VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wānanga o te Ūpoko o te Ika a Māui

School of Mathematical and Computing Sciences

Computer Science

A CASE STUDY: EXPLORING THE ROLE OF CUSTOMERS ON EXTREME PROGRAMMING PROJECTS.

Angela Martin

TECHNICAL REPORT CS-TR-03-1

January 2003

School of Mathematical and Computing Sciences
Victoria University of Wellington
PO Box 600, Wellington
New Zealand

Tel +64-4-463 5666
Fax +64-4-463 5045
Email Tech.Reports@mcs.vuw.ac.nz
http://www.mcs.vuw.ac.nz/research

VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wānanga o te Ūpoko o te Ika a Māui

School of Mathematical and Computing Sciences

Computer Science

PO Box 600
Wellington
New Zealand

Tel +64-4-463 5666, Fax +64-4-463 5045
Email Tech.Reports@mcs.vuw.ac.nz
http://www.mcs.vuw.ac.nz/research

A CASE STUDY: EXPLORING THE ROLE OF CUSTOMERS ON EXTREME PROGRAMMING PROJECTS.

Angela Martin

TECHNICAL REPORT CS-TR-03-1

January 2003

ABSTRACT

eXtreme programming (XP) is one of a new breed of methods, collectively known as the *agile or light* methods, that are challenging conventional wisdom regarding systems development processes and practices. The agile methods are specifically designed, by practitioners, to meet the business problems and challenges we face building software today in the fast-paced dynamic world of the Internet. As such these methods are receiving significant attention in practitioner literature. In order to effectively operate in the world of vague and changing requirements, XP moves the emphasis away from prescriptive processes into practices that enable people.

We used an interpretative in-depth case study to explore a successful XP Project. We obtained multiple perspectives on the implementation of the customer role within the planning process of XP and found the following. Firstly, the XP customer role, especially for larger organisations, is a demanding role. It requires preparation, skills, attention to detail, and the ability to make critical decisions. Secondly, obtaining regular feedback during the project allows the customer to make effective business decisions concerning the system. Finally, the development team must carry out key XP practices in order to enable feedback necessary for the customer to make effective decisions.

A CASE STUDY: EXPLORING THE ROLE OF CUSTOMERS ON EXTREME PROGRAMMING PROJECTS.

A research report
by
Angela M Martin

School of Communications and Information Management
Faculty of Commerce and Administration
Victoria University of Wellington
15 October 2002

ACKNOWLEDGEMENTS

I would like to acknowledge the assistance of a number of people, each of whom played an important role in the completion of this report:

- Dr Robert Biddle & Dr James Noble, my supervisors, for their contributions and encouragement throughout this study
- The participants at DevCorp and KiwiCorp for their time and insights into their XP project

Thank you.

TABLE OF CONTENTS

1. EXECUTIVE SUMMARY 6

2. INTRODUCTION 7
   - 2.1. THE PROBLEM 7
   - 2.2. THE CONTEXT 7
   - 2.3. THE PROPOSED SOLUTION 8
   - 2.4. STRUCTURE 8
   - 2.5. CONVENTIONS 9

3. XP – AN INTRODUCTION 10
   - 3.1. DEFINITION OF XP TERMS 12

4. REVIEW OF THE LITERATURE 13
   - 4.1. ISD METHODOLOGY – A GENERAL REVIEW 13
   - 4.2. USERS IN THE SOFTWARE DEVELOPMENT PROCESS 14
   - 4.3. XP CASE STUDIES 16

5. RESEARCH METHOD 18
   - 5.1. OVERALL STRATEGY AND RATIONALE 18
   - 5.2. FOCUSSING ON THE SPECIFIC SETTING 20
   - 5.3. DATA COLLECTION PROCEDURES 21
   - 5.4. DATA ANALYSIS 23
   - 5.5. VERIFICATION STEPS 24

6. DESCRIPTION OF THE CONTEXT OF THE EXPERIENCE 26
   - 6.1. THE PROJECT 26
   - 6.2. THE TEAM 27
   - 6.3. THE PROCESS 29

7. RESULTS FROM THE EXPERIENCE 36
   - 7.1. THE TEAM'S IMPRESSION OF XP 36
   - 7.2. ON-SITE CUSTOMER 36
   - 7.3. PLANNING 40
   - 7.4. AGILE MANAGEMENT 45
   - 7.5. SUMMARY OF INTERPRETATIONS 50

8. CONCLUSION 51
   - 8.1. CONTRIBUTIONS 51
   - 8.2. COMPARISONS WITH RELATED STUDIES 51
   - 8.3. FUTURE RESEARCH 52

9. REFERENCES 53

APPENDIX A: INTERVIEWEE BACKGROUND EXPERIENCES 55

APPENDIX B: DEVCORP PROPOSAL 56

APPENDIX C: PARTICIPANT CONSENT LETTER 58

APPENDIX D: INTERVIEW AGENDA 61

1. EXECUTIVE SUMMARY

eXtreme programming (XP) is one of a new breed of methods, collectively known as the *agile or light* methods, that are challenging conventional wisdom regarding systems development processes and practices. The agile methods are specifically designed, by practitioners, to meet the business problems and challenges we face building software today in the fast-paced dynamic world of the Internet. As such these methods are receiving significant attention in practitioner literature. In order to effectively operate in the world of vague and changing requirements, XP moves the emphasis away from prescriptive processes into practices that enable people.

We used an interpretative in-depth case study to explore a successful XP Project. We obtained multiple perspectives on the implementation of the customer role within the planning process of XP and found the following. Firstly, the XP customer role, especially for larger organisations, is a demanding role. It requires preparation, skills, attention to detail, and the ability to make critical decisions. Secondly, obtaining regular feedback during the project allows the customer to make effective business decisions concerning the system. Finally, the development team must carry out key XP practices in order to enable feedback necessary for the customer to make effective decisions.

2. INTRODUCTION

2.1. THE PROBLEM

eXtreme programming (XP) is one of a new breed of methods, collectively known as the *agile or light* methods, that are challenging conventional wisdom regarding systems development processes and practices. The agile methods are specifically designed, by practitioners, to meet the business problems and challenges we face building software today in the fast-paced dynamic world of the Internet. As such these methods are receiving significant attention in practitioner literature (Highsmith, 2001). In order to effectively operate in the world of vague and changing requirements, XP moves the emphasis away from prescriptive processes into practices that enable people (Beck, 2000).

One of the core roles within the XP team is the *Customer* role. Beck & Fowler (2001) describe a good customer as someone who:

- *Understands the domain well by working in that domain, and also by understanding how it works (not always the same thing)*
- *Can understand, with development's help, how software can provide business value in the domain*
- *Is determined to deliver value regularly and is not afraid to deliver too little rather than nothing*
- *Can make decisions about what's needed now and what's needed later*
- *Is willing to accept ultimate responsibility for the success or failure of the project. (p. 18)*

Practitioners (Cohn & Paul, 2001; Deursen, 2001) are questioning the practicality of implementing this role. However, the XP founders have provided little further guidance regarding this pivotal role in the XP team, but Beck and Fowler (2001) do acknowledge the risk associated with this role:

> *All the best talent and technology and process in the world will fail when the customer isn't up to scratch (p. 17)*

2.2. THE CONTEXT

Information Systems Development (ISD) methodology researchers (Fitzgerald, 2000; Nandhakumar & Avison, 1999; Russo & Stolterman, 2000) have expressed a growing concern that existing ISD methodologies and ISD methodology do not met the needs of today's business and software development environments. Studies (Baskerville & Stage, 1996; Myers & Young, 1997; Nandhakumar & Avison, 1999; Wastell, 1999) in this area have begun to explore practices in natural settings in order to begin to address these issues, however practitioners have not waited for this research, and have instead established agile or light methods, including XP.

XP relies on significant user involvement. Studies (Baskerville & Stage, 1996; Bostom & Thomas, 1983; Orlikowski & Gash, 1994; Urquhart, 1998) exist which cover the types of user and development team interaction issues and benefits proposed by this involvement.

No research of academic rigour has been undertaken that specifically addresses the use of XP on software development projects. This study begins to fill this gap by applying some of the suggestions coming from methodology researchers, including the use of rich descriptive studies that explore actual practice.

This study also specifically addresses practitioner concerns regarding the practicality of implementing the customer role.

2.3. THE PROPOSED SOLUTION

2.3.1. THE PURPOSE OF THE STUDY

The purpose of this case study is to explore the actual use of the XP method, particularly the practices utilised surrounding the Customer role, in a software development project.

The central question of this research is "How was the customer role addressed in the XP method implemented on an XP software development project in a New Zealand organisation?" The subquestions within this study focus on the issues (Creswell, 1998) surrounding the customer role within the planning practices of XP, and their development has been guided by reported practitioner issues (Beck & Fowler, 2001; Cohn & Paul, 2001; Deursen, 2001):

- The resolution of the competing needs of multiple customers represented by a single customer on the XP project
- The requirement for the customer to be on-site with the development team and the potential organisational impacts of this move

2.3.2. DELIMITATIONS AND LIMITATIONS OF THE STUDY

The scope of this study will be one XP project. The study will focus on the customer role in the XP planning processes. No attempt will be made to address other issues surrounding XP, such as the productivity of pair programming or the scalability of XP to large projects.

This proposed approach is exploratory research and the results are not expected to be generalised to all XP projects. This study, however, will empower later projects that will take a wider scope.

2.4. STRUCTURE

The report consists of the following sections:

- *Introduction to XP*. This section provides an overview of the background to XP and the key practices and roles of XP. It emphasises the planning practices over the development practices and also provides recommendations for further reading. It also provides a definition of the key XP terms used in the report.
- *Review of the Literature.* This section covers the background literature to the area under study including information systems methodology research, the users involvement in non XP software development processes and finally reviews some of the key XP studies and practitioner reports regarding the customer role.
- *Research Method.* This section outlines the research method undertaken to conduct this study including the perspective and method selected for the study, the data collection and analysis procedures and the verification steps undertaken to increase validity.

- *Description of the context of the experience.* This section describes the XP project studied including the background and structure of the project, the team and the actual processes followed.
- *Results of the experience.* This section relates the key findings and interpretations of the case. The results are divided into four key areas: the team, on-site customer, planning and agile management.
- *Conclusion.* This section summarises the report and includes future research opportunities.

The general structure of this report was guided by Creswell (1998). The case description and results sections have been influenced by the guidance to XP researchers from Marchesi et al (2002). Marchesi et al suggest the use of a specific structure and titles for XP cases. These titles have been used for the case description and results sections of this report.

2.5. CONVENTIONS

Quotes from the interviewees and photographs of the environment are used to support our description and interpreted findings. All quotes and photographs have been modified to retain the anonymity of the organisations and participants involved. Quotes have also been modified at times to enhance readability. These changes are clearly marked with square brackets.

Quotes appear as indented italicised text following the same conventions as Gittons, Hope & Williams (2002).

3. XP – AN INTRODUCTION

> *XP is the most important movement in our field today. I predict that it will be as essential to the present generation as the SEI and its Capability Maturity Model were to the last – Tom DeMarco (Beck & Fowler, 2001, p.xii)*

XP is described in a collection of books known as the XP series. The founder of XP, Kent Beck, published the initial text (Beck, 2000). In this book he proposes a new lightweight method for small to medium sized teams that he claims will allow us to deliver, and to deliver value, in the face of vague or rapidly changing requirements. He notes in this text that none of the XP practices are wholly new. Instead, the innovation of this method is turning these practices up to extreme levels, grouping them together and ensuring they are practised as thoroughly as possible. For example, code reviews have been an established software best practice for some time and the XP practice of pair programming turns this up to an extreme level. All production code is developed with two people side by side at the computer. The practice of collective code ownership is supported by "test-first" development, because changes made to the code can be verified by ensuring the tests still run correctly.

Beck (2000) notes that XP is distinguished from other methodologies by:

- *Its early, concrete and continuous feedback from short cycles.*
- *Its incremental planning approach, which quickly comes up with an overall plan that is expected to evolve through the life of the project.*
- *Its ability to flexibly schedule the implementation of functionality, responding to changing business needs.*
- *Its reliance on automated tests written by programmers and customers to monitor the progress of development, to allow the system to evolve and to catch defects early.*
- *Its reliance on oral communication, tests, and source code to communicate system structure and intent.*
- *Its reliance on an evolutionary design process that lasts as long as the system lasts. (p. xvii)*

Beck, in his first book, notes that this book is not a how-to manual. The second book in the series is of particular interest to our study in that it provides further guidance on the planning practices of XP (Beck & Fowler, 2001). Beck & Fowler emphasise the need to plan because:

> *We need to plan to ensure that we are always doing the most important thing left to do, to co-ordinate effectively with other people, and to respond quickly to unexpected events. (p. vii)*

One of the key aspects outlined by Beck & Fowler concerning the XP planning process is the clear separation of roles between the business and technical people. The business responsibilities concern the content of the system, that is what functions will be built and in what order. The technical responsibilities concern how the system will be built and how long it will take to build the system. It is also clearly recognised the two sets of decisions cannot be made in isolation, as one has a key role in determining the other. The key practices outlined to support this process are the *on-site customer, writing user stories, the planning game* and *yesterday's weather*.

The on-site customer is the person responsible for making all of the business decisions. This person is expected to know the domain well, be able to make decisions, and to be on-site with the rest of the XP team. The on-site customer is responsible for writing user stories and acceptance tests. A user story is a short paragraph description of a system requirement or feature that is:

> *… understandable to customers and developers, testable, valuable to the customer and small enough so that the programmers can build half a dozen in an iteration. …A user story is nothing more than an agreement that the customer and developers will talk together about a feature. (Beck & Fowler, 2001, p. 45-6)*

An acceptance test script proves the function works as expected by the customer.

XP projects are decomposed into small public releases of software, and within each release are iterations. Releases are small, typically two to three months. Iterations are smaller, typically two to three weeks. The *planning game* is run at both levels at different granularities. The standard process is:

- The customer prioritises the user stories[1] in business value order.
- The programmers signup to estimate the user stories they will develop. They estimate the time needed to develop the story by breaking it into tasks. Each task is estimated in ideal time[2] based on yesterday's weather. The principle of yesterdays' weather is that the best prediction of today's weather is yesterday's weather. This applies to task estimation in the following way, compare the new task to a similar task and use the actual figures from the previous task as the new task's estimate.
- The team uses the principle of yesterday's weather to determine the amount of ideal time available for this release or iteration.
- The customer confirms the final scope of the iteration or release. This step can include manipulating the stories, including breaking a story into parts, in order to ensure business value is delivered.

Once the scope is determined, the programmers are able to enter the coding phase. This phase primarily consists of discussing the story with the customer, implementing the story and ensuring the story passes all of the customer's acceptance tests. A number of XP practices support the coding phase, including metaphor, simple design, test-first coding, refactoring, pair programming, collective ownership, continuous integration, coding standards and the 40 hour week. These practices are outside the scope of this study. The initial text by Beck (2000), provides an excellent overview of these practices.

The customer is a pivotal role in an XP project and this report will discuss how the role was addressed on an XP software development project.

---

[1] Note that after the first iteration defects are included in this process as well as user stories

[2] Ideal time is the time without interruption where you can concentrate on your work and you feel fully productive. (Beck & Fowler, 2001, p. 61).

3.1. DEFINITION OF XP TERMS

The following section outlines the key terms used in this report.

Term	Definition
eXtreme Programming	XP is a lightweight methodology for small to medium sized teams developing software in the face of vague or rapidly changing requirements. (Beck, 2000, p. xv).
Customer	A role on the team for choosing what stories the system has to satisfy, what stories are needed first and what can be deferred, and for defining tests to verify the correct functioning of the system. (Beck, 2000, p. 177)
Planning Game	The XP planning process. Business gets to specify what the system needs to do. Development specifies how much each feature costs and what budget is available per day/week/month. (Beck, 2000, p. 178)
Programmer	A role on the team for someone who analyses, designs, tests, programs, and integrates. (Beck, 2000, p. 178)
User story	A user story is a short paragraph(s) description of a system requirement or feature that is understandable to customers and developers, testable, valuable to the customer and small enough so that the programmers can build half a dozen in an iteration. A user story is nothing more than an agreement that the customer and developers will talk together about a feature. (Beck & Fowler, 2001, p. 45-6)

4. REVIEW OF THE LITERATURE

This section reviews relevant literature in the area of information systems development (ISD) methodologies. The three topics covered within the area of ISD methodology research are:

- ISD methodology – a general review. A review of the recent challenges posed by researchers in this area and their suggested ways forward.
- Users in the software development process. The proposed study focuses on the customer role in the XP software development process, and this literature provides a background to this area allowing a deeper exploration of the proposed topic.
- XP case studies. A review of three relevant experiences reporting on the implementation of the customer role.

4.1. ISD METHODOLOGY – A GENERAL REVIEW

XP, is part of a new breed of methodologies that question the assumptions inherent in conventional methodologies. XP challenges conventional methodological wisdom (Beck, 2000) by emphasising:

- The need for specialist team members such as analysts, architects, programmers, testers and integrators. Team members should be generalists and able to undertake the full role of XP development team roles.
- The value of formal requirements specification documents as an effective representation of user requirements. Preferable is an on-site Customer who develops user stories (short paragraph(s) description of a system requirement). These user stories represent a promise of a conversation with the Programmer to elaborate the requirement. Short iteration cycles result in the customer reviewing these requirements in software almost immediately, enabling the requirements to be explored and refined.
- The cost of changing software does not necessarily rise dramatically over time.
- The use of best practices such as code reviews, if turned up to eXtreme levels, can result in increased productivity and quality.

Therefore, I explored articles within scholarly literature that also question conventional assumptions.

Fitzgerald (2000) discusses the need to re-examine the foundations of our system development methodologies. He uses the software development methodology literature to expose the fact that most of our concepts come from a period between 1967 and 1977. He explores the changes in the business environment and associated software development projects using a mix of literature, surveys and interviews. Of particular interest are the following trends:

- The increase of small scale rapid development projects to meet the needs of fast-paced business change(s).
- The increase of projects that do not use a methodology
- The most popular technique on projects, irrespective of whether the project uses a methodology or not, is prototyping

He concludes that in this field practice often drives theory, so one of his suggestions is that we explore the software development projects today to derive the next generation of methodologies.

Russo & Stolterman (2000) also challenge the existing research on software development methodologies. In particular they challenge the underlying assumptions and purpose of the research. They contend prescriptive methodologies rely on a number of assumptions that may not accurately reflect practice and the complexity of the software development environment. They recommend the need to create methodologies that facilitate practitioner reflection and learning. In particular, they suggest rich interpretative descriptions of practice are required to create these methodologies.

Nandhakumar & Avison (1999) use a case study to explore the application of a systems development methodology in a large scale executive information system (EIS) development project. They discover in practice that ISD methodologies are too prescriptive to be of use in the day-to-day activities of software development teams. In fact they find that the methodologies are used to present the necessary fiction of everything being under control within the project. They recommend ISD methodology researchers need to research supportive rather than prescriptive processes to assist ISD.

Wastell (1999) uses clinical research, a form of action research with multiple cases, to explore a new theoretical perspective in ISD methodology research, a learning perspective. He contents that many IS failures are due to anti-learning mechanisms that develop due to the inherently stressful nature of ISD. He suggests it is essential to create environments that are open and nurturing that will emphasize and promote learning. He suggests recasting ISD into a transitional space and to use transitional objects within this process to increase the potential for *problem solving learning* behaviors, rather than *problem avoidance* behaviors. An example of a transitional object is bicycle training wheels: the wheels give us the support and comfort as we move from one state to another. Of particular interest to this study is his comment:

> *Both IS professionals and users must engage in an intensive learning experience, the former to develop a thorough understanding of the business domain, the latter to reflect on current practices and to acquire an understanding of the potential of IT to transform the way the work is done. (Wastell,1999, p. 582)*

The study will need to consider the impact of XP principals and practices on the learning process, particularly that of the customer.

The consensus from both practitioners and academic researchers is that the assumptions behind conventional methodologies need to be challenged and explored using rich descriptions of practice. Two studies were explored that typified this direction, one examined the use of a ISD methodology in an EIS development project, and the other re-cast the ISD process as a learning one.

4.2. USERS IN THE SOFTWARE DEVELOPMENT PROCESS

In order to effectively operate in the world of vague and changing requirements, Beck (2000) claims XP moves the emphasis away from prescriptive processes into practices that enable people. XP relies on effective face to face and regular communication within the XP team. Hence a small relevant section of the existing literature regarding user and development team interactions was explored.

Bostrom and Thomas (1983) recognise the importance of effective communication between users and developers in developing computer systems that are on time, within budget and satisfy the needs of users. The authors used action research in a campus administration system to explore effective communication practices. The authors draw our attention to one of the ongoing issues in user and developer communications, the use of framing. Bostrom et al used an example to illustrate the concept of framing, which I found particularly helpful; this example is replicated below.

> *What is the ordering in the following numbers?*
>
> *8, 5, 4, 9, 1, 6, 3, 2*
>
> *Most of you were probably looking for a well-formed numeric sequence, a pattern of thinking installed since early childhood. If you were you probably had trouble answering the question since the ordering is alphabetic by first character of the name of the number (eight, five, four etc).(Bostom & Thomas, 1983, p. 2)*

This example highlighted to me how my background caused me to frame the question in such a way as to "miss the point". The study found the use of communication patterns was key to the to the development of effective communication patterns between users and developers. People needed to be trained to ensure frames were (a) understood and (b) expanded by gentle verbal prodding.

Urquhart (1998) used a multiple case research design to review issues in user-developer relationships during the early requirements gathering phases of ISD. Three public agencies in Australia participated in the study. Six people were interviewed, representing 3 relationships, all participants were internal employees. She outlines the different perceptions of the process organisational context (departmental relationships) to play an important role in the user requirements gathering process. However, the development of positive working relationships is likely to have a large bearing on the success of the ISD process.

McKeen, Guimaraes & Wetherbe (1994) used a positivist survey research approach of 151 ISD projects to discover the key determinants of user satisfaction with ISD. The key findings from this research are:

- Effective user-developer communication directly affects users' satisfaction with the system
- Users influence on the project also directly contributed to users' satisfaction with the system
- User participation in highly complex systems development is worthwhile however
- User participation in low complexity systems development may not be effective

The next two studies specifically concentrate on the role of the user in prototyping methods, particularly the evolutionary prototyping method. Gordon & Bieman (Gordon, 1995) collected 39 prototyping case studies (22 published and 17 first hand accounts) and analyzed them to determine commonalties that led to effective use of the prototyping method. One of the areas explored in the study was the role of the user in the prototyping approach. The users were typically able to participate in the process and explore and evolve their requirements as the system develops. This exploration opportunity however, could lead to users viewing incomplete functionality and becoming concerned about the quality of the end system. Alternatively it can set high expectations concerning the timeframe for delivery, particularly where a 'smoke and mirrors' prototype was developed. One of the limitations noted in the study was the ability to draw conclusive results from cases varying in rigor.

Baskerville & Stage (1996) used action research to explore the ISD technique of evolutionary prototyping. A small contact management system for a non-profit organization was the unit of analysis. Baskerville suggests one of the reasons for using evolutionary prototyping is that the technique emphasizes learning, and helps to facilitate meaningful communication between users and developers over concrete artifacts. The expected result is rapidly built flexible software. This study explored an approach to the planning of the evolutionary prototyping iterations and the approach was risk mitigation based. The approach described in this study had a number of elements in common with XP, including the short release cycles, building only what you need now, and the planning game technique (which ensures business functionality is delivered in business priority order).

The studies in this section highlight the importance of the users in the software development process. The communication and interactions between developers and users can significantly affect the outcome of a developed system. The prototyping studies reviewed explored the strengths and weaknesses of the users involvement in an incremental delivery approach. Note that one common misconception regarding XP is that XP *is* evolutionary prototyping. Ron Jeffries, one of the leading XP practitioners, in a recent personal communication (2002) has explicitly stated that XP is not the same as evolutionary prototyping

Beck in the first XP text (2000) pointed out that the techniques of XP are not new. The innovation of XP is the combination of the techniques, processes and roles that reinforce and support one another to deliver and to deliver value. The studies reviewed above provide valuable insights into the involvement of users in the process. However, the combination of XP practices is likely to change the dynamic of the user role and requires further exploration.

4.3. XP CASE STUDIES

One of the ways the XP community is establishing a body of knowledge is through the collection of real life experiences concerning the implementation of XP (Marchesi, 2002). This section outlines a few of the most relevant experiences available.

Gittins, Hope & Williams (2002) used action research to explore the implementation of XP in a development team at a medium sized application (software) service provider. The company produced enhancements to the existing software every two to four weeks in response to feedback from customers, sales representatives, marketing executives and corporate management. One of the findings of this study was the importance of the implementation of a customer proxy. This person sat with the development team as a substitute for the customer. The proxy understood the customer's needs in detail and always acted in the customer's interests. The report does not clarify, however, how the customer proxy developed a detailed understanding of the customer's needs or the attributes of the person undertaking this role.

Schalliol (2002) provide a practitioner report on the experience encountered by a team of eight analysts on a 50-person, multiyear development project that converted to using an XP process. On this outsourced development project, the customer company devoted a team of its employees full-time to the project. One of the issues reported was the need to handle diverse and competing user requirements inherent in the development of complex systems. The team added roles and procedures to XP that reduced the agility of the process, but allowed the team to handle this issue when it arose. The key aspects of the process change was the addition of a role to facilitate the communication and resolution of this issue. The team also introduced an issue card to allow the prioritisation and planning of this activity to occur within the XP planning processes. The report is a reflection on a real life experience, and does not contain the validity checks expected in case research.

Farell, Narang, Kapitan & Webber (2002) also provide a practitioner report on the implementation of an effective on-site customer practice. This report considers the case of two analysts playing the customer role. The report describes the ineffective practices at the start of the project that included the customer being off-site and only meeting the development team once a week for a formal progress report. It then takes us through the journey that allowed an effective on-site customer to be available to the team. The conclusion was that the location of the team is important because customers and developers must work as a single team. Another important conclusion was the importance of feedback in XP. Testing was identified as an activity that must occur early for the project to succeed. Regular reviews of the process allowed the team to identify the issues and to implement correction strategies that improve the final outcome of the project. As with the previous report, this practitioner report does not contain the validity checks required for case research.

XP is new. Little research exists regarding the method, and particularly the customer role. The experiences reviewed above highlighted a number of interesting insights that require further exploration with rigour.

5. RESEARCH METHOD

5.1. OVERALL STRATEGY AND RATIONALE

Research perspectives[3], such as positivist, interpretative and critical provide a researcher with multiple methods of viewing a research problem or topic. Each of these perspectives has its own set of assumptions that result in strengths and weaknesses of the research design approach. Orlikowski et al. (1991) challenge the dominance of the positivist perspective in IS research and suggest viewing the world through one lens is unnecessarily restrictive. There is a growing interest in the use of the interpretative perspective in both IS research generally (Benbasat, Goldstein, & Mead, 1987) and IS research methodology research specifically (Cockburn, 1999; Fitzgerald, 2000; Nandhakumar & Avison, 1999; Russo & Stolterman, 2000).

Orlikowski et al. note the interpretative perspective is most appropriately used to capture complex and dynamic social processes and quote Gibbons (1991):

> *The interpretative perspective attempts to understand the intersubjective meanings embedded in social life … [and hence] to explain why people act the way they do (p. 13)*

The interpretative perspective assumes people are inherently complex and that group interactions are complex social practices. It assumes the researcher interprets or shares in these meanings. These assumptions allow the interpretative perspective to explore complex social processes, without attempting to simply these processes into cause and affect laws of nature.

Orlikowski et al go on to critique the interpretative perspective. Their critique includes the following deficiencies:

- External conditions, including history, are not examined and could give rise to certain meanings and consequences.
- Conflict and contradictions within the social group are not explored.

One of the research methods available for use within the interpretative perspective is case research[4]. Case research has been used in a number of diverse schools of study including, law (case law), medicine, psychology and political science(Creswell, 1998; Robson, 1993). Benbasat et al. (1987) recommend it as an appropriate method for IS research and provide the following definition.

> *A case study examines a phenomenon in its natural setting, employing multiple methods of data collection to gather information from one or a few entities (people, groups, or organizations). The boundaries of the phenomenon are not clearly evident at the outset of the research and no experimental control or manipulation is used. (p. 370)*

---

[3] For the purposes of this paper, paradigm and perspective are treated as separate concepts. The terms qualitative and quantitative (paradigms) are used to refer to the type of data collected and associated collection methods, not the assumptions inherent in the perspectives (positivist, interpretative and critical) outlined by Orlikowski & Baroudi (1991).

[4] It is possible for case research to be either positivist or interpretative (Lee, 1999); it should be clearly noted the case research detailed here is based in the interpretative perspective.

Benbasat et al. (1987) also provide four reasons to guide researchers as to the applicability of case research to their intended research. The following table outlines these reasons and their applicability to this research.

Case research selection reason	Applicability to this research
Phenomenon of interest cannot be studied outside its natural setting	The complex social interactions and processes (Russo & Stolterman, 2000) inherent in a software development project make it difficult to replicate outside of its natural setting without the loss of the richness of the activity
Study focuses on contemporary events	The focus is on events in the immediate past.
Control or manipulation of subjects is not necessary	The key focus of this research is to richly describe the phenomenon of interest and not to intervene as might occur in action research (Baskerville, 1999).
Phenomenon of interest enjoy an established theoretical base	As outlined in the literature review, research exists in the software development methodology arena. However, it has recently been suggested (Fitzgerald, 2000; Russo & Stolterman, 2000) the existing research does not capture reality of practice and/or the foundations require new research. Finally, a review of research was undertaken and no research, containing academic rigour, was located for eXtreme Programming.

Benbasat et al. (1987) also recommend that:

> *Case research is well-suited to capturing the knowledge of practitioners and developing theories from it. (p. 370)*

This is the stated purpose of this research study.

Finally case research is the appropriate research method for this research as it provides the rich descriptions of existing practitioner practices that are being recommended by systems development methodology researchers (Cockburn, 1999; Fitzgerald, 2000; Nandhakumar & Avison, 1999; Russo & Stolterman, 2000) as the way forward in this research area.

Creswell (1998) and Benbasat et al. (1987) outline the typical data collection and associated analysis techniques of case research. The data collection techniques include interviews, observations, documentation analysis, physical artefacts analysis and visual image capture. The data collection and analysis techniques used in this research project fall within this typical list, and are detailed in later sections of the report.

Creswell (1998) notes case studies have no standard report format and the reports often reflect the analysis techniques employed. The general structure of this report was guided by Creswell (1998). The case description and results sections have followed the guidance to XP researchers from Marchesi et al (2002). Marchesi et al suggest the use of a specific structure and titles for XP cases. These titles have been used for the case description and results sections of this report.

5.1.1. RESEARCHER BIASES

In the interpretative perspective it is assumed the researcher will interpret the data based on their values and beliefs. It is generally accepted that our values and beliefs are shaped by our life experiences (Creswell, 1997) and researchers need to provide relevant experiences as part of a piece of research. Hence the following section provides some of my background that is most likely to affect my interpretation of the data.

During the last eight years of my career I have specialised in outsourced custom development projects in both the public and private sector in New Zealand. Although I have worked for both the client and the vendor in outsourced arrangements, I am typically employed, or more latterly contracted, by the vendor. I have undertaken a variety of project roles including: team lead, information architect, business analyst, systems analyst and developer.

My experience as a consultant allows me to easily relate to and understand the day to day complexities in the software development process experienced by project teams. It is possible this experience may lead to inconsistencies or conceal issues that I simply take for granted. However, the practices I am currently learning as part of this University course allow me to view the case holistically and consider my biases.

5.2. FOCUSSING ON THE SPECIFIC SETTING

The case study will focus on an XP software development project - the unit of analysis. Benbasat et al.(1987) recommend exploratory research studies like this one should utilise either a single or multiple case study approach. Given the initial timeframe of this research, and the importance of obtaining multiple perspectives in order to, compare and contrast views of the customer role, a *single case study* approach was selected. Creswell (1998) provides a typology of sampling strategies that are appropriate for case research, and for this project the *convenient* sampling technique will be used. A convenient sampling technique is useful given the apparent adoption rate[5] of XP in New Zealand, it also fits into the proposed research timeframe and budget constraints.

Given the above rationale the key criteria for the case selection was the XP project be based in Wellington, New Zealand. My supervisors, Robert Biddle and James Noble, identified the candidate project through their previous involvement with the company.

Benbasat et al. (1987) recommend the two key points a researcher should cover while seeking co-operation are confidentiality and benefits. The consultant responsible for the project was approached with these points in mind, and the opportunity for research was discussed. The resulting proposal for the research project is provided in the appendices. This proposal clearly outlines the agreed outcomes (benefits), scope, and approach to the research project with the company. Once organisational agreement was obtained the project team were approached to determine their interest in participating in the study.

Potential participants were informed of the outcomes of the project, and were assured all information would remain confidential, and all results would be reported so as to retain their anonymity. The information provided to the team members is provided in the appendices, and outlines the importance of confidentiality and the agreed rules. Five of the seven team members approach volunteered to be involved in the project. The remaining two team members declined due to other commitments.

---

[5] Based on initial personal discussions with Wellington based New Zealand organisations, there appears to be strong interest in XP but low adoption rates. No definitive reliable reported adoption rates in New Zealand were located.

The key benefit arising from the research was the opportunity for the organisation(s) and team members to learn and reflect on their existing practice. This learning opportunity provided the organisation(s) and team members with the learning experience more typically obtained via an implementation review. The research may result in replacing the implementation review, or reducing the scope of the implementation review. A second opportunity provided by the research is an opportunity to be seen as a practice leader, if the organisation chooses to be identified in any articles or conference proceedings published.

The costs to the organisation(s) were minimal and time-based only. The total time required of each of the five identified team members was between 3 and 6 hours over a period of three months. The maximum period in any one-day for any team member was 1 hour. All interactions were arranged around the team members' existing commitments, so disruption remained minimal.

This time was utilised to collect the data via initial interviews, and to later confirm the raw data and interpreted findings. The next sections on data collection and analysis provide further details on these activities.

The project was undertaken with Victoria University of Wellington human ethics approval.

5.3. DATA COLLECTION PROCEDURES

5.3.1. RESEARCHER ROLE

Lee (1999) and Creswell (1997) note researchers typically distinguish between four types of research participation in qualitative studies. The key elements that distinguish these types of participation are summarised below.

Research participation classification	Research role	Primary focus	Participation in environment
Complete participant	Hidden	Research & Participation	Active
Participant as observer	Public	Research & Participation	Active
Observer as participant	Public	Participation	Active
Complete observer	Public	Research	Passive

I will remain a complete observer in this research, no planned intervention or direct participation will occur, as the intention is to richly describe practice.

5.3.2. SAMPLING PEOPLE, PROCESSES AND DOCUMENTS

The XP team consists of the following roles: programmer, customer, tester, tracker, coach, consultant and big boss (Beck, 2000). Each of these roles will have a view on the implementation of the XP customer role:

Role	Reason
Customer	The team member fulfilling this role is the most intimately involved in the implementation of the XP on-site customer requirement. It is imperative their views and experiences are explored.

Role	Reason
Programmer	One of the key tenants (Beck, 2000) in XP is use of effective face-to-face communication between all team members but particularly between the customer and the programmers. A user story is a promise of a conversation between the customer and the programmer. This practice deviates from traditionally heavy methods, which require detailed user requirements documentation. Also a programmer's role is extended in XP by requiring them to work directly with the customer to prioritise user stories. It is imperative to consider the impact of the customer and associated customer-programmer communication assumptions of XP.
Tester	The tester in XP focuses on helping the customer to choose and write functional tests which provide feedback into the planning process.
Coach	The coach (Beck, 2000) is responsible for the process as a whole. They work with each of the other roles to help facilitate the correct use of XP. This person fulfilling this role will provide valuable insight into the adoption of XP in general and also the specific adoption of the customer role.
Big Boss	The big boss, often known as the project sponsor, is responsible for the project. The big boss will have a unique perspective on the customer role, particularly how competing customer requirements and needs were prioritised and whether that was acceptable to the organisation as a whole.

The interviewees covered the spectrum of the required roles, except for the "big boss". The project sponsor was not approached given the constraints on her time. Her views on the process were provided by the customer. Further information on the interviewees is provided in section 6.2.

The processes to be sampled are those that directly involve or impact the customer. The XP practices (Beck, 2000) that meet these criteria are the *planning game, small releases, testing and on-site customer*. These practices involve the following processes:

- specifying the requirements as user stories (a short paragraph),
- planning the releases or iterations based on the user stories,
- conversing with the developers about the user stories (user stories are the promise of a conversation between a customer and a programmer) and finally
- testing the resulting software (regularly) with acceptance tests.

The remaining XP practices (Beck, 2000), which will not be specifically sampled because they are related directly to coding. These remaining practices are: 40-hour week, metaphor, simple design, refactoring, pair programming, collective ownership and coding standards.

Participants were given the opportunity to provide information on these or other practices but they were not directly targeted during the interview.

The documents sampled included those described in the processes above: user stories and plans.

5.3.3. DATA COLLECTION TECHNIQUES

The previous section identified the people, processes and documents to be sampled. This section identifies how this data will be collected. The following data collection methods recommended for case research (Benbasat et al., 1987; Creswell, 1998) will be utilised: semi-structured one-on-one interviews, and document analysis. Each of these methods is described in further detail below.

SEMI-STRUCTURED ONE-ON-ONE INTERVIEWS

Semi-structured one-on-one interviews were held with each of the people identified in the previous section. These interviews allowed me to explore the team member's view of the XP process, particularly the practices and processes identified in the previous section as being of particular interest. The topics covered the background of the interviewee, the planning process followed and their view of any perceived issues and effective experiences encountered on the project. The topics covered are presented in the appendices.

The interviews occurred in a meeting room at the site of the interviewee. The on-site location minimised the disturbance to the interviewee's day-to-day activities. The meeting room also ensured the interviewee was able to speak freely without concern for other team members perceptions of their responses. All interviews were taped and later transcribed in detail. The interviewees were asked to validate both the transcription of the interview and the interpreted findings. One interviewee suggested changes to the initial interpretations and these changes were incorporated into the report.

DOCUMENT REVIEW AND ANALYSIS

Documents identified in the previous section were requested and reviewed. These documents aided my understanding of the artefacts produced, in line with the XP suggestions.

These documents are internal project deliverables and are of a competitive nature. As previously stated all documents will be kept confidential and only findings will be reported. For example, the changes to the user stories were hand written onto the user story sheet in the standard format described in the XP literature. The team members validated all findings regarding the documentation.

5.4. DATA ANALYSIS

Creswell (1998) notes there is no consensus on how to analyse qualitative data. However, he presents a useful set of activities that he has synthesised from multiple sources. The data analysis procedure used in this project utilised Creswell's advice. The amount of data to be analysed did not warrant the use of data analysis software, so a manual coding procedure was used based on Lee's description of open coding (Lee, 1999).

The interview data was transcribed from the tape recordings. Each interview was printed on different coloured paper to ensure the source of the data was not lost. Individual interview transcripts were analysed for patterns, themes or practices: the first level of coding. The transcripts were then cut into "coloured cards" that encapsulated the relevant text for each code. The coloured cards were then organised into like piles. A picture of this process is included for reference below. Each pile was then analysed and compared for contrasts and similarities to form the findings of this case study.

Figure 1: Coding process

The coding process and the interpretations made were then reviewed by my supervisors (key interpretations only) and the participants. The changes that were suggested were then incorporated and the data re-classified as appropriate.

5.5. VERIFICATION STEPS

Creswell (1997) notes that internal validity ensures the information is accurate and it matches reality. Internal validity, as summarised by Creswell, will be achieved as follows.

Recommended procedure	Implementation in this study
Accessing multiple sources of information	Multiple data collection methods were employed including interviews and document analysis. Multiple people on the same project were interviewed.
Auditing the decisions and interpretations given the raw data, by another researcher	Supervisors reviewed all transcripts and interpreted findings.
Obtaining feedback from the participants on both the raw and interpreted data to confirm the data has been interpreted accurately	Participants were given the opportunity to review the raw and interpreted data. Some of the participants confirmed the data.

Creswell (1997) notes external validity is the ability to generalise findings. The purpose of this study is to explore a situation not previously explored, hence the research has not focussed on external validity. However, it may be possible to generalise the findings (Creswell, 1998), in terms of identified themes, patterns or practices based on existing literature. This report provides future research suggestions, including the explanation of the research to multiple case studies using the *maximum variation* sampling technique recommended by Creswell (1998) to assist with external validity.

Creswell (1997) notes reliability as the last verification step. Reliability is met when sufficient information to allow another researcher to replicate the study is provided. All information expected to enable this replication to occur is provided, including researcher biases, data collection and data analysis procedures.

6. DESCRIPTION OF THE CONTEXT OF THE EXPERIENCE

In this section we explore the implementation of XP on an intranet content management system re-development project (CMS).

The following section describes:

- The project, outlining the beginnings of the project, the organisations involved in the project and at what stage in the project XP was utilised.
- The team, outlining the team members previous experience on software development projects including their exposure to software development methods and how the team changed over the duration of the project.
- The process, outlining the tailored version of XP used on this project.

Quotes and photographs have been used to support our description. The conventions used in this report are outlined in section 2.5. The quotes provided in this section, where appropriate, will be analysed in detail in the results section (section 7).

6.1. THE PROJECT

CMS was an outsourced software development project and involved the three organisations as described in the table below.

Pseudonym	Role in Project	Description
KiwiCorp	Customer organisation	A large New Zealand corporation with employees dispersed throughout the country.
DevCorp	Outsourced software development company	A New Zealand based company specialising in providing Internet, extranet and intranet solutions. The company is wholly owned by an international consulting company.
BureauCorp	Outsourced infrastructure services company	An international information technology services company that supplies facilities management services. This company is responsible for the physical infrastructure CMS will be tested and deployed on.

KiwiCorp's project lead on CMS describes the beginnings of the project below.

> *Internet redevelopment in [KiwiCorp] had sort of been mooted for a couple of years and various people had sort of had a go but nothing really worked too well, and then my manager got handed the job of sorting out the intranet – Customer, KiwiCorp*

CMS was established in the middle of 2001 with DevCorp as the outsourced software development vendor.

> *We felt we could probably only do it if we used [DevCorp] … because we had such a lot of confidence in them based on previous experience – Customer, KiwiCorp*

Initially the project was a traditional waterfall project and was divided into three phases, as outlined in the figure below. At the end of the planning phase it was decided to use XP for the development phase. It was recognised at that point that the formal implementation phase would remain as this approach was required by BureauCorp and meshed with the existing practices of KiwiCorp.

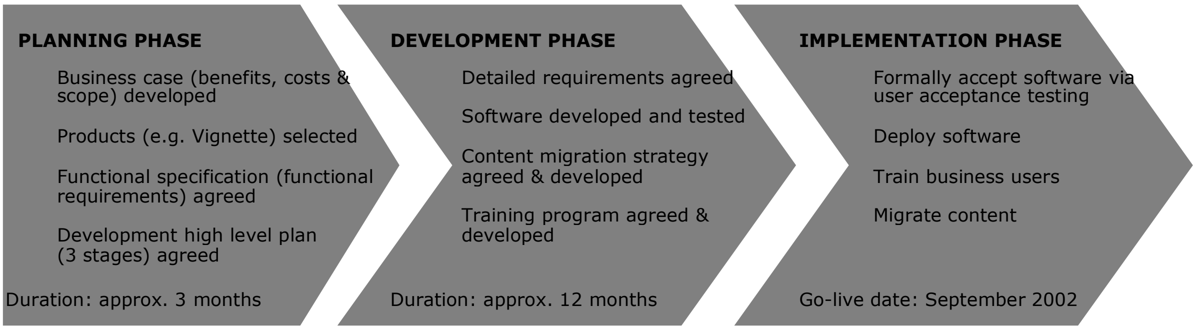
PLANNING PHASE
Business case (benefits, costs & scope) developed
Products (e.g. Vignette) selected
Functional specification (functional requirements) agreed
Development high level plan (3 stages) agreed
Duration: approx. 3 months
DEVELOPMENT PHASE
Detailed requirements agreed
Software developed and tested
Content migration strategy agreed & developed
Training program agreed & developed
Duration: approx. 12 months
IMPLEMENTATION PHASE
Formally accept software via user acceptance testing
Deploy software
Train business users
Migrate content
Go-live date: September 2002

Figure 2 : Outline of structured waterfall approach

CMS was sucessfully deployed in September 2002. The interviews to collect the data for this report occurred during the implementation phase, prior to deployment. The indications from KiwiCorp, based on the acceptance testing and training feedback, are that this project is considered a success:

> *This development approach [has meant] we were able to track things as we go and actually discover that things aren't quite what people wanted and complete them …The indication [from the testing] is really good … And the training feedback has been really good too because we've done some one on one training with the key publishers – about 50 to 100 odd people … [and] generally people find it easy to use and are going to be fine with it so that's really good – Customer, KiwiCorp*

6.2. THE TEAM

The development team consisted of KiwiCorp and DevCorp representatives. At its peak the team consisted of:

No	Status	Project position	XP roles played	Organisation	Research participation
7	Full-time	Developer	Programmer	DevCorp	Yes[6]
1	Full-time	Project manager	Coach, Tracker, Tester	DevCorp	Yes[7]
1	Full-time	Lead developer	Programmer, Coach	DevCorp	Yes
1	Part-time	Pre-sales consultant	Coach, Specialist	DevCorp	Yes
1	Full-time	Project lead	Customer	KiwiCorp	Yes
1	Part-time	Project sponsor	Big Boss	KiwiCorp	No

---

[6] One of the seven developers was interviewed

[7] Note that we interviewed the project manager and the lead developer responsible for the development phase of the project not the planning phase.

No	Status	Project position	XP roles played	Organisation	Research participation
1	Full-time	Tester	Tester	KiwiCorp	No
1	Full-time[8]	Content Migration Facilitator	Customer	KiwiCorp	No

The success of a project can be affected by a number of team factors including the consistency of team members and the experience level of the team both in terms of methods and projects. The following sections describe these project specific factors.

6.2.1. PERSONNEL CHANGES

The make-up of the project team changed between the planning phase and development phase. The team grew to include a number of additional programmers at its peak. The team also changed the project manager and lead developer between the planning and development phases. Four members of the team, a developer, pre-sales consultant, project lead and project sponsor remained consistent throughout the project.

6.2.2. PREVIOUS SOFTWARE DEVELOPMENT EXPERIENCE

The development team was comprised of experienced individuals.

The DevCorp members ranged in experience from 3 years of small web development projects to 23 years of wide ranging software development experience. All of the programmers, except one, were experienced computer programmers. The novice programmer, however, was an experienced business analyst with a background in KiwiCorp's industry.

The KiwiCorp members had recently been involved in another Intranet development project gaining an insight into the software development process.

Further information on the software development backgrounds of the team members interviewed is provided in Appendix A.

6.2.3. PRIOR EXPOSURE TO XP

All of the team members were new to XP. DevCorp had recently used XP successfully on a similar project. None of the team members on the earlier XP project transitioned to the CMS project. There was strong support in the team for adopting XP as evidenced below.

> *I'd had good reports about it in [project name], I'd read the book and the book made a lot of sense, it was common sense. [and later in the interview] I was willing to give it a go myself but the other thing was the team was willing to give it a go, if the team had pushed back, I probably would have folded … [goes on to note two particular team members] were very strong – Project Manager, DevCorp*

This support was also evidenced in the following comments regarding the more traditional rigorous processes.

---

[8] Note that the implementation phase required a full-time acceptance testing team. These team members are not represented in these figures - only the peak development phase personnel are described.

> *His idea of a timetable was putting a cross in a calendar and saying "we're aiming for that date" and we always did it … we did them more the XP style and we tended to be more successful than all of the other people who were running around with [their] MS Project [plan] updating it every other day and yet never ever seeming to reach the mark. … [and later in the interview regarding rigid projects] you sit there and write screeds and screeds of detailed information without really knowing whether or not you can do that. People who want a system … don't know what they want until they see something else. You have to have something for them to see before they can really figure out, … realise what they want – Programmer, DevCorp*

> *I've been engaged with what I think of as traditionally run projects where people have a Gantt chart and they have a list of deliverables and they keep nagging people until things are crossed off their list – Pre-sales consultant, DevCorp.*

The team gained an understanding of XP by reading the XP books, sharing XP experiences with other practitioners of XP and also by regularly reviewing their progress with the method throughout the project.

6.3. THE PROCESS

6.3.1. THE PLANNING PHASE

The activities undertaken and associated deliverables during the planning phase provide the background to the XP project. This section will cover the activities that had a strong influence on the project teams' decisions to customise XP and reflections on their experience with XP.

UP-FRONT REQUIREMENTS GATHERING

The planning phase involved the requirements gathering activity typical of rigorous software development methods. The requirements were gathered during a three month period with IT specialists driving the process and facilitating the workshops sessions:

> *[We had] heaps and heaps of meetings about the various aspects, various work strains [with DevCorp facilitating the workshops] … picking my brains and anybody else's brains that we felt we needed – Customer, KiwiCorp*

The customer also highlighted the importance of the senior management support during the requirements gathering process:

> *My manager … was responsible for getting buy-in from the sort of senior level in business and she had been talking to lots of people and had feedback from them that she relayed to me – Customer, KiwiCorp*

The result of this process was a functional specification, described by a DevCorp programmer:

> *The functional spec's for the client and it's in far less technical language [than the technical specification for the programmers]; ... Certainly for a web project it would be how things are supposed to work, what the user will see when they come into the situation, what sort of behaviour they can expect … [with] lots of screen shots of the user interface – Lead developer, DevCorp.*

The planning process used on the CMS project resulted in an intensive up-front requirements gathering and analysis process that is not typical of an XP project.

HIGH-LEVEL PLAN

The initial estimation and high-level planning activity occurred at the end of the planning phase. A day long estimation session occurred where DevCorp:

- Developed a list of system features based on the workshop sessions
- Divided the features amongst the development team
- Estimated each feature based on previous experience (development tasks required to implement the feature and associated time to accomplish each task)
- Reviewed the estimates as a group
- Released the expected cost and time to develop estimates to KiwiCorp.

As is often experienced in this process, there was a tension between the accuracy of the provided estimate and the information available to provide the estimate (McConnell, 1996).

The project manager from DevCorp then worked with the customer from KiwiCorp to develop a high-level plan based on business priorities of features. Three stages were agreed:

1) A skeletal content management system,
2) The workflow functionality and
3) The remaining features including security and administration facilities.

This estimate combined with the high-level plan set the expectation of KiwiCorp as to the timeframe, scope and budget of the CMS project. Although a similar process occurs in XP, termed the high-level release plan, the lack of detailed user requirements provide an indication that the estimate and plan are to be refined.

6.3.2. THE DEVELOPMENT PHASE

XP literature outlines how to develop a system using XP in a "green-fields" environment. This description allows us to grasp the generic concepts and lessons of XP so that we are able to apply these concepts to our own situation, and indeed the authors emphasise the need to tailor XP to your project. This section provides a description of the tailored XP process that was used on CMS.

CREATE USER STORIES

XP challenges the conventional wisdom (Beck, 2000) of a formal requirement specification document as an effective representation of user requirements. XP recommends instead an on-site Customer who expresses requirements as user stories. A user story is a short paragraph description of a system requirement or feature that is:

> *… understandable to customers and developers, testable, valuable to the customer and small enough so that the programmers can build half a dozen in an iteration. …A user story is nothing more than an agreement that the customer and developers will talk together about a feature. (Beck & Fowler, 2001, p. 45-6)*

Short iteration cycles result in the customer reviewing these requirements in software almost immediately, enabling the requirements to be explored and refined.

The first tailoring of the XP method on this project occurred in the development of user stories. The team needed to tailor XP to cater for the up-front requirements gathering that had already occurred, while retaining the change friendly and evolving requirements gathering process of XP. The team decided to develop the user stories from the existing functional specification.

The team was able to use the high-level release plan to narrow down the functional requirements to those appropriate for the iterations within the release. The programmer responsible for creating the user stories used her knowledge of KiwiCorp to select stories from a business user's point of view.

> *I took the view of thinking from a user point of view, you know, I'm sitting down at my PC, I've come to the homepage – "What do I expect to happen? What can I see? What can I click on? What happens next?" – Programmer, DevCorp*

The resulting user stories did not include all of the user stories for the release but did include sufficient stories for the customer to prioritise for the iteration.

The programmer provided the draft user stories to the customer prior to the iteration planning meeting. At this point the customer became the owner of the stories. She reviewed the batch of stories and modified or added new stories. During this period she was able to question the programmer's selection of stories based on both her business knowledge and the functional specification:

> *And [the user stories] were written by [DevCorp]. And then they'd give them to me to review and I'd just make sure that our requirements had been translated correctly to the user story. And then I had to prioritise everything – Customer, KiwiCorp*

PLAN AN ITERATION

An iteration planning meeting was held every three weeks on the first day of the iteration, after the user stories were created. The meeting involved the customer, the project manager, the lead developer and 1 or 2 programmer representatives. The meeting included the following steps:

- The user stories were reviewed to ensure the team understood what each story meant.
- The customer prioritised the stories
- The team estimated the stories
- The project manager determined the iteration capacity
- The customer decided the scope of the iteration, that is stories to be included in this iteration.

The development team customised the attendance at this meeting. XP recommends all programmers attend the meeting and sign-up for their tasks that they will estimate during the meeting (Beck & Fowler, 2001). The team streamlined this process to reduce the negative reaction of the programmers to this process while still maintaining their involvement.

> *People hated estimating, hated it; hated estimating and then putting tasks together … people just find it very, very dull. We did it as a group for the first couple of iterations and then they just didn't want to …What ended up happening was … we'd [project manager and lead developer] just basically try and rope one or two others in, just beg them, and I think if people felt they didn't have to do it every single time it was slightly more tolerable. So that way it wasn't just me [saying] it's going to be this way. At least there*

> *was one or hopefully two other people having input into how … things [were] going to work – Lead developer, DevCorp*

These steps of this activity are elaborated in further detail below.

Step: Prioritise the user stories

The customer then ranked the stories in order of priority, taking into account any technical or business dependencies outlined by the development team. The customer discusses the process from her perspective:

> *[At] the first [iteration planning meeting] no one quite knew what they were doing but we had the user stories and I had been given them the night before to go and prioritise them …[During the early iterations] I was guided by the lead developer [regarding prioritising]… she knew how the system had to be built and what were the fundamental things that had to be done. So usually at the beginning she'd say "oh well you know you can't do this until you've done this" and so we got various – we got the content management application – you know, the bare bones of it, so we could create a site and create a page and just really basic stuff. And then once the system got a bit more mature … a lot of my priority decisions were based … on the three-month phase [high level plan]. [Prioritising has] become a really key thing now and for the previous three months when we realised it was getting to the end and I've actually had to make decisions though probably can't afford to have this thing so we'll leave that till last – Customer, KiwiCorp*

She elaborates on the process she has used to make some of the recent difficult decisions regarding prioritisation in the following quote.

> *[I considered] how many people will I have to front up to and say "no actually it's not going to do this"… Usually my manager and I have done that together because she knows what's key to her based on her discussions of the business [it's] about common sense ... People say they want this but really, you know, there's [a] work around .... Usually I've done that myself and in conjunction with [my manager], when I can't bear responsibility anymore – Customer, KiwiCorp*

At this point the customer typically left the meeting, allowing the programmers to begin the estimation process. She was however, available for any further queries regarding the stories.

Step: Estimate the user stories

The estimation process began with a division of the user stories among the attendees. The attendee estimated the story by breaking the story down into tasks. Each task was estimated based on the ideal perfect engineering day (PED) concept detailed in XP:

> *The time without interruption where you can concentrate on your work and you feel fully productive. (Beck & Fowler, 2001, p. 61) .*

Attendees initially estimated all tasks to the nearest half day. However, the team quickly found that the zero tasks created by this process added up. The team settled on estimating tasks to the nearest ¼ day with very few zero tasks.

The group then reviewed the estimate for each story as a whole, including a review of the tasks to ensure completeness.

Step: Determine iteration capacity

The project manager calculated the team's capacity for this iteration, that is, how many perfect engineering days work could be accomplished in the next three weeks. The project manager used the capacity figures of a similar project to provide an initial estimate of the team's capacity. He modified this estimate to account for the unique aspects of this project, particularly the new and untried technologies of the project. The project manager continued to track these figures for each iteration. He applied the XP advice of yesterday's weather that is the best prediction of this iteration's performance should be the last iteration's performance.

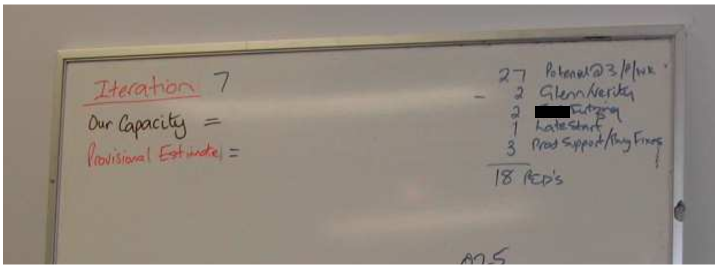

Figure 3: Capacity calculation

It should be noted the project manager tended to calculate the team's capacity prior to the iteration planning meeting.

Step: Decide the scope of iteration

All of the user stories were now estimated as to the number of PED's required to develop them and the capacity or number of PED's available in the iteration was also now determined. This allowed the team to go back to the prioritised list of stories and determine which ones would be included in the iteration. The customer was brought back into the process to confirm the user stories to be included in this iteration. This step allowed the customer to alter the priority of the stories, including deferring parts of a story to a later iteration, in order to obtain useful business functionality for this iteration.

ALLOCATE WORK

Once the planning meeting was completed the project manager wrote up the results of the meeting by expanding the user story to include the task sheets (refer to Figure 4). The stories were displayed in priority order (left to right) on the project wall (refer to Figure 5)

The lead developer used the story sheets to allocate work to the pairs. She was able to take into account their experience level, preferences and also recent workload similarities when allocating stories.

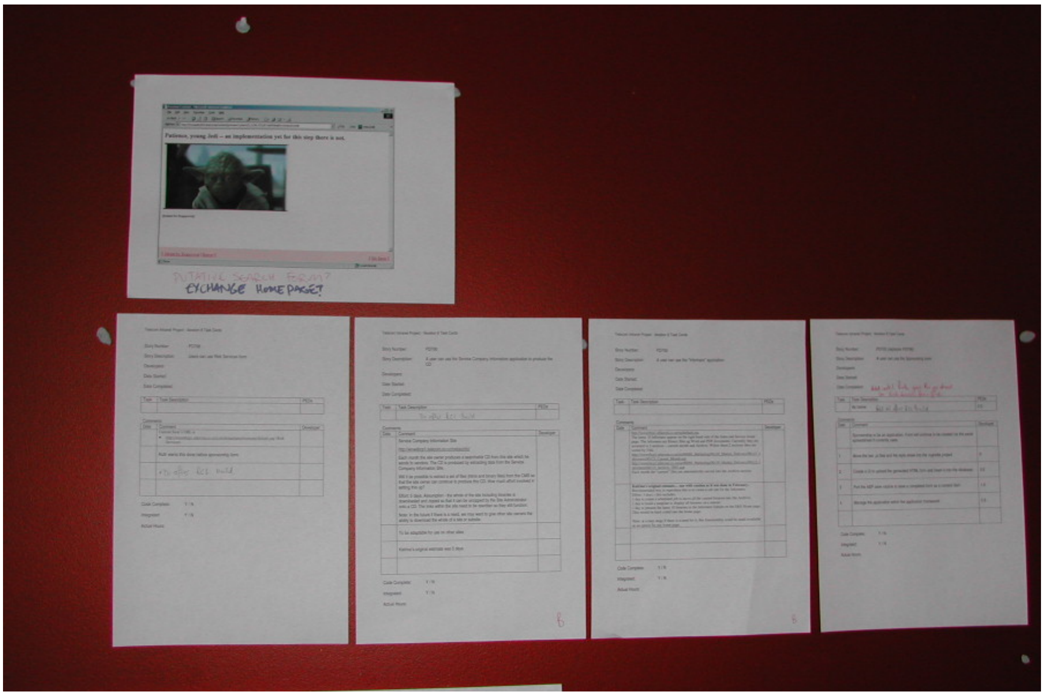
Patience, young Jedi -- an implementation yet for this step there is not.
PUTATIVE SEARCH FORM?
EXCHANGE HOMEPAGE?

Figure 4: A user story/task sheet

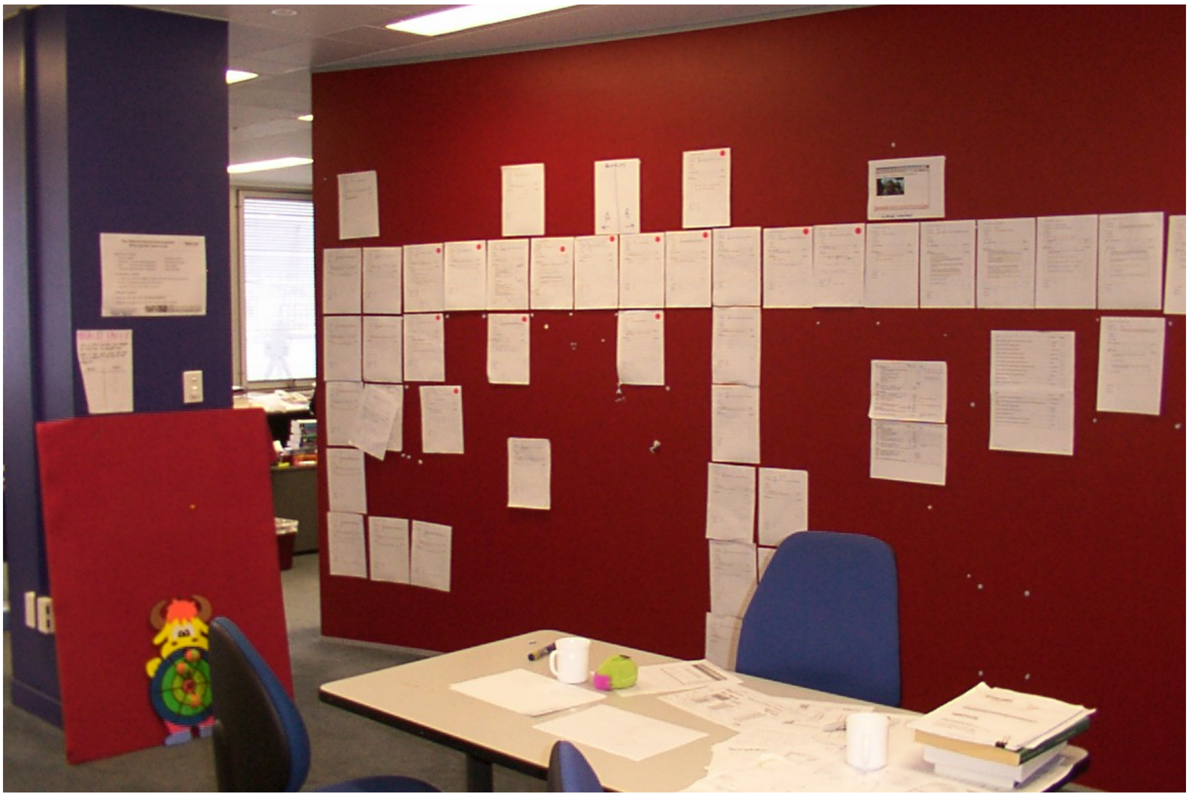

Figure 5: The Wall

DEVELOP THE STORY

During the iteration the pairs worked to develop the code to produce the story. The customer was available to each of the pairs throughout this process to provide any further clarifications on the business functionality. Any resulting changes to the user story were hand-written onto the user story sheet (refer to Figure 4). Once the story was complete a red dot was applied to the story on the wall and the actual development hours were recorded on the story sheet.

STAND-UP MEETINGS

Daily stand-up meetings occurred every morning. These meetings included all of the team, although the customer only attended 2 – 3 times a week. These meetings covered what each pair had done yesterday, what they were doing today and any issues affecting their progress.

The stand-up meetings proved to be a valuable activity for the development team as reflected in the quotes below.

> *The other thing that was really good was the stand-up meetings. Every morning at 9.15 we -- all of us -- everyone who was working on it stood there and said what we did the day before, whether we wrote any little functions that could be re-used and that sort of thing, so that you all knew what everyone else was working on so you didn't both find yourselves going and working on the same thing. … it just gave a really good team spirit, it kept everyone working together and also knowing what was going on too because you also found out quite a lot – because … you'd have your project manager there as well and … your client – Programmer, DevCorp*
>
> *Certainly the meetings in the morning worked really well. Again, we probably didn't use them the way they were supposed to be used. They tended to be longer than 10 minutes most of the time, especially when we had eight people, getting round four pairs did tend to take longer. … We did use those meetings to thrash out difficulties that people were having which often again wasn't what you were supposed to do. … People got to tell jokes and have a bit of a laugh. It was a bit of a bonding session - Lead developer, DevCorp*
>
> *[Discussing the effective team culture] … it grew out of the close knit nature of the team and some of the rituals that we had for example the 9:00 or 9:15 if we were lucky, the stand up meeting every single morning. [And on the rules of the meeting] … The doll that says yadda, yadda, yadda – that was the signal for this conversation has gone on too long in a stand up meeting because the whole concept of a stand up meeting is if you get into detail, some people take it off line and let the meeting go on but that yadda, yadda, yadda was a sort of light hearted indication that the conversation really has gone on - Project manager, DevCorp*

REVIEW THE PROCESS

At the end of the iteration any incomplete stories were put into the user story pile for the next iteration planning meeting. The team also used the end of each iteration to review the process as a team and to discuss what went well and what didn't go well and to incorporate these learnings into the next iteration.

7. RESULTS FROM THE EXPERIENCE

The following section outlines the key findings from this experience regarding XP practices involving the customer including the planning game, small releases and on-site customer. This section divides the findings into four areas: the team's impression of XP, the implementation of the customer role, the implementation of the planning practices, and finally the reliance of XP on timely reviews and feedback.

Quotes have been used to support our description. The conventions used in this report are outlined in section 2.5.

7.1. THE TEAM'S IMPRESSION OF XP

The team's reaction to the use of XP was very positive:

> *Overall – I love this approach to development and I'd certainly like to use it again in any future projects I am involved in – Customer, KiwiCorp*
>
> *I mean XP [is] very change friendly and to me that's one of the things I like about it the most … the client is always going to change their mind at some point in the process and why should we expect them not to - Project Manager, DevCorp*
>
> *[Programmer compares the XP process to other successful projects processes used in the past] we did them more the XP style and we tended to be more successful than all of the other people [using rigid methods] – Programmer, DevCorp*

The team members' enthusiasm for XP may influence their comments. A second factor to consider is the point in the lifecycle the data was collected:

> *There's a core of people who've actually been nose to the grind stones for like 10 months now and it's showing signs of … it doesn't look good so you sort of have to eat the last few drops of blood out of these people – Project Manager, DevCorp*
>
> *It's been my life for about a year so … look at these grey hairs – Customer, KiwiCorp*

Team members may interpret the entire experience positively due to their personal commitment to XP and the project, resulting in them glossing over any difficulties encountered. The interviewees did, however, relate both positive and negative experiences of the process. Their comments can help us understand the implementation of XP on a real project, including the biases to be found on a real project.

7.2. ON-SITE CUSTOMER

Earlier in this report we established that the customer is an essential role in an XP project. They are responsible for steering the project, and for deciding what is developed and when. There are three aspects of the on-site customer practice considered as part of this report, the characteristics of the customer, the location of the customer and the skills of the customer. These aspects are explored below.

7.2.1. CHARACTERISTICS OF THE CUSTOMER

Beck & Fowler (2001) have described the ideal preparation for someone playing the customer role. The customer representing KiwiCorp had close to the ideal, as we show in the table below.

Ideal Customer	Actual Customer Characteristics
Understands the domain well	The customer's librarian training, combined with her tenure at KiwiCorp, allowed her to understand how CMS must work to meet the diverse needs of the business users. To supplement her knowledge, she involved operational users of the existing system in the process.
Understands how the software can provide business value in the domain	The customer's existing knowledge of similar systems, combined with her perceived value of DevCorp's ideas (the possibilities of technology), allowed her to understand the value software could provide the domain.
Understands the importance of regular delivery	The customer perceived regular delivery as important as it allowed her to evolve the requirements and test the system with operational users.
Understands the importance of prioritising the functionality to be delivered	The customer perceived the importance of prioritisation and worked closely with her senior manager to ensure prioritisation decisions were made effectively. She learnt, as described in the prioritisation findings, that she needed to be tougher, earlier.
Accepts responsibility for the success or failure of the project	This aspect, as described here, was not covered sufficiently in any of the interviews to make an interpretation.
Is able to represent diverse users[9], termed "speaking with one voice"	The customer represented a small KiwiCorp project team including her senior manager, migration manager, testing team as well as the thousands of end users of the system. DevCorp team members all considered the customer as the sole source of requirements and decision making for KiwiCorp. No DevCorp interviewees indicated competing requirements or conflicting decisions occurred during the project.

7.2.2. SKILLS OF THE CUSTOMER

> *One of the things that is unsaid in the XP literature is how to be a customer (analyst) (Fowler, 2002)*

The guidance for customers provided in the literature concentrates on user stories and testing (Beck & Fowler, 2001). The customer on this project did not develop user stories, and a contract tester was hired to assist with the development of test scripts.

However, our customer noted, that despite these tasks not being her responsibility, she was still overloaded:

---

[9] Includes operational users, business management and IT operations

> *I was the main [KiwiCorp] person on the project, I think we needed some extra roles basically. We probably needed about three of me.*
>
> *[and later describing the three roles she played as the customer] The main areas from a business point of view were looking after the product, looking after other issues, technical or otherwise, and then another major area was content migration – actually communicating with the existing site owners and working out a plan for them to migrate their staff and all that – Customer, KiwiCorp.*

The customer elaborated that she quickly realised that she was unable to fulfil the content migrator role as well as the other roles, and a full-time migration project manager was employed. During the interview the customer also considered the potential of applying this lesson on her next project:

> *[Regarding determining that a content migration manager was required]*
>
> *So we should have realised that up front and ...*
>
> *[Interviewer interjected - Perhaps next time you will?]*
>
> *Maybe, I mean it's all so … I mean, money, you know – Customer, KiwiCorp*

It appears expecting the customer to be able to focus only on the requirements and testing may be unrealistic.

The customer also discussed the need for her to consider and understand the diverse needs of thousands of business users:

> *[KiwiCorp is a large organisation and has] such [diverse] needs, we had to … all the way along I had to be thinking "is this flexible enough", you know, will it fit this person, will it fit this person*
>
> *[and later] when I felt that I didn't have enough knowledge to make a call then we would – I'd ask questions of other people in business – Customer, KiwiCorp*

The customer also elaborated on the importance of understanding how to "get things done" in the organisation:

> *[My manager] was responsible for getting buy-in from the sort of senior level in business … Well we knew that if we actually got peoples' formal sign-off – business people to sign-off for everything – we'd never actually get anything done. [and a little later her understanding of how this approach mitigated that risk] … Constantly using this development approach we were able to track things as we go and actually discover that things aren't quite what people wanted and complete them. – Customer, KiwiCorp*

Further consideration needs to be paid to the less task orientated aspects of the customer role including the balancing of multiple roles, understanding the needs of diverse users and senior managerial support.

7.2.3. LOCATION OF THE CUSTOMER

> *A real customer must sit with the team, available to answer questions, resolve disputes, and set small scale priorities… The on-site customer will have the disadvantage of being physically separated from other customers, but they will likely have time to do their normal work. (Beck, 2000, p. 60 - 61)*

On this project KiwiCorp and DevCorp were situated in different buildings within the central business district of Wellington, approximately a 10-minute walk apart. Beck notes the importance of the time with the developers but assumes other customer separation is acceptable. The customer realised she needed to spend direct "face-to-face" time with the developers as suggested by Beck. However, she also needed to spend direct "face-to-face" time with KiwiCorp stakeholders as well. Her time with KiwiCorp people allowed her to represent them with a "single voice" and also to focus on non-software development project activities such as training and migration. Her decision was to spend 50% of her time at each building.

During the project, approximately 50% of her time was spent resolving technical integration issues with BureauCorp. The result was that she was unable to spend a significant portion of her time at DevCorp moulding the software to meet the business needs:

> *My life would have been easier if I could have been 100% devoted to requirements and testing … because I would have been right there when the developers were saying "shall I do it this way or do it that way" right before it had even got to build. – Customer, KiwiCorp*

Cockburn concurs with and elaborates on the customer's suggested impact of not being available to the development team:

> *Having a usage expert available at all times means that feedback time from imagined to evaluated solution is as short as possible, often just minutes to a few hours. Such rapid feedback means that the development team grows a deeper understanding of the needs and habits of users, and start making fewer mistakes … with a good sense of collaboration, the programmers will test the usage experts idea's and offer counter proposals. This will sharpen the customer's own ideas for how the new system should look. The cost of missing this sweet spot is a lowered probability of making a really useable product and a much higher cost for running all the experiments. (Cockburn, 2001, p. 150)*

It is clear that the practice of an on-site customer has obvious intended value. How to achieve this requirement and still obtain the required "single voice" of a customer over multiple locations has to be determined.

An alternative solution, locating the DevCorp developers at KiwiCorp, was briefly touched upon:

> *I would make explicit the fact that typical outsourcing arrangements are what drive development away from the client's premises, thus breaking one of XP's implicit assumptions. – Pre-sales consultant, DevCorp*

Schalliol (2002) also worked on a project where they encountered significant customer overload. This team introduced analysts to facilitate the communication activities involved with complex systems with a large diverse user base. This role complemented the existing XP customer role. The communication activities were also planned as part of an iteration. A issue card was introduced and placed into the prioritisation sessions in a similar manner to story cards. The roles and procedures introduced in this team were done for a large project, however these suggestions may be relevant to smaller projects, such as CMS, particularly when a large diverse user base exists.

Note that the outsourcing arrangement of CMS may complicate the issue further. Not only were three organisations involved, but it could be difficult for the DevCorp team to suggest

KiwiCorp are not fulfilling their XP role appropriately without causing contractual or good will issues.

7.2.4. INTERPRETATION

The customer on this project was clearly overloaded. She had the ideal preparation for the role but the requirement to represent thousands of diverse users with a "single voice" took a considerable portion of her time. This is particularly significant when you consider:

- This project is the recommended size for XP
- The customer did not do all of the typical tasks expected of an XP customer including writing user stories and acceptance tests

The different locations and the technical integration issues that arose exacerbated this situation.

7.3. PLANNING

> *Our planning process relies on clearly separating the roles of business people and software people. This ensures that business people make all the business decisions and software people make all the technical decisions. (Beck & Fowler, 2001, p.15)*

The business decisions during the XP planning phase include what to develop (requirements) and in what order (prioritisation). The technical decisions include how to develop the functionality (design) and how long it will take to develop (estimation). These decisions are not independent:

> *A decision of business priority cannot be made without knowing the cost of the feature. An estimate of cost cannot be made without knowing what the business expects from the feature. Thus, the development plan is assembled from the detailed decisions made by the business and technical people (Martin, 2002)*

These activities are explored in further detail below. Note the design activity falls outside of the scope of this project and is only covered in as much as it affects the estimation process. The prioritisation and estimation activities, however, are key activities that involve the customer role.

7.3.1. REQUIREMENTS

The user stories allowed the customer to adapt or evolve the initial requirements as they watched the system develop and learnt what does and does not work and/or matter. Both the customer and development team members commented on the approach:

> *If we'd had to come up with requirements to the [nth] degree right at the beginning, I think it would have been very difficult because I hadn't worked with intranet content management systems so a lot of the time I was sort of learning as I went and once … I saw the bare bones thing, it was much easier to make more detailed decisions for the next level – Customer, KiwiCorp*

> *[On rigid projects you] sit there and write screeds and screeds of detailed information without really knowing whether or not you can do that. People who want a system … don't know what they want until they see something else. You have to have something for them to see before they can really figure out, … realise what they want – Programmer, DevCorp*

The team did not follow the standard XP process for documentation because the user stories were based on a functional specification. The client formally accepted the functional specification. No formal change control was implemented to track the changes between the user stories and the functional specification. Neither was there a formal agreement that the user stories superseded the functional specification. The DevCorp team noted a potential issue with this approach, particularly in an outsourcing arrangement, is scope debate between the organisations.

The customer noted the issue with user stories and communicating the requirements with conversations was an issue of knowledge transfer. The training material development required an understanding of the system typically obtained from the functional specification, but the functional specification no longer reflected the system functionality. No documentation existed that specified the system to the level of detail required by the trainers. Cockburn (2001) discusses this issue and suggests it is best to determine a need for the documentation rather than simply producing it. This suggestion indicates new adopters of XP should consider their knowledge transfer requirements (conversations or documentation) and plan accordingly.

The other change to the process was that the customer did not write user stories and instead the stories were based on the functional specification. The project manager emphasised the time it took the development team to write user stories. He went on to discuss the potential impact of relying on the customer to create the stories:

> *I know strictly speaking that's the clients job if you read the books but I actually did that to sort of streamline the process a little … I don't think the client would … have had enough stories prepared on the day iteration kicked off so we'd have delayed started the iteration and I think … delay[ing the] iteration is the slippery slope – [and against the] .. whole point .. of time blocks and time approaches – Project Manager, DevCorp*

The project manager did not clarify his concern further. Possible interpretations include:

- The time available to the customer *on this project* was insufficient to include this task. The customer noted she needed more time to effectively contribute to requirements development and testing
- The difficulty maybe the technical skills required to write usable user stories for developers as suggested by Beck & Fowler (2001)

It is interesting to note that no one interviewed noted that the development team writing the user stories and the customer reviewing and confirming the stories was an issue or deficiency in the process.

7.3.2. PRIORITISATION

> *I used to think time-boxing was about time, but I've learned that instead it is about forcing hard trade-off decisions through-out the project – Jim Highsmith cited by Beck & Fowler (2001, p. 85)*

One of the key roles of a customer is to make business priority decisions about what functionality is delivered when. During the interview the customer demonstrated her strength of feeling regarding prioritisation by using strong emotional words, shown in bolded typeface in the quote below to provide her verbal emphasis; she also repeated these words. Her words appear to concur with Highsmith's suggestion regarding the difficulty of this process, the trade-offs:

> ***I hate it, I hate** that word prioritise*

> *[and later in the interview]…[I considered] how many people I will have to front up to and say "no actually it's not going to do this" … Usually my manager and I have done that together because she knows what's key to her based on her discussions of the business [it's] about common sense … People say they want this but really, you know, there's [a] work around … Usually I've done that myself and in conjunction with [my manager], when* ***I can't bear the responsibility*** *anymore – Customer, KiwiCorp*

One of the key elements in her ability to successfully make these decisions, as described above, was the support and knowledge of her senior manager:

> *[My manager] was responsible for getting buy-in from the sort of senior level in business and she had been talking to lots of people and had feedback from them that she relayed to me. – Customer, KiwiCorp*

The customer also learnt the importance of not assuming all functionality would be delivered on the high level plan, and so in hindsight should have started prioritising from the start of the process:

> *We thought we'd bitten off what we could chew but we worked out that our eyes are bigger than our stomachs – it works for IT to … so I guess one way of doing it would have been to … do the prioritising right at the time – what can we actually leave till last? And actually leave that till last. – Customer, KiwiCorp*

This point concurs with the XP philosophy of delivering functionality in business priority order. However, this study also suggests that it is a learning process for the customer. The customer will learn the importance of prioritisation as the project progresses. So it is important to review the information learnt (velocity, size of work and so on) and reapply this information to the high level plan regularly. It is important that the customer learn, as soon as possible, the importance of prioritising according to business value.

This project did not prioritise defects and stories within iterations, primarily because defects were not uncovered until late in the process. The development team agreed this was a deviation to XP:

> *Estimating [and] rectifying [bugs] … should have just been … brought into the estimation story task process … because you should be prioritising both in exactly the same way that you prioritise stories.*
>
> *[He goes on to elaborate, however that] I think the whole question of how you manage bug fixes in the story task oriented thing is one that we need to answer – do bug fixes become new tasks in a story called "Bug Fix". I don't know - Pre-sales consultant, DevCorp*

Kini & Collins (2002) discussed a similar finding and draw out a potential issue of this approach that may affect the outsourced XP project:

> *The "green book" (Planning Extreme Programming) suggests that significant bugs should become stories in future iterations. We probably should have tried to convince our client that this was the right course to follow – there's a natural tendency for the client to feel that bug fixes are "owed" to them … above and beyond our work on new stories. (p. 368)*

7.3.3. ESTIMATION

> *Estimating software development is hard; they're [programmers] doing the best they can and they will get better. (Beck & Fowler, 2001, p 18)*

In the process of "getting better" at XP estimation, the programmers on the CMS project learnt a number of lessons. The programmers discussed these lessons in detail during the interview. The following table outlines the lessons learnt using their words.

Lesson	Experience as described by the team
Apply yesterdays weather to all tasks, including non-XP tasks	*The generic lesson I guess is you can apply, even though you are doing part of your project in a non XP way, you can still apply some of the disciplines of XP and yesterday's weather is a very good discipline … [if] it's blatantly obvious, in your experience doing something like increasing the size of a disc volume so that it doesn't run out of disc usually takes 2 hours and if you do it with [an external company who] takes 2 days … – it if was a oncer you could go okay well it took 2 days but if on the second time or the third time it happened, you know what will happen*
Add up-front design tasks for external interfaces because interfaces can not always be easily refactored	*Some things don't get designed particularly well because it's like the opposite of peeling the onion, it's like putting the onion skins on so the one thing [specific example] we came up with a fairly quick solution, it will work and didn't put enough architectural thought into it and that's now come back to haunt us in that there's a lot of sticky tape and glue holding it together. Now normally that's not a problem because you can refactor but this is one component where [BureauCorp] wrote a proportion of it and [BureauCorp] are not working within this process so we're actually sort of stuck with a component of the system where we're not in full control of it, so we can't just decide to refactor the whole thing, we have to re-litigate with the partners involved and there's a whole bunch of why do we have to do this, whose going to pay for this* *[So you need to] identify [external interfaces] up front and … put more effort into [them at] design time, we did put a little design effort up front into [some of them], and that's actually paid dividends.*
Be wary of zero tasks – they add up	*It generally got the point that you tried not to do zero tasks because we found that a lot of our [story] estimates are going over because all the zero tasks were adding up. Even though they might only be half an hour or an hour, they were adding up. So we generally started always assigning some time to it.* *[Another team member noted the importance of] tracking on a day by day basis [to find] a lot of this type of stuff, [including the issue of] zero*

Lesson	Experience as described by the team
	estimate[s]
Programmers should sign-up and estimate tasks – NOT doing so may result in: A lack of ownership Under-utilising the experience of the team	*I think [the estimation process] would have been improved by rigorously adhering to group estimation and tie estimates to individual developers and developers to individual stories.* *[The interviewee provided the following example earlier in the interview regarding this point] .. [one developer] in particular was fuelled by the need to crank stuff out and he was felt like free from the constraints of specs and stuff, so he was really happy about the latitude that it gives you, but he wasn't prepared to do any of the discipline that compensates … So the whole kind of peer pressure part of it, which I kind of like about XP, didn't seem to work for him as well as it should, and again I feel not having a sense of ownership in the estimates might be part of that* *[The interviewee at an earlier point re: developers not estimating their tasks] And I feel that's had a negative impact on quality too in that if you're a person who will take longer to do something than the estimate says, you'll do a crummy job so you can get it done in time.* *[A programmer commenting on meeting the task estimates] we [the pair] looked at the task and thought actually that's not the best way of doing it*

The last lesson is a recommendation in hindsight, and unlike some other lessons, was not implemented later during the project. The XP practice of programmers volunteering for and estimating their tasks has also been considered in other studies. Schalliol (2002) agreed with the CMS projects recommendation. Their experience determined the following benefits of programmers volunteering for and estimating their tasks:

- It was much more difficult for someone to acknowledge that something could not meet a deadline when the confession also implied that the person had estimated the task badly.
- It made it possible for several individuals to employ their particular intelligence to solve many problems (p. 414).

Johansen (2002) on the other hand discovered the centralised approach worked well for their project. The key to their approach was the knowledge of the "central funnel", an experienced lead developer, who was responsible for tracking the estimates as well. The CMS project is similar to this project in terms of the person responsible for the estimation and allocation, so it may be an acceptable ongoing approach. Further consideration of this matter is recommended in order to understand the factors on the project that allow each process to succeed.

Note that these difficulties relate primarily to the developers, but they also concern issues of great importance to the customer to enable them to perform the role successfully.

7.3.4. INTERPRETATION

One of the core practices of XP is separating the business decisions from the technical decisions during the planning process. However, the decisions made by each party are not independent and the two sides must communicate effectively to ensure effective decisions are made. The key lessons learnt from each perspective are outlined below:

Owner	Decision on	Lesson(s)
Business	Requirements	Incremental development allows the customer to *adapt* their requirements as they see the system develop, and understand what works and does not work A new adopter of XP should consider their knowledge transfer requirements (conversations and/or documentation) and plan accordingly If a scope document exists outside of the user stories you should agree the user stories and prioritisation decisions supersede the scope document Customers need to participate in developing the stories but may not need to actually write them
Business	Prioritisation	Prioritisation is difficult Customers learn the importance of prioritisation as the project progresses you should provide opportunities for them to learn the importance early in the process Both defects and user stories should be prioritised as part of the iteration process
Technical	Estimation	Estimation is difficult The team should apply "yesterdays weather" to estimate all tasks not just coding tasks The team should add up-front decision tasks for external interfaces The team should be wary of zero tasks – they add up! The programmers should sign-up and estimate tasks – not doing so may result in a lack of ownership and under-utilising the experience of the team

7.4. AGILE MANAGEMENT

In an agile approach such as XP, projects are developed to maximise the opportunities for regular feedback. Regular feedback allows the team to quickly adjust the course of the project, react to change and learn from mistakes. As Cockburn (2001) notes:

> *There is no substitute for rapid feedback both on the product and on the development process itself, incremental development is perfect for providing feedback points. Short increments help both the requirements and the process itself get repaired quickly. (p. 150)*

This section highlights two activities where the feedback loop malfunctioned, and one where the feedback loop was very effective.

7.4.1. BUSINESS PLANNING RELIES ON PROJECT PLANS

During the planning process on the XP project, the high level plan was not revisited regularly. The impact of not revising the cost, scope and timeline of the project every few months affected both KiwiCorp and DevCorp:

> *I think that overall the really top level crystal ball gaze is really good … and probably accurate enough, but where its been really hard with the estimating is as [we've got closer to the end, my senior manager needed to know] what's the bottom line …[how many days of work left to do before releasing the software". [To get the answer] we would have had to stop doing the work and do detailed estimation on all the user stories, but that didn't really sort of fit the process…So finally we had to leave that to [the DevCorp Project Manager] … he suggested that we do a semi-detailed [estimate of the remaining functionality] so we did that and it was a little bit easier after that … it was still a bit vague but that was something to work with …- Customer (KiwiCorp)*

> *We didn't know the project was going to be this long, but had we known the project was going to be this long I might have thought about site cleaning up – cycling out some of the key people and actually cycling some new blood in, … there's a core of people who've actually been nose to the grind stones for like 10 months now and it's showing signs of – well they've been showing signs for the past few months now of just ? but if I had known early enough we could have cycled people through a lot earlier but it's hard to cycle someone like the lead developer, you can't replace them like a month to go live, from a customer's perspective it doesn't look good so you sort of have to eat the last few drops of blood out of these people. If I had known it was going to be as long, I'd try to cycle some people through – Project Manager (DevCorp)*

XP typically uses a public release every few months as an opportunity to revise the high level plan (Beck & Fowler, 2001). The XP release is a public release, which was not appropriate for this project as it was replacing an existing system and did not fit into the existing release practices of KiwiCorp. The project did divide the high level plan into stages (non-public releases) but did not utilise the high level re-planning activities associated with XP.

Implementing the release planning practice, despite the release being private, would have resulted in an increasing accuracy of the high level plan. The increased accuracy would have allowed the organisations to undertake the required business planning:

- KiwiCorp would be able to plan their implementation tasks, for example, training and migration
- DevCorp would be able to manage their staff effectively to ensure no staff are burnt-out or demotivated

7.4.2. TESTING EARLY IS CRITICAL

One of the key pieces of feedback in XP to agree the functionality of the system is acceptance testing. In XP acceptance testing should occur as part of the three-week iteration (Beck & Fowler, 2001). On this project the acceptance testing was not immediate:

> *We actually got three months into the program before any serious testing was done – Project Manager DevCorp*

The impact of the customer not testing during each iteration was a decrease in the effectiveness of the XP feedback mechanisms. The result was an increase in the cost and timeline of the project due to lost learning opportunities and an inaccurate picture of progress:

> *I think because some of the bugs got left to the end …the time between when a bug was introduced to the system by the function being [built] and the time it gets completed[tested] because things are still fresh in people's memory's, there's still an opportunity to learn from that so we don't repeat it. A good example, [input text fields]. We had either omitted or forgotten in the coding standards to say that all input fields should be checked for the maximum size and it was only after most of the functionality had been written that we found that they … hadn't been checked … now we've got 150 text boxes [to change] whereas if that had come up within the first batch of testing, [no-one] would… have written a text box that didn't have [a check] on it so that would have saved a lot of re-work – Project manager, DevCorp*
>
> *[Regarding defects and starting the process early] You've effectively got a bucket of unscoped work in your plan … So that gave us an improper over-confident view of our velocity – Pre-sales consultant, DevCorp*

7.4.3. REGULAR COMMUNICATION AND FEEDBACK WORKS

The development team was unanimous in their feedback on the success of the regular and informal review opportunities provided by XP including the stand-up meetings, the wall and the post-iteration reviews:

Lesson	Experience as described by the team
Regular stand-up meetings ensure work is not duplicated	*Every morning at 9.15 … everyone who was working on it stood there and said what we did the day before, whether we wrote any little functions that could be re-used and that sort of thing, so that you all knew what everyone else was working on so you didn't both find yourselves going and working on the same thing – Programmer DevCorp*
Regular stand-up meetings facilitate the growth of an effective team culture	*[Discussing the effective team culture] … it grew out of the close knit nature of the team and some of the rituals that we had for example the 9:00 or 9:15 if we were lucky, the stand up meeting every single morning. [And on the rules of the meeting] … The doll that says yadda, yadda, yadda – that was the signal for this conversation has gone on too long in a stand up meeting because the whole concept of a stand up meeting is if you get into detail, some people take it off line and let the meeting go on but that yadda, yadda, yadda was a sort of light hearted indication that the conversation really has gone on – Project Manager DevCorp* *[Regarding the stand-up meetings] people got to tell jokes and have a bit of a laugh. It was a bit of a bonding session. - Lead developer*

Lesson	Experience as described by the team
	DevCorp *[Regarding the stand-up meetings] it just gave a really good team spirit, it kept everyone working together and also knowing what was going on too – Programmer DevCorp*
Minimising technical/design discussions in stand-up meetings is important	*I didn't find the stand-up meetings very useful. I think if I'd been there the whole time then that would have been good and I probably could have got more out of this and contributed more to the stand-up meetings, but because my time at DevCorp was so precious I felt that that wasn't the best way to spend it. If I'd come down after the stand-up meetings and just go through detailed things with [the lead developer]… if I'd had the time to get more involved in the technical stuff, you know, and know what all the jargon means it probably would have helped the end result because I'd be able to see the implications of a[discussion] – Customer KiwiCorp* *Again, we probably didn't use [the stand-up meetings] the way they were supposed to be used. They tended to be longer than 10 minutes most of the time, especially when we had eight people, getting round four pairs did tend to take longer. … We did use those [stand-up] meetings to thrash out difficulties that people were having which often again wasn't what you were supposed to do – Lead developer DevCorp*
Tactile and visible progress indicators work	*The wall was really good, too. Having everything up on the wall was good and the dots and people being able to, because then people can really have a measure of progress. They can actually see how far along the wall they have got. – Team Lead DevCorp* *We actually represented things on the walls in terms of priority left to right, which was the client set's priority and the way we indicated on the stories the progress when it was completed, putting a red dot on it which was like a very visible artefact, as soon as you walk in the room, you know when the iteration started because as soon as you walk in the room, you've actually got a feel for where you're up to and it doesn't take a rocket scientist to work out that if you're two weeks through the iteration and only one third of the stories have got dots on, you're actually*

Lesson	Experience as described by the team
	behind… so everything is a lot more tactile and visible – Project Manager DevCorp
Review the process at the end of each iteration and change the process to fit your needs.	*One of the biggest learning that I got straight up was look if the process isn't working for you, change it and at the end of the iteration we tried to get – even if it was just a couple of hours, … the team together and do a what worked well and what didn't work.*

7.4.4. INTERPRETATION

These findings reinforce the existing experiences outlined in XP literature regarding the importance of regular and constant feedback. Beck and Fowler (2001) liken it to driving a car, where we use small regular changes to take us to our intended destination.

Regular feedback was effective and allowed the team to adjust their course quickly. For example, the regular project reviews allowed the team to determine zero-tasks were affecting the accuracy of their estimates, resulting in them under-estimating stories. The flow-on effect of the increased accuracy of the estimates allowed the customer to make more informed trade-off decisions regarding the stories to include in an iteration.

Conversely delay to the feedback loop impacts the effectiveness of XP. For example, the testing was substantially delayed and resulted in the programmers being over-confident of their velocity. The impact was that no-one was aware the project was off-course. The correction, when it came, resulted in a longer path to reach the destination as the team had already travelled an incorrect route. In the testing example, the programmers needed to re-code 150 fields rather than 1. The delay in the testing feedback loop resulted in rework and the customer making less effective planning decisions due to inaccurate information.

Regular and timely feedback is an important and effective component of XP. XP relies on knowing the project is off-course almost immediately, three weeks is the longest time any activity should occur without review. Delays to this loop will reduce the effectiveness of XP. Teams should be very wary of delaying any feedback mechanisms in XP including testing and iteration reviews.

Another interesting finding was the different reactions to the wall and stand-up meetings demonstrated by the customer and the programmers. One of the key things noted by the DevCorp team is that the use of these rituals developed a team culture. The customer comments that she may have found these rituals more useful had she been on-site 100% of the time. She also comments on the technical information contained in the sessions.

Farell, Narang, Kapitan, & Webber (2002) discuss the division between the programmers and customers on their project. One of the changes implemented to reduce these barriers was placing the customers on-site with the programmers and including the customers in all of the rituals such as the stand-up meetings. The impact of the CMS customer not being on-site may be reducing the cohesiveness of the team as a whole.

Kini & Collins (2002) discuss the intrusion of technical design issues into the stand-up meetings on their project. The increase in length of the meetings caused by this intrusion is an ineffective use of people's time. Their suggestion is to note on the whiteboard during the meeting the design issues as a "promise to have a conversation" after the stand-up meeting. This procedure ensured the programmers were able to raise the issue and obtain the right people involved in the process without stretching the stand-up meetings into long meetings. Shorter stand-up meetings that are not overloaded with technical content are likely to make the meeting more relevant to the customer and not "waste" their time.

7.5. SUMMARY OF INTERPRETATIONS

The study reviewed a small to medium sized[10] project with a team and organisations new to XP. XP was not adopted for the entire lifecycle of the project. The project retained both an up-front requirements gathering phase and an implementation phase, only the development phase of the project used XP. On this case we have found:

- The XP customer role, especially for larger organisations, is a demanding role. It requires preparation, skills, attention to detail, and the ability to make critical decisions:
- The customer was overloaded, despite her apparent ideal preparation for the role and enthusiasm for the process. The time required to represent thousands of diverse users was significant and diminished the time the customer could spend with the programmers. The impact of the diminished time affected the quality of the product and may have increased both the cost and duration of the project due to long feedback loops. The outsourcing nature of this project may have affected this finding but it appears the predominant factor was the large diverse user base.
- The requirements process appears to be resilient to some changes regarding who does the preparation work for the decision. In this project the programmers had significant control over the requirements practice.
- Prioritising functionality is hard as trade-offs need to be made, and not everyone will be "happy" with the decision made by the customer. Customers must have everything necessary to make such decisions effectively.
- Obtaining regular feedback during the project allows the customer to make effective business decisions concerning the system:
- The planning process continues to be a difficult process for both the business people and the technical people, despite the clear distinction between the two roles. To ensure software delivers value, the functionality must be developed in business priority order (highest value first). The team, business and technical representatives, are likely to be optimistic and assume more functionality will be delivered for dollars and time than will actually be possible. Review the high level, as well as the low level, priority decisions regularly to ensure the optimism is tempered as early as possible. Note this review requires all iteration activities to occur, including testing.
- The evolving development approach facilitates learning and easy change. However, it does require forethought regarding on-going activities such as training and maintenance. A project adopting XP should consider the associated non-development activities associated with the project and agree on the knowledge transfer process. These requirements, for documentation or for conversations, will need to be planned.
- The software development estimation process does improve over time but this improvement will be hampered if regular testing does not occur.

---

[10] Using the definition provided by Beck (2000) for XP projects

8. CONCLUSION

> *XP is the most important movement in our field today. I predict that it will be as essential to the present generation as the SEI and its Capability Maturity Model were to the last – Tom DeMarco (Beck & Fowler, 2001, p. xii)*

This study explored what is claimed to be one of the most promising movements in information systems development today, XP. The intention of this study was to investigate an issue critical for whether XP will live up to its promise, and allow us to deliver software that provides business value in the face of vague and changing requirements. One of the pivotal roles in XP is the customer, but little guidance is provided in the literature on the practicalities of succeeding in this role. We have investigated the implementation of the XP customer role and focussed on the critical planning activities of the project.

We believe this is the first study of its kind, and a valuable contribution to the study of information systems development.

8.1. CONTRIBUTIONS

We used an interpretative in-depth case study to explore a successful XP Project. We obtained multiple perspectives on the implementation of the customer role within the planning process of XP and found that:

- The XP customer role, especially for larger organisations, is a demanding role. It requires preparation, skills, attention to detail, and the ability to make critical decisions.
- Obtaining regular feedback during the project allows the customer to make effective business decisions concerning the system.
- The development team must carry out key XP practices in order to enable feedback necessary for the customer to make effective decisions.

The XP project team have participated in a process that allowed them to learn and reflect on their existing process. DevCorp has expressed their on-going interest in the study and they have requested a seminar on this study.

Of course, this study was based on multiple perspectives of a single case and the subject deserves full attention, as we discuss below.

8.2. COMPARISONS WITH RELATED STUDIES

XP is relatively new and as such little research exists on XP. The majority of the studies available on XP are practitioner reports. Such reports (Farell, 2002; Schalliol, 2002) have provided valuable insights into the issues encountered during the implementation of XP, including that of the customer role. However, these reports have not used multiple perspectives, that of the customer and the programmers, to review the effectiveness of XP as this study has done. One of the few research reports (Gittins, 2002) that considers the implementation of the customer role provides little guidance on the detail involved in the customer role. Our study considers the required characteristics and skills of the customer and explores the day-to-day challenges encountered in this role.

8.3. FUTURE RESEARCH

This study was exploratory in nature, it provides an initial foundation from which future research can be conducted. Suggestions include:

- Increasing the depth of this study with a follow-up longitudinal study on the CMS project in six months time. What further experiences have been undertaken by the team to facilitate the effectiveness of the customer role?
- Increasing the depth of this study by widening the perspectives. A full 360 degree review of the system and process which would include the perspectives of the project sponsor, the users of the system, the acceptance testers and BureauCorp personnel.
- Increasing the breadth of this study by exploring multiple cases using a maximum variation sampling technique to increase the validity of the findings of this research.
- Increasing the breadth of this study by undertaking a quantitative investigation using the findings of this study.

9. REFERENCES

Baskerville, R. L. (1999). Investigating Information Systems with Action Research. *Communications of the Association for Information Systems, 2*, 1-23.

Baskerville, R. L., & Stage, J. (1996). Controlling Prototype Development Though Risk Analysis. *MIS Quarterly, 20*(4), pp. 481 - 502.

Beck, K. (2000). *eXtreme Programming Explained: Embrace Change*: Addison Wesley.

Beck, K., & Fowler, M. (2001). *Planning Extreme Programming*: Addison Wesley.

Benbasat, I., Goldstein, D., & Mead, M. (1987). The Case Research Strategy in Studies of Information Systems. *MIS Quarterly, 11*(3), pp. 369 - 386.

Bostom, R. P., & Thomas, B. D. (1983). *Achieving excellence in communications: a key to developing complete, accurate and shared information requirements.* Paper presented at the Twentieth annual computer personnel on research conference, Virginina, United States.

Cockburn, A. (1999). *Characterising people as non-linear, first order components in software development.*Unpublished manuscript.

Cockburn, A. (2001). *Agile software development*: Addison-Wesley.

Cohn, T., & Paul, R. (2001). *A comparison of requirements engineering in eXtreme programming (XP) and conventional software development methodologies.* Paper presented at the Seventh Americas Conference on Information Systems.

Creswell, J. (1997). *Research design: qualitative and quantitative approaches*. Beverley Hills, CA.: Sage Publications.

Creswell, J. (1998). *Qualitative Inquiry and Research Design: Choosing among five traditions*. Thousand Oaks: Sage Publications.

Deursen, A. (2001). *Customer involvement in eXtreme programming.* Paper presented at the Second international conference on eXtreme programming and flexible processes in software engineering, Italy.

Farell, C., Narang, R, Kapitan, S & Webber, H. (2002). *Towards an effective onsite customer practice.* Paper presented at the Third International Conference on eXtreme Programming and Agile Process in Software Engineering, Italy.

Fitzgerald, B. (2000). Systems development methodologies: the problem of tenses. *Information technology and people, 13*(3), pp. 174 - 185.

Fowler, M. (2002). *XP Customer Quotes*. WikiWikiWeb. Retrieved, from the World Wide Web: http://www.c2.com/cgi/wiki?XPCustomerQuotes

Gittins, R., Hope, Sian & Williams, Ifor. (2002). Qualitative Studies of XP in a Medium-Sized Business. In M. Marchesi, Succi, G, Wells, D & Williams, L (Ed.), *EXtreme Programming Perspectives* (pp. 421 - 435): Addison-Wesley.

Gordon, V. S. B., James. M. (1995). Rapid prototyping: lessons learned. *IEEE Software, 12*(1), 85-95.

Highsmith, J. (2001). The great methodologies debate, part 1: opening statement. *Cutter IT Journal, 14*(12), pp. 2 - 4.

Jeffries, R. (2002). XP and evolutionary prototyping. In A. Martin (Ed.).

Johansen, K., Stauffer, Ron and Turner, D. (2002). Learning by Doing: Why XP Doesn't Sell. In M. Marchesi, Succi, G, Wells, D & Williams, L (Ed.), *EXtreme Programming Perspectives* (pp. 411 - 420): Addison-Wesley.

Kini, N., Collins, S. (2002). Lessons learnt from an XP project. In M. Marchesi, Succi, G, Wells, D & Williams, L (Ed.), *EXtreme Programming Perspectives* (pp. 363 - 374): Addison-Wesley.

Lee, T., W. (1999). *Using qualitative methods in organizational research*. Thousand Oaks: Sage Publications, Inc.

Marchesi, M., Succi, G, Wells, D & Williams, L. (2002). XR: Extreme Reality - real - life experiences. In M. Marchesi, Succi, G, Wells, D & Williams, L (Ed.), *EXtreme Programming Practices*: Addison-Wesley.

Martin, R. (2002). *XP Customer Quotes*. WikiWikiWeb. Retrieved, from the World Wide Web: http://www.c2.com/cgi/wiki?XPCustomerQuotes

McConnell, S. (1996). *Rapid Development: Taming Wild Software Schedules*: Microsoft Press.

McKeen, J., Guimaraes, T., & Wetherbe, J. (1994). The Relationship Between User Participation and Use Satisfaction: An Investigation of Four Contingency Factors. *MIS Quarterly, 18*(4).

Myers, M., & Young, L. (1997). Hidden agendas, power and managerial assumptions in information systems development: an ethnographic case study. *Information Technology and People, 10*(3), pp. 224 - 240.

Nandhakumar, J., & Avison, D. E. (1999). The fiction of methodological development: a field study of information systems development. *Information Technology and People, 12*(2), pp. 176 - 191.

Orlikowski, W., & Baroudi, J. (1991). Studying information technology in organizations: research approaches and assumptions. *Information Systems Research, 2*(1), 1-28.

Orlikowski, W., & Gash, D. (1994). Technological frames: making sense of information technology in organisations. *ACM Transactions on Information Systems, 12*(2), pp. 174 - 207.

Robson, C. (1993). *Real world research: a resource for social scientists and practitioner researchers*. Oxford: Blackwell Publishers Ltd.

Russo, N. L., & Stolterman, E. (2000). Exploring the assumptions underlying information systems methodologies: their impact on past, present and future ISM research. *Information Technology and People, 13*(4), pp. 313-327.

Schalliol, G. (2002). Challenges for Analysts on a Large XP Project. In M. Marchesi, Succi, G, Wells, D & Williams, L (Ed.), *EXtreme Programming Perspectives* (pp. 375 - 386): Addison-Wesley.

Urquhart, C. (1998). *Analysts and clients in conversation: cases in early requirements gathering.* Paper presented at the International Conference on Information Systems, Helsinki, Finland.

Wastell, D. (1999). Learning Dysfunctions in Information Systems Development: Overcoming the Social Defenses with Transitional Objects. *MIS Quarterly, 23*(4), pp. 581 - 600.

APPENDIX A: INTERVIEWEE BACKGROUND EXPERIENCES

Role	Previous experience
Project manager	The project manager has over 23 years of IT experience in a variety of roles including mainframe computer operator, programmer, consultant and latterly as a project manager for web development projects. He has been involved in a number of different projects of varying sizes in both out-sourced and internal software development projects. He has been exposed during this period to a variety of software development methods.
Lead developer	The lead developer has over 3 and a 1/2 years of IT experience, primarily as a developer or lead developer on small web development projects. The project under study was the largest project she had worked on. She has primarily been exposed to semi-structured methods.
Programmer	The developer has over 6 years of IT experience, primarily as the interface between the business representatives and the development team. This project is her first project as a developer. She has primarily been involved in projects with lightweight, non-rigid software development methods. Her experience has been this type of approach is much more likely to deliver than rigid development processes.
Pre-sales consultant	The consultant has over 10(?) years of IT experience in a variety of roles including as a developer, team leader, technical writer, help desk operator, informal project manager and pre-sales consultant. He is typically heavily involved in the scoping, estimation and proposal development at The Company. He tends to be the interface between the business people and the development team. He has been exposed to a variety of software development methods and has a keen interest in exploring ways to improve the process of developing software, particularly the peopleware aspects of software development.
Project lead	The customer is a trained librarian. Her IT experience includes being the business representative over the last four years on an on-line library system, including a major 6 month re-development project. She has been primarily involved in projects with structured software development methods. She has a keen interest in IT projects and is considering developing her analysis and testing skills.

APPENDIX B: DEVCORP PROPOSAL

INTRODUCTION

The purpose of this document is to establish a research project with DevCorp. In order to achieve this purpose this document covers the following:

- Objectives of the project
- Scope of the project and
- Approach to the project.

This proposal reflects an initial project concept that focuses on the *planning and estimation* practices of the Collaborative Development Method light (CDM Light). We look forward to working with you to mould it into a project that will meet our mutual objectives.

OBJECTIVE

- To recommend the key planning & estimation practice improvements to be implemented on your next CDM light project
- To increase academic and industry knowledge of the planning & estimation practices that work and do not work when implementing agile development practices.

SCOPE

Is	Is not
To review the recent KiwiCorp intranet content management project.	To review other projects implementing the CDM light method.
To explore the planning & estimation practices utilised on the project including: ▪ Release & iteration planning ▪ Writing user stories ▪ Estimation process ▪ Priority planning ▪ First plan versus the on-going plans ▪ Planning meetings ▪ Contractual issues ▪ On-site customer involvement	To explore the development specific practices including: ▪ Pair programming ▪ Refactoring ▪ Metaphor ▪ Testing ▪ Collective ownership ▪ Continuous integration ▪ 40-hour week ▪ Coding standards
To understand and document the implementation of the 'text book' processes of planning & estimation including: Activities that worked well for the team when using these processes (what we must keep on the next project) Issues and difficulties the team encountered with these processes (what we need to change on the next project)	To implement change or prepare a change management plan.

Is	Is not
Suggested improvements to these processes (how we might change these processes on the next project)	
To provide recommendations for improvement based on literature.	To provide recommendations for improvement based on the consultant's experience.

APPROACH

We will utilise a mixture of semi-structured one-on-one interviews with the project team and project artefact review to understand the project team's experience with this method.

The *interviews* will focus on the key project roles associated with the planning & estimation practices including:

- Customer.
- Programmer(s)
- Coach
- Big Boss

The interviews will occur in a meeting room at DevCorp's premises. The on-site location will minimise the disturbance to the interviewee's day-to-day activities. All interviews will be taped and later transcribed. The interviewees will be asked to validate the transcription of the interview and the interpreted findings.

The *artefact review* will focus on the key planning & estimation artefacts including user stories, plans and project walls. This review will allow us to understand the adherence and useful variations to the CDM light and eXtreme Programming standards.

The *costs* to DevCorp are expected to be minimal and time based only. The total time required of each identified team member (maximum of 7) is expected to be between 3 to 4 hours over a period of 6 weeks. The maximum period in 1 day for any team member will be 1 hour. All interactions would be arranged around the team members existing commitments, so disruption should remain minimal.

All raw data obtained during this project will remain *confidential* to the researcher, Angela Martin and her direct supervisors, Robert Biddle and James Noble. All findings will be reported at an aggregate level with the company and individuals retaining their anonymity. However DevCorp is welcome to be identified in subsequent publications should they wish to be acknowledged as a practice leader.

The research report will be provided to DevCorp at the completion of the project. This report is due for university appraisal by 15 October. A summarised version targeted at the practitioner will be made available to DevCorp on 30 October, if requested.

APPROVAL TO PROCEED

DevCorp authorises this project to proceed under the conditions stated in this proposal.

Name: Signature:

Date:

APPENDIX C: PARTICIPANT CONSENT LETTER

[Participant's name]

[Company name]

Wellington

Dear [Participant's name],

INFO 408: Project in Information Systems

This letter follows up our recent email conversation in which I expressed an interest in conducting an interview with you as part of my research project requirements for INFO 408: Project in Information Systems. Prior to conducting the interview, Victoria University of Wellington requires that I obtain your written informed consent. This consent is a normal part of any research project and forms one criterion of the Faculty of Commerce and Administration's Human Ethics Committee Guidelines that I must meet.

The record of the interview (tapes or notes) will not contain a personal identifier - only generic titles will be used - Mgr 1 of Coy 1. I will keep a separate log of the link between your personal identity and generic identifier. This log will be kept separate from the survey data. The log will be destroyed once the report is assessed.

Attached for your information, therefore, is:

- A copy of the *proposal* submitted to [Pre-sales consultant]. This proposal outlines the purpose, scope & approach to the project.
- An *information sheet* on the proposed interview. This sheet complements the proposal by highlighting the academic aspects of this research, including your rights.
- A *consent form* for your signature. The consent form informs me that you agree to the interview and the proposed use of your information and opinions, and that you are aware of the research conditions, including the purpose and use I will make of your comments.

I have also enclosed a copy of the interview questionnaire so that you will have a good sense of the nature of the questions I intend to ask. Please fell free to contact me on 021 668 511 or my supervisors (Robert Biddle on 463 5833 or James Noble on 463 6736) if you require further information of the informed consent requirement, or if you would prefer not to be an interview participant.

Yours sincerely

Angela Martin (Student)

ETHICAL APPROVAL FOR INTERVIEW: INFORMATION SHEET

Nature and purpose of research

This research project contributes towards the course requirements for a Honours or Masters degree in Information Systems. The purpose of the research is to investigate one application of a software development process, and to critically assess the issues associated with the use of this process.

The research will be carried out by interviewing up to six staff members of DevCorp and two staff members of the client organisation, KiwiCorp. An interview questionnaire has been attached to this information sheet outlining what information will be sought during the interviews: the interviewee's own opinions, viewpoint and experience of the software development process within the KiwiCorp Intranet Project. No personal details or information will be collected.

Purpose of the material collected

Material collected from the interviews will be used within my research project as evidence of the application of a software development process and the organisational issues associated with its use. The final research report will only be accessible to the course supervisor, Dr Beverley Hope, and the broad findings of the research will be presented during a class seminar. Written interview notes will be stored at my home and destroyed at the completion of the project.

Before the interview proceeds, I need to be sure that I have permission of your employer for you to be interviewed. The consent form will have a statement to this effect. I will request the attached proposal is signed and hence provides that permission.

Confidentiality

All raw data will be kept confidential to my supervisor and me. The research report (or any conference papers or journal articles that may result from the study) will not identify you or your employer. There will be an opportunity to review any written notes that result from the interview to ensure factual material is recorded accurately.

Timing of the Project

Interviews are scheduled for August and September, but the report will not be finalised until mid October 2002. The interview will take about one hour. The possibility of a further interview will be discussed with you at the time of the first interview

University insurance cover

Please note that the University retains insurance cover against claims relating to harm, loss or damages suffered by participation in research projects as a result of any negligent act, error or omission by or on behalf of the university.

Victoria University of Wellington

Consent to Participation in Research

Project Title: Exploring the implementation of the planning & estimation practices on an XP project

I have been given and have understood an explanation of this research project and the confidentiality conditions. I have had an opportunity to ask questions and have them answered to my satisfaction.

I agree to be interviewed by Angela Martin for the purpose of this research for her Honours/Masters degree, and I consent to the collection and use of my perceptions, experiences, opinions and information in this research.

I understand that I may withdraw from this research or request anonymity at any time without penalty or explanation.

I confirm that I

DO DO NOT

have approval by my employer to participate in this research project.

Do you agree to have interviews tape-recorded?

YES NO

Name: ____________________________

Signed: ____________________________

Date: ____________________________

APPENDIX D: INTERVIEW AGENDA

INTERVIEW DETAILS:

Date:	To be decided
Topic(s)	The implementation of the planning & estimation practices on the KiwiCorp Intranet Project

OUTCOME:

- To recommend the key planning & estimation practice improvements to be implemented on your next CDM light project
- To increase academic and industry knowledge of the planning & estimation practices that work and do not work when implementing agile development practices.

AGENDA:

No.	Description	Duration
1.	Agree outcome, agenda & rules for the interview	05 mins
2.	Previous project and software development process experience of interviewee	10 mins
3.	Planning & estimation practices & processes (what we did on the project)	15 mins
4.	Activities that worked well for the team when using these processes (what we must keep on the next project)	10 mins
5.	Issues and difficulties the team encountered with these processes (what we need to change on the next project)	10 mins
6.	Suggested improvements to these processes (how we might change these processes on the next project)	05 mins
7.	Wrap-up Review progress Next steps	05 mins

RULES:

- The interview will be tape recorded to reduce the risk of the interviewer not obtaining all of the information provided by the interviewee during the interview.
- The transcription of the interview and subsequent findings will be provided to the interviewee to ensure all information has been recorded and interpreted accurately.
- To ensure all items are covered in sufficient detail any discussion concerning development practices will be noted in a register for further discussion.