Applied Multi-Case Research in a Mixed-Method Research Project: Customer Configuration Updating Improvement

Slinger Jansen and Sjaak Brinkkemper
Institute of Information and Computer Sciences
Utrecht University, the Netherlands
{S.Jansen, S.Brinkkemper}@cs.uu.nl

Keywords: Multiple Case Research, Mixed Method Research, Customer Configuration Updating, Research Method Engineering, Case Study Protocol, Case Study Database, Case Study Reporting

Abstract
Even though information systems is a maturing research area, information systems case study reports generally lack extensive method descriptions, validity defense, and are rarely conducted within a multi-case research project. This reduces the ability to build theory in information systems research using case study reports. In this chapter we offer guidelines, examples, and improvements for multi-case studies. If information system researchers stick to these guidelines, case study reports and papers will get published more often, improving the rapidly maturing research area of information systems.

1 Multiple Case Study Research

It is our belief that IS research is quickly getting more mature and deserves more attention in the area of methods and practices. In this chapter we report on our experiences (van de Weerd, Brinkkemper, Souer & Versendaal, 2006), (van de Weerd, Brinkkemper & Versendaal, 2007), (Jansen, Ballintijn, Brinkkemper, & van Nieuwland, 2006), (Jansen & Brinkkemper, 2007) with multi-case studies in the area of Information Systems (IS) research. We believe that case study evidence is fundamental for building theories in this rapidly maturing field (Darke, Shanks & Broadbent, 1998). Case studies encourage students and researches to train themselves in understanding large complex systems (for system maintenance) and organizations. Furthermore, case studies are generally appreciated by practitioners, and provide a popular method in IS research to disseminate and explain phenomena in the field. These case studies are, however, often reported without much validation. This chapter provides a detailed description of a multi-case research project to help researchers to document their research methods, compare their research approach, and provide them with helpful experience reports. We believe such descriptions will improve the overall quality of multi-case study research and reports, thus contributing to IS research at large.

1.1 Case Study Context

The experiences reported in this chapter concern a multi-case research project that lasted from 2003 until 2006 and concerns the customer configuration updating practices of product software vendors.

To date product software is defined as a packaged configuration of software components or a software-based service, with auxiliary materials, which is released for and traded in a specific market (Xu & Brinkkemper, 2005). One area that is specific to product software vendors is the fact that they have to release, deliver, and deploy their products on a wide range of systems, for a wide range of customers, in many variations. Furthermore, these applications constantly evolve, introducing versioning problems. An increasingly important part of product software development thus is customer configuration updating (CCU). CCU is *the combination of the vendor side release process, the product or update delivery process, the customer side deployment process, and the activation and usage processes* (Jansen, Ballintijn, Brinkkemper, & van Nieuwland, 2006). The release process describes how products and updates are made available to testers, pilot customers, and customers. The delivery process describes the method and frequency of update and knowledge delivery from vendor to customer *and* from customer to vendor. The deployment process describes how a system or customer configuration evolves between component configurations due to the installation of products and updates. Finally, the activation and usage process concerns license activation and knowledge creation on the end-user side.

Product software vendors encounter particular problems when trying to improve these processes, because vendors have to deal with multiple revisions, variable features, different deployment environments and architectures, different customers, different distribution media, and dependencies on external products. Also, there are not many tools available that support the delivery and deployment of software product releases that are generic enough to accomplish these tasks for any product. Case studies have shown (Jansen, 2005;

Ballintijn, 2007; Jansen, Ballintijn & Brinkkemper, 2004; Jansen, 2006a; Jansen, 2006b; Jansen, Brinkkemper, Ballintijn & van Nieuwland, 2005) that many issues remain unsolved. Large parts of the CCU process are still performed manually, such as quick fix distribution and deployment, license file creation, and error feedback reporting. Next to this, surveys have shown that up to 15 percent of deployments of products are unsuccessful, due to missing components and configuration errors by the deployers. System administrators for large networked environments too, experience many problems with respect to deployment, caused by heterogeneous environments, faulty configuration update tools, and lack of knowledge about system and software constraints. These problems require attention, due to their large overhead costs (implementation, testing, configuration evolution, etc) and due to the fact that no quality guarantees can be given at deployment or update time.

To solve the problems stated above a model to improve the CCU process of a software vendor has been developed. This model is based on a number of theories which, when applied correctly, improve the CCU process for any product software vendor and its customers (Roberts, Cater-Steel & Toleman, 2006). Over the last four years we have conducted mixed-method research to evaluate these theories. The methods that have been applied are case studies, tool evaluations, prototype building, design research and two surveys. The different methods were used to confirm or refute the theories. Case study research proved particularly useful where "research and theory are at their early formative stages" (Benbasat, Goldstein & Mead, 1987). We would like to address the role of case studies in mixed-method research.

The CCU improvement model, although intuitively addressing the right issues with the right amount of attention, needed to be evaluated and validated. The model consists of a process model and maturity measures in each of the mentioned process areas. From 2003 Utrecht University and CWI[1] have performed a number of case studies to establish problems in CCU and to test the model for improvement of CCU on the software vendor, system administrator, and end-user sides. These case studies led to successful research output (Ballintijn, 2007; Jansen, Ballintijn, Brinkkemper, & van Nieuwland, 2006; Jansen & Rijsemus, 2006; Jansen & Brinkkemper, 2007) and contributed to the area of IS research and more specifically software product management. To further confirm conclusions and observations drawn

[1]http://www.cwi.nl

in the case studies and make a stronger claim for generalisability, a survey has been undertaken during February of 2007 (Jansen & Brinkkemper, 2008). The case studies were performed with care and rigor, using Yin's (2003) case study method and guidelines. To avoid common pitfalls we have used guidelines and pointers from Kitchenham, Pickard and Pfleeger (1995) and Flyvbjerg (2006).

1.2 Contributions and Structure

The main aims of this chapter are to encourage researchers in IS to more explicitly define and describe their research methods in publications. Secondly, we wish to encourage both industrial and scientific researchers to publish case study reports in case study databases. Finally, we wish to encourage researchers to do more emperical research and rigorously apply validity criteria to their research.

The chapter reports on a multi-case research project that lasted from 2003 until 2006 and concerns the customer configuration updating practices of product software vendors. In the next section the research methods and validity criteria are described. Furthermore, section 2 provides a detailed overview of the used research methods. In section 3 the case study protocols, databases, and reports are described in detail. In section 4 an experience report is provided on the case studies and the strong points of this research and its findings are highlighted. In section 5 the cases are discussed and some of the points for improvement are stated. Section 6 describes the conclusions of this research and discusses the future of case studies in education and IS research.

2 The Research and Its Method

The aims of these case studies were to determine the state of the practice of CCU, to explore what problems were still open, and to test the hypothesis that the CCU model potentially efficiently and cost-effectively improves the CCU processes. At the time of conception we were already aware of the fact that this was a mixed-method research project, where the team would be doing tool prototyping and surveys as well as the case studies. To make sure the case studies would follow rigorous methods we extensively used Yin (2003). This research followed positivistic principles, where we believe that there are pre-existing regularities that can be discovered, investigated, and characterized relatively unproblematic using constructs devised by us (Orlikowski & Baroudi, 1991). We did attempt to closely guard the principles of validity, by explicitly using the same case study protocol, case study database structure, and case study report structure.

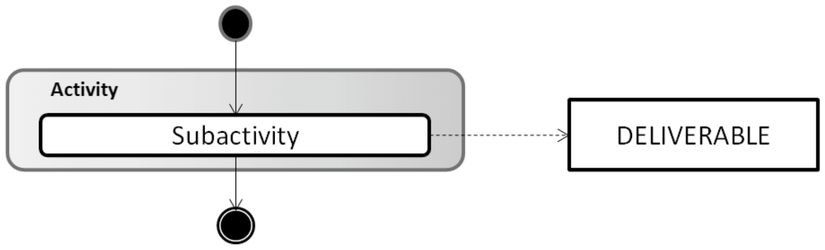


Figure 1: Process-Deliverable Diagram Example

The units of study of the cases were product software vendors, and more specific the CCU processes of different product teams within these product software vendors. The case studies had the specific aims to uncover problems that were potentially of interest to the research community, to establish the state of the practice by documenting the practices and processes of product software vendors, and to find potential points for improvement for the participating product software vendors. The deliverables of each case study are the case study report (listing potential points for improvement), the case study database, and a list of publishable results. We find that our work is similar to the work of Gable (1994). Gable, however, does not elaborate on the specific methods on how data was gathered for his study and does not provide the same level as detail as this chapter. Furthermore, he defined his first study as a pilot project because no dependent variables had been specified. In another multiple-case study, which has strong survey elements, Radford and Kern (2006) only shortly describe their data collection and analysis.

2.1 The Method

As IS research is rapidly maturing we are eager to present our research method for future reference, usage, and criticism. It is our firm belief that a further standardization and documentation of IS research methods contributes to the field as a whole and enables better research. To document the research method we use process-deliverable diagrams, a technique taken from the field of method engineering (Saeki, 1994). Method engineering enables a method developer to take parts of a method called method fragments from a method base and compose them into new methods that are applicable to the developer's situation. With our work we wish to contribute to such a method base for IS research.

The process-deliverable diagrams define activities, sub activities, and deliverables from the activities. The left-hand side is a meta-process model based on UML activity diagrams and the right-hand side is a meta-data model based on UML class diagrams. The dotted arrows indicate which concepts (deliverables) are created or adjusted in the corresponding activities. Please see van de Weerd, Brinkkemper, Souer & Versendaal (2006) for a more detailed overview of this modeling technique.

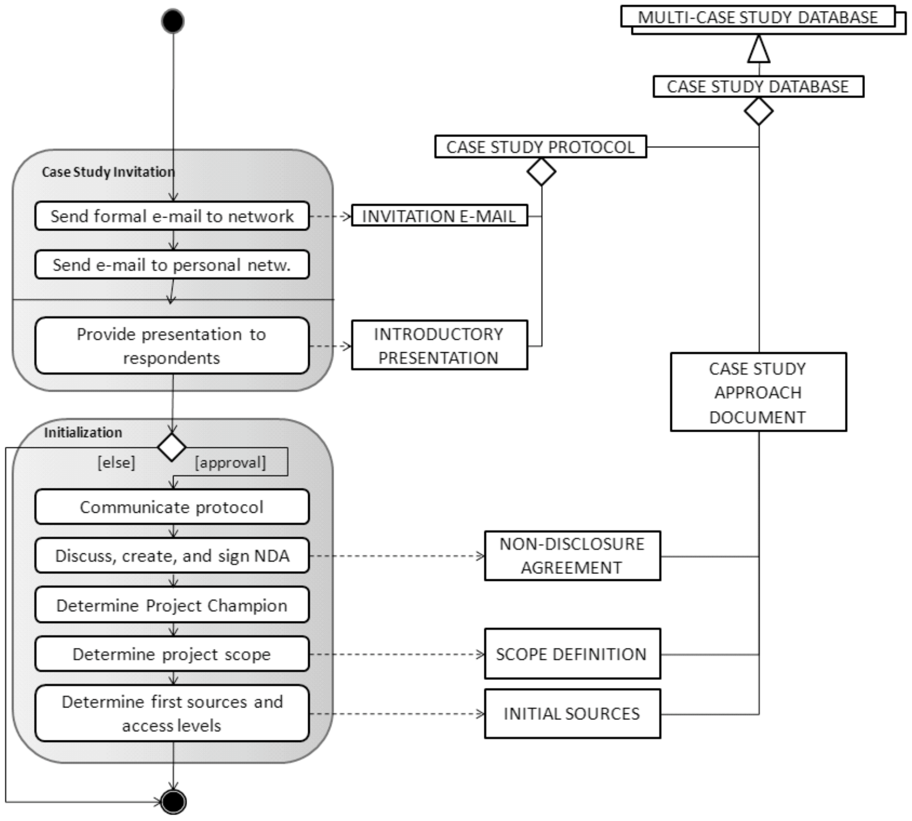

MULTI-CASE STUDY DATABASE
CASE STUDY DATABASE
CASE STUDY PROTOCOL
Case Study Invitation
Send formal e-mail to network
INVITATION E-MAIL
Send e-mail to personal netw.
Provide presentation to respondents
INTRODUCTORY PRESENTATION
CASE STUDY APPROACH DOCUMENT
Initialization
[else]
[approval]
Communicate protocol
Discuss, create, and sign NDA
NON-DISCLOSURE AGREEMENT
Determine Project Champion
Determine project scope
SCOPE DEFINITION
Determine first sources and access levels
INITIAL SOURCES

Figure 2: Invitation and Initiation Activities

The case studies have all been conducted using the same method, consisting of four activities: case study invitation, case study initiation, research execution, and case study finalization. The main deliverables of each case study were the case study protocol (see section 3.1), case study database (see section 3.2), and case study report (see section 3.3). We now describe each activity and each deliverable in detail.

During the **case study invitation** (see Figure 2) activity we attempted to find as many interested participants as possible by contacting our professional and personal networks. The e-mail sent out to different mailing lists described the motivation behind the research, the research and case study process, the possibilities for cooperation, and invited interested parties to reply. Once a number of suitable candidates had replied we contacted them by phone to further establish the details and to make an appointment with the initiator (usually someone from higher management) and the potential *project champion* for an introductory presentation. At these meetings we would generally also introduce the Non-Disclosure Agreement (NDA), to clarify that we would treat results as confidential but had the intention to publish interesting results. After the introductory presentation both the research group and the product software vendor had 2 weeks to consider the possibilities for cooperation and initiate further contact.

During the **case study initiation phase** (see Figure 2) everything was set up such that after this phase the case study researcher could start with the research undisturbed. First a protocol was created to demonstrate potential impact on the organization and to show that the intention was a win-win situation for both the company, in the form of advice and publicity, and the research team, in the form of the results of the case study and the opportunity to (anonymously) publish the results of the research. As part of the protocol the NDA was finalized and signed together with the project champion. The project champion also played a part in determining the scope of the research within the company (who to interview, access to which systems, documentation, access to the intranet, etc.) Finally the first documents were shared and discussed with the project champion to get the researcher started. These activities generally provided the researcher with a running start once the researcher moved its working place to the company for the case study period.

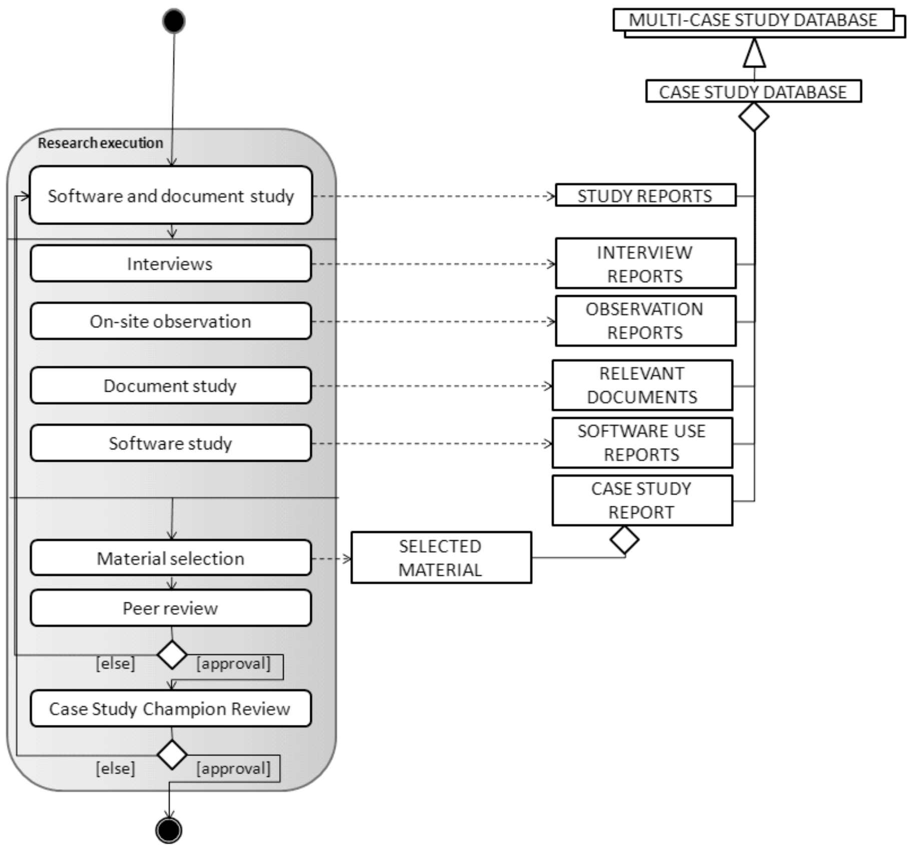


Figure 3: Research Execution Activity

Once the researcher arrived at the site the **case study research execution phase** (see Figure 3) was formally started. The case study database was set up (both a digital folder and a paper binder) and the peer reviewer was assigned. The researcher would then start by studying mostly documents and software that was available on the intranet. After the researcher had built a database of assumptions (mostly in the form of schematic process pictures and lists of facts) the first phase of the interviews was started. After most of the interviews and observations were done the researcher would, while still at the product software vendor's site, write up a draft of the case study report. This report would be extensively reviewed by the peer researcher and after his approval the report would be reviewed by the project champion. The project champion's approval of the case study report indicated the end of the research execution phase. The researcher would then return to its research institute to finalize the case study report.

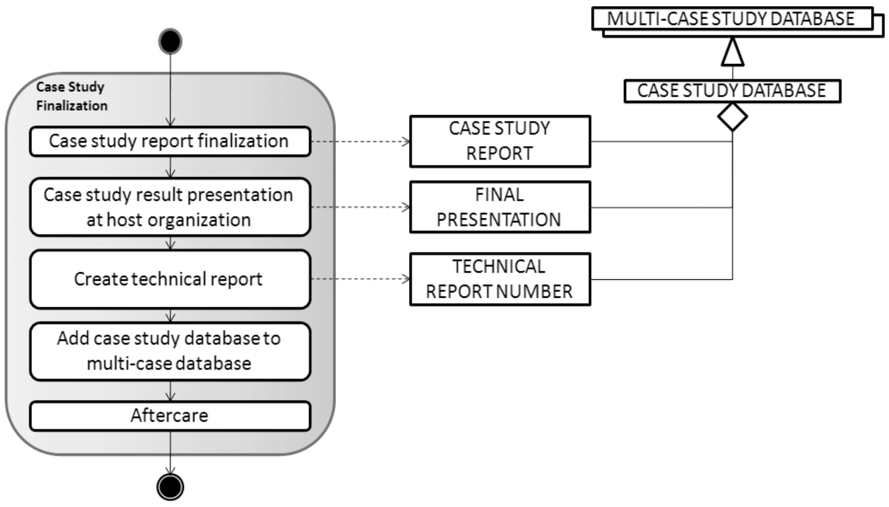

Case Study Finalization
Case study report finalization
Case study result presentation at host organization
Create technical report
Add case study database to multi-case database
Aftercare
CASE STUDY REPORT
FINAL PRESENTATION
TECHNICAL REPORT NUMBER
MULTI-CASE STUDY DATABASE
CASE STUDY DATABASE

Figure 4: Case Study Finalization Activity

The **finalization phase** (see figure Figure 4) of the case study was usually started after the case study project champion had approved of the report. Final changes would be made by the researcher at his home institute where a final presentation was also prepared. Both the report and the presentation contain advice, based on CCU literature for the product software vendor on how to improve their CCU practices. As a final step the case study report was published as a technical report within the research organization, the database was added to the large multi-case database, possibilities for further cooperation were discussed, and publishing opportunities for the research were considered.

The researcher used three methods to find answers to the research questions, being document study, software study, and interviews.

2.1.1 Document Study

The project champion would generally supply the researcher with a number of relevant documents before the research was started. These documents would not only provide insights into the CCU processes of a software vendor but more importantly provide an understanding of the terminology used in the company. The researcher would write down terms that were confusing, and bring them up during the first interviews. We noticed that development documentation is often out of date and frequently presents a vision instead of the truth. Document study was the second most powerful but also time consuming method. Documents were very helpful in supporting both software study (manuals, mostly) and the interviews. During the interviews documents were often used to provide a foundation for discussion. Typical documents were implementation manuals, deployment manuals, development instructions, and implementation/sales presentations.

2.1.2 Software Study

An easy way to study how software is delivered and deployed is by actually installing the product on a workstation. For five of the cases (the sixth is a large hospital information system) we actually installed the products locally. This type of research provides deep insight into the deployment process (dependencies, minimum requirements, etc.) and more importantly proved to the interviewees that the researcher had sufficient knowledge about the delivery, licensing, and deployment processes. The software vendors freely provided licenses for research purposes.

2.1.3 Interviews

Interviews consisted of two sessions of approximately one hour. The first interview consisted of three parts. First, the interviewee was interviewed for approximately 15 minutes with predefined questions. After the short round a longer round of 45 minutes followed, during which the researcher and the interviewee would discuss different aspects of the CCU process. During the last 5 minutes time was taken to see who else was of importance to the researcher. Approximately one week after the first round with an interviewee, a second interview was arranged to double check the facts distilled from different sources, such as software study, documentation study, and material from other interviews. We found the two round system powerful, especially when sources seemed to disagree. After gathering a new list of potential interviewees the researcher would generally call or visit that person to arrange a new interview. In about ten percent of the interviews the conversation was ended prematurely, mostly because the person did have any knowledge of interest to the researcher. Special forms were created for the interviews on which the researcher could make notes and write down standard answers, such as the person's name and job position. Furthermore, the interview notes were written up within 24 hours after doing the interview (mostly immediately after the interviews) (Walsham, 1993).

We found that e-mail discussions are only useful before interviews. If an e-mail discussion becomes too detailed it is best to call or meet that person, even after the interviews have been completed. We also noticed that a good researcher brings a notebook and pen to lunch.

2.2 Data Analysis

During the process of analyzing the data gathered through software study, document study, and the interviews we found that a concrete scope, triangulation, and a strong belief in the CCU model were most necessary to create valuable deductions in the case study reports. During the first case study the scope of the research had only been vaguely defined, leading to a case study project without boundaries. By re-evaluating the scope of the case study during the project a final list of to do items could be made, enabling the researcher to define when enough data had been gathered. Scoping enabled us to clearly define the boundaries of the research, especially after seeing what results were interesting to the community at that time.

One large part of data analysis was theory shaping, where the different descriptions and facts were modeled in different process diagrams. By triangulating these diagrams with data from interviews, software, and internal documents these models were changed until they represented the practice of an organization fully. This method of data analysis is described by others as constant comparative analysis (Hewitt-Taylor, 2001).

It was hard to identify weaknesses within the product software vendor's practices without having a clear view of what we considered mature CCU. As the project progressed more knowledge became available about good and bad CCU practices. As the researchers became more knowledgeable on the subject the quality of our data analysis improved. We found the length of the first case study contributing in this aspect, because it enabled us to explore the case study practice.

2.3 Validity

There exist four validity criteria for emperical research (Yin, 2003; Flyvbjerg, 2006; Eisenhardt, 1989) and we have attempted to satisfy all. They are the following:

Construct Validity - Construct validity is satisfied when concepts being studied are operationalized and measured correctly. A glossary of terms was created before actually starting the multi-case research. Furthermore, we attempted to identify and compare different constructs within the case. Also, the same protocol was applied to each case study. To gain correct and consistent facts for the case study database, we did double interviews and cross checked our results from the document and software study with these. Periodic peer review of the case study database ensured that the right information was being collected and stored. Finally, to minimize personal bias the case study reports were reviewed by fellow project members, the researcher was required to stay at the site, and the researcher had to inform participants of the exact aims of the project.

Internal Validity - Internal validity is defined as establishing a causal relationship and distinguishing spurious relationships. To make sure we correctly extracted facts from the case study database the case study report was reviewed extensively by informed outsiders, the project champion, and one other employee from the product software vendor. We mostly used explanation building for establishing causal relationships (Stake, 2005). To make sure the facts and relationships in the case study reports were complete, towards the end of the case study we would formally invite the project champion to make suggestions to complete and prioritize lists, process overviews, and the list of improvements.

External Validity - External validity is defined as establishing the domain to which a study's findings can be generalized. The cases are found to be representative for the Dutch product software vendor market domain because the vendors have come from a list of the Platform for Productsoftware[2], an organization that aims to share knowledge between research institutes and product software vendors in The Netherlands, with over 100 members. The list shows that the cases are typical for the Dutch software industry with respect to size, number of customers, and income.

Empirical Reliability - Empirical reliability is defined as the ability to demonstrate that the study can be repeated with the same results. A lot of work has been put into defining the procedures of the research (the case study protocol), the case study databases, and the case study report. We expect that the cases and the results can be repeated if the circumstances are at least similar (same interviewees, same documents, etc).

We found these validity criteria guiding beacons in designing the research and would advise others to discuss them frequently during the progress of the research.

3 The Case Study Protocol, Database, and Report

In this section the case study protocols, databases, and reports are described in detail.

[2]http://www.productsoftware.nl/

3.1 The Protocol

The case study protocol serves three purposes. First, the aim of the case study protocol is to clearly define the aims of the case study as to avoid confusion and conflict in the future. Secondly, we use the protocol to convince the participants of the usefulness of the research. Thirdly, the case study protocol is a useful document to instruct different researchers at different sites and enabled us to reuse the results from the different cases. The protocol has been specifically adjusted for each case but only to change the company details in each. The peer-reviewer and the researcher must agree on the protocol during the initiation process. This document is kept deliberately short, to enable product software vendors to quickly decide whether or not to cooperate.

The document consists of four parts, the introduction, the description of the research, the description of this specific case, the conclusions, and a summary and related work of the case study. In the first two parts the research was described non-specific to the product software vendor. The research description consists of a description of the different participants, the conceptual model and terminology used, the CCU model on which this research is based, the research questions, the applied research methods, and potential participants. In the third part was explained how we would go about our research and what confidentiality aspects would be taken into account. In the fourth part the research is summarized and related work is provided.

The document, generally no longer than 7 pages, needed to take away any doubts that we would be wasting the product software vendor's time. To do so the research aim is stated in short, a short description is given of the overall research, and a lot of attention is given to the description of the CCU model. We also note that product software vendors spent the most time over the list of potential participants (in the form of job titles, since we had no inside information on the company itself yet) and the list of used research methods (software study, document study, and interviews). Some of the job titles are *configuration manager, developer, supportdesk employee, and development manager*. The document was used as a checklist once the project was ending to determine whether points had been missed.

3.2 The Database

At the end of each of the case studies the researcher was responsible to deliver his case study database. The case study database, as depicted in Figure 5, was structured similarly for each case and consisted of four parts. During the case study documents and materials were obtained from the product software vendor on which the researcher had had no influence. These were either available on paper or available digitally. The third and fourth parts of the case study database consist of materials created by the researcher, such as interview and observation notes and screenshots and -captures. Because the case study databases never grew over approximately 150 items the documents were kept in two digital folders and two binders. We had the largest problem storing e-mails and digital discussions, due to the many different formats in which they were delivered. We also found that meticulous record keeping (Benbasat, Goldstein & Mead, 1987) is essential for storing items in the case study databases. Due to the fact that the researchers are not at their normal working place and were travelling by public transport or car documents were indeed lost. Also, due to insufficient record keeping a lot of duplication was found in the databases, especially among the digital folders and paper binders.

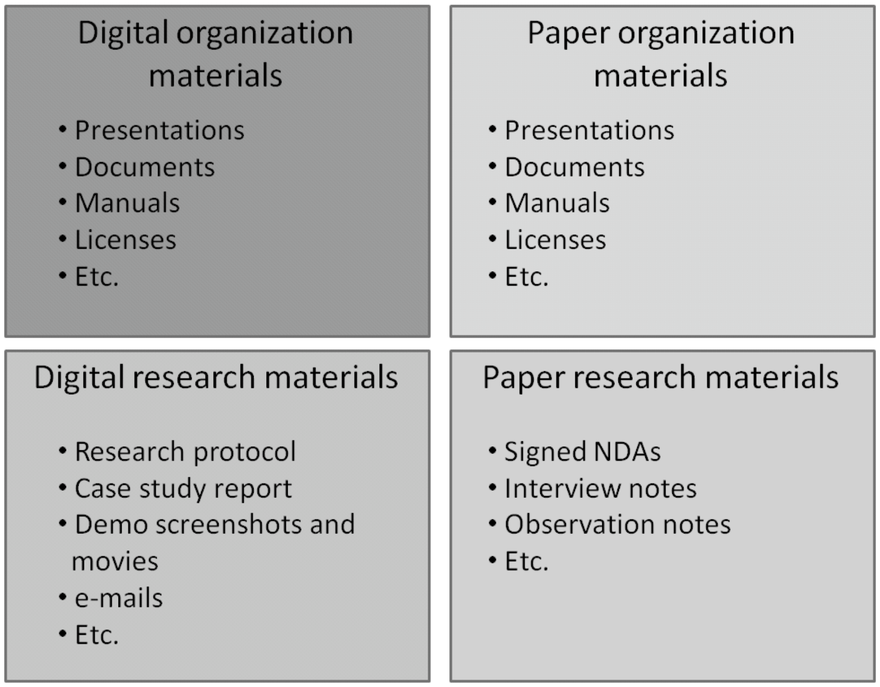
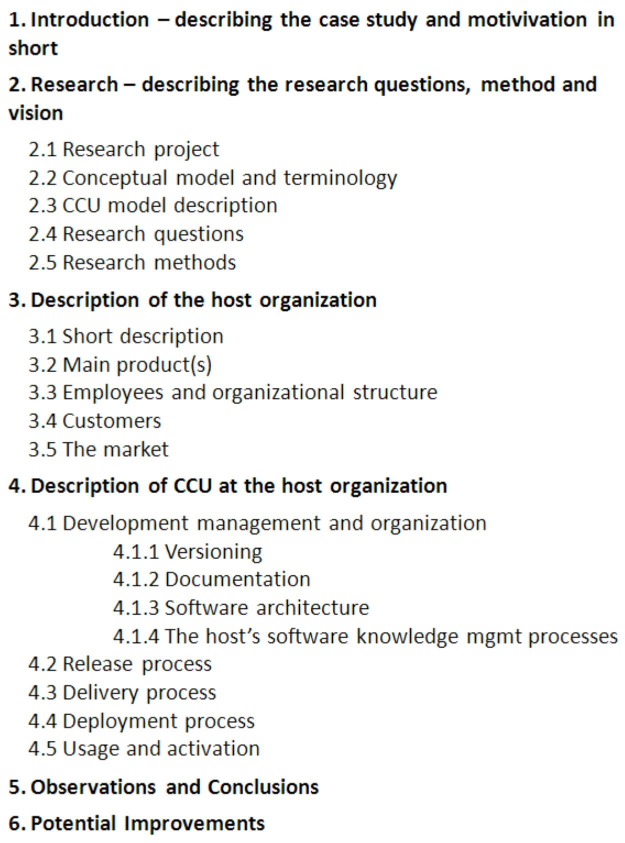


Figure 5: Case Study Database

3.3 The Report

Please see Figure 6 for the template outline of a case study report. The aim of the case study report is to provide future researchers with concise material about the case studies done. The case study reports, varying from 18 to 50 pages, all adhere to this outline, thus using a uniform method to present the varying results and observations of the different researchers.



1. Introduction – describing the case study and motivivation in short

2. Research – describing the research questions, method and vision

- 2.1 Research project
- 2.2 Conceptual model and terminology
- 2.3 CCU model description
- 2.4 Research questions
- 2.5 Research methods

3. Description of the host organization

- 3.1 Short description
- 3.2 Main product(s)
- 3.3 Employees and organizational structure
- 3.4 Customers
- 3.5 The market

4. Description of CCU at the host organization

- 4.1 Development management and organization
  - 4.1.1 Versioning
  - 4.1.2 Documentation
  - 4.1.3 Software architecture
  - 4.1.4 The host's software knowledge mgmt processes
- 4.2 Release process
- 4.3 Delivery process
- 4.4 Deployment process
- 4.5 Usage and activation

5. Observations and Conclusions

6. Potential Improvements

Figure 6: Case Study Report Outline

4 Conducting the Research

We found that the research took approximately 24 weeks per case study. Of these 24 weeks, only a small part was done at the companies (see table 1). The other weeks we were waiting for responses from software vendors or writing up reports and processing material. We found that it generally took product software vendors one week to send an answer to the invitations. It then took another week for them to respond to our formal invitation letter by mail. Putting the NDA together took between one and four weeks. After the NDA had been signed we generally gained access immediately to the product software vendor's resources. The research itself took between three and ten weeks and became progressively shorter as we gained more experience. Once the first phase of research had been done the writing of the case study report took an average of two to four weeks. Getting it approved, however, took between one and 10 weeks, depending on the quality of the report, the sensitivity of the data, and the detail of process descriptions. In general we would present our results to the case study participants approximately two weeks after the case study report had been approved.

Table 1: Case Studies Overview

Identification Code	**Time**	**Formal interviewees**	**Informal interviewees**	**Organization size**	**Duration at the host**
ERPComp	Early 2004	15	24	1504	10 weeks
OCSComp	Early 2005	4	8	114	7 weeks
HISCComp	Mid 2005	7	12	107	6 weeks
FMSComp	Late 2005	8	8	164	4 weeks
CMSComp	Early 2006	4	8	65	3 weeks
TDSComp	Mid 2006	4	5	62	3 weeks

4.1 Finding the best Fitting Case

A strong point of this research was the selection procedure for case study participants. This research project was supported by the Platform for Productsoftware, providing us with an excellent case selection source. At first the project was mostly opportunity driven where we would grasp any organization prepared to share their experience. As the results of our research were spread, however, we were soon invited by a larger number of product software vendors than we could handle. This enabled us to pick software vendors with different products, practices, and technologies. Furthermore, it enabled us to pick software vendors who had very mature processes and practices (Jansen, Ballintijn, Brinkkemper, & van Nieuwland, 2006) and others who were actually looking for improvements in the area of CCU (Jansen, 2006a). Screening is a powerful tool when selecting a product software vendor and we recommend being selective and avoiding any organization that is undergoing rapid change. Finally, we noticed that once your foot is in the door of these organizations, they will be more receptive to new initiatives. With four of the six companies we dealt with we have continued our research in the form of research grants, outplacing master students, case studies in other fields, etc.

4.2 The Publications

The work led to a number of publications. The best case was the research conducted at a large bookkeeping product software firm, who had already implemented many different aspects of the CCU model. The research conducted there led to two papers being accepted at a leading conference (Jansen, Brinkkemper, Ballintijn & van Nieuwland, 2005) and in a journal (Jansen, Ballintijn, Brinkkemper, & van Nieuwland, 2006). The research from the third case study was published in the industrial forum of a large conference (Ballintijn, 2007). Finally, an overview paper of all the cases was published at a leading conference by Jansen & Brinkkemper (2007).

4.3 From Protocol to Theory

The multi-case research was conducted to determine the state of the practice in CCU, to define the scope of CCU, and to contribute new findings to this field. The underlying hypothesis was that proper knowledge management over the distributed locations where a product is being used and developed (at the vendor, end-user, system administrator, etc.) can reduce deployment failures and enable quicker fault location in a software product.

To find new contributions, four methods were used. First, we formulated long lists of hypotheses, stating such things as “a software vendor cannot support more than 1000 end-users without tool support for updates and deployments”, “software vendors are unaware of the hardware that is used by customers” and “vendors with more than 100 customers automatically release their software”. Secondly, we performed cross-case comparison, taking different CCU processes and trying to abstract similarities and differences between the cases (Jansen & Brinkkemper, 2007). Also, we labeled findings in the case study databases in relation to the formulated theories. Finally, a tool evaluation framework was created (Jansen, Brinkkemper & Ballintijn, 2005) and applied to the various tools found during the cases, to see whether and how these tools supported the processes.

The processes defined in the CCU model were validated (proved to be similar to those of each subject of study) by the case studies, for the first time providing us with a validated overview of the CCU processes and practices, i.e., all the processes involved with knowledge creation and distribution around a software product. The CCU model enabled the linking of each process to best practices from the different cases. Finally, those hypotheses that had not yet been accepted or refuted were processed (where possible) by comparing and contrasting the different case studies.

4.4 Findings

During the case studies we found a number of factors contributing positively (and in some cases even essential) to the end results of the case studies.

To begin with, **the elevator pitch is your introduction to the company**. Especially the first couple of days you will have to explain why you are visiting the company. If you can clearly and shortly convey the message of why you are there to random people, we found it was much easier getting access to unexpected sources. In one case it even lead us to the packaging department where only a couple of years ago batches of up to 20 floppy's were formatted, printed, wrapped and sent off to customers at 5 packages per minute.

A good relation with the project champion is essential for a successful case study. We noticed that the success of the case studies was greatly dependent on a project champion within the company who was responsible for the success of the case study within the product software vendor. For each case study performed we noticed that the project champion would perform as an intermediary between the vendor's organization and the research institute. Furthermore the project champion would reconfirm the importance of the research to interviewees and remove information blockades. During the case studies there were six project champions guiding us through the process of conducting the research at their product software vendors. Among these six were five development managers and one product manager. These people were generally hands-on professionals who had spent between five and ten years at the company. We found that the project champions had to support the project above all else. Their motivation to approve of our research was mostly that we would propose improvements to their processes and compare them to those of others. We also found that project champions, after signing the NDA, gave complete openness and were not afraid to show weaknesses of the organization.

After the cases were complete we **organized meetings for practitioners to report on the results of our research**. A working group was started that meets four times per year to share experiences and knowledge about the CCU processes. The companies are willing to share their knowledge of these processes because their strategic value lies in the fact that they build great bookkeeping software or content management systems (for example) and not in their development and CCU processes. Currently two of the product software vendors still participate in the working group, and we maintain strong ties with the other four.

4.5 CCU Conclusions and End Products

Here we give examples to the reader of some of the contributions of the research, to show the types of deliverables that can come from multi-case research. The research has been undertaken with two main research questions in mind: "What are the concepts and is the state of affairs of customer configuration updating for product software? " and "Can customer configuration updating be improved by explicitly managing and sharing knowledge about software products? ".

The first research question was answered providing an overview of the current state of the art (research), and the practice (multi-case project). Also, the CCU model has been created, providing practitioners with a method to evaluate its processes, tools, and practices. The second research question was answered by stating the methods used by product software vendors to explicitly manage and share knowledge. Furthermore, a tool was developed (currently under validation at one of the case study participants) that enables a product software vendor to share all types of knowledge in a software supply network with its customers (Jansen, 2007b). Besides the tools, CCU model, and publications, one of the main deliverables of this research was a PhD thesis (Jansen, 2007a).

5 Discussion

As can be seen in table 1 the duration of the case study projects became progressively shorter. As the experience of the researchers grew, so did their ability to quickly get to the point. Unfortunately, also did their sloppiness. We found that towards the final two cases the study had become a fairly straightforward assignment in filling in the standard structure for case study reports found in Figure 6. This did, however, lead to longer periods of discussion with the peer reviewer (who was essential for the quality of this process) and the project champion.

Some of these discussions were extensive and lengthy. The NDAs stated that we would not publish anything without the product software vendor's explicit permission. In one case we were given the choice to remove sensitive details or to fully anonymize the report. Obviously we chose to anonymize the report, but that lead to other problems. The company is quite unique in the Netherlands and already the report stated that the vendor was a member of the 100 members of the Platform for Product Software, enabling further identification (and even if we had removed that fact the informed reader would know, since we are the founders of the Platform). Fortunately, as the discussion went into its sixth week, the project champion gave up and approved the report. In the future we would not change the NDA, however, since the vendor's approval of all outgoing publications was a key selling point for the case studies.

To deal with the issue of self-selection, where the results from the cases are influenced by the fact that vendors respond to an invitation and thus find the research questions relevant, we took two measures. First, the invitation stated explicitly that we were looking for both product software vendors who found their own processes mature and those who really wished to improve their CCU processes. The second measure is that we actively approached two software vendors who initially had not replied to the invitational e-mail.

During the research a list of hypotheses was formulated. Not all hypotheses were accepted or refuted using the multi-case project. A large part could only be processed with the results from the survey, and some challenges still remain, such as determining the "ideal" moment for a product release (encompassing both CCU and requirements engineering), applying the CCU model to software vendors selling their applications in a web service model, etc.

According to Eisenhardt (1998) four to ten cases appear to be a sufficient amount for theory building. However arbitrary this suggestion may sound, the presented six cases fall into this range nicely.

5.1 Case Studies in Mixed-Method Research

The multi-case study has been the larger part of a mixed-method research project into the CCU practices of software vendors. The research furthermore consists of prototype tool building (Jansen, 2007b; van der Storm, 2007) and a survey done at the beginning of 2007. Gable (1994) states that mixed-method research is not new, but underappreciated in the IS community. Some journals even specifically target one type of research, encouraging further polarization of research methods. In Klein's five fundamental attitudes towards research methods we identify ourselves as pluralists, i.e., we believe that different approaches can be brought to bear on the same problem domain and that there exists no single universally valid way to delineate objects of study or to match the strengths and weaknesses of research approaches with contingent features of the object of study (Klein, Nissen & Hirschheim, 1991).

While using mixed-methods to establish empirical evidence for theories within our model, we noticed that case studies were responsible for the larger part of the success of our research. These cases confirmed hypotheses, provided anecdotal evidence and counterarguments, and strengthened the applicability of the research work in practice. Furthermore, the multi-case study project uncovered problems of interest to the scientific community. The case studies certainly delivered in that respect and we hope to validate the claims about our prototype tools in production environments of the case study participants. Furthermore, the case study subjects were brutally honest about limitations to and feasibility of the CCU model. On the other hand the case studies were lacking in two areas. The feasibility of certain tools and protocols within the model could never be proven until a prototype was built. Secondly, it was hard to generalize conclusions from the case studies to other organizations. This research was undertaken as mixed-method project, where different hypotheses were tested using different methods. This is different from method triangulation, where one hypothesis is tested using different methods.

Due to the limited use of rival hypotheses and the fact that we were studying a relatively small subset of software vendors to validate the CCU improvement model, we encountered problems in generalizing our conclusions (Gable, 1994). To further validate our research, a benchmark survey was launched in February of 2007. The benchmark survey takes a "double survey" approach, where the results from the first survey are published in a report (including improvement advice) to the submitter. A second survey is attached to the report to see whether it was of use to the software vendor. The cases were the source for all improvement advice given in the survey reports. Both the advice and the reports were deemed useful and informative by the survey submitters. We find the use of different methods extremely useful to test hypotheses and become experts in the field of CCU. Furthermore, the use of mixed-method research has enabled us to generalize the conclusions from our multi-case research.

5.2 Multiple vs Single Case Study Research

The research project was always intended to consist mostly of multi-case research. There are three reasons for this. To begin with, we wanted to find out the current state of the practice in CCU, not the one company that does things differently (Yin, 2003) calls these “black sheep” cases). Secondly, we needed a solid base to launch other research from, such as surveys. This was reached through multi-case research, which is considered to deliver more “robust” results than a single case study (Herriott & Firestone, 1983). Finally, we needed measurable constructs and a conceptualized overview of the CCU research field. Such constructs had hardly been defined in previous research by others (Jansen, 2007a), enabling us to take a green fields approach. In a single case study we potentially had not been able to define constructs as well, simply because one study would not have provided a full overview of the different concepts.

In practice the different case studies unearthed much more than a single one would have. The multiple cases enabled the discovery of one particular exotic case (Jansen, Ballintijn, Brinkkemper, & van Nieuwland, 2006), more theory building, and the definition of similarities and differences between the different CCU processes of the organizations. The exotic case is described further in section 5.3. Some of our early hunches, such as the fact that the number of customers a vendor has is detrimental to the process of license management, could only be confirmed by looking at multiple case studies. The same holds for product complexity: a more complex product leads software vendors to having a more elaborate CCU process with more manual steps, leading to our conclusion that such complexity comes at a price.

The identification of differences and similarities between the cases enabled us to define processes and best practices that are standard in regards to the CCU process for software vendors. For example, all companies use the major, minor, and bugfix release model, where major releases happen every one to three years, minor releases happen every six months to a year, and bug fix releases are released anywhere between one per day to one per six months. Another find was that all cases used a release scenario to make sure that on the day of a release the product was released in full without any missing parts (documentation, languages, libraries, etc.) An interesting sidebar to this find is that later in the survey we uncovered that this holds only for those organizations with more than 5 employees (all organizations from the cases had 62 employees or more). Similarly, differences were found between the cases. Some of the organizations had strict well-planned release schedules set up to map out the different versions that would be released over the following 3 year period, whereas others had vague targets and flexible dates. Also, when it comes to license management of their software products all vendors used different mechanisms and support tools. One final reason to do multi-case research was that we wished to give something back to the organizations from previous experiences and from some of the research on CCU in academia, in the form of advice and consultancy.

Yin states that one should try to avoid to put “all eggs in one basket”, i.e., one should always attempt to get more case studies or be ready to defend the choice for that specific case fiercely. The one downside that we did find in this research was that a lot of work is involved in doing six case studies. If one is going to do such research, our advice is to distribute the work over different researchers. Also, in the future, we hope to combine different forms of research to get more data from one organization in different research fields. Finally, the selection of the cases deserves at least as much attention as with one case because a case study in a multi-case project must at all times contribute to the data. See section 5.4 for an example of a case that contributed only marginally to our work.

5.3 The Best Case

In 2003, when this research started, there were no reports on how the industry performed in the area of CCU. To establish whether there were new and innovative practices in this area an invitation was sent out for case study participants with either a mature or immature process. One organization, Exact Software, contacted us immediately. They believed their practices were innovative and industry leading. They had, for example, their own updater that kept customers up to date, maintained a customer base of 160.000 with low overhead, and had several product lines. The last founder of the company, that presently has 2644 employees, was leaving the organization and wished to contribute to the research community. This put the organization in an extraordinary position, where in the past the company was well known to reject any type of scientific cooperation, the company was now open for our research.

Within 5 weeks our researcher started at Exact to study the processes. The product manager at that time facilitated the research completely. The researcher obtained a login id for the intranet from any location and started work at the development department in Delft, the Netherlands. The case study did bring forward new and innovative ideas and these results were published in two papers (Jansen, Ballintijn, Brinkkemper, & van Nieuwland, 2006; Jansen, Brinkkemper, Ballintijn & van Nieuwland, 2005) (the founder is a co-author of one of these). The case study was a success for three reasons. First, the company was undergoing changes and was open to research and new ideas. Secondly, the company displayed novel phenomena worth reporting. Finally, the company wished to provide openness to the outside world making the publication process mucheasier.

5.4 The Worst Case

In 2004 a software vendor providing an on-line statistics package responded to our invitation for case studies. The vendor had three products, with 100.000, 1000, and 100 customers respectively. One of these products was a pure software product with a full delivery and deployment process and was of interest to our research. The other two products were provided as an on-line service and were thus less interesting. The management of the organization was undergoing changes, but the case study champion proved to be a reliable character with strong interests in our research. At face value we approved of the case study and would start, due to the fact that the research team was busy, in 2005.

When the case study began all seemed well. Unfortunately the smallest product (100 customers) was the product of interest, which we only discovered in 2005. Furthermore, the project champion was away a lot, making it hard to get access to all sources. As the case study progressed the product with the largest amount of customers was sold to another company and the project champion left the company. These changes made the employees wary of changes and thus were reluctant to participate in a case study. Fortunately the project champion was replaced by a new equally competent project champion, but the results proved to be only marginally relevant. These problems could have been avoided by better screening of the organization before the initiation of the case study.

5.5 Case Studies in Education, Industry, and Research

The case studies undertaken contribute to our educational program, to our industrial partners, and to our research initiatives. In the research methods course for our information systems students the multi-case example is discussed in one out of twenty two hour lectures. Many of our students do projects with industrial partners and we find that case study education is essential for these students. Case study education has proven fruitful, since many of the student projects have lead to publications (Brinkkemper, Soest & Jansen, 2007; Bakhshi, Versendaal, van den Berg & van den Ende, 2007).

The multi-case research project would have been much harder without the Platform for Productsoftware. The Platform enabled us to handpick case study participants, organize expert meetings, and the Platform provided us with an outlet to the industry when there were best practice findings. The research we have undertaken in the multi-case project has created credibility with the software vendors in the Platform. The credibility has lead to software vendors approaching us when encountering fundamental problems, creating long-lasting relations between our research group and some of the industrial parties that participated in the case studies.

With regards to research, the multi-case research project now serves as an example to new PhD students and starting researchers in our research group. We advise new PhD students to start with a practical case study, as to gain an understanding of the problems in industry, compared to the problems and challenges these students gather from literature. We caution them however, that what is interesting to an organization is frequently not aligned with our interests.

6 Conclusions

Case studies are a useful tool for IS and technology (IS/IT) research and improvement projects. More and more of our students are joining large consultancy firms after they obtain their master’s degree. The use of case studies will improve a student’s value in research and in the consultancy industry. We therefore encourage educational institutes to put case studies on the curriculum as one of the main drivers of IS/IT research. Case studies can also be of vital importance in new (PhD) research projects to give the researchers a feel for the problems and terminology used in the particular field of IS research.

There are three reasons for reporting on the research conducted in a multi-case study project in this chapter. First, we wish to propagate that case studies must be centrally stored to enable further theory building in the field of IS. Secondly, we want to formally report on our research methods, in a structured and fragmented fashion, to enable re-use and comments on our methods. Finally, we wish to share our experiences in case study research to improve future case study research.

7 Future Research Directions

Case study research undoubtedly contributes to the field of IS. All too often, however, case studies are published as mayflies. There should be a collaborative research database in which teaching cases and research case studies reside and can be retrieved through a simple interface. A good example is the eXperience (http://en.experience-online.ch) e-business case study database. Such databases enable IS practitioners and researchers to learn from others and to extract theories from literature research.

This work is only a first attempt at formally documenting research methods. In the future we hope to contribute to the method base in other forms, such that IS researchers can develop a research method toolkit. We found the mixed-method approach for this research successful and hope to repeat it in different projects to further evaluate mixed-method research.

8 Additional Readings

For the beginning case study scientist we recommend reading Yin's (2003) case study method and guidelines. Common fallacies and pitfalls are described in regards to information system case studies in Kitchenham, Pickard and Pfleeger (1995) and Flyvbjerg (2006). Describing research with method engineering techniques enables re-use and discussion about research methods. This requires comprehensive descriptions (van de Weerd, Brinkkemper, Souer & Versendaal, 2006) of method engineering techniques. In regards to CCU we recommend reading more on the benchmark survey for Dutch product software vendors (Jansen & Brinkkemper, 2008) and the changing product software industry (Brinkkemper, Soest & Jansen, 2007).

References

Bakhshi, N., Versendaal, J., van den Berg, J., & van den Ende, D. (2007) Impact of the extensible business reporting language on the administrative burden of organizations: A case study at the Dutch water boards. In *Proceedings of the 4th International Conference on Enterprise Systems, Accounting & Logistics*.

Ballintijn, G. (2007). A case study of the release management of a health-care information system. In *proceedings of the IEEE International Conference on Software Maintenance, ICSM2005, Industrial Applications track*.

Benbasat, I., Goldstein, D. K., & Mead, M. (1987). The case research strategy in studies of information systems. *MIS Q.*, 11(3):369–386.

Brinkkemper, S., van Soest, I. & Jansen, S. (2007). Modeling of product software businesses: Investigation into industry product and channel typologies. In *The Inter-Networked World: ISD Theory, Practice, and Education, proceedings of the Sixteenth International Conference on Information Systems Development (ISD 2007)*. Springer-verlag.

Darke, P., Shanks, G. G., & Broadbent, M. (1998). Successfully completing case study research: combining rigour, relevance and pragmatism. *Information Systems Journal*, 8(4):273–290.

Eisenhardt, K. M. (1989). Building theories from case study research. *Academy of Management Review*, 14:532–550.

Flyvbjerg, B. (2006). Five Misunderstandings About Case-Study Research. *Qualitative Inquiry*, 12(2):219.

Gable, G. G. (1994). Integrating case study and survey research methods: an example in information systems. *European Journal of Information Systems*, 3(2):112–126.

Herriott, R. E. & Firestone, W. A. (1983). Multisite qualitative policy research: Optimizing description and generalizability. *Educational Researcher*, (12):14–19.

Hewitt-Taylor, J. (2001) Use of constant comparative analysis in qualitative research, *Nursing Standard*. 15, 42: 39-42.

Jansen, S. (2005). Software Release and Deployment at Planon: a case study report. In *Technical Report SEN-E0504*. CWI.

Jansen, S. (2006). Software release and deployment at a content management systems vendor: a case study report. Institute of Computing and Information Sciences, Utrecht University, Technical report UU-CS-2006-044.

Jansen, S. (2006). Software release and deployment at Stabiplan: a case study report. Institute of Computing and Information Sciences, Utrecht University, Technical report UU-CS-2006-041.

Jansen, S. (2007). *Customer configuration updating in a software supply network*. PhD Thesis, Utrecht University.

Jansen, S. (2007). Pheme: An infrastructure to enable any type of communication between a software vendor and an end-user. In *International Conference on Software Maintenance 2007, tool demonstration*.

Jansen, S., Ballintijn, G., & Brinkkemper, S. (2004). Software release and deployment at exact: a case study report. In *technical report SEN-E0414*. CWI.

Jansen, S., Ballintijn, G., Brinkkemper, S., & van Nieuwland, A. (2006). Integrated development and maintenance for the release, delivery, deployment, and customization of product software: a case study in mass-market ERP software. In *Journal of Software Maintenance and Evolution: Research and Practice*, volume 18, pages 133–151. John Wiley & Sons, Ltd.

Jansen, S., & Brinkkemper, S. (2007). Definition and validation of the key process areas of release, delivery and deployment of product software vendors: turning the ugly duckling into a swan. In *proceedings of the International Conference on Software Maintenance (ICSM2006, Research track)*.

Jansen, S., & Brinkkemper, S. (2008). A benchmark survey into the customer configuration processes and practices of product software vendors in the Netherlands. In *proceedings of the 7th IEEE International Conference on Composition-Based Software Systems (ICCBSS)*. IEEE.

Jansen, S., Brinkkemper, S., & Ballintijn, G. (2005). A process framework and typology for software product updaters. In *Ninth European Conference on Software Maintenance and Reengineering*, pages 265–274. IEEE.

Jansen, S., Brinkkemper, S., Ballintijn, G., & van Nieuwland, A. (2005). Integrated development and maintenance of software products to support efficient updating of customer configurations: A case study in mass market ERP software. In *Proceedings of the 21st International Conference on Software Maintenance*. IEEE.

Jansen, S., & Rijsemus, W. (2006). Balancing total cost of ownership and cost of maintenance within a software supply network. In *proceedings of the IEEE International Conference on Software Maintenance, Philadelphia, PA, USA, September*.

Kitchenham, B., Pickard, L., & Pfleeger, S. L. (1995). Case studies for method and tool evaluation. *IEEE Software.*, 12(4):52–62.

Klein, H. K., Nissen, H.-E., & Hirschheim, R. (1991). A pluralist perspective of the information systems arena. *Information Systems Reserch: Contemporary Approaches and Emergent Traditions*, pages 1–20.

Orlikowski, W. J., & Baroudi, J. J. (1991). Studying information technology in organizations. *Information Systems Research*, 2:1–28.

Radford, M. L., & Kern, M. K. (2006). A multiple-case study investigation of the discontinuation of nine chat reference services. *Library and Information Science Research*, 28(4):521–547.

Roberts, D., Cater-Steel, A., & Toleman, M. (2006). Factors influencing the decisions of SMEs to purchase software package upgrades. In *Proceedings of the 17th Australasian Conference on Information Systems (ACIS 2006)*.

Saeki, M. (1994). Software specification and design methods and method engineering, *International Journal of Software Engineering and Knowledge Engineering*.

Stake, R. E. (2005). *Multiple Case Study Analysis*. The Guilford Press.

van de Weerd, I., Brinkkemper, S., Souer, J., & Versendaal, J. (2006). A situational implementation method for web-based content management system-applications: method engineering and validation in practice. *Software Process: Improvement and Practice*, 11(5):521–538.

van de Weerd, I., Brinkkemper, S., & Versendaal, J. (2007). Concepts for incremental method evolution: Empirical exploration and validation in requirements management. In *Proc. of the 19th Conference on Advanced Information Systems Engineering*.

van der Storm, T., (2007). The Sisyphus continuous integration system. In *Proceedings of the Conference on Software Maintenance and Reengineering*. IEEE Computer Society Press.

Walsham, G. (1993). *Interpreting Information Systems in Organizations*. John Wiley & Sons, Inc., New York, NY, USA.

Xu, L., & Brinkkemper, S. (2005). Concepts of product software: Paving the road for urgently needed research. In *First International Workshop on Philosophical Foundations of Information Systems Engineering*. LNCS, Springer-Verlag.

Yin, R. K. (2003). *Case study research - design and methods*. SAGE Publications, 3rd ed.