Packaged software implementation as requirements engineering practices at SMSDCs

Issam Jebreen,[a] Robert Wellington[b]
[a] SERL, School of Computing & Mathematical Sciences, Auckland University of Technology
Auckland 1142, New Zealand
ijebreen@aut.ac.nz
[b] School of Computing & Mathematical Sciences, Auckland University of Technology
Auckland 1142, New Zealand
rwelling@aut.ac.n

Abstract. An enduring problem in PS implementation has been the misalignments between software functions and users' needs. This problem is exacerbated by the fact that most current requirements engineering approaches are appropriate when software is to be developed from scratch, not for packaged software implementation. However, it is now the case that in most organizations, new software is created by integrating functionality from existing software and components or by implementing packaged software. To explore this area, we study cases of packaged software (PS) implementation in two software development companies. Our research design follows an interpretive approach, in which analysis was undertaken using an inductive approach. From an analysis of the cases, we induced that when implementing packaged software, there is a greater need for certainty regarding whether what the packaged software infrastructure requires and what the user's IT infrastructure has match each other. Furthermore, analysts may use work-arounds, but this is in order to minimize customization, rather than to reduce conflicts between requirements.

Keywords: requirement engineering, packaged software implementation.

1 INTRODUCTION

In recent years the market for large packaged software (PS) has become saturated (Morabito et al., 2005). PS companies and vendors have therefore begun to target the small to medium-sized PS market (attempting to sell packages to small – medium enterprises (SMEs), and various midrange or less complex software packages have been developed. SMSDCs are considered to be fundamentally different from large PS companies in several respects. In addition, studies of PS implementations have argued that findings about implementations in large companies cannot be applied to SMSDCs & SMEs (Buonanno et al., 2005; Laukkanen et al., 2007). Some distinguishing characteristics of SMEs include ownership type, culture, structure, and market orientation (Laukkanen et al., 2007). Other researchers have found that when it comes to IT/IS adoption, SMEs are constrained by limited resources and limited IS knowledge, or by a lack of IT expertise (Buonanno et al., 2005; Laukannen et al., 2007). These distinguishing characteristics of SMEs may influence the PS implementation issues they face (Zach et al., 2012). PS implementation remains a challenge for many SMSDCs (Malhotra & Temponi, 2010; Olson & Staley, 2012; Zach et al., 2012). Despite the importance of PS implementation being recognized by former studies, there has been little research exploring this issue further. In particular, discussions about SMSDCs rarely occur in the literature about PS implementation, and how the structure of SMSDCs shapes the software throughout its life cycle of implementation is rarely mentioned (Zach et al., 2012).

Previous researchers have also highlighted that there is a lack of knowledge about the requirements engineering practices that assist PS implementation in these types of companies, and due to the particular characteristics of SMSDCs, several software engineering researchers

have argued that most current requirements engineering practices are unsuitable for SMSDCs (Cox et al., 2008; Quispe et al., 2010).

The use of poor requirements engineering (RE) practices has often been identified as one of the major factors that can jeopardize the success of a software project. Meanwhile, researchers have also recognized that following appropriate RE practices contributes to the success of software projects. For example, Aranda et al. (2007) stated that gathering and managing requirements properly are key factors when it comes to the success of a software project. There is a general critical consensus that RE practices plays a very important role in the success or failure of software projects. However, it is not possible to improve RE practices until areas that need improvement in an organization's current RE practice have been identified. Meanwhile, the solution for improving RE practices will be different in each company; it has been found that a one-size-fits-all approach does not work in such a scenario (Cox, 2009; Quispe, 2008; Aranda et al., 2007).

This study presumes that the specific characteristics of SMSDCs may influence the RE practices in PS implementation. The recent literature has paid little attention to RE practices of PS implementation from the perspective of Small to Medium-sized Software Development Companies (SMSDCs) (Wagner et al., 2010; Zach et al., 2012).

2 RELATED WORKS

In this article Haddara & Zach (2011) reviewed the existing literature that relates to the adoption and running of ERP systems in SMEs. Noting that ERP systems have now been almost universally adopted by large organizations, Haddara & Zach (2011) stated that ERP vendors have now begun to turn their attention to small-medium sized organizations (SMEs). While ERP systems may be of benefit to SMEs, "the risks of adopting an ERP system are different for SMEs since SMEs are likely to have limited resources, and have business characteristics that are different from those of large organizations". Haddara & Zach (2011) shed light on the areas that are lacking in current research into ERP adoption in SMEs, and provide information intended to help 'practitioners, suppliers, and SMEs when embarking on ERP projects'. In fact, "SMEs have been recognized as fundamentally different environments compared to large enterprises" (Welsh & White, 1981), no reviews had been published of literature the deals with ERP implementations within SMEs (Haddara & Zach, 2011).

Haddara & Zach (2011) stated literature shows that there has been a gradual increase of academic interest in ERP usage within SMEs, and that the most frequent methods employed within research articles on this topic are case studies and surveys. They found that the implementation phase was the most discussed phase in the literature on ERP use in SMEs – a finding that accords with the main discussion topics of literature on ERP systems within larger organizations. However, the adoption decision, the acquisition phase, and the use and maintenance phase are also given reasonable degrees of attention within the literature on ERP use in SMEs. The phases for which literature was very scarce or non-existent are ERP evolution and ERP system retirement (Haddara & Zach, 2011). Moreover, Haddara & Zach (2011) stated that only two research papers considered 'in-house developed systems' to be a feasible option for SMEs, even though "standard ERP packages could compel rigid structures and inflexibility on niche SMEs". Hence, it reasonable to assume that the recent literature has paid little attention to RE practices of PS implementation from the perspective of SMSDCs.

The literature on implementation issues surveyed by Haddara & Zach (2011) found that "project activities, coordination, and project sponsors (Muscatello et al., 2003), employee behaviour, individual characteristics of ERP project management's team, and organization culture have a great effect on the success of ERP implementations in SMEs (Chien et al., 2007)". One study conducted by Newman & Zhao (2008) investigated the importance of business process modeling and business process re-engineering during implementations carried out in SMEs (Haddara & Zach, 2011). The conclusion of Newman & Zhao (2008)

study was that "in some cases, ERP systems should be customized to fit with niche SMEs and not vice versa, as they might lose their competitive advantage by complying with standard ERP processes".

After offering such brief discussions of the literature reviewed, Haddara & Zach (2011) make some further comments about the literature reviewed and suggest further avenues for study. First, they suggest that despite the fact that they found and reviewed 77 articles, this was still a very small number of articles to be published on the topic within 10 years, given the growing importance of ERP systems in relation to SMEs. They believe that "SMEs did not receive appropriate attention in comparison with ERP in LEs". They also identify specific gaps in the literature. These include a lack of studies that look at "ex-ante cost estimation, financial feasibility, and investment evaluation studies of ERP projects", lack of comparison between "SME's-specific ERP and general ERP systems" or between "industry-specific ERP packages vs. general ERP ones". Haddara & Zach (2011) found that very few studies had been made relating to the evolution of ERP systems within SMEs, and no studies had considered the retirement phase of an ERP system in relation to SMEs. Lastly, Haddara & Zach (2011) stated that while they did find 77 articles relating to ERP systems within SMEs, most of the SMEs were involved with traditional manufacturing, and it could be interesting to obtain results pertaining to different types of industries, or it could be of benefit if studies relating to ERP system use within SMEs were more explicit about exactly what kinds of manufacturing or industry the SME was involved with. They also noted that most of the studies conducted have considered companies located in America, Australia, Europe, and Asia. There was a shortage of studies investigating SMEs in Africa or in the Middle East. In general, existing literature have adopted a one sided perspective (in data collection) e.g. customer side, while other perspectives could enhance the understanding of certain phenomena. Also missing are any studies which investigate cases of failed ERP implementations within SMEs.

An enduring problem in PS implementation has been how to identify misalignments between software functions and users' needs. The current requirements engineering approaches (traditional RE) are appropriate when software is to be developed from scratch (Sommerville et al., 2012). However, in most organization, new software is now created by integrating functionality from existing software and components or by implementing packaged software. In such cases, it makes little sense to specify requirements in terms of what the software should do – the functionality is already defined in this software (Sommerville et al., 2012). Rather, requirements engineering when companies implement packaged software more frequently involves looking at what functions software provides, who needs those functions to do their job, and at what misalignments occur between software functions and users' needs.

According to Karlsson et al. (2007) there are "several studies that concern or include RE issues. However, none of these focus primarily on PS development and implementation. Furthermore, in most of these studies, the studied projects and organisations are mainly large, both in terms of the number of persons and requirements involved, and in terms of the duration of the projects". Quispe et al. (2010) highlighted that "there is a lack of knowledge about the requirements engineering practices in these types of companies [small-medium]". This lack of knowledge is particularly apparent when it comes to packaged software companies. It is in fact difficult for researchers to gain much knowledge about how SMSDCs carry out RE given that most SMSDCs seldom request external support, probably due to limited finances. However, RE research should eventually enable those companies to become aware of more state of the art or innovative RE techniques and to be able to improve their RE practice without external help (Merten et al., 2011).

Several questions remain unanswered. One core question that remains is: How are the RE practices of packaged software implementation enacted in SMSDCs?

2 RESEARCH METHOD

An ethnographic research method has been applied in relation to the two software development companies who participated in this research, Organisation 1 and Organisation 2. The business of both organizations considered in this study is dominated by the provision of packaged software solutions. This study follows Hammersley & Atkinson's (2007) discussion of various features of ethnographic research method:

- People's actions are studied in their everyday context, rather than under conditions created by the researcher. In other words, the research takes place in the field.
- Data is gathered from a range of sources, including documentary evidence of various kinds, but participant observation and/or relatively informal conversations are often the main means of collecting data.
- Data collection is, for the most part, relatively 'unstructured' in two senses. First, it does not involve following through with any specific fixed and detailed research design. Secondly, the categories that are used to interpret what people say or do are not built into the data collection process through the use of observation schedules or a questionnaire. Instead, they are generated through the process of data analysis.
- The focus is usually on a few cases, generally of a fairly small scale, perhaps a single setting or single group of people. This is to facilitate in-depth study.
- The analysis of data involves interpretation of the meanings, functions, and consequences of human actions and institutional practices, and how these are implicated in local, and perhaps also wider, contexts.
- What are produced, for the most part, are verbal descriptions, explanations, and theories; quantification and statistical analysis play a subordinate role.

4 RESULT & DISCUSSION

Our inductive analysis of the collected data, across both organisations, provided a rich set of findings to inform an alternative view of RE. Two main processes that emerged from the analysis of the collected data are presented. The first process is pre-implementation. The second process is during implementation that development of an in-depth understanding of client's needs. This involves two sub-processes. First, is the identifying misalignment between packaged software functions and users' needs, which involves conducting discussions with users to determine the future requirements of the software (what functions are desired by the users).

The pre-implementation RE practices stage in this study resembles such feasibility studies as those used in traditional RE practices at a high abstract level. This is because feasibility studies in traditional RE and the pre-implementation stage discussed here are similar in terms of their purpose, such as dealing with software objectives, time and budget. However, at the practical level, pre-implementation RE practice has its own specification. Table 1 shows feasibility study in traditional RE (Sommerville, 2004) vs. pre-implementation RE practices.

Table 1 Feasibility study vs. Pre-Implementation

Elements	Traditional RE Practices Feasibility study	Pre-Implementation RE Practices Feasibility study

Goals	Are the overall objectives of the organization satisfied by the proposed system? Can the system be developed with the proposed budget and timeline?	What are the client issues? What is the possible solution? Is the possible solution within the scope of the company's domain? What are the cost and time required for a possible solution?
Business dimension	Worthiness of proposed system.	Instilling confidence in the client, securing business, and creating a software offer.
Software analysis dimension	Information gathering to assist in the assessment of proposed system.	Information gathering to identify client's issues, new requirements and new features needed (if any) to assess cost and time for proposed solution implementation.
Stakeholders	Management of departments, experts, technical professionals, and people who are familiar with such a system.	Potential client, client's issues, client analysis information and client company structure information
Tools	Interviews, questionnaire.	Live Scenario, and discussion and negotiation.
Domain knowledge	The development organization and the customer can cooperate to ensure that the domain is understood.	The development organization has to be an expert in the domain.
Assessment criteria	Objectives of the organization are satisfied by proposed system. System is developed with the proposed budget and timeline.	A new future level, Customization level and Output level.
Critical Decision	Considers the worthiness of the proposed system, or regards changes, development decisions, seclude and budget.	The possible solution is within the company's domain.
Output	Feasibility study report and recommendations.	Packaged software offer, assessment report, and client issues, organization structure and analysis.
Scoping Factors	Budget, timeline, technical and development issues.	Packaged software assessment level, elements, and imitation of work domain, client organization size, and client's issues.

Table 2 follows a practice adopted from Sawyer et al. (1997) and Cox et al. (2009) and shows the requirements document practices in traditional RE vs. during implementation RE. The practices are mentioned in the table in terms of how they were used by analysts; the result is therefore based on an ethnographic account. When describing the during implementation processes, we use the four levels of assessment (as theorized by Sawyer et al. (1997)) in relation to the requirements document practices. These levels of assessment are the following: standardised use, Common use, discretionary use, and never use.

- Standardised use (SU): This practice has a documented standard and is always followed as part of the organisation's software development process i.e. it is mandatory. Followed when practices are perceived as having a high value.

- Common use (CU): This practice is widely followed in the organisation but is not mandatory. Followed when practices are perceived as having a medium value.
- Discretionary use (DU): This practice is used at the discretion of individual project managers. Some may have introduced the practice for a particular project. Followed when practices are perceived as having a low value.
- Never used (NU): The practice is never or rarely applied. Followed when practices are perceived as having a no value.

The table uses guideline classifications relating to 'good requirements practices' that consist of 'basic', 'intermediate', and 'advanced'. In this case, the 'basic' practices can continually be repeated, and it is possible to estimate costs, time, and resources associated with these practices. Meanwhile, 'intermediate' practices are more complex and lead to a 'defined' requirements engineering process. Lastly, 'advanced' practices are designed to help support the continuous improvement within any RE process. Some of these practices involve advanced technology and advanced methods which require specialist knowledge. They may also involve guidelines for organizational change. The requirements document itself is a document that effectively communicates requirements to customers, managers and developers.

Table 2 Requirements documents in Traditional RE vs. During Implementation

Requirements Documents Practices			
Type	No	Traditional RE Practices	During Implementation
Basic	RD1	Define a standard document structure	Standardised use
Basic	RD2	Explain how to use the document	Common use
Basic	RD3	Include a summary of the requirements	Standardised use
Basic	RD4	Make a business case for the system	Standardised use
Basic	RD5	Define specialised terms	Discretionary use
Basic	RD6	Make document layout readable	Common use
Basic	RD7	Help readers find information	Common use
Basic	RD8	Make the document easy to change	Common use
New Requirements Documents Practices			
Basic	RD9	Users' needs/Misalignments specification document	Standardised use
Basic	RD10	Estimating time needed for users' needs document	Standardised use
Basic	RD11	Estimating cost needed for users' needs document	Standardised use
Basic	RD12	Include users' needs validation document	Standardised use

Table 3 shows the requirements elicitation practices in traditional RE, using terminology and concepts adopted from Sawyer et al. (1997) and Cox et al. (2009) vs. during implementation RE practices. Requirements elicitation is defined as a group of practices designed to help discover the requirements for a system. These practices are followed by analysts in order to elicit requirements from the stakeholders related to the system. However, the requirements elicited also depend on the application domain and on the organizational and operational environments of the system.

Table 3 Requirements elicitation in traditional RE vs. During implementation

Requirements Elicitation Practices

Type	No	Traditional RE Practices	During Implementation
Basic	RE1	Assess system feasibility	Standardised use
Basic	RE2	Be sensitive to organisational and political consideration	Standardised use
Basic	RE3	Identify and consult system stakeholders	Standardised use
Basic	RE4	Record requirements sources	Standardised use
Basic	RE5	Define the system's operating environment	Standardised use
Basic	RE6	Use business concerns to drive requirements elicitation	Standardised use
Intermediate	RE7	Look for domain constraints	Discretionary use
Intermediate	RE8	Record requirements rationale	Common use
Intermediate	RE9	Collect requirements from multiple viewpoints	Discretionary use
Intermediate	RE10	Prototype poorly understood requirements	Standardised use
Intermediate	RE11	Use scenarios to elicit requirements	Standardised use
Intermediate	RE12	Define operational processes	Discretionary use
Advanced	RE13	Reuse requirements	Standardised use
New Requirements Elicitation Practices			
Basic	RE14	Use live software demonstration to elicit users' needs	Standardised use
Basic	RE15	Use a user manual	Standardised use

Table 4 shows the requirements analysis and negotiation in traditional RE practices, using terminology and concepts adopted from Sawyer et al. (1997) and Cox et al. (2008) vs. during implementation RE practices. Requirements analysis and negotiation are defined as practices that help analysts to identify and resolve problems associated with the elicited requirements. These may include identifying and resolving misalignments, incompatibility issues, and missing information.

Table 4 Requirements analysis and negotiation in traditional RE vs. During implementation

Requirements Analysis and Negotiation			
Type	No	Traditional RE Practices	During Implementation
Basic	RA1	Define system boundaries	Standardised use
Basic	RA2	Use checklists for requirements analysis	Discretionary use
Basic	RA3	Provide software to support negotiations	Standardised use
Basic	RA4	Plan for conflicts and conflict resolution	Standardised use
Basic	RA5	Prioritise requirements	Discretionary use
Intermediate	RA6	Classify requirements using a multi-dimensional approach	Standardised use
Intermediate	RA7	Use interaction matrices to find conflicts and overlaps	Discretionary use
Advanced	RA8	Assess requirements risks	Standardised use
New Requirements Analysis and Negotiation			
Basic	RA9	Use print-out of a screen shot to clarify conflicts, and engaging in conflict resolution	Standardised use

Basic	RA10	Use live case scenarios to support negotiations	Standardised use

As can be seen from information provided in Table 1, there are a number of differences in practice, and differences of purpose, between the elements of feasibility studies carried out in traditional RE and for Pre-Implementation RE. Traditional RE and Pre-Implementation RE share similarities as both can be seen as comprised of the same kinds of elements, and as, to some degree, sharing similar objectives and being influenced by similar business concerns and technical concerns. For example, the stages involved in both processes can be divided into the same ten dimensions, which involve goals, the business dimension, the software analysis dimension, stakeholders, tools, domain knowledge, assessment criteria, critical decision, output, and scoping factors. However, within these dimensions, important differences appear. The analyst concerned with carrying out RE for packaged software implementation will be concerned with accessing different information and meeting different objectives than the analyst concerned with building custom-made software.

For example, when building a bespoke system, traditional RE will focus on identifying whether the timeline and budget that have been proposed are feasible, and then with making sure that the organization's objectives can actually be met by the system that has been proposed. With Pre-Implementation RE, however, the analyst must instead think about what the client's specific issues are and identify whether any existing packages offered by the analysts' company can offer a solution. The analysts engaging in Pre-Implementation RE must also consider the possibility of refusing a request for a particular solution if that solution falls outside the scope of the company or outside the scope of the company's current products. Part of the process of identifying whether the solution is within the company's scope may involve thinking about the time and cost involved with implementing a particular package or with making requested changes to that package.

With traditional RE, the main goal of the 'business dimension' of RE is concerned with establishing whether the proposed system is 'worthy': whether it can be created and whether it will actually satisfy the demands of the business and be the best possible system for the business. The analyst carrying out Pre-Implementation RE, however, will be engaged with different concerns, such as actually selling the proposed packaged system to the client by showing them how the package operates and how it could fulfill their requirements. The analyst carrying out Pre-Implementation RE must actively instill confidence in the client, secure the client's business, and create a software product offer.

The software analysis dimension in traditional requirements engineering and pre-implementation requirements engineering is quite similar. The analysts in both forms of requirement engineering carry out a range of activities that find out the client's issues that need solving and that help them to find initial requirements. They will later need to follow up on such requirements by checking in case new requirements are needed or new features need to be added to the proposed solution. If new features are required, they will again need to assess the cost and time involved with such requirements. However, there are some differences between the two forms of requirements engineering. In pre-implementation requirements engineering, analysts need to consider the modifications to existing functions that have been requested by clients. However, such considerations do not concern analysts practicing traditional requirements engineering.

The stakeholders involved with the two different forms of RE are also different. As shown in Table 1, with traditional RE, the analyst generally interacts only with people who manage departments within the client business, or with experts, technical professionals, and people who are familiar with such a system. These are the people whose needs or input the analyst will be concerned with. With Pre-Implementation RE, however, the analyst's considerations will be somewhat broader, as they need to first identify potential clients, then gather as much

information as possible about the potential clients, and then prepare to attract the clients by identifying the client's issues that need solving. This is done via the use of forms relating client analysis information and by using the analyst company's databases that contain information about potential clients' client company structures.

The tools used in the two forms of RE also differ, since different forms of information need to be collected. In traditional RE the analyst is able to collect the required information by holding interviews and by using questionnaires. However, in Pre-Implementation RE the analyst engages in Live Scenarios to demonstrate the proposed solution, or will find out what the requirements for the solution are, and at the same time, sell the solution, by carrying out discussions and various forms of negotiation. The Pre-Implementation analyst is engaged in designing a system that meets the client's requirements, and in demonstrating and selling that proposed solution.

The level of domain knowledge required for the analyst engaging in these different forms of RE also changes. With traditional RE, the analyst can gain sufficient knowledge of the client's domain by interacting with and listening to the client. The client is more active in advising the analyst what is needed in the system. With Pre-Implementation RE, however, the client will expect the development organization to already be an expert in the domain and to offer them the best possible solution or a range of viable solutions.

The assessment criteria used to design and develop the system also differ between traditional RE and Pre-Implementation RE. With traditional RE, the feasibility of the system is seen to depend on whether the objectives of the organization will be satisfied by the proposed system. If it is considered that they will be, the system will then be developed in accordance with the proposed budget and timeline. Pre-Implementation RE for packaged software involves its own set of assessment criteria. As detailed in discussions earlier, these assessment criteria involve a new future level (which assesses proposed changes to the existing package), a customization level which assesses the impact that may result from modifying existing functions to fill gaps in requirements, and an Output level which consists of creating new reports or modifying existing reports.

With traditional RE, the main Critical decision that needs to be made usually relates to confirming the worthiness of the proposed system. Other critical decisions, or factors in the critical decision may relate to changes to the proposed system, or to budgetary factors or company developments. The analyst engaging in Pre-Implementation RE will make a Critical decision when deciding whether the solution needed by the potential client is within the domain of the analyst's company.

When considering the Output dimension of traditional RE, the analyst will rely on a feasibility study report and on recommendations. Meanwhile, the analyst working with Pre-Implementation RE considers the project feasibility and responds to its feasibility by means of the assessment report, information gained about client issues, organization structure and analysis, and the packaged software offer that is made to the client.

The last element of comparison between feasibility studies in traditional RE and Pre-Implementation RE is Scoping Factors. Again, the Scoping Factors involved in the two different forms of RE are not the same. In traditional RE, scoping is guided mainly by the budget that has been set for the project, and by its timeline, and also by technical and development issues. Pre-Implementation RE practice differs from this, as Scoping for packaged software is influenced by a number of factors, including assessment levels, the packaged software offer elements, and the limitation of the work domain, the client's organization size, and the client's issues.

As can be seen from Table 2, in traditional RE, all of the levels of requirements documents practices remain at 'basic', whereas many of those practices associated with during implementation RE are actually at the same level as in traditional RE. The results show that the most common standardised requirements documentation practices are to define a

standard document structure (RD1), to include a summary of the requirements (RD3), and to make a business case for a project (RD4). The practices of explaining how to use the document (RD2), making the document layout readable (RD6), helping readers find information (RD7), and making the document easy to change (RD8), can be considered as 'Common use' practices in during implementation RE. This means that these practices are widely followed in the organisations but are not mandatory. We also found that during implementation RE documentation practices approached defining specialised terms (RD5) with 'discretionary use'.

However, during my field work, we also discovered a range of new practices that were carried out, that are related to the requirements document. We have listed these 'New Requirements Documents Practices' in the lower half of Table 2. These new practices involve creating a users' needs/misalignments specification document, estimating the time and cost related to creating the users' needs/misalignments document, and the users' needs validation document. All of these practices are carried out at a 'basic' level. They are a part of during implementation RE, being practiced with 'standardised use'. These new practices have not been identified during previous studies of traditional RE and packaged software requirements engineering practices.

In Table 3 we see that in traditional requirements elicitation, many RE practices are carried out at the 'basic' level; that is, they are almost always practiced. However, just over half of the practices operate at the 'intermediate' or advanced levels. In during implementation RE, most of those practices that are basic in traditional RE are standard practices. Practices RE 7 through RE 13, as shown in the table, are practiced rather differently in traditional RE and during implementation RE. In traditional RE, a large range of practices could be considered as 'intermediate' practices, that is, they are more complex and not always practiced. Those practices regarded as 'intermediate' include looking for domain constraints, recording the requirements rationale, collecting requirements from multiple viewpoints, prototyping poorly understood requirements, using scenarios to elicit requirements, and defining operational processes. In during implementation RE, however, these practices are carried out at a range of levels. For example, prototyping poorly understood requirements and using scenarios to elicit requirements are carried out as standardised practices, but looking for domain constraints, collecting requirements from multiple viewpoints, and defining operational processes are practices that are only carried out with discretionary use. One other requirements elicitation practice in this group, recording the requirements rationale, can be considered as having 'normal' use in during implementation RE. Another difference occurs with reusing requirements (RE 13). The practice of reusing requirements is an advanced practice in traditional RE, a practice used to improve a system, whereas in during implementation RE, it has a completely standardised use.

We have also identified some new requirements for traditional RE and during implementation RE, in terms of requirements elicitation practices. The new practices are using a live software demonstration to elicit the users' needs, and using a user manual. These practices are carried out at the 'basic' level, and have 'standardised use' in during implementation RE. Therefore, they are almost always practiced during documented standards when used in during implementation RE.

We can see from Table 4 that in traditional requirements analysis and negotiation, many of the practices are considered basic elements of RE. For example, RA 1 through RA 5, which involves defining system boundaries, using checklists, providing support to support negotiations, planning in case of conflicts, and prioritizing requirements, are all listed at the basic level in the table above. In during implementation RE, all but two of these practices have a standardised use, RA 1, 3, 4, and 6 therefore have documented standards that are followed. Two practices, however, are approached differently, with RA 2 (using checklists for requirements analysis) and RA 5 (prioritising requirements) receiving discretionary use. This

is understandable, since analysts, during packaged software implementation use screenshots to validate user needs, rather than using a checklist. This is because the software has already been created and only needs modification. During packaged software implementation, prioritizing requirements is not a basic practice. Rather analysts collect requirements in a circular process and develop those requirements that are agreed upon at the time or that their managers agree they should give priority to (i.e. their managers act with 'discretion' regarding the requirements). There are further differences in how RA 6, 7, and 8 are practiced within the two approaches. In traditional RE, classifying requirements using a multi-dimensional approach is an intermediate practice, therefore not always performed, however, in during implementation RE, it is a standardised practice. In traditional RE using interaction matrices to find overlaps or possible conflicts is considered an intermediate practice. In during implementation RE, this practice is discretionary, not universal. Lastly, while traditional RE treats assessing requirements risks as an advanced practice (RA 8), this is a standardised practice in during implementation RE.

Once more, we identified some new practices related to requirement analysis and negotiation. These include using print-outs of screen shots to clarify conflicts, and engaging in other forms of conflict resolution (RA9), and using live case scenarios to support negotiations (RA10). The use of printouts and other forms of conflict resolution, and the use of live case scenarios are carried out at the 'basic' level in traditional RE, and have 'standardised use' in during implementation RE.

5 CONCLUSION

The research reported in this paper is one of few empirical studies focused on requirement engineering practices for packaged software implementation. It offers an in-depth, qualitative view of requirement engineering to implement packaged software. In relationship to the existing literature on packaged software, our focus is on activities near the end of a software package's lifecycle. Given the growing importance of packaged software, and the apparent inevitability of packaged software implementation, it is increasingly necessary to understand the requirement engineering practices for packaged software implementation. Our contribution to this effort is a parsimonious theoretical result portraying the interactions among requirement engineering practices and for packaged software implementation. The result draws its inspiration from earlier literature on requirement engineering and other related literatures.

Packaged software implementation is a unique type of IS software, with characteristics that distinguish it from requirement engineering, traditional system development and initial adoption of a commercial system. In a packaged software implementation, the software company has substantial control over the development of packaged software, and the client organization becomes vulnerable to software company actions. Future research could consider problematic area within existing RE tools is that they do not support a distributed collaboratively collection and analysis of requirements, which can be said is necessary in the packaged software context since packaged software requirements for implementation comes from defendable requirements. In such cases, it makes little sense to specify requirements in terms of what the software should do – the functionality is already defined in this software. Rather, we argue that requirement engineering practices for PS implementation should be approached from a misalignments perspective, which focuses on what functions software provides, who needs a particular function in order to do their job, and what misalignments exist between software functions and users' needs.

References

Aranda, J., Easterbrook, S., & Wilson, G. (2007). Requirements in the wild: How small companies do itIEEE. Symposium conducted at the meeting of the Requirements Engineering Conference, 2007. RE'07. 15th IEEE International

Buonanno, G., Faverio, P., Pigni, F., Ravarini, A., Sciuto, D., & Tagliavini, M. (2005). Factors affecting ERP system adoption: a comparative analysis between SMEs and large companies. Journal of Enterprise Information Management, 18(4), 384-426.

Chien, S.-W., Hu, C., Reimers, K., & Lin, J.-S. (2007). The influence of centrifugal and centripetal forces on ERP project success in small and medium-sized enterprises in China and Taiwan. International Journal of Production Economics, 107(2), 380-396.

Cox, K., Niazi, M., & Verner, J. (2009). Empirical study of Sommerville and Sawyer's requirements engineering practices. IET software, 3(5), 339-355.

Haddara, M., & Zach, O. (2011). ERP systems in SMEs: A literature review IEEE. Symposium conducted at the meeting of the System Sciences (HICSS), 2011 44th Hawaii International Conference.

Hammersley, M., & Atkinson, P. (2007). Ethnography: Principles in practice: Taylor & Francis.

Karlsson, L., Dahlstedt, Å. G., Regnell, B., Natt och Dag, J., & Persson, A. (2007). Requirements engineering challenges in market-driven software development–An interview study with practitioners. Information and Software technology, 49(6), 588-604.

Laukkanen, S., Sarpola, S., & Hallikainen, P. (2007). Enterprise size matters: objectives and constraints of ERP adoption. Journal of Enterprise Information Management, 20(3), 319-334.

Malhotra, R., & Temponi, C. (2010). Critical decisions for ERP integration: Small business issues. International Journal of Information Management, 30(1), 28-37.

Merten, T., Lauenroth, K., & Bürsner, S. (2011). Towards a new understanding of small and medium sized enterprises in requirements engineering research. In Requirements Engineering: Foundation for Software Quality (pp. 60-65): Springer.

Muscatello, J. R., Small, M. H., & Chen, I. J. (2003). Implementing enterprise resource planning (ERP) systems in small and midsize manufacturing firms. International Journal of Operations & Production Management, 23(8), 850-871.

Newman, M., & Zhao, Y. (2008). The process of enterprise resource planning implementation and business process re- engineering: tales from two Chinese small and medium- sized enterprises. Information Systems Journal, 18(4), 405-426.

Olson, D. L., & Staley, J. (2012). Case study of open-source enterprise resource planning implementation in a small business. Enterprise Information Systems, 6(1), 79-94.

Quispe, A., Marques, M., Silvestre, L., Ochoa, S. F., & Robbes, R. (2010). Requirements Engineering Practices in Very Small Software Enterprises: A Diagnostic StudyIEEE. Symposium conducted at the meeting of the Chilean Computer Science Society (SCCC), 2010.

Sawyer, P., Sommerville, I., & Viller, S. (1997). Requirements process improvement through the phased introduction of good practice. Software Process Improvement and Practice, 3(1), 19-34.

Sommerville, I., Lock, R., & Storer, T. (2012). Information requirements for enterprise systems. arXiv preprint arXiv:1209.5246.

Sommerville, I. (2004). Software engineering (Seven Edition ed.): Addison-Wesley Publishers Limited.

Wagner, E. L., Newell, S., & Piccoli, G. (2010). Understanding Project Survival in an ES Environment: A Sociomaterial Practice Perspective. Journal of the Association for Information Systems, 11(5).

Xu, L., & Brinkkemper, S. (2007). Concepts of product software. European Journal of Information Systems, 16.

Zach, O., & Munkvold, B. E. (2012). Identifying reasons for ERP system customization in SMEs: a multiple case study. Journal of Enterprise Information Management, 25(5), 462-478.