Status Record, Correction Register, and Manual User. It is worth noting that project managers considered that introducing more process control would result in more documentation and thus bureaucracy in the organization, a finding supported by the literature in respect of SMEs (Coleman and O'Connor 2006).

Although *software quality standards* are not a major concern for very small entities, all interviewed have a sincere interest in improving their way of working as one of the project managers stated *"This is a road [software process improvement] that has no end ... we have to go throughout our lives"*. In fact, same as other previous studies (section 2.3), the current study also found a low *level of acceptance*. Despite lack of knowledge about quality standards, the project managers have *standard benefits awareness* and *barriers towards its adoption*. In spite of this, they agree that ISO/IEC 29110 could be helpful, even though they will not immediately adopt it.

6.1   Implications, limitations and future research

If the companies in this study are broadly representative of the small software product community in Ecuador then the absence of knowledge about the software quality standard like ISO/IEC 29110 amongst practitioners should be a cause of concern amongst its [standards] founders and advocates. The reasons behind that the very small companies studied do not adopt software quality standards is because they do not feel ready to do so and further argue that they do not have time to implement such standards. However, they want to know more about how to improve their way of work step by step with minimum overhead of resources, especially time.

The main contribution of this study is an expanded understanding of actual software process, areas of concern, with a focus on why very small companies are not prepared to adopt or even experiment with lightweight software process standards.

A limitation of this study is related to the reliance on qualitative data, as all data is derived from interviews and focus groups are known to be prone to bias. We attempted to overcome this potential threat to validity was using three strategies. Firstly, the guidelines used were revised by two experts in this area and there were discussions with the research supervisors. Secondly, we achieve a whole perspective in each company of the issues by interviewing their project manager and carrying out focus groups with their software developers (developer, tester, business analyst). Finally, the data derived from their artifacts did not contradict, but rather supported the prior data, thereby strengthening it.

A further limitation is related to data collection issues. There were 3 companies but almost all the staff involved took part that means we have a rounded view. It creates some limitation in the research results. However the

appropriate type of companies and respondents were identified for this research, have produced a valid indicator and the results could accurately characterize the context studied which in turn was in some ways limited by our access to them. The project managers made the final selection of respondents therefore there is the possibility that the project managers intentionally selected respondents that would present a picture of software process development, which they could approve. However, there was no reason to let such political factors play a part in the selection of respondents. The project managers were informed that the study had no intention to examine the organization as such. The purpose was to gather information on how system development takes place in practice and to examine which role methods play in the process. They were also informed about the anonymity of the organization in the report. Thus, there was no real motive why they should wish to present a wrong picture of the situation in their organization.

An issue not addressed directly in this study is the cross-cultural comparison of software process and process improvement in action (Niazi et al. 2010) as different countries who poses different cultures may have a different pre-disposition to varying levels of formality in process implementation (Cater-Steel and Toleman 2006). Therefore there is potential to place the present study in the context of similar studies in other geographical locations (Sanchez-Gordon et al. 2015).

The immediate future research is to expand the number of research participants, which would help to produce more reliable and generalized results. From there, very small companies could get the information and gain awareness in managing their software development process as the first step to address the challenge to deal with the software process improvement.

References

Abrahamsson, P., Oza, N., & Siponen, M. T. (2010). Agile Software Development Methods: A Comparative Review. In T. Dingsøyr, T. Dybå, & N. B. Moe (Eds.), *Agile Software Development* (pp. 31–59). Springer Berlin Heidelberg.

Baldassarre, M. T., Caivano, D., Pino, F. J., Piattini, M., & Visaggio, G. (2012). Harmonization of ISO/IEC 9001:2000 and CMMI-DEV: from a theoretical comparison to a real case application. *Software Quality Journal*, *20*(2), 309–335. doi:10.1007/s11219-011-9154-7

Basri, S., & Connor, R. V. (2011). A study of knowledge management process practices in very small software companies. *American Journal of Economics and Business Administration*, *3*(4), 636–644.

Basri, S., & O'Connor, R. V. (2010). Understanding the perception of very small software companies towards the adoption of process standards. In A. Riel, R. V. O'Connor, S. Tichkiewitch, & R. Messnarz (Eds.), *Systems, Software and Services Process Improvement* (Vol. 99, pp. 153–164). Springer.

Basri, S., & O'Connor, R. V. (2011). Knowledge Management in Software Process Improvement: A Case Study of Very Small Entities. In M. Ramachandran (Ed.), *Knowledge Engineering for Software Development Life Cycles: Support Technologies and Applications*. IGI Global.

Benbasat, I., Goldstein, D., & Mea, M. (1987). The Case Research Strategy in Studies of Information Systems. *MIS Quarterly*, *11*(3), 369–386. doi:10.2307/248684

Boehm, B., & Turner, R. (2003). *Balancing Agility and Discipline: A Guide for the Perplexed*. Boston: Addison-Wesley.

Caballero, E., Calvo-Manzano, J. A., & Feliu, T. S. (2011). Introducing Scrum in a Very Small Enterprise: A Productivity and Quality Analysis. In R. V. O'Connor, J. Pries-Heje, & R. Messnarz (Eds.), *EuroSPI 2011* (pp. 215–224). Roskilde, Denmark: Springer Berlin Heidelberg.

Cater-Steel, A. P. (2000). COTS developers lead best practice adoption. In *Conference on Software Engineering*. (pp. 23–30). Presented at the IEEE Computer Society, Los Alamitos, CA. doi:10.1109/ASWEC.2000.844555

Cater-Steel, A., & Toleman, M. (2006). Exploring national culture in software development practices. *EuroSPI 2006 Industrial Proceedings*, 4–1.

Clarke, P., & O'Connor, R. V. (2012a). The situational factors that affect the software development process: Towards a comprehensive reference framework. *Information and Software Technology*, *54*(5), 433–447. doi:10.1016/j.infsof.2011.12.003

Clarke, P., & O'Connor, R. V. (2012b). The influence of SPI on business success in software SMEs: An empirical study. *Journal of Systems and Software*, *85*(10), 2356–2367.

Coleman, G., & O'Connor, R. (2006). Software Process in Practice: A Grounded Theory of the Irish Software Industry. In I. Richardson, P. Runeson, & R. Messnarz (Eds.), *EuroSPI 2006* (pp. 28–39). Springer-Verlag.

Coleman, G., & O'Connor, R. (2007). Using grounded theory to understand software process improvement: A study of Irish software product companies. *Information and Software Technology*, *49*(6), 654–667. doi:10.1016/j.infsof.2007.02.011

Coleman, G., & O'Connor, R. (2008a). Investigating software process in practice: A grounded theory perspective. *Journal of Systems and Software*, *81*(5), 772–784. doi:10.1016/j.jss.2007.07.027

Coleman, G., & O'Connor, R. V. (2008b). An investigation into software development process formation in software start-ups. *Journal of Enterprise Information Management*, *21*(6), 633–648.

Eurostat. (2014). *Annual enterprise statistics by size class for special aggregates of activities (NACE Rev. 2)*. http://epp.eurostat.ec.europa.eu/statistics_explained/index.php/Information_and_communication_service_statistics_-_NACE_Rev._

Glass, R. L. (2003). The state of the practice of software engineering. *IEEE Software*, *20*(6), 20–21.

Hansen, B. H., & Kautz, K. (2005). Grounded Theory Applied - Studying Information Systems Development Methodologies in Practice. In *38th Annual Hawaii International Conference on System Sciences, 2005. HICSS '05* (p. 264b–264b). Presented at the 38th Annual Hawaii International Conference on System Sciences, 2005. HICSS '05, Hawaii. doi:10.1109/HICSS.2005.289

Herranz, E., Colomo-Palacios, R., de Amescua Seco, A., & Yilmaz, M. (2014). Gamification as a disruptive factor in software process improvement initiatives. *Journal of Universal Computer Science*, *20*(6), 885–906.

Hoda, R., Noble, J., & Marshall, S. (2012). Developing a grounded theory to explain the practices of self-organizing Agile teams. *Empirical Software Engineering*, *17*(6), 609–639. doi:10.1007/s10664-011-9161-0

Hofer, C. (2002). Software development in Austria: results of an empirical study among small and very small enterprises. In *Proceedings of the 28th Euromicro Conference* (pp. 361–366). IEEE.

Hove, S. E., & Anda, B. (2005). Experiences from conducting semi-structured interviews in empirical software engineering research. In *Proceedings of the 11th IEEE International Software Metrics Symposium* (pp. 10–32). IEEE.

ISO/IEC. (2011). *Software engineering – Lifecycle profiles for Very Small Entities (VSEs) Part 5-1-1: Management and engineering guide: Generic profile group: Basic Profile* (No. ISO/IEC TR 29110-5-1-2:2011(E)). Geneva.

Jeners, S., O'Connor, R. V., Clarke, P., Lichter, H., Lepmets, M., & Buglione, L. (2013). Harnessing software development contexts to inform software process selection decisions. *Software Quality Professional*, *16*(1), 35–36.

Johnson, J. (2006). *My Life is Failure: 100 Things You Should Know to be a Better Project Leader*. Boston, MA: Standish Group International Publisher.

Khankaew, S., & Riddle, S. (2014). A review of practice and problems in requirements engineering in small and medium software enterprises in Thailand. In *Empirical Requirements Engineering (EmpiRE), 2014 IEEE Fourth International Workshop* (pp. 1–8). IEEE.

Kontio, J., Bragge, J., & Lehtola, L. (2008). The Focus Group Method as an Empirical Tool in Software Engineering. In F. Shull, J. Singer, & D. I. K. Sjøberg (Eds.), *Guide to advanced empirical software engineering* (pp. 93–116). London: Springer.

Kontio, J., Lehtola, L., & Bragge, J. (2004). Using the focus group method in software engineering: obtaining practitioner and user experiences. In *Proceedings of the 2004 International Symposium on Empirical Software Engineering, ISESE'04* (pp. 271–280). IEEE.

Kroeger, T. A., Davidson, N. J., & Cook, S. C. (2014). Understanding the characteristics of quality for software engineering processes: A Grounded Theory investigation. *Information and Software Technology*, *56*(2), 252–271. doi:10.1016/j.infsof.2013.10.003

Langford, J., & McDonaugh, D. (2003). *Focus groups: Supporting effective product development*. London: Taylor and Francis.

Laporte, C. Y., Alexandre, S., & O'Connor, R. V. (2008). A Software Engineering Lifecycle Standard for Very Small Enterprises. In R. O'Connor, N. Baddoo, K. Smolander, & R. Messnarz (Eds.), *Proceedings of EuroSPI* (Vol. 16, pp. 129–141). Heidelberg: Springer. doi:10.1007/978-3-540-85936-9_12

Laporte, C. Y., & O'Connor, R. V. (2014a). Systems and Software Engineering Standards for Very Small Entities: Implementation and Initial Results. In *Proceedings of the 9th International Conference on the Quality of Information and Communications Technology (QUATIC)* (pp. 38–47).

Laporte, C. Y., & O'Connor, R. V. (2014b). A systems process lifecycle standard for very small entities: Development and pilot trials. In B. Barafort, R. V. O'Connor, & R. Messnarz (Eds.), *Systems, Software and Services Process Improvement* (pp. 13–24). Springer-Verlag.

Laporte, C. Y., O'Connor, R. V., & García Paucar, L. (2015). Software Engineering Standards and Guides for Very Small Entities: Implementation in two start-ups. In *to appear in proceedings of 10th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2015)*. Spain.

Lee, S., & Yong, H.-S. (2013). Agile Software Development Framework in a Small Project Environment. *Journal of Information Processing Systems*, *9*(1), 69–88. doi:10.3745/JIPS.2013.9.1.069

McCaffery, F., & Coleman, G. (2009). Lightweight SPI assessments: what is the real cost? *Software Process: Improvement and Practice*, *14*(5), 271–278. doi:10.1002/spip.430

McFall, D., Wilkie, F. G., McCaffery, F., Lester, N., & Sterritt, R. (2003). Software processes and process improvement in Northern Ireland. In *16th Int. Conf. Software & Systems Engineering and their Applications (ICSSEA 2003)* (pp. 1–10). Paris.

Misra, S., Fernández, L., & Colomo-Palacios, R. (2014). A simplified model for software inspection. *Journal of Software: Evolution and Process*, *26*(12), 1297–1315. doi:10.1002/smr.1691

Mora, M., Gelman, O., O'Connor, R., Alvarez, F., & Macias-Luevano, J. (2009). An Overview of Models and Standards of Processes in the SE, SwE, and IS Disciplines. In A. Cater-Steel (Ed.), *Information technology governance and service management: Frameworks and Adaptations* (pp. 371–387). Hershey, PA: IGI Global.

Moreno-Campos, E., Sanchez-Gordon, M.-L., Colomo-Palacios, R., & Amescua Seco, A. (2014). Towards Measuring the Impact of the ISO/IEC 29110 Standard: A Systematic Review. In *Proceedings of 21st EuroSPI 2014 Conference* (Vol. 425, pp. 1–12). Luxembourg: Springer Berlin Heidelberg. doi:10.1007/978-3-662-43896-1_1

Niazi, M., Babar, M. A., & Verner, J. M. (2010). Software Process Improvement barriers: A cross-cultural comparison. *Information and Software Technology*, *52*(11), 1204–1216. doi:10.1016/j.infsof.2010.06.005

O'Connor, R., & Basri, S. (2012). The Effect of Team Dynamics on Software Development Process Improvement: *International Journal of Human Capital and Information Technology Professionals*, *3*(3), 13–26. doi:10.4018/jhcitp.2012070102

O'Connor, R., & Coleman, G. (2009). Ignoring" Best Practice": Why Irish Software SMEs are Rejecting CMMI and ISO 9000. *Australasian Journal of Information Systems*, *16*(1).

O'Connor, R. V. (2012a). Evaluating Management Sentiment Towards ISO/IEC 29110 in Very Small Software Development Companies (Vol. 290, pp. 277–281). Presented at the 12th International Conference SPICE 2012, Spain: Springer Berlin Heidelberg. doi:10.1007/978-3-642-30439-2_31

O'Connor, R. V. (2012b). Using Grounded Theory Coding Mechanisms to Analyze Case Study and Focus Group Data in the Context of Software Process Research. In M. Mora, O. Gelman, A. L. Steenkamp, & M. Raisinghani (Eds.), *Research Methodologies, Innovations and Philosophies in Software Systems Engineering and Information Systems:* (pp. 256–270). IGI Global.

O'Connor, R. V., & Laporte, C. Y. (2014). An Innovative Approach to the Development of an International Software Process Lifecycle Standard for Very Small Entities. *International Journal of Information Technologies and Systems Approach*, *7*(1), 1–22. doi:10.4018/ijitsa.2014010101

Paternoster, N., Giardino, C., Unterkalmsteiner, M., Gorschek, T., & Abrahamsson, P. (2014). Software development in startup companies: A systematic mapping study. *Information and Software Technology*, *56*(10), 1200–1218. doi:10.1016/j.infsof.2014.04.014

Paulk, M. C. (1998). Using the software CMM in small organizations. In *Proceedings of the Pacific Northwest Software Quality Conference and the Eighth International Conference on Software Quality* (pp. 350–361). Portland, Oregon.

Pino, F. J., García, F., & Piattini, M. (2008). Software process improvement in small and medium software enterprises: a systematic review. *Software Quality Control Journal*, *16*(2), 237–261. doi:10.1007/s11219-007-9038-z

Pressman, R. (2009). *Software Engineering: A Practitioner's Approach* (Seventh edition.). New York: McGraw-Hill Science.

Richardson, I., & von Wangenheim, G. C. (2007). Why are Small Software Organizations Different? *IEEE Software*, *24*(1), 18–22.

Ruiz-Rube, I., Dodero, J. M., & Colomo-Palacios, R. (2015). A framework for software process deployment and evaluation. *Information and Software Technology*, *59*, 205–221. doi:10.1016/j.infsof.2014.12.001

Sanchez-Gordon, M.L., O'Connor, R. V., & Colomo-Palacios, R. (2015). Evaluating VSEs Viewpoint and Sentiment Towards the ISO/IEC 29110 Standard: A Two Country Grounded Theory Study. In A. Dorling, T. Rout, & R. V. O'Connor (Eds.), *Software Process Improvement and Capability Determination*. Springer-Verlag.

Schoeffel, P., & Benitti, F. B. V. (2012). Factors of Influence in Software Process Improvement: a Comparative Survey Between Micro and Small Enterprises (MSE) and Medium and Large Enterprises (MLE). *IEEE Latin America Transactions*, *10*(2), 1634–1643.

Staples, M., Niazi, M., Jeffery, R., Abrahams, A., Byatt, P., & Murphy, R. (2007). An exploratory study of why organizations do not adopt CMMI. *Journal of Systems and Software*, *80*(6), 883–895. doi:10.1016/j.jss.2006.09.008

Strauss, A., & Corbin, J. M. (1998). *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory* (Second Edition.). Thousand Oaks: SAGE Publications, Inc.

Taylor, P. S., Greer, D., Sage, P., Coleman, G., McDaid, K., Lawthers, I., & Corr, R. (2006). Applying an agility/discipline assessment for a small software organisation. In *7th International Conference, PROFES 2006* (Vol. 4034, pp. 290–304). Amsterdam, The Netherlands: Springer Berlin Heidelberg.

UNCTAD. (2012). *Information economy report 2012: the software industry and developing countries*. New York: United Nations Publications.

Verner, J. M., Babar, M. A., Cerpa, N., Hall, T., & Beecham, S. (2014). Factors that motivate software engineering teams: A four country empirical study. *Journal of Systems and Software*, *92*, 115–127. doi:10.1016/j.jss.2014.01.008

Von Wangenheim, C. G., Anacleto, A., & Salviano, C. F. (2006). Helping small companies assess software processes. *IEEE Software*, *23*(1), 91–98. doi:10.1109/MS.2006.13

Yilmaz, M., O'Connor, R. V., & Clarke, P. (2015). Software development roles: a multi-project empirical investigation. *ACM SIGSOFT Software Engineering Notes*, *40*(1), 1–5.

Yilmaz, M., O'Connor, R. V., & Collins, J. (2010). Improving software development process through economic mechanism design. In *Proceedings of the 17th European Systems and Software Process Improvement and Innovation (EuroSPI 2010)* (pp. 177–188). Springer-Verlag.

Zahran, S. (1998). *Software Process Improvement – Practical Guidelines for Business Success*. Boston, MA: Addison Wesley.

Author Biographies

Mary-Luz Sánchez-Gordón is a PhD student in Information Science and Technology at Universidad Carlos III de Madrid, Spain. She holds a Master's degree in Information Science and Technology from the same university. She studied Computer Engineering at Universidad Central del Ecuador, Quito, Ecuador. She also got her Master's degree in Education at this University. She has more than 10 years' experience in the software industry and five years in research and teaching in Ecuador. Her research interests are Software Process, Software Process Improvement and Knowledge Management

Prof. Rory V. O'Connor is a Professor of Software Engineering at Dublin City University where he is currently serving as the Head of the School of Computing. He is also a Senior Researcher with Lero, the Irish Software Engineering Research Centre. He has previously held research positions at both the National Centre for Software Engineering and the Centre for Teaching Computing, and has also worked as a software engineer and consultant for several European technology organizations. Prof. O'Connor is also Ireland's Head of delegation to ISO/IEC JCT1/ SC7, a member of ISO/IEC JTC1/SC7 Working Group 24 and editor of ISO/IEC 29110-2. His research interests are centered on the processes and standards whereby software intensive systems are designed,

implemented and managed. His focus is on researching methods, techniques, tools and standards for supporting the work of software project managers and software developers in relation to software process improvement, and the management of software development projects.