Technological University Dublin
ARROW@TU Dublin

Dissertations

School of Computing

2009-07-01

Knowledge transfer within a software development team

Geraldine Conway
Technological University Dublin, brendan.tierney@dit.ie

Follow this and additional works at: https://arrow.dit.ie/scschcomdis

Part of the Computer Engineering Commons

Recommended Citation
Conway, Geraldine, "Knowledge transfer within a software development team" (2009). *Dissertations*. 24.
https://arrow.dit.ie/scschcomdis/24

This Dissertation is brought to you for free and open access by the School of Computing at ARROW@TU Dublin. It has been accepted for inclusion in Dissertations by an authorized administrator of ARROW@TU Dublin. For more information, please contact yvonne.desmond@dit.ie, arrow.admin@dit.ie, brian.widdis@dit.ie.

This work is licensed under a Creative Commons Attribution-Noncommercial-Share Alike 3.0 License

Knowledge Transfer within a Software Development Team

Geraldine Conway

A dissertation submitted in partial fulfilment of the requirements of
Dublin Institute of Technology for the degree of
M.Sc. in Computing (Knowledge Management)

July 2009

I certify that this dissertation, which I now submit for examination for the award of MSc in Computing (Knowledge Management), is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the test of my work.

This dissertation was prepared according to the regulations for postgraduate study of the Dublin Institute of Technology and has not been submitted in whole or part for an award in any other Institute or University.

The work reported on in this dissertation conforms to the principles and requirements of the Institute's guidelines for ethics in research.

Signed: _____

Date: ***29th July 2009***

ABSTRACT

Knowledge in a software company is the blood of the company, not surviving without the valuable knowledge running through its veins. Knowledge is no longer seen as an object of competitive advantage but is the core of how an organisation operates.

The transfer of tacit knowledge into a format that is usable and can be transmitted to other individuals is a challenge in any organisation due to the nature of this knowledge. Tacit knowledge is a challenge to convert into an explicit form as it is hard to contextualise this knowledge in order for it to be reusable for others. Investigation is carried out into whether explicit and tacit knowledge that has been documented throughout a software project can be recorded and easily transferred to someone outside of the original project team.

The research demonstrated how a Wiki, if used in the correct way, is a sufficient and effective tool for managing important tacit and explicit knowledge within a software team. [1] Knowledge exists in a spiral and grows and matures as it moves though this circle of communication. The people and the technology using it need to be able to deal with its ever-changing nature. The first issue highlighted as the main barrier to knowledge transfer was ensuring that knowledge is in a context that is going to be understood by users who had no previous experience with it. The second issue is regarding the addressing of factors that might result in the contributor of tacit knowledge not participating in the knowledge transfer process.

The transfer of project knowledge is possible to users outside of a project team, but only if the issue of user contribution and the contextual knowledge issue are addressed.

Key words: *knowledge transfer, spiral of knowledge, Wiki, Semantic Wiki, software projects, knowledge management project.*

---

[1] A Wiki is a new technology that supports “conversational” knowledge creation and sharing. A Wiki is a collaboratively created and iteratively improved set of web pages, together with the software that manages the web pages. (Wagner 2004)

ACKNOWLEDGEMENTS

I would like to express my sincere thanks to Ciarán O'Leary who has provided great guidance and help throughout the entire time of the dissertation. I would also like to thank those within my organisation who assisted with the knowledge management project and provided support throughout the project. I would also like to thank Patrick Kiernan, Ronan O Hicidhe, Michael Cunningham and all of family who provided great support through out the research period.

TABLE OF CONTENTS

1 INTRODUCTION 1

1.1 INTRODUCTION 1
1.2 BACKGROUND 5
1.3 RESEARCH PROBLEM 8
1.4 INTELLECTUAL CHALLENGE 8
1.5 RESEARCH OBJECTIVES 9
1.6 RESEARCH METHODOLOGY 10
1.7 RESOURCES 12
1.8 ORGANISATION OF THE DISSERTATION 14

2 KNOWLEDGE MANAGEMENT 17

2.1 INTRODUCTION 17
2.2 WHAT IS KNOWLEDGE MANAGEMENT? 17
2.3 TYPES OF KNOWLEDGE 20
2.3.1 Explicit Knowledge 20
2.3.2 Tacit Knowledge 20
2.3.3 The Spiral of Knowledge 20
2.3.4 The issues regarding Knowledge Transfer in the Spiral of Knowledge 22
2.4 CONCLUSION 25

3 KNOWLEDGE TRANSFER 27

3.1 INTRODUCTION 27
3.2 KNOWLEDGE GENERATION 27
3.3 KNOWLEDGE SORTING AND CLASSIFICATION 28
3.3.1 Email 29
3.3.2 Instant Messaging 33
3.3.3 Face to Face 36
3.4 KNOWLEDGE TRANSFER 37
3.5 KNOWLEDGE RE-USE 38
3.6 CONCLUSION 39

4 KNOWLEDGE LOSS IN A SOFTWARE LIFECYCLE 42

4.1 INTRODUCTION 42
4.2 STANDARD SOFTWARE LIFECYCLE 42
4.3 SPIRAL LIFECYCLE 43
4.4 KNOWLEDGE LOSS IN STANDARD SOFTWARE LIFECYCLES 44
4.5 METHODOLOGIES TO ASSIST KNOWLEDGE TRANSFER IN A SOFTWARE PROJECT 48
4.5.1 Extreme Programming 49
4.5.2 SCRUM 50
4.6 CONCLUSION 50

5 KNOWLEDGE TRANSFER TOOLS 52

5.1 INTRODUCTION 52
5.2 SOCIAL SOFTWARE 52
5.3 WIKI OVERVIEW 53
5.4 WIKI PRODUCTS 55
5.5 FUNCTIONALITY OF A WIKI 57
5.5.1 Articles 57
5.5.2 Categories and Linking 58
5.5.3 Searching 59
5.5.4 Discussion Forums/Talk Pages 59
5.6 ADVANCED FUNCTIONALITY OF WIKIS 60
5.6.1 Wiki Mind Maps 61
5.6.2 Document Templates 61
5.7 PEOPLE ISSUES 62
5.8 WIKI AND KNOWLEDGE TRANSFER 63
5.8.1 Semantic Wikis 64
5.8.2 Annotations 65
5.8.3 Predictive Annotations 66
5.8.4 Knowledge Wikis 66
5.9 TAG CLOUDS 68
5.10 CONCLUSION 70

6 RELATED CASE STUDIES 72

6.1 CASE STUDY REVIEW 1 72
6.2 CASE STUDY REVIEW 2 73

6.3 CASE STUDY REVIEW 3 75
6.4 CASE STUDY REVIEW 4 76
6.5 CASE STUDY REVIEW 5 78
6.6 CASE STUDY REVIEW 6 79
6.7 CASE STUDY REVIEW 7 81
6.9 CONCLUSION 83
6.9.1 Process 84
6.9.2 People 85
6.9.3 Technology 87

7 EXPERIMENTATION & EVALUATION 88

7.1 INTRODUCTION 88
7.2 THE APPROACH 88
7.2.1 The organisational culture 89
7.2.2 Requirements of the Organisation 90
7.2.3 Management Buy In 92
7.2.4 Team Presentation and Brainstorming 92
7.2.5 Lessons Learned Documentation and User Feedback 92
7.2.6 Surveying the Team 93
7.2.7 Requirements Definition and Solution Design Document 93
7.2.8 Communication Strategy 94
7.2.9 Implementation Approach 94
7.2.10 Rollout All Tools 95
7.2.11 Match of requirements against Existing 96
7.3 APPROACH ON UPLOAD OF EXISTING INFORMATION 98
7.4.1 Tagging 100
7.4.2 Tag Cloud results 100
7.4.3 Categories in Media Wiki 101
7.4.4 Semantic Wikis 101
7.4.5 Document Templates 102
7.4.6 Communication 103
7.4.7 Integration of other systems 103
7.4.8 Functionality of the Wiki 104
7.5 OVERALL EVALUATION APPROACH 104

7.5.1 Questionnaire Analysis *105*
7.5.2 Questionnaire Goals *105*
7.5.3 Target Audience *105*
7.5.4 Project Team Questionnaire *106*
7.5.5 Pre Project Survey Results *107*
7.5.6 Post Implementation Survey *110*
7.6 CONCLUSION 118

8 CONCLUSION 120

8.1 INTRODUCTION 120
8.2 RESEARCH DEFINITION & RESEARCH OVERVIEW 120
8.3 CONTRIBUTIONS TO THE BODY OF KNOWLEDGE 123
8.4 EXPERIMENTATION, EVALUATION AND LIMITATION 127
8.5 FUTURE WORK & RESEARCH 128
8.6 CONCLUSION 130

BIBLIOGRAPHY 133

APPENDIX A 144

APPENDIX B 146

APPENDIX C 152

APPENDIX D 161

APPENDIX E 163

APPENDIX F 165

APPENDIX G 166

APPENDIX H 167

TABLE OF FIGURES

FIGURE 1-1 WATERFALL METHODOLOGY (KASSER 2002) 3
FIGURE 2-1 EXAMPLES OF TACT AND EXPLICIT KNOWLEDGE IN THE SOFTWARE LIFECYCLE 18
FIGURE 2-2 SPIRAL OF KNOWLEDGE (GIRARD 2006)..... 21
FIGURE 2-3 – SOFTWARE TRANSFER PROJECT WITH NO KNOWLEDGE TRANSFER ISSUES 23
FIGURE 2-4 – ISSUE OF TACIT TO TACIT COMMUNICATION OUTSIDE THE SPIRAL OF KNOWLEDGE..... 24
FIGURE 3-1 TACIT TO EXPLICIT KNOWLEDGE CONVERSION (TALISAYON 2008)..... 29
FIGURE 3-2 EMAIL CLASSIFICATION - OUTLOOK RULES..... 31
FIGURE 3-3 FORMAL AND INFORMAL KNOWLEDGE 32
FIGURE 3-4 EMAIL TO FEATURE – MEDIA WIKI..... 33
FIGURE 3-5 TYPES OF CONVERSATIONAL KNOWLEDGE..... 35
FIGURE 3-6 EXPLICIT TO TACIT TRANSFER TALISAYON (2008)..... 38
FIGURE 4-1 ROYCE WATERFALL MODEL (BOEHM 2006) 43
FIGURE 4-2 KNOWLEDGE LOSS IN THE WATERFALL LIFECYCLE 46
FIGURE 5-1 HOW TO CREATE A WIKI PAGE 57
FIGURE 5-2 REVISION HISTORY - WIKIPEDIA..... 58
FIGURE 5-3 KNOWLEDGE REFINING (WIKIS) 60
FIGURE 5-4 WIKI MINDMAP.ORG 61
FIGURE 7-1 COMPANY B-2-B WIKI HOMEPAGE 98
FIGURE 7-2 RESULTS OF TAG CLOUDS PROCESS ON EMAIL FOLDER 100
FIGURE 7-3 VIEW OF WIKI ARTICLES CREATED 100
FIGURE 7-4 WIKI MIND MAP 104

TABLE OF TABLES

TABLE 1-1 ISSUES PREVENTING EFFICIENT KNOWLEDGE TRANSFER..... 5
TABLE 5-1 WIKI TOOL COMPARISON..... 56
TABLE 7-1 COMPANY REQUIREMENTS V'S MEDIAWIKI FUNCTIONALITY 97
TABLE 7-2 PRE PROJECT IMPLEMENTATION QUESTIONNAIRE RESULTS 108
TABLE 7-3 POST PROJECT IMPLIMENTATION QUESTIONNAIRE RESULTS 111
TABLE 7-4 UAT TESTING RESULTS ROUND 1 115

1 INTRODUCTION

1.1 Introduction

As highlighted by Turk (2008) a person who graduated yesterday and stops studying today is uneducated tomorrow. Individuals need to continually learn to ensure that they are operating with up to date information. If individuals make decisions on information that is out of date, the rippling effect can have a commercial and detrimental effect to your software business (Goodman 2003). An example of this would be spending time fixing a bug in the system for a customer that was already fixed in a previous release but never distributed at the time of fixing. The results of this mistake would be time spent redoing something already implemented.

The goal of this research is to investigate if a knowledge management tool can successfully capture tacit and explicit knowledge relating to a software project and if this knowledge can be reused and merged with knowledge of others, inside and outside of the project team.

Many modern software engineering projects suffer because of a lack of understanding of what another team member knows or what was achieved previously in a project, resulting in repeat incorrect behaviours (Charette n.d). Ruhe (2000) highlights that members of a project team are not always aware of knowledge held by other team members and in other areas of the team and, therefore are often required to reinvent the wheel. The Standish Report (1995) showed after a study was completed on 8000 software projects in 352 companies that 32% of the projects failed. The remaining 70% failed to deliver what was expected. In another study by Jones (1991) it was highlighted that 35% of requirements change throughout the software lifecycle. It is important to realise that if project team members can not effectively communicate what they know, requirements will be elicited without key information, developers will develop without the correct knowledge and as a result customers will not get what they asked for. It will then prove a real challenge to transfer this undocumented knowledge to users who were not a part of the project team. The goal of the research is to review how tacit knowledge

can be transmitted from project team members to users who were not part of the original project team.

Following a study over a three month period of a software project, which took place in a small Software Development organisation specialising in E-Learning (Company E-Learn), the issues regarding the tacit knowledge transfer process were investigated. A knowledge management initiative and a knowledge management technology were put into place to try to overcome these issues. A Wiki tool was put in place to create a record of all project knowledge of both a tacit and explicit nature for one particular customer, Company B-2-B. [2] This Wiki technology was intended to serve as a record of all project knowledge in order to facilitate users outside of the project team who needed easy access to this knowledge without asking individuals that were involved in the project. Examples of these users were new employees within the organisation or users that were allocated to different customers within the organisation.

Knowledge within a software project is ever changing due to requirements shifting and designs changing due to the requirements being altered. Only some changes are reflected in documents and some changes are discussed via face-to-face communication. As a result of poor communication of tacit knowledge at the time of execution of a software project, little documentation or record is taken from face to face interactions. As a result of this, knowledge can never be fully transmitted to others outside of the project team.

As highlighted by Cloutier (2004) software projects had been traditionally implemented using the well-known software methodology, the Waterfall Methodology. The Waterfall Methodology involves taking customer requirements, documenting these in a design document and the development team implementing these requirements to produce the required solution. Upon development of the

[2] Company B-2-B is a Global organisation that contracted Company E-learn to build a Learning Management system for their employees. Company E-Learn tried to store Company B-2-B's tacit and explicit knowledge in a tool that was available to all users inside Company E-Learn.

requirements, testing takes place after development and once tested the solution is delivered to the customer (Khalifa & Verner 2000).

The Waterfall Methodology is an approach to software development in a sequential matter, i.e. phases happen one after another. Jones (2001) highlighted in *Applied Software Measurement* that 63% of projects failed in 1993 that used the Waterfall Methodology. Within Company E-Learn, the Waterfall Methodology was used for all projects, which meant that knowledge was transferred from one phase to another in a sequential manner. As face-to-face conversations were had throughout the project lifecycle, less and less was documented as time moved on. As a result of this methodology, any tacit knowledge that was not recorded did not reflect the format at which it was created.

Figure 1-1 Waterfall Methodology (Kasser 2002)

The research within this paper examines:

- What are the goals and aims of knowledge transfer within a software team?
- Why tacit knowledge is difficult to transfer to an explicit format?
- Can a tool, a Wiki tool, manage the tacit knowledge of a project?
- What are the main barriers to knowledge management project and knowledge transfer?
- Can tacit knowledge be transferred to individuals who were not a member of the original project team?

The following approach was taken to try to transmit the project knowledge of Project B-2-B to an explicit format for reuse by users outside of the project team.

1) Review of past implementations of similar scope was undertaken.
2) An approach in terms of a plan to organise, collect and disseminate project knowledge was undertaken which involved the following:

    a. Every phase within the software lifecycle was reviewed and the knowledge that might exist was classified into either tacit or explicit knowledge.
    b. Communication media for the transfer of tacit knowledge was identified.
    c. Tacit knowledge that existed in the different communication media was classified depending on its content into categories of knowledge.
    d. Both the explicit and tacit knowledge was transferred into a project Wiki for all users to access.
    e. Experiments were carried out with users inside and outside of the project team in order to analyse the success of knowledge transfer.

It was proven that knowledge transfer of both tacit and explicit knowledge is possible but certain barriers such as organisational culture, environment for transfer and processes for transfer have to be addressed.

As proven from the implementation in Company E-Learn and other knowledge management projects such as highlighted by Madden (2008) and McMahon (2008), there are barriers to the transfer of tacit knowledge such as resistance by users to share, technology to support knowledge sharing activities and an overall communication of what is expected in terms of the outcomes of sharing. As highlighted by Collison & Parcel (2001) the main issues fall under certain categories of organisational issues. Collison and Parcel highlighted that issues relating to knowledge management can be broken down into 70% of issues relating to people, 20% relating to process issues and 10% relating to technology issues. As highlighted below, each component of success for knowledge management does not exist on its own and each component needs to work in unison with each other to contribute to overall successful knowledge management. Knowledge transfer to others outside of the project team is achievable via a Wiki technology if all barriers, people, process and technology are addressed.

Examples of these barriers relating to people, process and technology, are highlighted below and discussed in the evaluation chapter of this research.

People	Processes	Technology
Knowledgeable individuals not sharing knowledge	Process of sharing tacit knowledge with others not being defined	Technology to support storage of Tacit Knowledge not existing
Individuals not wanting to learn	Process of converting tacit knowledge to explicit knowledge not being defined	Technology to support communication not existing
Knowledge Project Managers not knowing what should be shared	Process of merging old and new explicit knowledge together to form something new not being defined	Technology to support Collaboration not existing
Knowledge Marketing individuals not selling the initiative internally	Process of translating explicit knowledge into the practical doing of what has been documented not being defined	Technology to support Searching not existing
Senior Management not providing buy in for the project		Technology to support Changing Knowledge not existing
People outside the team not wanting to learn what happened on the project		Technology to support dispersed users not existing
		Technology to support different version of knowledge not existing

Table 1-1 Issues preventing efficient knowledge transfer

1.2 Background

Company E-Learn is a small E-Learning software company consisting of twenty-seven staff. Most projects teams vary from 5-6 team members from various disciplines, including Project Managers, developers and solution architects. The organisation sells an off-the-shelf Learning Management System. Projects normally range from three to four months in duration depending on the customer requirements. There are a diverse set of customers from domains such as financial services, oil refineries and domestic appliance producers.

A Wiki had been used in the company before this knowledge management project but was only used on occasion by 2-3 people in the organisation. Other users did not use the Wiki as they were unaware of the current uses and knowledge that was stored on this Wiki. The more knowledgeable workers were users who had been in the company for the longest duration and experienced the most project implementations. Users with less experience opted to obtain this knowledge through face-to-face conversation rather than searching on the Wiki.

A Waterfall Methodology was used for all projects within the organisation. This methodology meant that knowledge was being passed on at every stage of the lifecycle in a series of phases. The sequential approach normally started with a business analyst defining the requirements, passing these requirements onto the solution architect ending with the developers developing the proposed solution for the client. Depending on the individuals themselves, different communication media were used to transfer knowledge, which included email, face to face to instant messaging. It was often the case that knowledge was lost in emails that was not updated in a document or transferred to the developer. Lessons learned and brain-storming meetings proved on several occasions that documents were not being correctly reflected. Knowledge exchanges in other media such as email and instant messaging were not being reflected in project documentation.

As every stage of the lifecycle from requirements to design involved interaction and conversations about the customer's requirements both with the client and internally, many pieces of important knowledge were lost along the way. A previous phase of a project for customer B-2-B had showed that requirements conversations had gone undocumented and, as a result, when the project went to delivery, the customer was unsatisfied that they had received all of the functionality.

Coughlin (2002) highlighted that when discussing issues regarding requirements elicitation, knowledge acquisition and sharing can only be achieved through effective communication between the various stakeholders.

If users were not directly involved in communication in the project, users were unaware of certain agreements if they were not discussed at a project meeting. The requirements phase is only an example of how key knowledge learned from one

experience could be lost as different team members become involved. As highlighted by Bontis (2001), it is hard to really quantify loss in terms of statistics and financial loss as a result of knowledge loss due to its intangible nature but it can be proven on a less tangible scale. From a previous implementation of a project for Company B-2-B, knowledge loss from a requirements perspective resulted in an unhappy customer, which, as a result, could lead to a financial loss as the customer will not return for more business if unhappy with the service.

It is important to understand that new people provide new ways of doing things and are always required throughout a software project implementation and after its delivery from a support perspective. Users outside of the project team are beneficial in times of crisis, when certain users are not in the office and a problem needs to be solved. These users review documentation and compare with other implementations so designs are not documented with biased opinions. Users from a support team know exactly what happens at every phase of the project so that they can answer support questions without having to repeatedly contact the person who was on the project.

The focus is on the communication between the old and the new, and the creation of something usable from a combination of both. It is not only important to share this knowledge with other users outside of the project team to get opinions but to reduce risk. For example, if a company had to reduce costs and offer redundancy, often the highest paid workers are those who have the most knowledge. If valuable resources leave a company, this competitive knowledge is lost.

The challenge is the reliance on individuals to ensure that this knowledge is passed on and reused and does not become out of date and to specify what element of this management can be carried out by a tool.

The term *knowledge transferral* is potentially misleading, as it suggests that knowledge is moved from one location to another resulting in its removal from the original location. In reality, it should more accurately be viewed as knowledge reuse and renewal, encompassing new and old knowledge, and tacit and explicit knowledge. This is the knowledge that puts us ahead of our competitors as we are reusing what has been learnt on other projects and the successes of other implementations. This first step

is to ensure that this knowledge is firstly shared, i.e. that users inside and outside the team come together and share this knowledge but to also create a place for sharing. There must be a tool that will allow the communication and collaboration of this tacit to tacit relationship. This tool must support the merging of this tacit knowledge with the already existing explicit knowledge, to create something new.

1.3 Research problem

The research problem involved identifying tacit and explicit knowledge for Company E-learn and ensuring that this knowledge existed in a place that was available for reuse for users that existed both in the project team and for users outside of the project team. This knowledge needed to be made available so that it was available to everyone, updated and merged with new experiences and once again reused by everyone.

Tacit and explicit knowledge existed at every stage of the software lifecycle. The challenge involved identifying both types of knowledge and who were the largest owners of this knowledge and in what format that this knowledge existed in. Tacit knowledge existed in many conversational formats including email, face-to-face and instant messaging.

In order to have one place for project knowledge, all knowledge had to be transferred to a technology that could manage this ever-changing project knowledge. The real challenge was not managing the explicit knowledge, as this was less changeable in nature, but the management of this changeable tacit knowledge.

This research attempted to maintain a record of this project knowledge (both explicit and tacit) for future references for all employees within the organisation, for those who existed in the team and those who were outside of the core project team.

1.4 Intellectual challenge

The overall goal of the knowledge management exercise was to ensure that both tacit and explicit knowledge gained initially and at all stages throughout a software development lifecycle for Company B-2-B was retained in one place and reused by those within and outside the project team.

The intellectual challenge was to:

1. Highlight and expose tacit knowledge at all phases of the software lifecycle.
2. Collect the tacit knowledge and explicit knowledge and place in a central location.
3. Ensure that the central location can manage the changeable nature of the tacit knowledge.
4. Ensure that the tacit knowledge and explicit knowledge are clearly merged and visible.
5. Ensure that all users within the project team know how to contribute, where to contribute, when to contribute and what to contribute.
6. Ensure all users outside of the project team can clearly see and obtain all project knowledge at any time during and after the project implementation.

There is also a challenge to overcome any of the barriers that may exist to the transfer of knowledge, which may include people barriers in terms of using or not wanting to share or contribute to the knowledge management project people. There may be challenges about devoting time and resources to such a project.

There also may be limitations in terms of what the technology can do in terms of supporting the knowledge transfer process, i.e. not being able to correctly represent the knowledge that exits within users or not being able to show users when knowledge has changed or has been updated.

1.5 Research objectives

The focus of the research objective is to prove whether there is an appropriate way to manage the transfer of project knowledge in a software development project, so that this knowledge is available to all users within the organization. This project knowledge needs to be documented in such a way that it is easily referenced for those who were and were not involved in the original implementation. The goal of the research is to prove that this transfer of project knowledge into a central area for distribution is possible. The objective is to prove that a knowledge management tool can manage this

ongoing communication of project knowledge, even when the project has been implemented. If one can prove that all project knowledge, both explicit and tacit, can be easily referenced, then results of this successful implementation have to be reviewed.

In summary, the research investigates the following:

1. Investigation was carried out to see if Company E-Learn was able to share tacit knowledge inside and outside the project team. The research investigates the issues regarding knowledge loss through the software lifecycle and reviews at what stage the knowledge loss is at the highest point.
2. What new processes and technology addressed knowledge loss at certain phases of the software development lifecycle?
3. What barriers existed to challenge the research objective?
4. How these barriers to knowledge transfer were overcome?
5. As a result of the findings of the research, what future recommendations could be made regarding the knowledge transfer process within the organisation?

1.6 Research methodology

In order to evaluate hypothesis regarding reasons for lack of knowledge transferral in the past and to decide on an approach moving forward, several different techniques were taken on board. As highlighted below, the entire knowledge transfer project was split into requirements, design, implementation and review. The tasks and approach for each phase is highlighted below.

<u>Requirements</u>

The requirements of the knowledge transfer process were decided by the Project Manager of the knowledge transfer process and agreed by Senior Management. Requirements were gathered by reviewing lessons learned from previous software project implementations and reviewing where the largest knowledge loss within the project was and who was responsible for this loss. Questionnaires were filled allowing team members to voice their opinions on what they thought the issues were. Meetings were held when the requirements of the knowledge transfer project were defined and a

solution had to be designed to propose how to address the knowledge transfer issues within the project team.

Design

Upon review of other implementations and review of documentation on the subject of knowledge transfer, an analysis of technologies to cater for knowledge transfer was taken on. The requirements were mapped to each technology and a suitability matrix allowed a tool to be chosen based on the most amount of requirements fulfilled.

A design document was put in place to highlight how the knowledge transfer project in terms of a new technology and a new process were going to be implemented.

Implementation

A wiki technology called Mediawiki was used as the tool to record all explicit and tacit knowledge for the project. Knowledge that existed from a previous phase was uploaded to the wiki, and the wiki was supposed to be the place for new knowledge moving forward, including knowledge gained post implementation.[3]

Review

Upon implementation of the wiki, a review in terms of usage of the tool was carried out. Questionnaires were filled out by team members asking questions about the knowledge tool and the knowledge sharing process. Open Interviews with team members allowed individuals to discuss why the project was or was not a success and to highlight barriers to the knowledge sharing process. Test scenarios were set up which asked users who were and were not members of the project team to extract knowledge from the wiki. Several tests were carried out over a period of time, reviewing user's experiences as the quality of the knowledge improved over time.

A review of the success of the knowledge transfer process was carried out by comparing the original requirements with the solution that was implemented and how it was accepted by the team members.

---

[3] MediaWiki is a free wiki technology developed using the scripting language PHP. MediaWiki is a type of technology that allows users to easily add content to a internet page without any administration rights or web development experience needed.

1.7 Resources

Before the dissertation commenced, an overall approach was decided on, regarding a flow of tasks and prerequisites before implementation of the overall knowledge management project. Both primary and secondary research was taken on at various stages to assist in the preparation for the project.

At the initial stages of the project secondary research was carried out post analysis of the initial high-level requirements. Secondary research involved reading websites, and communities of practice, other dissertations and journals to see if other companies and individuals had recognized a similar problem.

Without review of the same problems within other organisations, opinions of the overall issue would be quite narrow. Reading other experiences allowed for expansion of thoughts on the implications of existing issues.

Primary research was also carried out before the implementation by creating questionnaires to obtain peoples opinions on the existing issues before any changes were made. Interviews were carried out with key project members that existed within the project team and outside of the project to get an overall picture of the success of the knowledge transfer process. Reviews were carried out with the Project Sponsor intermittently throughout the project. Questionnaires and interviews were carried out at the end of the implementation when the project had been in use for a period to obtain feedback on the success and failures of the system.

A usability test was carried out with members who were outside of the project team to ensure knowledge was transferable to new project members. When certain knowledge was not clearly taken on by users outside of the project team, changes were made to the Wiki and several regression tests were carried out to ensure that the knowledge was effectively passed on.

Technical Resources

Access was needed to the knowledge management technology that was already in place within the organisation. Assistance from the individuals who originally implemented the existing tool was required.

Access to various online resources was required for literature review regarding previous implementations or research on similar topics.
Internet access was required at all times for research and also for back-up of documentation.

All Microsoft packages, such as Excel (statistics), Word (documentation of results), PowerPoint (internal and final presentations), and Visio (flow of information in experiment) were used as a part of the research project.

Human Resources

Access to a Project Sponsor within Company E-Learn on an ad hoc basis was required to verify at the early stages that the requirements that were documented were correct and in line with what was necessary.
Time was required at the development and testing of the new knowledge management solution to ensure that the functionality had not wavered from the requirements in any way.

Time from a technical representative was also required to assist with any issues that may have arisen in terms of issues with a database, user interface, development or server connectivity etc.

Access to users for a period of time in terms of feedback before, during and after the implementation was required.

Access to users who had documented and implemented similar projects was also required to compare thoughts and implementations.

Scope and limitations

As part of the research, a concrete decision on what was achievable in the time period had to be decided upon. The overall objectives were clear but a decision on what could be achieved in the time period had to be agreed. The decision was made to focus on one client, in terms of creating a knowledge-sharing tool that would manage the tacit and explicit knowledge of a project but the analysis was mainly on the tacit to tacit and

tacit to explicit process. It was not feasible to include all of the organisation's knowledge as a part of the scope of work as the time it would take to sort, upload and disseminate this information would be too large in the scope of time given. The existing Wiki architecture that was already in place was used but was re-branded and customised in one area for the chosen client.

Rolling the new process and knowledge management tool out to anyone outside the defined project team, giving the client access and rolling out the knowledge management tool to any other projects within the company was out of scope for this project.

The three and a half month time period only allowed research in open source tools, which meant that the functionality in terms of what it could provide were quite limited as there was no cost allowed for the implementation.

As the project did not have a budget there were no possibilities to implement a tool that had any support of licensing costs, which again resulted in risks in terms of implementing something that could result in issues and a lack of support being provided.

1.8 Organisation of the dissertation

The research is broken down into the following chapters.

Chapter 2 provides an overview of knowledge management, the different types of knowledge that exist in a software project, both explicit and tacit knowledge.

An overview of how knowledge is managed throughout its life as it passes through the spiral of knowledge is provided.

This chapter highlights how explicit and tacit knowledge is passed through the spiral of knowledge and issues regarding tacit knowledge at the socialisation process are investigated. Investigation into how this knowledge is effectively represented for organisational use is carried out.

Chapter 3 addresses the issue of knowledge transfer in a software project to users inside and outside of the project team. Chapter 3 highlights how knowledge passes through the different corners of the spiral of knowledge and where the main issues of

transfer occur. Chapter 3 also addresses the different media at which knowledge transfer can occur and how to clearly represent this knowledge when it comes together.

Chapter 4 discusses the main areas of knowledge loss in the knowledge transfer process in a software project. Analysis in term of loss of knowledge in a traditional lifecycle methodology such as the Waterfall Methodology is compared against newer agile methodologies such as XP and SCRUM. Investigation into the possibilities of introducing an agile method to improve the knowledge loss in a software project is discussed.

Chapter 5 examines the various technologies that can assist in the transfer of knowledge in a software project. This chapter focuses on Wikis as a tool to manage project tacit and explicit knowledge. This chapter highlights how Wikis can address the issues of making tacit knowledge explicit in a software project.

Chapter 6 explores the previous implementations of knowledge management tools that tried to overcome the transfer of tacit knowledge within a software project. Case studies of Wiki implementations, Semantic Wikis and Knowledge Management Projects were reviewed and discussed.
This analysis contributed to the approach taken at the experimentation stage.

Chapter 7 looks at the implementation of a Wiki as the knowledge management tool to manage the project's tacit knowledge. Discussion about the approach, how requirements were defined and how the implementation of the tool was carried out is also documented. Chapter 7 focuses on the results of analysis taken after the project was released the team, and carries out tests to review the success of the Knowledge Management Project.

Chapters 8 focuses on the evaluation on the goals of the knowledge tool and highlights the lessons learned from the implementation. Overall evaluation in terms of the aims set out at the start of the project is evaluated and the conclusion on the best way to manage the transfer of tacit knowledge in a software project is provided. Future work in terms of what was out of scope but would be of interest to investigate further was also discussed.

2 KNOWLEDGE MANAGEMENT

2.1 Introduction

As the CEO of Pepsi once said the soft stuff is always harder than the hard stuff (Enrico 1995).

It is always going to be a challenge to represent knowledge that has not been codified and does not exist in an explicit format. As a result, pockets of knowledge exist around the organisation and are not easily spread amongst others. This chapter investigates what knowledge management is, how knowledge is managed throughout its lifecycle and the issues regarding transferring knowledge to others and keeping this knowledge up to date. This chapter identifies how and where knowledge is lost throughout the software development lifecycle and will attempt to answer the question, how can knowledge be shared and reused?

2.2 What is Knowledge Management?

As highlighted by Young (2005) Knowledge Management is the discipline of enabling individuals, teams and entire organisations to collectively and systematically create, share and apply knowledge, to better achieve their objectives.

As Young highlights, knowledge management is about individuals, teams and organisations collectively sharing knowledge. Within the software development lifecycle, a vast amount of knowledge is created and reused, such as approaches on version control, coding reuse and software project management methods.

Depending on the approach and methodologies of the environment and company cultures where the software is being developed, this knowledge may be passed on to others or lost, updated or become out of date or incorrectly translated. Knowledge may exist within an individual and never be shared because someone might never know it existed. This tacit knowledge is informal knowledge not documented in a physical format (Polanyi 1967). As discussed by Nonaka (1995) explicit knowledge is knowledge that is saved in a document or has been recorded in some physical format and can be passed onto others.

In a software development team, there is a large amount of knowledge transmitted via emails, instant messaging software and in documentation. As shown in the diagram below, knowledge becomes less formal, and of a more tacit nature as a software project progresses from requirements through to development.

Figure 2-1 Examples of Tact and Explicit Knowledge in the Software Lifecycle

Knowledge is stored in both an explicit and tacit format at every stage of the project lifecycle. Knowledge is created at the start of a software project with customer requirements and passed onto different team members as it passes through the

development, testing and release phase. Knowledge changes format and is manipulated and lost depending on how well it is managed.

The challenge is to collect all of the project information, both explicit and tacit, and ensure that it is converted into a format that is usable for the entire organisation. The knowledge in this central location is not stored once and never reviewed by all but it is stored in such a way that it is self-maintaining. It should be used as the place for ongoing project collaboration and storage during and after implementation. Users should never be in a position to have to check if the data is up to date and correct.

Plato once said the learning and knowledge that we have, is, at the most, but little compared with that of which we are ignorant.
If we examine this statement closely, one realises that there is always the potential to never know what we don't know exists, therefore being ignorant to what we could know. Our challenge is to expose knowledge that exists without actually searching for this knowledge or being aware it even exists.

The main challenge investigated in this research is the transfer of knowledge from one source or media to another. Investigation into the issues regarding knowledge transfer of the knowledge that exists in a tacit format to those who were not directly involved in the project has been carried out. Investigation as to whether this knowledge management tool can effectively manage this ever-changing knowledge has been carried out.
If something is not communicated or communicated ineffectively then, the cost to the software project is huge by the time is gets to the end of the software lifecycle as it does not clearly represent what the original picture looked like.

As highlighted by Nonaka & Takeuchi (1995), in an economy where the only certainty is uncertainty, the one source of lasting competitive advantage is knowledge.
The issue under research involves bringing together all project information from the various media and classifying this knowledge in order to allocate it a topic for searching and reuse. Within a software development lifecycle, knowledge should be accessible from a central location. Knowledge should be kept up to date with minimum effort from the software development team. Re-use within software development

projects is crucial to profit and success as there is the risk of continually re-inventing the wheel. As Benjamin Franklin once said *time is money* (Franklin 1748).

2.3 Types of Knowledge

2.3.1 Explicit Knowledge

Explicit Knowledge is knowledge that can be articulated and is visible to others. As highlighted by Sanchez (2000) this type of knowledge can be disseminated to others in many formats.

Explicit knowledge could be seen as easier to transfer to others as it exists in a physical format and can be easily passed on. Explicit knowledge also has the same issues that are highlighted in the spiral of knowledge with tacit knowledge, in that it needs to be kept up to date and when it is used by others so that it evolves into new knowledge. Examples of explicit knowledge in a software projects are design documents or test plans.

2.3.2 Tacit Knowledge

Tacit Knowledge is knowledge that is not codified or made explicit. It is knowledge that exists in minds of individuals through experience or learning. . Tacit Knowledge is a mix of experiences and is a build on information as it contains this mix of experiences and skills (Polanyi 1967).

Tacit Knowledge within a software development cycle is particularly difficult to codify as knowledge passes through many cycles and phases and changes over time and never exists in one format. It is a real challenge to pass knowledge onto others that is continually changing. As Nonaka (2000) said, the word “manage” highlights control, but can one ever control an ever-changing tacit environment? Bringing knowledge carriers together is the first step in the knowledge management process (Sanchez 2000).

2.3.3 The Spiral of Knowledge

Knowledge exists in both an explicit and tacit format. Knowledge does not stay in one place or never change but moves through many stages of communication and changes in state as it passes through.

The spiral of knowledge created by Nonaka and Takeuchi (1995) highlights that tacit and explicit knowledge circle and reconnect in a spiral motion.

There are 4 processes or exchanges at which knowledge is transferred, communicated or changed.

- Socialisation is the exchanging of tacit to tacit knowledge. An example of this would be a conversation between a solution architect and a developer on how to set up labels in the version control tool clear case.
- Externalisation is making this tacit knowledge explicit; for example, documenting how to set up the labels in a procedure document.
- Combination involves merging this explicit knowledge with other explicit knowledge. In our example, this would be using this procedure document with a detailed design document to understand how to develop a piece of functionality.
- Internalisation is the changing of the explicit knowledge back to tacit knowledge by the practical implementation of doing what was specified in the explicit form.

Figure 2-2 Spiral of Knowledge (Girard 2006)

As highlighted in the Spiral of Knowledge, the process does not stop after the internalisation phase but begins again at the socialisation phase. The knowledge becomes more refined as it moves through the knowledge spiral. For example in the context of a software development environment, documentation is completed, reviewed with others and the solution is developed as per the requirements. If the documentation is then reviewed again to ensure the solution is in line with the requirements, the

solution will be more in line with what was required, as the knowledge will have moved through each stage of the spiral of knowledge several times.

In the Spiral of Knowledge, Nonaka looks at how from sharing and discussion, knowledge is created at the socialisation process. Nonaka believed that knowledge appears in both a tacit and explicit format. Knowledge is then translated into a format where it is externalized or made available to others. Though various techniques, and merging with other knowledge, knowledge is combined to become new knowledge and in the last step is reused and internalised by others (Nonaka et al 2000).

2.3.4 The issues regarding Knowledge Transfer in the Spiral of Knowledge

Although the knowledge is passed through the spiral of knowledge there exist many issues or potential issues at the different stages of its lifecycle.
The processes involved in Company E-Learn example are similar to the components of the spiral of knowledge in that knowledge is created at the socialisation stage, externalised and then issues arise at the stage of internalisation of reuse as the knowledge has not been clearly represented (Nonaka & Takeuchi 1995).

Within a software development project, knowledge is created at the initiation phase with face-to-face requirements gathering. Knowledge is then translated into an explicit format when it is translated into a requirements document and subsequently a design document. After the design stage, developers work on the implementation of the solution. At this point the system is tested and then delivered to the customer. Conversations regarding requirements normally happen outside of documentation as, when requirements are translated into design, clarifications and questions arise. When developers then go to develop the solution, there is further clarification from the business analyst or from the customer. The goal of this research is to try and successfully translate this tacit knowledge into an explicit format and merge this with existing explicit knowledge. Another issue that exists is post implementation of the project when decisions regarding functionality or changes are made to the system. In order to have a clear picture at any time of where the project is and what functionality is has, this tacit knowledge needs to be clearly documented and added to the already existing project knowledge. You will see from the picture below that in an ideal world

all requirements would be clarified at the requirements stage, everything needed in the design stage would be documented in a design document and everything would be developed as per what the customer wanted.

Figure 2-3 – Software transfer project with no knowledge transfer issues

In reality, conversations do happen outside the normal documented process, and do not get clearly reflected in the explicit format that is required for all users to gain benefit from this knowledge. As you can see below, conversations do take place outside of the documented project knowledge. If this knowledge is not documented, users outside of the project team will never be able to understand exactly what was delivered and are operating on out-of-date knowledge.

Figure 2-4 – Issue of tacit to tacit communication outside the spiral of Knowledge

In order to overcome this loss of tacit knowledge, from either one-to-one conversations with clients or internally within the group, firstly this tacit knowledge needs to be made global and secondly it needs to be converted into an explicit format and merged with existing explicit project knowledge. This tacit knowledge needs to be made expressible in a format that is comprehendible by others not involved in the original acquisition of the knowledge (Neef et al 1998).

Also, depending on the environment, there are certain other factors that limit the success of the spiral of knowledge. Nonaka (1997) highlights 5 enablers for knowledge management which are vision, strategy, structure, system and staff.

Vision is about creating the goals without any limitations. This is creating a global atmosphere for sharing without inhibitions. The strategy is about managing this vision and putting a plan in place to manage this vision and identifying the keys ways to make your knowledge rise to the top. Structure means keeping the goals of the project on track without inhibiting the knowledge management initiative itself. The system highlights how to do it, i.e. if it is a system itself or a new process and finally the staff are all the people involved in the entire knowledge management project. In a software

environment, knowledge will not be effectively expressed or transferred if all of these factors are not considered. The knowledge management initiative needs to be thought of as a project in itself with a strategy to manage itself, a system or process to manage the knowledge and to also identify the key players.

2.4 Conclusion

Some opinions are that to have knowledge about something is to experience it and so it is really nothing new but rather a build-up over time of something that has happened already either to you or to others. It is important that others who have not experienced something can also be as enriched with this knowledge without actually experiencing it. As Nonaka (2005) said, *knowledge management is nothing new, the difference is that people who pass on knowledge and accept knowledge gain out of passing on and receiving this knowledge*.

It is important to realise that you can not always make people accept knowledge or use the processes or tools put in place, but you can only try and make things better for people and make things as automated and make knowledge as accessible as possible. For something to be a success, no matter how complex it is or what you are trying to do, it needs to be easy to use and accept from a user's point of view.

This involves making the environmental factors more in tune with the initiative, i.e. ensuring the system is in line with what needs to be done and the organisational culture is in tune with the requirements.

Davenport and Prusak (1998) provide three reasons why organisations implement knowledge management systems: To enhance visibility of knowledge in organisations, to build a knowledge-sharing culture, to develop a knowledge infrastructure, not confined solely to technology, but to create an environment that permits collaborative work.

As Davenport and Prusak highlight, you can only try to expose the knowledge where it exists and create avenues of open and shared communication. After this point, the rest is up to the users themselves.

Knowledge or the technologies are not the key factors to a successful knowledge management project. As Collison and Parcel (2001) highlighted, the largest percentage of the success of a knowledge management is about the people. As highlighted below people, process and technology do not sit out alone but are all closely linked. In essence it is the people who push the process and build the technology and use the technology and so without their involvement, knowledge transfer would never take place.

The main concern of this research is regarding the tacit to tacit relationship, also called the socialisation process of the spiral of knowledge. Losses or misunderstanding in the software development lifecycle tend to be where conversations have not been recorded or translated into explicit knowledge. There exist many media and many categories of tacit knowledge that need to be classified and sorted before merging together at the internalisation process. In terms of risk and greatest loss, this socialisation process proves the most difficult to manage.

Knowledge has to be expressed in a format that is understandable by others. In reality this tacit knowledge that is expressed and not codified anywhere needs to be codified and categorised and merged with what is already there. In Chapter 3, investigation into the different types of tacit knowledge generated through out the software lifecycle is investigated. The different media of transfer are also investigated, and ways around classifying this knowledge are reviewed. Knowledge is generated through tacit to tacit interactions, sorted and classified via technologies such as rules or manual sorting. Knowledge becomes less explicit the more it moves through the spiral and the further down the software development lifecycle.

This is why the need for one tool to possibly remove all of the communication media where knowledge is being classified and have one tool for sharing, collaboration of project knowledge. Even if it is not possible to remove the communication mechanisms, the investigation of transferring all of this tacit and explicit knowledge to one place is the next focus of the research.

3 KNOWLEDGE TRANSFER

3.1 Introduction

As Nonaka and Takeuchi (1997) highlighted, knowledge differs from information as it adds the human interpretation and reasoning to it. They describe it *as a flux mix of framed experiences, values, contextual information and expert insight that provides a framework for evaluating and incorporating new experiences and information*. Knowledge is the incorporation and merging of new and old experiences to provide new knowledge. Knowledge rarely stays in the same state or is understood in the same context. Knowledge passes through the spiral of knowledge and gains weight over time or in some cases losses weight. In a software project, knowledge starts at the beginning of a cycle and flows through each cycle of the waterfall changing from tacit to explicit back to tacit before resulting in an explicit format this is deliverable to a customer. This chapter investigates the types of knowledge that exists in each phase of the waterfall lifecycle and investigates the issues regarding the transfer of tacit knowledge. Analysis is completed on the four areas of the spiral of knowledge and at what area the knowledge is generated, updated, changed and reused.

3.2 Knowledge generation

Knowledge is created by using information obtained previously and putting this knowledge into practical experience. Tacit knowledge in its purest form is knowledge gained by an individual due to a mixture of information and organisational experience. In a software project, knowledge is created by past experiences of other projects and practical experience of processes from other companies. When this knowledge is shared with other individuals in a social matter, it is enhanced and merged and refined to create new knowledge. This stage of the spiral of knowledge is called the socialisation stage. Knowledge at this point is of a tacit nature and proves the most difficult to translate to others and put in an explicit format.

There are different types of knowledge that are relevant to a software project, which can be categorised into personal knowledge and organisational knowledge. Personal knowledge within the organisational context is knowledge that users use to do their job

but is of little benefit to others within the organisation. An example of this would be procedural knowledge, such as creating a project plan from a project management perspective. Organisational knowledge is as it suggests, is knowledge obtained by a users regarding the organisation. This organisational knowledge would be of little use to a user if they left the company (Neef 1998).

There is much research on the differences between knowledge and data. Realistically one would not have knowledge without data, as knowledge is what one draws experiences and conclusions from. One must think that knowledge is created at the generation stage from tacit conversations and then converted to an explicit format. In a software project, there are face-to-face conversations with clients, which result in the challenge of trying to codify this tacit knowledge and put it in an explicit format. This document is then discussed and amended depending on outcomes of discussions. The issue is building all of these conversations into a big picture of what has been said or agreed outside of a document. In order for this tacit knowledge to be used, it needs to be sorted or classified into a format that can be easily understood. Every stage within the spiral of knowledge has an input and an output. The input at this stage is tacit knowledge about a subject, shared with another to produce a new piece of knowledge.

3.3 Knowledge Sorting and Classification

Before making something explicit, structure or format has to be applied to it. Intentionally or unintentionally, one creates headings in documents or folders on servers to classify our knowledge. Classification or association is not something that is new or something that has been created due to knowledge management. Classification is to class or categorise something. A type of classification originally invented in 1945 was called hyper linking. Hyper linking was invented by Vannervar Bush (1945). He first highlighted the idea of hyper linking as a way of an *associative index.*

Before hyper linking, classification started with the 'Dewey Decimal System', developed in1876 by Melvil Dewey. This was a paper card based classification system. As discussed by Gupta (1997) after several decades this form of classification itself was too large for a person to use.

Both attempts at classification sought to make things easier for those searching through an abundance of knowledge. At the time, the Dewey decimal system served a purpose

as its aim was to sort a certain amount of knowledge. The difficulties arose when the knowledge increased as the classification task itself became too big a task to manage. As with the Dewey Decimal system, in software development teams, the amount of knowledge to store, manage and share is becoming unmanageable.

Figure 3-1 Tacit to Explicit Knowledge Conversion (Talisayon 2008)

Before investigation into the classification of knowledge, investigation into the tacit to tacit conversations within a software development project has to be carried out. The media, from which the knowledge comes, has to be reviewed. Within most software projects and within Company E-Learn, the main sources of communication are email communication, face to face and instant messaging.

3.3.1 Email

As highlighted by Pelfry (2002) when Tomillson invented email in 1971, he was not thinking of how, in 20 years time, knowledge would be sorted and reused at a later stage. Most types of project data in a software development team are communicated via email, from project release information to project schedules to lessons learned documentation.

Information communicated on, originally lies in project folders on a shared folder on server and some information resides with individuals and is discussed over email. In order to find something that has been sent to someone, a person has to be a part of the communication and also know what you are looking for and who sent it.

This highlights the issue of occasions in which one doesn't always know what they're looking for and one's search is limited to what they do know. It also presents the issue that if someone outside of the project team is looking for a piece of project data; this is stored in one person's email database and is not in a central location. This knowledge needs to be made public so all company members can be enriched with this knowledge.

Before this knowledge is made public and transferred to a central location, it needs to be sorted into an index of terms so that it is easily referenced and searched.
A good example of classification of email is highlighted by Kassim et al (2008). Kassim suggests that it is hard without large amounts of time spent searching to find an outcome of an issue discussed via communication. Kassim et al highlights how categorisation of emails using metadata helps in what is called a semantic email system. Here, the focus is on using the email system as the knowledge base rather than categorising the knowledge so that it can be exported somewhere else.

The study focuses on the use of semantics through back end logic looking for key words in email text and grouping based on certain words.
A piece of functionality that is of benefit, which was undertaken, was the rating of emails in order of importance. Without any type of technical advances to what is already there, users could rate information 1- 3 in order of importance and if there was a large amount of emails, these could be sorted in order of importance.
Within Outlook, there are also features that allow users to set up rules based on the information received. You can set up folders based on key words in the subject when emails are received. It important to clearly define your ontology first; it is hard within a standard email inbox to define ontology's as it becomes more an index of terms without real relationships between each other. Ontology is more of a web on interconnected relationships linking concepts based on metadata (Grubber 1993).

Aery & Chakravarthy (2005) highlighted 3 high level categories of how to classify knowledge, *Rule based classification to classify emails into folder, Information Retrieval based classification and Machine Learning based classification techniques – e.g. Naive Bayes for training.*

As the focus of this research is on a small sized software development team, the need for changes to the email architecture to add algorithms based on frequency of words etc. is not required. The case study is only focused on a subsection of the company, where knowledge may be created daily but not so much that it is not sort-able without some preparation using rules in the email application itself.

Although the capabilities of the rules functionality within Microsoft Outlook may be quite extensive, the assumption is that one person gets all emails, and sorts this knowledge using the rules technique. The assumption cannot be made that the communication and transferral of knowledge is a many-to-many relationship. The knowledge manager needs to always be copied on every email, which is not always going to be the case.

As highlighted by Wagner (2004), e-mail is more of a one-to-one or one-to-many conversation tool without the central knowledge repository

Figure 3-2 Email Classification - Outlook Rules

An email software tool will never exist as a knowledge management tool because there needs to be areas for real time group collaboration and discussion, and all discussions need to be made public for its users to gain value out of the knowledge.

Email consists of knowledge that is unstructured and unformatted. Internal communication via email is especially unsorted and messy as users are aware that it is

not going to be distributed to a client. If a tool is used to sort this knowledge or even ask someone to manually sort through emails to classify, the amount of effort normally outweighs the gain.

Figure 3-3 Formal and Informal Knowledge

As highlighted in Figure 3-3, in terms of formality, knowledge that appears in email databases and in folders on servers is unstructured and uncategorised. Due to the conversational nature of the knowledge shared, a large amount of conclusions on a topic are not something that can be immediately seen. Users who are participating in this knowledge need to be aware that this contribution is going to be reused and put this knowledge in a format that can be easily reused. A clear process in terms of user contribution needs to be highlighted. It is a real challenge to ask users to share a conversation after it has been completed due to privacy issues and sometimes depending on the content of the email.

The best approach is to dedicate or ask someone to document an outcome of a situation. This person could then email a central area directly, so that it is put in a place that can be seen by all and has a defined structure.

Another feature that removes the need for classification in an email inbox is a feature within the technology MediaWiki called *emailtoWiki*. This allows a user to email a Wiki article already existing in a Wiki or else create a new Wiki article.

This removes issues regarding one-way communication. The data may not be classified correctly on the Wiki itself, but investigation into the classification techniques within the knowledge tool itself are investigated later.

Figure 3-4 Email to Feature – Media Wiki

It has been proven that it is possible to classify knowledge in an email inbox based on rules. The next focus should be on transferring this classified information in the email inbox to a central location.

3.3.2 Instant Messaging

As the software development lifecycle progresses, requirements have been clarified and design documents have been delivered, conversations and communication become less formal and organised. Questions and clarification tend to be posed over communication media such as instant messaging tools like MSN or Skype.[4]

A large amount of valid knowledge is lost as conversations are carried out on a one-to-one basis. Some of these communication tools do have the multi-user capability but tend not to be used.

Even if this functionality is used, it is rarely passed onto any other media for distribution to others. The type of knowledge discussed over instant messaging is

[4] Instant Messaging Software offers real time communication between people. Instant Messaging are sent by users typing text into a command box and the message being delivered immediately to the other user over the internet.

normally carried out between developer and solution architect. Valuable questions regarding clarification on design documents are carried out but not reflected in the design document itself.

The emphasis is more on technology as one moves away from paper-based communication or asynchronous communication and move towards a more real time communication. Communication is moving faster, more questions are being asked and the answers are being delivered faster than ever before.
Instant messaging moves ahead from the old-fashioned telephone conversations, where one was not guaranteed a response and could possibly play telephone tag until someone calls them back. Due to the technology developed in instant messaging technology, one can tell if someone is online and if you do not receive a response, the person does not want to talk to you (Hasse et al 2005). There is still the same issue of asking questions that have been asked somewhere before and are answered either in someone's heads or in a file on someone's hard drive. The challenge is to ensure that this communication is integrated with a larger centrally used system where everyone can either share or see the results of the conversation.

One disadvantage of instant messaging communication is that it takes away from people's principles of ensuring that what they are saying is correct and well written. When something is written in an email, a subject is given to it, its purpose stated in the first paragraph and, in the next paragraph, a question is answered or a question is posed. If someone needs to reference an email, they would have to remember who sent them the message and the subject. If someone were to be quoted from an instant message, they would have to spend time formatting it to take out badly phrased language and rearrange it to put some structure on it. It is important to note that some users do not agree with being referred to in an instant message without their approval.

Marianne Foley (2002) carried out a study investigating the use of Instant Messaging in Academic Libraries for the purpose of answering questions that would have been previously answered at a library desk. This study highlights the advantages of quick response on information retrieval but also highlights the unstructured and unsorted knowledge difficulties. Foley emphasises that face-to-face communication is probably the better form of questioning and answering. You can never fully replicate human

interaction as there is always the possibility to misinterpret something without facial expressions or visual representations of something, but from a knowledge management perspective, it is harder to pass on a conversation than a formal documented version of something (Foley 2002).

In the Cluetrain Manifesto (Locke et al. 2000), Locke details that the real cultivation of knowledge is through conversational activities; less knowledge is extracted through data mining, as it is more a dump of data than a growing of knowledge.
There is some truth to this conclusion as people will offer more and comment more on a topic if they are free to not be judged about opinions and knowledge they share. If someone knew that they would have to make a subject about which they were talking more public, then the amount of knowledge shared might be significantly less.

As Wagner highlights, conversational *knowledge creation is suitable for environments where the knowledge is not centralized, but resides with multiple owners who may be located far apart.* Investigation into this point will be carried out later, but the challenge of this research will try and overcome many knowledge owners and try and create a more central view of the project knowledge.

Table 3. Conversational Technology Overview

Technology	Communication	Knowledge Repository	Knowledge Catalog
E-mail	1-to-1, 1-to-many, person-to-person	Local e-mail archives possible	Local index possible
Static and DB backed web pages	1-to-many, approaching many-to-many, "dialog" between web pages through hyperlinks	Local archives	Local index possible, web rings create larger catalog
Discussion forum	Many-to-many in web based forums, repeated 1-to-many in list servers	Central repository if web based, local if list server	Central index if web based
Internet chat	1-to-1, many-to-many	Frequently none, transient communication	None
Video / audio streaming	1-to-many	Central host or decentralized streamers	None, streams not indexed.
Video / audio conference	1-to-1, 1-to-many	Local repository if content is recorded	None, content typically not indexed.
GDSS	Many-to-many	Available, but GDSS sessions often treated as one-off.	Typically none, but possible
Web Log	1-to-many, can approach many-to-many (similar to web pages)	Local repository within each weblog. "Metablogs" now emerging	Yes, local index, metablog may provide larger catalog
Wiki	Many-to-many	Yes, current knowledge and history ("temporal database")	Yes

Figure 3-5 Types of conversational knowledge

As highlighted above, knowledge can be transferred in email, in instant messaging tools, in video and all can be carried out either face-to-face or in a group context as a presentation etc. All are recorded in different formats and have their own indices to catalogues of information.

If knowledge gained in instant messaging software is to be reused, users have to be on board and know that they do not feel threatened and also that the sharing of the knowledge is made easy for them. There is ongoing research into classifying chat conversations based on certain characteristics, such as utterance level, i.e. grouping into a semantic unit and reading keys words to produce a grouping of a subject. Ivanovic (2005) highlights how grouping these conversations can be a challenge as sometimes users do not always do what they should, i.e. sometimes one does not carry out a conversation in sequence, or our grammar is not easily interpretive. This causes confusion when our program is set out to perform certain classifications based on a pre-programmed set of rules. Also, this type of development would be too large in a small software development company where the emphasis is to generate knowledge based on low-cost changes and minor changes to the way people do their daily business. Using this example, there are also issues regarding privacy and access to all messages within instant messaging conversations.

A simpler technique would be to firstly leave the responsibility to its users and allow them to choose what knowledge they would like to export. They could choose to export the conversations via any simple add-on to export data into html format. The person responsible for the knowledge management system could then carry out the classification from this point.

3.3.3 Face to Face

In order to extract knowledge from face to face communication, it will have to be turned into something that is explicit and easily codified. In a project environment, these conversations are rarely formalized as people do not have the time or may not want to put their name against something that can be referenced in the future. What is possible to document from a face to face conversation is the items discussed in a group context, such as a meeting or presentation. If there are clear and concise minutes of the meeting taken, then these can later be reviewed and sorted, based on its content.

There will always be a challenge to get people to document face-to-face conversations. Before transferral of this knowledge to an explicit format, there has to be recognition that this knowledge is going to be made public. As part of a knowledge management initiative, as users move through the software lifecycle, there is always recognition of what key subjects should be documented. Instead of ad hoc decisions about making conversations public, decisions and actions need to be decided before conversations have been held. If it is agreed and time is invested into knowledge documentation, this knowledge can be easily made explicit.

3.4 Knowledge Transfer

Davenport and Prusak (1998) highlighted knowledge transfer as one of the largest difficulties in knowledge management. One has to realise that sometimes the challenge exists in the cultural acceptance of a knowledge management initiative rather than a tool to manage the knowledge.

As discussed by Pyra (2003) *Knowledge transfer is seen as a process (not a one-time act).* In the spiral of knowledge, knowledge transfer is not a one-off thing and is always taking place (or not taking place in some cases). This is not a stagnant process; it is ever evolving.

In a software cycle in Company E-Learn, an example of a tacit to explicit process is the transfer of customer requirements into a document that will then be used as a design document. This solution design document will be the basis for what the developers use for the development of the solution. Normally at the design documentation stage, 98% of the requirements are defined in a requirements document and then this document is signed off by a customer. As the solution design document takes shape, requirements are clarified and questions arise. Outputs of conversations between customers and architects do not get reflected in documentation. When developers then use this documentation, key pieces of knowledge are lost and a piece of the requirements is not developed. In order to capture this knowledge, the responsibility of making this knowledge at the time of realisation needs to take place in order to ensure that it trickles through the cycle. Ownership of this update needs to take in order for the gap to be made in the spiral.

The media of transfer which facilitates the transfer also has to be investigated to ensure that this is an aid to the communication process, rather than a hindrance.
Coupled with the culture and want to share and the media for sharing, the type of knowledge is important also. Users find it easier to share and pass on explicit knowledge. As there are still challenges with knowledge management projects as a whole, sometimes users are just not given the time to share.

As highlighted in the diagram below, this knowledge transfer process is about making explicit knowledge tacit and putting into practice what one has learned. Lessons learned from this practical experience are discussed with others at the knowledge re-use phase. Examples of this in software projects are when users and solution architects review what has been implemented and discuss the outcomes.

Figure 3-6 Explicit to Tacit Transfer Talisayon (2008)

3.5 Knowledge Re-use

In the knowledge transfer process there is a giver and a receiver in this transfer process. This relationship multiplies the issues of knowledge transfer, as the giver may not want to give and the receiver might not care to receive (Hansen 1999).

One way to address the issues of knowledge transfer is to move away from the one to one relationships and move towards a community effort to share. It will not always be ideal to have group conversations about topics but sometimes, bringing someone into a

room to shame them into sharing will have to be done. Communities of practice are becoming the forum to share and exchange knowledge.

As highlighted by Nonaka (1997) the processes involved in Company E-Learn are similar to the components of the spiral of knowledge in that knowledge is created at the socialisation stage, externalised and then issues arise at the stage of internalisation of reuse as the knowledge has not been clearly represented at the externalisation stage.

The main problem that is under discussion in this research is to investigate how effectively knowledge transfer from one media to another in a software development project is achieved, focusing mainly on the tacit knowledge. Knowledge is created at the knowledge generation stage through experience and then converted in to a usable format at the externalisation phase. Knowledge grows and changes as it moves through the process. When this knowledge again is translated back into a tacit format by the practical implementation of something, it again takes shape.

The spiral of knowledge does not only happen once, but should be repeated and repeated until the knowledge does not require to be used anymore or is in its final state. Knowledge only gets better over time and by feeding this knowledge from the spiral back into the socialisation process, the knowledge is passed on again and reused by others.

In a software environment, design documents need to be reviewed by knowledgeable people to share thoughts on proposed ideas; repeat mistakes need to be avoided at development stage by clean handovers from people who have been there before.

As the saying goes, a problem shared is a problem halved.

3.6 Conclusion

Preparation is the key word in any successful execution of a task. Before you start a knowledge management project, it is important to set out the roles and tasks that the users are expected to carry out. If the rules based system is to be used for email, the rules should be clearly defined at the start of the project. Users should be told what key words to mention and format of information to display. Simple items such as emailing a group email address to make something public should be highlighted. If expectations are not communicated, users cannot know what to do or how to do it.

Sometimes users have tacit knowledge and are unaware that they have this knowledge. Smith (2000) explains that there is proven facts that if people were allowed figure something out themselves, without explicit explanation, they would have a better comprehension of the issue compared to reading the explanations of how to do it. Smith argues that users do not have to do the task to get a better understanding of what is involved as it is better to try this without any previous experience. One cannot agree with this point entirely as, without having previous experience, it sometimes takes a couple of attempts to really put this in perspective. For example, if a developer is given a design document and has already completed a similar piece of functionality for another user, then they are in a better situation than someone who has no previous knowledge of the functionality. As in the spiral of knowledge, the user has to put this theory into practice to thoroughly get the knowledge.
Adding the additional knowledge into a document to aid the user before they begin only expedites the time to acquire the knowledge and move things along faster. Smith suggests that there are three components of tacit knowledge, which are problem solving, problem finding and creativity and innovation.

Smith highlights that you cannot truly problem solve without the past experience and map of a pattern that you are going to follow in your head. Problem solving does require practical experience of a scenario to thoroughly understand an issue and really see where things could arise. In a software development project, knowledge is shared at every stage of the waterfall lifecycle.

Knowledge is reviewed by a minimum of two people at every stage. For example, when a solution architect is developing a design document, he/she will ask someone with knowledge from a previous implementation to review this document to make sure all facts are correct. In this scenario, the reviewer has to find both the issue through problem finding and problem solving, but the person doing the document can partake and learn from the problem solving. The creativity and innovation is brought out of these two people by marrying together their ideas to produce something that could not be produced on their own, as their ideas existed in isolation.

There is also the assumption that all companies that share knowledge consist of members that have a great working environment and all communicate regularly. A

large amount of tacit knowledge is lost due to fear of a work colleague, a culture where no ones talks to each other or a culture where everyone leaves after a short period, therefore forming no relationships (Nonaka 2000). As Nonaka highlights, the way knowledge is shared depends on the “ba”, the Japanese word for place. This could be virtual, physical or mental. A large amount of software development teams exist in a virtual space with the only way to articulate knowledge from one person to the next is via telephone or instant message conversation. Sometimes a barrier to transferral of knowledge is from the person communicating it. Some of the best geniuses were bad communicators.

The issue with tacit knowledge and explicit knowledge in a sense is that it is ever changing. In a software environment, we expect that requirements do not change, issues do not arise and that things go as planned. If this were the scenario, there would be a clear transition from tacit to explicit, to tacit again. The difference is the scope of our research is the knowledge that is outside of this circle. Sometimes extra requirements are discussed with the customer; processes are agreed upon, which are not fed back into the spiral. As a result knowledge is lost and the wheel is constantly reinvented. As a result of this loss, the client relationship is damaged, time is spent redoing requirements and as a result the project loses on turn over.

Upon investigation into the spiral of knowledge, it is realised the knowledge is generated, transferred and reused via a series of processes in the spiral of knowledge.
In the context of a software project, knowledge is generated and lost at various stages of the software lifecycle, depending on what lifecycle is used. Before investigation into what tools can mange knowledge transfer, the knowledge generated and lost at various phases of the software project have to be investigated? It may also be possible that, with the introduction of new process and methodologies in terms of software development, knowledge transfer could be improved with little introduction of new technologies.
Chapter 4 looks closely at knowledge created in a software project, at what point this knowledge is lost and how new methodologies could overcome this loss.

4 KNOWLEDGE LOSS IN A SOFTWARE LIFECYCLE

4.1 Introduction

As with the spiral of knowledge and with the standard lifecycle methodology there are processes to manage knowledge that exist in a software project. Processes exist to highlight to users how to do something. Within standard software projects, a methodology exists to carry out the software project but there is not always a methodology that sits within the software development lifecycle to transfer knowledge. It is important to have a methodology for knowledge transfer so knowledge is identified, the users who partake in the transfer are identified and the environment is set up for the knowledge transfer process.

This chapter identifies where knowledge is lost in the standard software lifecycles and at what phase the biggest lost is felt. Methodologies such as agile methodologies are reviewed that may assist in the knowledge transfer process. Knowledge transfer tools are discussed in Chapter 5 and reviewed at the experiment stage to review if these tools are sufficient to manage the knowledge transfer process.

4.2 Standard software lifecycle

Methodologies existed and are shown in examples in the 1950's where the Waterfall Methodology was used for the information processing project SAGE for US and Canadian air defence (Boehm 2006).

As time passed in the software development world, people realised that the waterfall lifecycle was a process that consisted of sequential phases that were never reviewed once finished. In 1970 the Royce Waterfall Methodology was documented showing that each phase in the waterfall cycle had to be revisited to ensure that the end result was as expected and documented at the requirements phase (Boehm 2006). Verification allowed for risk to be reduced as it was easier to attempt to resolve an issue and fed it back into the waterfall lifecycle than previous versions of the waterfall lifecycle.

The Royce model tried to overcome the sequential nature of the Waterfall Methodology and reduction in risk in terms of knowledge lost as it moved through the lifecycle.

As knowledge was transferred from one stage to another, more work in terms of development and testing and release was completed. If something was missed and had to be completed again, the costs were higher the later it was completed in the lifecycle (Boehm 2006).

Figure 4-1 Royce Waterfall Model (Boehm 2006)

In the 1980's Hewlett Packard found that its competitors were taking 2.75 years for their product lifecycles compared to 4 years for Hewlett Packard who were using the waterfall lifecycle methodologies.

As the shift was moving from a more focused requirements clarification, it was realised that requirements were changing but never being fed back into the knowledge would be fed back into the lifecycle.

4.3 Spiral Lifecycle

The spiral methodology moved on from the Waterfall Methodology by splitting pieces of functionality up into blocks of work and ensuring these blocks worked before moving onto the next stage. Although the methodology was still carried out in a sequential motion, it reduced risk in terms of issues being highlighted at an early stage in the project, rather than at the end when the entire project had been built and it was harder to remove components from each other (Boehm & Hansen, 2001).

Anchor point milestones within the spiral methodology are agreements or points made with the customer on certain functionality and requirements that have to be met.
The spiral methodology does not overcome the factor that if there are conversations outside of what was agreed as the anchor points, issues still occur as they did with the Waterfall Methodology as they are still sequential in their approach. The spiral lifecycle does not remove assumptions regarding requirements, but splits the development time up into chunks and delivers key chunks first to ensure these are correct before moving ahead with others. The spiral methodology does not overcome the issues of excluding key project team members at certain stages of the lifecycle and at critical stages where key pieces of knowledge may have affected a go-no-go decision.

4.4 Knowledge Loss in Standard Software Lifecycles

Knowledge is generated at the requirements stage with a customer in all projects. No matter what methodology is used, the knowledge is generated at this point. Depending on the methodology, if it is the Waterfall Methodology, spiral methodology or an agile method, requirements need to be passed onto team members in order to develop the required solution. As discussed by Ruhe (2000) if the requirements are passed on through several chains of communication, the risk is higher in terms of knowledge loss.

Tacit and Explicit knowledge exists at all stages of a software project.
Knowledge is of a more explicit nature at the initial stages as documentation is signed off by clients and conversations are recorded for referencing purposes in the future. When solutions are developed later and issues are only realised at the later stage, it is a struggle to continuously update a document to ensure the record of the requirements are always at an up to date state.
All project members should be a part of the requirements stage as they are building the actual solution and can ask the questions they need to ask at this stage rather than at the more costly development or testing phase (Grünbacher Briggs 2001).
As you can see below, the “I” icon is a visual representation of the knowledge and the image getting smaller displaying this loss as knowledge as it passes from person to person.

One of the biggest losses of knowledge transfer appears at the development into testing phase. At this point, testers are testing the solution against the original requirements and highlight that the solution does not do what it is supposed to do and has been developed outside the documentation and based on side conversations. Each project team member has a different opinion due to the fact that the tacit knowledge was never made explicit and was not taken on board by all project team members (Grünbacher Briggs 2001).

One of the main issues of knowledge transfer takes place at the initial requirements phase as incomplete requirements are documented due to issues not being captured, hidden client expectations due to assumptions not being articulated and conflicts due to these assumptions not being clarified (Grünbacher Briggs 2001).
As highlighted by Grünbacher & Briggs, requirements need to become more of a group process with a list of requirements documented by the customer, and then a series of group discussions regarding constraints issues and priorities.

Figure 4-2 Knowledge Loss in the Waterfall Lifecycle

The type of knowledge lost is the knowledge regarding customer requirements and not knowledge such as the practical knowing of how to do something.

<u>Knowledge Loss at the testing Phase and Implementation Phase</u>

As a software project progresses from requirements through to development especially in traditional software lifecycles such as the waterfall lifecycle, the solution gets limited exposure to client interaction. As the client starts to test the solution, it is then that requirements change and become altered to what was originally agreed. It is always a challenge to represent a solution from high-level requirements without creating some type of prototype first. As discussed by Osbery et al (n.d) in a survey carried out in 1996, it was highlighted by 71% that keeping a track of changing requirements was one of the biggest issues in terms of requirements capture.

At the testing and maintenance phases of the software project, customer requirements are continually changing and deviating from what was originally discussed. For users outside of the project team or for those inside the project team who were not involved in the original requirements, it is hard to know what was agreed if the requirements at all stages are not clearly documented. Even if an environment was created where normal non-client facing team members were allowed to interact with the client, a tool or environment would need to be generated so all interaction of both a tacit and non-tacit nature could be recorded and visible to all team members.

Knowledge regarding customer requirements becomes more haphazard once a design document has been signed off by a customer. Project Managers don't see the need to document any changes outside of what was agreed at earlier stages, as there are documents to refer back to on what was agreed. What is not considered is the longer-term maintenance of the product. The opinion could be that, after the solution is delivered, the product is signed off and it is no longer the team's responsibility. Rarely from a maintenance perspective are documents referenced and updated. Issues are discussed at the time and decisions made without real consultation of the original requirements or confirmation from original team members (Batista Dias et al. 2003).

In order to ensure that knowledge is passed through the software lifecycle, requirements need to be kept track of at all stages and not just at the actual requirements capture stage itself. In most software projects, the requirements documents are the basis for referral in later stages of the lifecycle or post go live of the project. There needs to be the realisation that requirements clarification does not stop at the actual requirements stage and that requirements form and change when the customer uses the system with real life business scenarios (Osberg et al n.d).

In terms of knowledge transfer, it is easy to review knowledge that has been documented in a design document and absorb and translate this knowledge into a solution. The challenge lies is at the user acceptance testing phase and maintenance phase where requirements are not stagnant, as suggested in older lifecycles models such as the waterfall model. There needs to be backward and forward traceability in terms of new requirements. New requirements need to ensure that they fit in with what has already been documented and developed (Orelena & Gotel n.d).

There needs to be a tool or process to manage the every changing knowledge within a software project. The biggest challenge appears at the user acceptance testing and maintenance phases. In older methodologies, the focus was largely on the process rather than the quality of the solution. One of the principles of agile development was to steer the focus away from the process and concentrate more on the customers needs.

As highlighted by Osberg et al, requirements traceability is critical to the success of a software project. There exist tools to track actual changes and give these changes priority and weight in comparison to the existing requirements. This tool is an improvement on change control management via documentation but does not cater or allow for the capture of the tacit knowledge associated with the change. There needs to be a process and a tool to manage the customer requirements from the start of the project and throughout the software life.

4.5 Methodologies to assist Knowledge transfer in a software project

Agile methodologies try to instil a faster time to client with small iterations and quicker customer use than previous methodologies. More bugs may be presented at the earlier stages as the software may not have had as much time in user acceptance testing, but if issues are reviewed by the customer at this stage, the cost is less than at a later stage (Paetsch 2001).

Agile methods are sometimes referred to as light in weight. Agile methods do not try to remove the methodologies such as the standard Waterfall Methodology but try to streamline these methods. As a result of this streamlining of methods, the focus is pushed more towards the people; end users rather then the actual processes themselves. A large focus of the standard methodologies is the documentation and getting customers to sign off on this documentation. In agile methods, the focus is more on a quality, working piece of software than getting the documentation right.
The main focus of agile methods is being more prepared if risks are realised. It is easier in an agile environment to make changes if there are misinterpretations about requirements as the customer is involved from the early stage throughout all stages.

Loss of requirements seems to occur and be realised as the customer spends more time with the solution and realises that their business need was not fully understood. Changes are required at the testing and live stage and are implemented by developers.
If these developers were not a member of the original project team, it is hard to pass on the experience of the original project without working closely with the person who originally developed the solution. Agile methods seek to pass on the experience rather than enforce processes for each phase. Cockburn once highlighted that software development was like a cooperative game of invention and communication – With the keywords being "cooperative" and "communication". As highlighted by Cockburn (2000) agile methods seek to involve the client and the developer at all stages to directly communicate what is required.

4.5.1 Extreme Programming

Extreme Programming is based on the values of simplicity, communication, feedback and courage. Extreme Programming was brought about due to the issues relating to long development cycles relating to the standard iterative software lifecycles.
Extreme Programming did not introduce anything new in terms of process but uses common sense principles. There are 5 stages used, which involve mocking up requirements into stories and continuous customer interactions until these stories are clarified.

The first phase of the XP process is called the Exploration Phase. This is where the customer highlights several stories, which eventually turn into functionality of the system. The second phase of the XP process is where these stories are estimated by the developers and prioritised by the client in the planning phase. The first release is scheduled at this point. What is different to standard processes is that there is not just one release to the customer when all of the stories are put together. There are several iterations until the requirements are correct, and then there is one release to the customer with all requirements. The death phase is where storyboards are not used anymore as the customer requirements are clarified (Abrahamsson et al 2002).
Different roles and responsibilities are defined which include the customer, a tracker, a developer and so forth. The tracker is an important role as they keep a track of what the developer estimated for the customer stories.

4.5.2 SCRUM

Another Agile methodology frequently used is called SCRUM. SCRUM comes from the principle of people scrumming together and passing the ball back down the line, similar to a rugby game. In a software project this can be attributed to users continually coming together with ideas and the responsibility being passed around the team (Schwaber & Beedle 2002).

Within the SCRUM methodology, what is called a backlog list is developed, which is a list of features required in the new system. The system is developed on what is called a black box to see what types of issues arise out of the requirements documented. The next phase consists of what are called SPRINTS. SPRINTS are similar to the phases of the standard software lifecycle in that it has requirements, design, development and testing. The Scrum methodology tries to achieve as with XP Programming is the realization of uncovering of risk at the early stages (Abrahamsson et al 2002).

Both SCRUM and XP try to overcome the staged phased approach seen in the earlier methodologies such as the waterfall methodology. These new agile methods try and remove the communication gaps between the client and the team members. These agile methods realise that requirements gathering does not just happen as a first stage in a software project but happens on an ongoing basis.

4.6 Conclusion

The main issues regarding knowledge loss in a software project is where requirements are discussed and lost and not fed back to the developer who is developing the solution. This happens when face-to-face conversations are not documented or made explicit or discussed with the project team. With a standard Waterfall Methodology a business analyst gathers requirements, which are passed onto the solution architect to design and then this knowledge eventually comes to a developer to develop.

As the developer is so far down the chain in terms of communication, items may have been lost or morphed depending on how many loops of communication the knowledge has come through. The largest knowledge loss only appears evident at the testing stage or even the maintenance phase when the customer realises requirements have been missed.

Agile methods do not try to achieve an introduction of a new technical process but a lighter way of managing the project so the solution is in line with the requirement. Taking both the XP and the SCRUM methods, similar traits are outlined in both methods, which are prioritizing requirements, and customer involvement to reduce risk.

Tacit conversations are still a part of both methods, with the focus taking place more so on the communication rather than the documentation. This communication style averts risk in terms of requirements getting lost when side conversations are had between the Project Manager and the client and never brought to the team's attention. Requirements conversations become before tacit when the customer has seen the solution and realise that items need to change. The challenge at this point is to ensure these requirements conversations are recorded so there is traceability of requirements moving forward. It is a challenge for a team member to come on board if the only form of requirements documentation is an out of date requirements document. Agile methods try to get the requirements right at the start by allowing team members to work directly with the client.

If an agile method was used with a software project, it would have to be very clearly managed in terms of scope and cost. Documentation and reference points such as the backlog list mentioned in the SCRUM methodology would have to be used to avoid scope creep. In order for the knowledge in this type of methodology to be reused for others outside of the Project team, a place for storage of this communication between client and vendor would have to be created. Even though all team members have a closer communication process and are aware of changes, there still needs to be a communication tool to record all project data so others outside of the project team could gain benefit and assist in the project without having little knowledge of what was going on in the project itself.

In Chapter 5, knowledge transfer tools are discussed and reviewed in terms of their collaborative nature and how these tools effectively distribute project knowledge to those outside of the project team.

5 KNOWLEDGE TRANSFER TOOLS

5.1 Introduction

As highlighted by Hunter (2002) classification is a *grouping together of like things according to common qualities or characteristics.* Items are grouped together so that when they are needed at a later date, all of the information is in one place. As discussed by Hunter (2002) there would be no need for classification if things were never going to be retrieved again. Without a common understanding of a meaning of similar items, there may be many groups of things that are the same but called something different. This chapter investigates the technology available to manage the classification of knowledge.

This chapter will discuss the Wiki technology in terms of a tool to manage the ongoing support for the sharing of tacit and explicit knowing for a project on an ongoing basis. Issues regarding a Wiki as a knowledge management tool will be highlighted and discussed.

5.2 Social Software

Social Software provides foundation for conversations and collaboration allowing for knowledge creation, sharing and publication and creating avenues to expert opinions worldwide (Avram 2006).

Examples of such social software are Wiki, blogs and forums. These tools allow for the communication and collaboration on the knowledge that is ever-changing and that can be captured, added and discussed throughout the whole software project lifecycle.

Social Software technologies seek to be the housing for the spiral of knowledge, where older technologies such as shared folders, content management systems did not have the collaborative, sharing capabilities. Social Software seeks to manage this previously undocumented tacit knowledge and feed this into existing explicit knowledge, adding value and depth to lesser quality knowledge. It seeks to be the facilitator of the spiral of knowledge by forming the bridge between the tacit and explicit knowledge and the regeneration of this knowledge back into the spiral.

In this chapter, investigation into certain social software technologies focusing mainly on Wiki technology is carried out. This chapter will provide an overview of this technology in terms of its capabilities for sharing knowledge. Several types of Wiki technology will be reviewed along with the features of each technology. Certain functionality within Wikis will be reviewed in terms of its ability to deal with tacit knowledge within a software project. Issues with Wiki technology in terms of the human difficulties such as resistance and usability will be discussed. Ways to overcome these difficulties will be addressed during this chapter also.

5.3 Wiki Overview

A Wiki is a website that allows anyone to create, edit and delete content from its pages.[5] A Wiki allows communication and collaboration on certain subjects and allows for the fine-tuning of knowledge by its users over a period of time. Wikipedia is a free, multilingual encyclopedia project supported by the non-profit Foundation, Wikipedia. There are 12 million Wiki articles (May 2009) written collaboratively by users all over the world. Wiki articles are articles regarding a certain subject that have been refined over time by the Wikipedia users. There are many Wiki technologies available to users to adapt to their own requirements. Most Wiki technologies are open source. Examples of companies that use Wikis as part of their core business and company intranet are Yahoo and Nokia (Theony 2005).

There are various add-on features and customisations available to download to add to your existing Wiki. Wiki technology is similar to any website, in that you need to set up a backend database, an application server and sometimes there is a requirement to install a scripting language, such as PHP.

Wiki technology focus is ease of use from a user's perspective. Users do not need to have prior web development experience and can easily contribute to their website with little or no assistance from a technical person. WIKI is a Hawaiian word that means "quick" or "fast". Wiki also translates as "What I know is." Wikis were originally

[5] A wiki is a collection of web pages created and edited by any user that has access to the internet. A wiki page is edited by adding text to a page for review and possible further editing by other users. A wiki was first introduced by Wikipedia in 1994.

created for the sharing of what we know. As the name suggests, Wikis are the quick way of sharing this tacit knowledge that reside with individuals into an explicit format (Ruman 2006).

Leuf and Cunningham (2001) define a Wiki as a *freely expandable collection of interlinked Web pages, a hypertext system for storing and modifying information.*
Wagner (2004) highlighted that *knowledge is the blood of your company* and a Wiki is a means of holding onto that essential life of your company. In the software industry, knowledge is core to generating revenue. In a software development company, time is spent developing code for a specific purpose for a specific customer or many customers, depending on your business. If you have someone or something that can reduce the time on producing a product, it results in lower resource cost and increased revenue. Imagine if this knowledge and expertise could be passed onto others without the years of experience. This is the goal of a Wiki, to harness knowledge regarding a particular subject in one place for re-use by others, reducing time spent trying to do similar tasks without the knowledge that could have been gained from another.

Investigation into the possibilities of extracting knowledge and classifying knowledge through many different media such as email and instant messaging has been carried out but there is still not one place to find all answers to questions posed in the software project. There are many tools out there that manage information and knowledge but one of the most user friendly and capable knowledge management tools is the Wiki.

The first Wiki system, WikiWikiWeb, was created by Ward Cunningham in 1995, so it is relatively new in terms of use and feedback (Brännström & Mårtenson 2006).

A Wiki differs from a normal content management system in that it takes the responsibility from a data entry person or a web developer and gives this control back to its users. A Wiki is like a collaborative website allowing anyone to change, update and delete content. A Wiki has the functionality to undo incorrect updates or deletes allowing the open environment to work without large supervision. In terms of credibility of information and in terms of scalability there is no control, unless constantly monitored (Brännström & Mårtenson 2006).

It is hoped that people do not distribute incorrect information on a Wiki as, if the system is in use, users will see incorrect information and it will be corrected and refined. Sometimes people are afraid to contribute at all and do not want to be seen as making a mistake and so they will normally check information first.

One could look at a Wiki and state that security is an issue. However, if there are issues with the information uploaded, someone can correct this (Brännström & Mårtenson 2006). A study was completed comparing Wikipedia and Britannica and only 8 serious errors were reported from the comparison of both systems. Errors were not just reported on the Wikipedia side with 4 errors being proved on both sides (Giles 2005). There are always going to be issues regarding quality, but there are ways to highlight or rate the information and give it a certain level of scoring. Ratings of quality can be restricted to only knowledgeable people in that field.

Start with some seeds and watch it grow, and the Wiki will become moderated by its users' community, respect and trust the users, leave them anonymous in order to avoid ego problems, good things happen when you trust (Buffa n.d). The entire principle for which Wikis were created and developed is based on trust of its users. A small percentage of internet users want to actually harm or remove information from a website but this may only be a small amount. If users are given more freedom to contribute what they want to share, they should feel more inclined to contribute than if work has to be redirected to management or approved (Cunningham 2001).

5.4 Wiki Products

Looking at companies such as Yahoo, Nokia and Disney and their implementation of Wikis, one has to realise that Wikis are not just for small sized companies requiring cheap open source software.

There are hundreds of Wiki tools available, a large percentage being open source.

The scope of the research is to allow for the transfer of tacit knowledge to a place that could be used by other users within and outside of the project team.

The Wiki tool required for this research needed to be open source as there was no budget assigned to this project for Company E-Learn.

The Wiki had to allow sharing between multiple users; it had to allow for collaboration of multiple documents at the same time. Functionality needed to include ways to

convert tacit knowledge into explicit either by integration with existing communication tools or management of the communication itself.

A list of tool requirements were set out below and, based on the amount of requirements that were met, a tool was selected.

Based on reviews from the internet and opinions from fellow Wiki enthusiasts, the following 3 technologies were reviewed.

Tool Requirements	**TiWiki 4.3.0**	**MediaWiki 1.14.0**	**PMWiki 2.2.1**
Open Source	Yes	Yes	Yes
Multiple User Collaboration	WYSIWYG Editing	WYSIWYG Editing	WYSIWYG Editing
Revision History	Yes	Yes	Yes
Discussion Capabilities	Yes	Yes	Yes
Easy to Download	No	Already Installed	No
Support	Open Source Support from users	Open Source Support from users	Open Source Support from users
Email Functionality	WebNotify	Watch Functionality	Email Notification
Categorisation	Webs (Groups of Pages)	Links/Categories	Categories
License	General Public License	General Public License	General Public License
Searching	Full Search	Full Search	Full Search

Table 5-1 Wiki Tool Comparison

TWiki is an open source Wiki that allows for easy edit and creation of content on the internet. TWiki has powered large organisations such as Yahoo, Nokia and Oracle. TWiki, MediaWiki and PMWiki all provided the features that were required to effectively transfer knowledge from one person to another on a regular basis.

MediaWiki was chosen as the tool of choice as it was already installed in the company. Time was spent trying to install the Wiki technology and it proved a challenge to set up the database, install PHP and set up a web server for someone who was not largely

technical. As the focus of the time was not on the installation of a Wiki, but about knowledge transfer in a Wiki, it was decided to use the current Wiki architecture that was already in the company.

5.5 Functionality of a Wiki

Wikis allow the ability of expression to highlight when someone is not right and it allows independent authors to share opinions and reconfirm rather than relying on someone's word. They allow the expression of this undocumented tacit knowledge to be merged with this explicit knowledge.

There are many scenarios from which the Wiki can obtain its knowledge, classified or unclassified from email, instant messaging, and upload of documentation of thoughts coming straight from a users head. Users can email a Wiki directly and an article is created based on what is after the @ sign. No matter where the knowledge has come from, it will never come from one place or one person and it will never all come together to form one concept. Due to Wikipedia open source nature, in that all users from everyone type of background can contribute, it does not discriminate against anyone in terms of cost or usage rights. It allows contributions of opinion and facts from all of the world's internet users (Kitter et al, 2007).

5.5.1 Articles

Wiki articles are created with simple options such as "create page". In Wiki technology, a large text area is presented to the user where they can simply type as much text as they like without need to submit this text to a webmaster (Stvilia et al 2005).

Figure 5-1 How to Create a Wiki Page

Wikis do allow certain types of formatting such as creating table of contents and text formatting through a type of WYSIWYG editor. A WYSIWYG editor is an editor that

allows non-technical users to format text on a webpage without any coding background. WYSIWYG stands for “what you see is what you get.”

Once pages are created, there is an ability to link to other pages using a linking option in the WYSIWYG editor. The goal of creating links within Wikipedia is to allow different concepts to be linked together in order to try to overcome the duplication of the same topic on several occasions. As Wikis save both opinion and fact, articles are designed to be easily manipulated and corrected so that they improve in quality over time (Stvilia et al 2005). As highlighted below, a clear indication of revision history allows users to continually change articles over time.

Figure 5-2 Revision History - Wikipedia

There are also different levels of articles in terms of quality within Wikipedia. A featured article is an article that is of the highest quality according to users. A clean up article is an article that is not entirely correct and requires work and a normal article is an article with an average amount of correct information in it (Zeng n.d).

5.5.2 Categories and Linking

There is functionality within the Wiki technology called categories. This allows users to classify and link pages to certain categories. It allows for easier searching in the future as it groups pages based on a certain category with which they are associated. Categories within Wiki technology seek to address the previously discussed topic of classification of tacit knowledge.

These categories do not cater for a different meaning for the same categories and the linking of these categories together to form an overall concept. As discussed later, an extension to the out-of-the-box Wikis, called Semantic Wiki, tried to overcome this

issue (Buffa 2006). As highlighted by Decker (2007) there is also the functionality called Text-Cast, which lets users add text, highlighting that the page belongs to a certain category using several pages at once. This allows ease of use from a user's point of view in that it reduces the labour-heavy tasks relating to Semantic Wikis, as discussed later.

5.5.3 Searching

There is certain search features available within a Wiki that allow users to search based on key words. In order to facilitate better searching, pages should be associated with different categories. One feature, which is not at the technical searching level of a tool such as Google, is Wikipedias search capabilities. There are no "did you mean" or spell checker facilities which would greatly improve the search features of a Wiki.

5.5.4 Discussion Forums/Talk Pages

Wikis have features such as discussion forums or "talk pages", as they are called, that allow for the discussion of a topic. These pages are to discuss current articles and remove the one-to-one conversations originally had with the author.

Through the use of easy uploading of content, categorisation of content, searching and discussion of content, knowledge becomes refined over time.
Through the combination of revisions and discussion regarding articles, inaccurate information is corrected and refined over time. Please see below where articles are discussed and updated, similar to the spiral of knowledge, where tacit knowledge is translated to explicit and tacit to tacit again. With a Wiki there is a record of this tacit knowledge.

Figure 5-3 Knowledge Refining (Wikis)

5.6 Advanced Functionality of Wikis

The first step is to overcome the issue of extracting tacit knowledge and getting this knowledge into a place of sharing and collaboration and sorting this knowledge into a category.

The scope of our issue is to try to sort and classify the tacit knowledge within a software project into sort-able pieces of knowledge for later use and update. Article validation within a Wiki is an impressive way to increase the quality of knowledge over time. It is hard to say that real value from knowledge can be obtained without ensuring that a knowledgeable person has reviewed users' contributions to verify their work. Article validation within a Wiki allows for what are called Editors and Reviewers to rate and make revisions of articles based on their level of content. It allows users to give feedback and this feedback in turn contributes to the learning process for all, as the original contributor gets to learn from someone else correcting their work. In most scenarios, it is not the intent to post incorrect information but to share views and possibly incomplete knowledge with others. This is what makes Wiki technology so useful and successful compared to a standard WebPages. Users are informed of when they are incorrect with their information and can learn from their mistakes. The intention of a Wiki is to collaborate as a large group of people, compared to a website where one user pushes information and the other user pulls.

5.6.1 Wiki Mind Maps

Mind maps originally served the purpose of letting people visually represent what they were thinking with a series of visual pictures.[6] By displaying these pictures, linkages were formed on paper and sub-concepts were formed. It also was a means to provide fun, interactive and collaborative brainstorming.

There is currently an extension to the technology MediaWiki that allows users to add their content with a mind map, showing the overall concept of the subject in a mind map format. It is proposed by Eric Larcher. It is based on the tools from www.Wikimindmap.org.

It takes an existing Wiki, asks the user to enter a search word and returns a mind map of all concepts of the word in the Wiki. This type of technology makes learning easy and also displays the relevant knowledge without extensive searching. This technology is currently still under review and there are no real life examples of its success to date. When you choose to click on a topic, it will bring you to the Wiki page in question.

Figure 5-4 Wiki MindMap.org

5.6.2 Document Templates

One of the main things that deter users from using Wikis is the complexity of adding tags within Wikis and asking users to categorize the text. It may be as simple as

---

[6] A wiki Mindmap is a visual representation of common topics on a wiki webpage displayed as a map of interrelated categories to easily search and find information.

creating a template for a user in each subsection of your Wiki, where users can fill in these missing pieces and fill out the actual text, rather than the formal Wiki text.

This template could contain tags that need to be filled in by the user, put clearly explain what the user has to do. Decker et al, (2007) highlight simple text as a requirements gathering technique that lets the users concentrate on the actual input and sharing of issues rather than the technology itself.

Simple items such as naming conventions for sections and documents allow items to be easily retrieved. These principles are nothing new for a Wiki but do encourage users to share and search if the process becomes seamless for them.

5.7 People Issues

The internet originally allowed for only technical people to add information to the internet. Being knowledgeable does not necessarily mean you are technical and a Wiki allows knowledgeable users to add content without any technical training (Buffa n.d).

If the types of users of the internet and Wikis are reviewed two types of users are identified: novice un-knowledgeable users and knowledgeable users of the context in question.

Social software promotes a group atmosphere where knowledge is shared and users feel rewarded when they contribute. This is the case in some scenarios but most users of the Internet are pullers of information rather than pushers. People in general don’t have time to sit around and contemplate what they know and want to share with others. In order to receive, one has to give and in order for the World Wide Web or even for a Wiki to succeed, someone and everyone at some point decided to share (Reinhart 2005).

It is why the success of Wikis from a global internet perspective is particularly strange. In the Wikipedia example, there is no goal from a commercial perspective for its users to share knowledge.

Companies invest in knowledge management tools to retain valuable organisational knowledge in order to decrease retraining costs or to decrease costs from a service point of view as the required knowledge is readily available.
Knowledge management from an organisational point of view is moving towards innovation, and capitalising on its knowledge rather than seeing it as just another initiative or application (Reinhart 2005).

Larry Prusak, executive director, IBM Corp.'s Institute for Knowledge Management said *in the past that the promise of knowledge as the origin of sustained competitive advantage encourages corporations to examine their current processes and implement knowledge capturing initiatives* (Reinhart 2005).

Wikipedia is exceptional as it removes the need for a large amount of administration staff to look after it as it looks after itself.
There is the opinion in some organisations that a knowledge management initiative is only a success if it is supported by management, owned by a knowledge manager and communicated to all its users. Without a Wikipedia representative coming into every person's home telling them how to use Wikipedia, shows the benefits, how and why was it such a success? Really the answer is its simplicity. Its success is similar to why Google achieved such success. If you ask Google a question with a word in it, one of the first results is Wikipedia with a definition of what it is. It is such a success as it is akin to reading a book and finding what page you are on through the use of a table of contents. The knowledge and information stored within a Wiki gets better because users want to know more and see how easy it is to share with users.

A Wiki removes the hierarchy of authority from the knowledge transferral process. Users may attempt to exert authority by hoarding the knowledge. If users are not rewarded for their contributions, it may lead to potential hoarding of knowledge in the future.

5.8 Wiki and Knowledge Transfer

There exists a real challenge to capture tacit knowledge, unless there is constant recording, monitoring by video cameras and streaming into a Wiki. Knowledge has the potential to become redundant if it is not kept up to date. Tacit knowledge, the know-

how, is often hidden in the practical procedures or thoughts of individuals (Reinhart 2005).

Nelson & Cooprider (2001) highlighted that users only share when they trust someone, so it important that a sharing environment is created and that the knowledge is communicated effectively.

According to Nonaka and Takeuchi (1995), and as highlighted within the spiral of knowledge, the process of changing tacit to explicit knowledge begins with first socialising internal knowledge; then creating new knowledge by merging it with what already exists and then creating more new knowledge by sharing it with others. As highlighted in the spiral of knowledge, most of the knowledge shared reused and passed on is in a tacit form (Reinhart 2005). The challenge within any knowledge management tool is not the refining of explicit knowledge but the converting of the tacit knowledge gained at the extremely beneficial socialisation process to a knowledgeable explicit format.

Maybe knowledge management systems will become so advanced that they won't need humans in the future to manage the knowledge that is inputted and ensure that, when it is used by someone else, it is still as valuable as when it went in and still in the context of what it was meant. There may be techniques and tools out there that can contextualise what a human being intended to mean and interpret that without significant interaction from a human.

The components of a successful knowledge management system consist of people getting the requirements correct, the people using it and gaining value from it. If the assumption is made that the requirements for the system are gathered correctly, and implemented as per the design, can one assume after this point the tool is self managing?

5.8.1 Semantic Wikis

Without a technology to comprehend a context of a subject, users will have to find a tagging system that will result in easy grouping of data based on a tag or labelling of information. A new extension of Wiki allows users to add meaning using tags so chunks of information do not sit out in isolation. A Semantic Wiki tries to form

relationships with information and form webs of concepts rather than islands of information.[7] Although the concept would work if all users followed the rules, adding semantic annotations to information pieces requires effort. Users will only make this effort if they benefit from it (Eyal Oren n.d).

The overall issue of the problem being reviewed as part of the case study is creating usable Wiki articles from an abundance of tacit knowledge so that is easily transferred to users inside and outside of the project team. Wikis have grown in popularity because of their ease of use and abundance of information, but this popularity has also led to a bigger issue. With all of the information available and all of the people using it, it gives back to one of the main crises of knowledge management which is duplication of information and bad data due to naming conventions of data. Semantic Wikis try to annotate content so that is represented easily and simply navigated through.

Without adding meaning to the context of the article, there is no way to say that the same article with the same information is hidden within the same Wiki. In order for a Wiki to be beneficial, the information needs to be searchable and reusable.

5.8.2 Annotations

Annotations are termed as “data about data”. Annotations are the categorisation of information and summary of its subject (Oren &Del n.d).

Annotations define content by using what is termed as a predicate and object. For example, capital city and Dublin would be the predicate and object in this scenario as Dublin is the capital city of Ireland. In order for Semantic Wikis to work, the descriptions of the content on each page have to be linked to the descriptions on another page to provide an overall context of a subject.

In a Wiki, you are either searching for knowledge you know you want or you are brought to knowledge you would never of thought of without the suggestion of linking

[7] A Semantic WIKI is WIKI that adds content and meaning to categories created within the Wiki. Semantic wikis try and classify and group information so that topics can be linked, and duplication is avoided.

from the existing search. The issues of context in terms of what you detail in your annotation will never change. In terms of the amount of detail, you can add to a general body of knowledge and the way it is returned has advanced with the use of semantic technology.

5.8.3 Predictive Annotations

Depending on the type of Semantic Wiki procured, some Wikis have the functionality to provide extensive meaning to the definition of the knowledge entered in the Wiki. As highlighted, one of the challenges of semantic technology is to get users to add context to their content with little technical experience and extra steps compared to what they are used to. An auto completion feature suggests alternatives and explanations that have already been entered for existing text entered by the Wiki users. This avoids creation of similar ontology's within the same Wiki, and also creates a sense of ease for the Wiki users. Once users see and realise the benefit of this auto completion functionality in terms of retrieval of information at a later stage, they will be encouraged to use this feature.

5.8.4 Knowledge Wikis

Knowledge Wikis have two ways to retrieve knowledge; either by normal browsing of pages and standard retrieval of information or by an interactive menu which guides a user through the system via a type of question and answer style of communication through window pane navigation. This is called "in–place answers".[8] As a result of a series of questions, tags are created in the navigation panes, which describe the content of the subject's concepts. The window on the right shows the edit pane of a topic, which implements a decision table. Through the use of an easy to use interface, users can quickly add context and link to other pieces of context with ease.

---

[8] Knowledge Wiki is a Wiki that tries to build on the standard wiki by adding more content and context to the information on the Wiki. This type of Wiki asks the end users a list of questions and as users add content and answer questions over time, categories of knowledge are created based on the answers to these questions and past answers by other users..

Through the use of WYSIWYG editors, the process of users needing prior information of web authoring tools is eliminated and with the use of questions posed regarding the content, the actual context is refined down to a cleaner format. [9]

An addition to WYSIWYG editors is the ability to categorize Wiki pages on creation of an email. How this is implemented is quite simple; every time a user wants to add some piece of new knowledge to the Wiki, they do not have to leave the email package. Depending on the information after the @ sign in an email address, a user can create a new entry to a page. The page name will relate to whatever is after the @ sign in the email address.

Sometimes it is as simple as too many clicks to actually do something. If we click a link on a web site that leads to another website, to another website we normally forget the reason we were on the first website. This advancement in technology allows for easy submission of knowledge to a Wiki and allows the Wiki classify the knowledge for you. It may not always be a success in terms of the classification of the knowledge but it does make sure people contribute.
If the knowledge sharing process can be integrated as part of an employee's work process it will take one of the road blocks out of the knowledge sharing process (Hunter 2002).

A lot of functionality with search engines such as Google has been provided to us and, as a result, our expectations in terms of information retrieval are extremely high.
Google provides the most relevant in terms of what it thinks you meant and how many other users thought the same thing in comparison to the basic searching of the wiki.

A Wiki fails in comparison to a search engine of Goggle's standards is in the actual searching itself. A Wiki is easy to use and it is easy to publish content to. When the filtering out the knowledge is the goal of the wiki is lacking in this area. Without really

---

[9] A WYSIWYG editor stands for what you see is what you get. It is an editor that allows users add text to a webpage with the addition of simple formatting options such as bold, italic and underlined. This editor removes the need for a user to have previous web scripting experience to add information to a webpage.

integrating the existing tools such as email, and instant messaging it would be a challenge to see how people would use a Wiki as a communicative tool rather than a collaborative tool.
Until the searching technology becomes as advanced as a Google search engine, users will not be encouraged to use it.

One barrier that exists for Wikis is that there are many applications existing in the software development world that exist for certain purposes. For example Bugzilla is used in some organisations to track issues relating to software. Valuable knowledge can be gained about pieces of code that can exist on its own but will never be transferred into a Wiki and will probably be lost. If we take an example of a software project, we will probably use many tools to track issues before and after the system goes live.

Semantic Web technologies and tools require considerable technical expertise, and are thus not well suited for users outside the field of computer science.
Also, knowledge engineering tools for the Semantic Web are currently mostly single user and do not provide good support for the collaborative creation of formal knowledge. This makes it hard for domain experts and knowledge engineers to work together on knowledge engineering tasks (Schaffert et al 2004).

5.9 Tag Clouds

All communication that is text based, and that can be exported in bulk and imported, can be run through what are called tag clouds. [10]Tag cloud technology is where a visual representation of repeated word is produced with a block of words is fed into the tag cloud (Hayes & Avesani 2007). Tag clouds use a series of algorithms to check for the most frequently used words in a piece of text. The *tag* does not try to understand the word or link it to any type of category but merely highlight its regular use. If the text used in our email and instant message is reviewed, patterns of used terms are

[10] A tag cloud is a piece of technology that creates groups of words depending on how many times they are used in a body of text. Tag clouds do not categorise similar words, rather highlight the use of the same word in larger font compared to use of another word that is used less in a smaller font. Tag clouds simply highlight frequency of words through sizing of font.

highlighted and focus can be given to key topics. The technique can be used before and after the transfer of knowledge from one media to another, i.e. when the knowledge management tool is constantly being uploaded to, you can continuously run the tag cloud software to see where the focus of the knowledge is moving to. Media that has a text-based output has the potential to be used with tag cloud software. Tag cloud software highlights the most popular topics but does not sort the actual information for you, it is up to you as the user to then sort the data topics based on your search.

An inherent problem of human indexing process is the called "inter-indexer inconsistency. This is where different people use different indexes to tag the same words (Montero & Solana 2006).
This research highlights that with specific changes to the way the logic of the way the tag cloud works, the tag can be programmed to remove unnecessary words, such as "hello" and a person's name and then only display relevant words. One suggestion is to semantically link words that are similar to each other, therefore avoiding duplication of the same data.

Without the management of the tagging system, i.e. suggestions on tags, how would the users ever know what tags to use, which may result in hundreds of tags, all different meaning the same thing? (Hayes & Avsensi 2007).

Tag clouds are designed to classify similar words together, depending on their occurrences in a piece of text. Tag clouds do not seek to understand contexts of conversations, or come to conclusions of a subject within a conversation. The benefits of tag clouds seek just to group like things, and so instead of manually reading every email or message, a pattern of regular words can be recognised. Difficulties in tag clouds do exist, where different words are used for the same subject. Without semantic understanding, built into the tag cloud software, this type of occurrence cannot be avoided. What tag clouds do not seek to overcome is transferral of knowledge into an explicit format. They only serve the purpose of classifying tacit knowledge that has been made explicit. Tag clouds do try to overcome the issue of unclassified knowledge existing in many different media. They have the possibility to sort text based on manual input of text. Tag clouds do not try to overcome the communication process between email or instant messaging to a central storage area. The link between all

communication tools is still a manual process and, as highlighted, tag clouds only seek to overcome the classification issue at the externalisation stage of the spiral of knowledge.

5.10 Conclusion

Wikis try to overcome the issues discussed, such as the transmittal of all of the tacit knowledge to explicit knowledge. Wikis serve as a media for transmittal and also serve as a place to share and express opinions on tacit knowledge. Wikis are the foundations of the spiral of knowledge as it caters for transfer throughout the spiral. In a software project, Wikis allow for the discussion of new projects, they allow for this discussion to turn into documentation and for further discussion regarding this document.

Although Wikis serve as a media for communication on this tacit knowledge, standard Wikis do not try to understand what is being discussed. Wikis can very easily take in knowledge and store this knowledge, but the real challenge exists between merging old and new knowledge highlighted at the internalisation stage of the spiral of knowledge.

If a Wiki is not supported and sold by management and allocated to a specific resource, both technical and managerial, it is a challenge to keep the system in use and to gain advantage from it. Within the context of any organisation or team there is going to be a varied comprehension of the benefits of the new systems. The ranges of use are varied from a technical knowledge base to a process index of documents. One must not forget that, no matter its intensions of use, users have to see a need for use. When something becomes optional for a user, it needs to be seen as beneficial or else reward needs to be given for the use of the system.

Knowledge in a Wiki should be clear and should stand out. It should not go out of date and it should be easily discussed. It should be as organised as possible. This should not be left to a human to decide but the tool should do this for you. Via the use of a well-programmed document management system with an excellent communicative tool and the people to drive it, there is a higher change of success.

The goal of the collaborative element of a Wiki tool must promote learning.

Within all of the tools available, such as message boards, forums, instant messaging, it is important that the system and knowledge shared is still saved down in a structured format, i.e. that the knowledge stored down is easily retrievable and understandable by someone else, i.e. it is still organised and codified but now explicit rather than tacit.

They key to a successful Wiki is communication, learning, sharing, recognition, ease of use, ease of access to information and up to date consistent information.
In almost every situation, the less time that it takes to do something the more someone will use it. If there are more clicks involved than people are used to, they will not use the system. If people have to contribute to two areas, they will not use the system.
Users have to see gain in contributing to the system, such as making an improvement in their day to day jobs, financial gain etc.

The mains issues lie in terms of users and Wikis is the way and what a user contributes to. All users have different techniques and style of working. Everyone has difference personalities. Everyone's perspectives on something or experiences are different; this is what makes us human beings. Without the use of technologies, such as Artificial Intelligence, a tool will never to able, in my opinion, to interpret what you actually meant to say and what you mean. There will always be a need to introduce a *Knowledge Manager* to govern its use, contextualize and monitor the content.

The tool should be built is such a way that it manages the organisation of the knowledge but the person needs to make sure everything is in context. The concentration should be placed on the people in terms of use and collaboration rather than the focus being on the complex capabilities of the technology. The tool should be a wide platform with the ability to allow large collaboration across individuals. "*We must be systematic, but we should keep our systems open*" (Whitehead 1938).

6 RELATED CASE STUDIES

In order to prove existing theories in this research, investigation was completed into recent deployments of Wiki systems, Semantic Wikis and Knowledge Management Wikis. Opinions and lessons learned from previous implementations carried out helped channel thoughts into the approach for the knowledge management project for Company E-learn. This chapter seeks to apply lessons learned from previous implementations to form an approach for Company E-Learn. Each of the case studies was chosen due to the similar characteristics in requirements in terms of knowledge management and the following questions were asked of each case study:

1. What was the scope of their work?
2. What are the results or findings?
3. How could their results help shape my requirements and approach?

6.1 Case Study Review 1

Implementing a small, but scalable knowledge management initiative in a software development team (Higgins 2008).

What was the scope of the work?

Higgins carried out an implementation of a Wiki in a small scale software development team in a financial services company. Higgins addressed the issues of tacit knowledge residing in certain individuals within a software development team without others being aware of the same knowledge and subsequently reinventing the wheel. Higgins highlights the financial repercussions of the failure of knowledge transfer in the context of a software development team. The initiative came about due to the nature of the organisation being a financial services company and that the amount of profit reduction, which sought the reason to cut costs in training, hiring and by promoting reuse in terms of organisational knowledge.

What were the results or findings?

The project was a success due to its small sized approach and constant management from Higgins. A large percentage of the organisation's knowledge was in a tacit format.

Learning within the organisation was carried out via face to face communication. Higgins highlights the need to promote a knowledge sharing community in order to first get people to socialise and share knowledge and share within communities of practice. Higgins emphasises that knowledge is just information if it has not been put into practice. Higgins highlights that through the use of Wiki technology, communication and collaboration is achieved and users can take tacit knowledge that has been made explicit on the Wiki and put this knowledge into practice.

How could the results help shape my requirements and approach?

The results showed that due to the constant involvement and time put into the set-up and constant management of the Wiki, it was a success and managed to effectively share and reuse the organisation's tacit and explicit knowledge. The same issues which were highlighted in Company E-learn are highlighted by Higgins in terms of trying to save lost tacit knowledge in a software team. Higgins was trying to avoid duplication of tasks, reoccurring issues and, as a result, a loss of time and credibility. The size of the project team and type of business they were working at were similar to that of Company E-Learn. Higgins makes a good point in that knowledge management is not all about the documents that are shared on a Wiki; it's more about one person talking to another about the topic and gaining benefit for it. The project was a success because Higgins was directly involved and could communicate closely with the team. The project was also a success because there was already a knowledge sharing culture within the organisation, which meant that users wanted to share before the tool was put in place.

Higgins showed through the usage statistics that the Wiki was used for both contributing and receiving of new knowledge. Higgins statistics showed that contributions grew over time. Although Higgins did not clearly indicate what made the Wiki such a success in terms of exact procedures, evidence was proven by Higgins to show that the users did see benefit in sharing and, as a result, uptake on the system grew very quickly.

6.2 Case Study Review 2

The process of knowledge transfer, a diachronic analysis of stickiness (Szulanski1999).

What was the scope of their work?

This research refers to knowledge transfer as stickiness and examines an example of knowledge transfer within a cycle of product development in an electronic company. As a result of this research completed, a model of knowledge transfer has been proposed. This case study highlights that the knowledge transfer process is a 2-way relationship, especially in a tacit to tacit environment. Depending on the media that is accepting the knowledge, this may be unable to absorb or soak in the transferred knowledge. Depending on the media, the knowledge may appear as a different pattern in someone's head, and not in the way that it was meant.

As Szulanski highlights, *lack of motivation may result in foot dragging, passivity, feigned acceptance, hidden sabotage, or outright rejection in the implementation and use of new knowledge.*

Success may only be obtained through what is termed value-setting or providing incentives to share. The principle of learning by doing is discussed; in terms of you will never really learn something until you put this into practice, highlights that you have to convert this explicit knowledge to tacit knowledge before really saying you are knowledgeable about the subject.

What were the results or findings?

Questionnaires were filled out by companies who took part in a knowledge transfer exercise. In all case studies undertaken, knowledge got to what was called the implementation stage of the 4-stage knowledge transfer process, highlighting no real issues in the transfer process. There was evidence of less stickiness of knowledge for those who had previous experience on a subject, resulting in some way of rejection of the new knowledge. Results showed that people learn when they were not interrupted or the knowledge transfer is not interrupted. Knowledge transfer needs to be treated with the time and management of a real software project.

How could their results help shape my requirements and approach?

This research showed that each stage of the spiral of knowledge needs to be broken down into its own process or stage. The focus of this paper was on knowledge transfer and subsequently this was broken down into a 4 step stage. Knowledge transfer seemed to succeed when time and communication was dedicated to the process. Ownership from both sides needed to be taken in the process and when issues occurred; they were

addressed at the time and were not put off or swept under the carpet. In this case study, knowledge transfer is not an easy process and the relationship between the giver and the receiver needs to be addressed before the process starts. As highlighted previously, knowledge transfer is about the seller and the buyer and the media and motivation between both parties.

6.3 Case Study Review 3

Improving software Quality by reusing knowledge and experience (Basili & Calderia 1995).

What was the scope of the work?

Quality is improved in the software development lifecycle by reusing components of knowledge and actual software that has already been developed.

There is a known advantage of companies reusing personal experiences but this could be extended to reusing components of software. A Software engineering lab was set up in NASA and dedicated to reuse from a software and organisation knowledge perspective. Results were that cost was increased at the start of the reuse lab but decreased at the development stage. It was found that there was 30% reuse across projects in terms of software reuse.

What were the results of the findings?

Results showed that existing processes could be changed in terms of management and reuse of organisational knowledge. The SEL project managed to change and instill new processes in terms of knowledge reuse.[11] Effort was needed, which meant increased cost at the start of the project but rewards was gained over time. By repeating success stories already experienced, they were able to install quality into products and services.

How could their results help shape my requirements and approach?

---

[11] SEL (Software Engineering Laboratory) was a laboratory set up in 1976 at NASA by the Department of Computre Science in Maryland University, NASA and the Computer Sciences Corporation. It was set up to improve software development process in an organisation, Flight Dynamics Division.

From this case study, it was highlighted that it is possible to re-use what already exists in terms of knowledge. There needs to be dedicated time and effort and a project set up especially for this purpose. A lot of organisations cut knowledge management initiatives as they do not feel immediate reward for the input investigated. It is important to note that return on investment in knowledge management projects takes time.

The core problem within this research was not about transfer of tacit knowledge to users outside a project team but of reuse in terms of software and actual organisational knowledge. It highlighted the fact that, although there could be return on investment over a period of time, it is hard to prove quick wins and cost reductions until several implementations of similar projects have occurred. The same applies to knowledge reuse in a knowledge management project, as in it is a challenge to highlight quantitative measures of success at the start of the project as most rewards can not be measured until the end of a project. It is hard to put a price on positive results, such as employee satisfaction and speed in terms of tasks.

6.4 Case Study Review 4

A synthesised knowledge mapping framework to embed a KM strategy using topic maps and Wikis: the Tao of Wiki (McMahon 2008).

What was the scope of their work?

McMahon looked at implementing a knowledge management strategy in BIM (Board Iascaigh Mhara) that tried to provide an overall company overview of the organisations knowledge. McMahon highlighted that it was easier for the human brain to remember a visual representation of something rather than a block of text. McMahon's approach was to use topic maps as a representation of the company's knowledge but, after some investigation into this technology and the knowledge within the company, it was realised it was too large a task to translate the company's knowledge into topic maps within the time of the project. A company Wiki was then taken on board as the Wiki chosen satisfied a large amount of the knowledge requirements but did not visually represent the knowledge within the organisation.

What were the results or findings of the case study?

McMahon was concerned not just about showing a visual representation of where knowledge lay but how knowledge related to each other.

One of the main issues regarding McMahon's findings was that users were more concerned with how the Wiki looked and how aesthetically pleasing it was. Users liked the ease of use of the Wiki but had concerns about ownership of documentation.

One important point that McMahon highlighted was the issue of users "putting themselves out there". There needed to be encouragement and communication that it was ok if information was wrong as it can be easily changed in the future. Although users did share their own tacit knowledge, users liked to keep their own documentation and did not like their powers being taken away.

How could their results help shape my requirements and approach?

McMahon makes an excellent point that knowledge management projects are constantly being treated like software development projects in that, there is a quick review of the requirements, rushed into development and then delivered. McMahon highlights the most important parts of a knowledge management project is the cultural awareness and the explanation of the benefits of what knowledge management can do for your role.

McMahon makes an important point of defining a framework within your Wiki so that you are not back to the start in terms of sorting your knowledge once users have spent time contributing to it. Unfortunately because we have moved so fast, if we don't find something quick enough we tend to move on and start again.

Although the knowledge map was abandoned, there was still a need to visually display the abundance of knowledge. Although part of the original goal was to visually display the company's knowledge, the Wiki did not satisfy that goal. The Wiki was deemed as a success within the organisation, so maybe the idea is to take baby steps and the Wiki was a first step.

Upon review of this body of work, it was realised that selling the system and the communication of its purpose was probably as important as its technical abilities to deal with the tacit knowledge that existed. Although its purpose was to sort all knowledge

into one area and create Wiki articles, the most important factors to start the project was to ensure that the user interface was easy to navigate and the knowledge stood out in terms of an index or table of contents so users knew where to go when searching. Possibly the next step would have been to link all knowledge together so that it was represented clearly and semantically but maybe it is not possible to get to a semantically defined Wiki in such a short period and possibly the results of something of this size could only be seen over a bigger time frame.

The technology should not hinder people and people should not become laden down with the amount of knowledge they have to create, share and upload. If the knowledge management tool is to be a success, it does need the capabilities to manage ongoing tacit knowledge. McMahon highlights that before you get to this point, you need to be able to sell its usefulness to the team first. Although the focus of McMahon's research was not identical to the core problem within this research, in that McMahon's core focus was not to transfer the tacit knowledge of a software development team to others outside of the team. The similar characteristics included rolling out a Wiki to manage organisational knowledge of both a tacit and explicit nature.

6.5 Case Study Review 5

Case Study Title - AceWiki: A Natural and Expressive Semantic Wiki (Kuhn n.d).

What was the scope of their work?

The ACE Wiki deals with the user interface issues within Semantic Wikis in terms of asking users to contribute tags to classify knowledge. This case study was completed on a group of users that were not familiar with Semantic Wikis. A population of 20 students with different technical and non-technical ability was surveyed. Results of contributions, in terms of sensible sentences, were also evaluated to see if a change to standard Semantic Wikis in terms of the way information was presented to the users in the user interface had improved the user contribution.

What were the results or findings?

It was found that existing Semantic Wikis made assumptions that users of the Wikis were all of a certain technical level. This assumption puts users off using the system and, as a result, ends up not being used.

The results indicate that the three design principles — naturalness, uniformity, and strict user guidance assisted in the contribution of knowledge to the Wiki.
The results show that almost 80% of the created sentences made sense.

How could their results help shape my requirements and approach?

Again, no matter what the technology and the tool; it needs to be easy to use as the Wiki is something that will be used by all levels of users.
Semantic Wikis will always struggle to overcome the usability issues that are associated with their functionality. The goal of this study is to try to translate the tacit knowledge of users in a software team to an explicit format. It has been shown that this is achievable by looking at previous case studies. As a result of a successful sharing environment being created and users contributing to one place, there still exists the issue of making this information usable continually from an ongoing perspective. In this case study, the focus was to structure this knowledge upon addition to the Wiki, so that it was automatically sorted upon upload. This case study proves as previously stated, we do not always know what we know. With this series of questions, this Semantic Wiki seeks to overcome this and pull knowledge from the corners of people's minds that they were not even aware that they had.

6.6 Case Study Review 6

Wiki Pedagogy – A Tale of Two Wikis (Bower et all 2006)

What was the scope of their work?

In this research, two different Wikis were deployed in the area of the management of knowledge in Information Technology. This research tried to investigate the issues around sharing problems and issues of an IT nature in a real time environment. The goal was to edit content synchronically with others.
A number of extremely valid and useful questions were asked such as "how are Wikis as a learning tool?" and "how much learning of how to use a Wiki is needed before users use a Wiki as a learning tool?" Two types of learning environments were created on two different Wiki applications. One environment consisted of asking questions on object oriented language on Moodle Wiki and the other consisted of a semester long project on Advanced Web Technology.

What were the results or findings?

In the Wiki that required weekly interaction, piecemeal contributions were added. Students did not link to pages within the Wiki. According to the survey that was completed, students found the Wiki a good way to communicate, rather than as a learning tool. Minimal training was required, which meant the user interface was easy to navigate and use without minimum training.

Additional features were required from the semester long group, which consisted of white board functionality, update notification functionality etc. Students in the semester long group found it difficult to fathom the idea of a blank canvas to share knowledge so templates were suggested as a starting point. Students still felt that effective face-to-face communication is still needed and better than a tool. Those collaborating on a weekly basis found it a waste of time to communicate via the week and felt that the time would be better spent face-to-face.

One student member said, *"I see the Wiki pages as a tool, which makes it easier to communicate between lecturer and students and also between the group members, but not as a tool that enhances learning experience."*

How could their results help shape my requirements and approach?

The environment for which this Wiki was set up was very much a learning and training environment, rather than a knowledge share. Possibly more of a scheduled mentoring environment or e-learning environment could have been chosen. Wikis do not force users to answer or pose questions and so items can be left open and unanswered.
Great lessons were learnt from this paper in that you need to make clear what information goes where in a Wiki or it will become out of control. Discussion points need to go into discussion forums and less conversational knowledge needs to be published in an area that is ready to use and to be reviewed section for all users. Depending on the scenario, templates should be used to create a starting point. Nothing can beat a face-to-face conversation with someone, if there is time. Providing feedback, in terms of ensuring the knowledge created is correct and ensuring people know that their contributions are being recognized and being rewarded. The author of

this paper poses a valid question of will the knowledge users ever gain from the Wiki collaboration as the will they always give knowledge and never receive?

6.7 Case Study Review 7

Eureka: web based tool for collaborative problem solving and project work documentation of knowledge creation and discovery
(Tan & Wong 2008).

What was the scope of their work?

The scope of this work was to create a place that facilitated learning and knowledge sharing in the institute, NTU. The application was called *eureka*
The tool was focused on managing all of the institute's project work and to create a space where all of the students could share and transfer their project knowledge via this Eureka tool. The focus of the content of the system was not just college course material but was on college project work, ensuring that users could think for themselves and complete a project without much assistance from lecturers. The focus of the research and the focus of the Eureka tool were knowledge discovery and the management of this knowledge.

What were the results of their findings?
The portal aimed to mimic face-to-face communication between students and supervisor, translating what were once meetings of a tacit nature into something that would be explicit and traceable. It allowed all project team members to contribute equally and allow for progress to be monitored all of the time.
The research was not just an internal project for the institute but also served as a repository for organisations associated with the institute.
Discussions were managed via discussion forums. Forums were categorised into subject and documentation from the project was associated with the tool.

How could the results help shape my requirements and approach?
The tool Eureka proved that knowledge transfer was possible through the use of a tool designed for this purpose. Knowledge that students owned had to be articulated in order

to communicate progress and issues to superiors. Supervisors that were not in the project had to try to comprehend what was completed in order to mark the student.

This case study assisted in the knowledge transfer research as it showed that it is possible to not just transfer knowledge from one person to another via a tool, but it is possible to collaborate and share using a tool. This case study showed that a tool did not have to be abandoned and that face-to-face conversations were not resorted to.

6.8 Case Study Review 8

Previous Implementations on agile methods to improve knowledge transfer. Is Internet-Speed Software Development Different? (Baskerville & Ramesh 2003).

What was the scope of their work?

A study was carried out in 10 software development projects and a discussion group was also created regarding standard software development lifecycles versus new methodology called Internet speed software. Internet speed software methodology is similar to the agile methodologies. Analysis was carried out using open end interviews, followed up with discussion groups to review issues regarding the methodology in question. The goal of the case study was to highlight where internet speed software methodology could overcome the failures of the standard sequential software methodologies.

What are the results or findings?

In order to compete within the software development industry, companies need to be able to get their product to market quickly. It was shown that there were several characteristics that were highlighted as import with the new agile process, which were not evident in previous older software methodologies.

Examples of this included parallel development and allowing the customer to be a part of the project team. This allows for a customer to really express what they want in terms of requirements and highlight risks throughout the project, as opposed to waiting until the end product is delivered. One important factor that was highlighted was that the software and knowledge should not be thrown away but should be reused. Sometimes there is the perception that agile methods abandon any type of structure or moves away from documentation. This is not the case. It was shown that documentation

is still a large factor of software development and knowledge needs to be reused and the "throw away" attitude needs to be improved on. Baskerville makes a good point that agile methods are no different to software methods but they are extreme in their nature. They seek to readjust the flows of activities and the practical elements are just hastened by all members being in close contact with each other and cycles being shorter and the customer getting to see a product quicker.

How could their results help shape my requirements and approach?

It was highlighted in this study that agile methods work well with an ever-changing project that is volatile in nature. When customer requirements are continually changing, agile methods allow for quick action, based on new information rather than having to start again from scratch with standard principles. If releases are split up, there is less risk in terms of having to throw away code as the client is not waiting on one release at the end of a long development cycle.

The focus on agile methods is focused on the relationship between the customer and the entire project team. If this environment of trust is set up, there is a better understanding of the customer needs as requirements are not based on a 2-day workshop, where the customer attempted to document everything they needed in the system. In terms of knowledge transfer in a software environment and in terms of Company E-Learns example, there is the possibility that an agile method could assist in both an internal and external knowledge transfer point of view. If an agile method was employed between Company E-learn and Company B-2-B, items that were discussed outside of requirements would not be lost because the communication would go straight to the project team rather than the business analyst. If there was some sort of a tool such as a Wiki tool that could manage this communication between the client and the vendor in an agile environment, this knowledge would then be available to those outside of the project team.

6.9 Conclusion

As highlighted in past implementations of knowledge management projects, these types of implementations do not always run smoothly. There are various factors that have to be considered before kick-off of a knowledge transfer project. These factors are discussed below. Upon review of previous implementations, it was realised that these

issues were not just technical issues or the approach taken but largely the people who took part in the knowledge management projects. As (Collison and Parcel 2001) have proven in the past, knowledge management is not just about the technical issues that may arise but it is about the combination of the people, process and technology.

The scope of the research was to investigate if any other organisations or groups had taken on board similar issues and what could have been learnt from their research. Higgins had a similar issue and approached the issue with the implementation of a Wiki to manage the knowledge of a software team. This proved that knowledge transfer was possible via a Wiki tool but barriers to this process had to be addressed. Wikis in other organizations, such as in Board Iascaigh Mhara, were utilised as a sufficient tool to manage the knowledge of certain teams. A large amount of the case studies involved only small teams to prove that knowledge could be managed via a Wiki tool.

Factors that were discussed as barriers to the knowledge transfer process are seen below and fall under the issues of process, people and technology:

6.9.1 Process

A process is an agreed way of doing something. Processes in terms of management of project knowledge highlighted as a result of previous case studies of similar scope are outlined below.

Communication Process

The communication process needs to be ongoing through the implementation of the knowledge management project and post implementation. The technology and the owners need to ensure users are aware of new knowledge and can gain benefit from it. Success and failures of the system need to be frequently reviewed and changed. Feedback needs to be listened to, discussed and acted upon.

As highlighted in the case study where an agile method was used as methodology for the management software projects, these principles could assist in the knowledge management project. If the goal of the exercise is to retain knowledge from a software project so that others outside of the project could gain benefit from it, an agile method would be more suited to trap this potentially lost knowledge seen in older traditional

software methodologies. With the close communication between the customer and the team, requirements are carried over to the team without getting lost in the loop of communication.

Roles and Responsibilities

Before any project starts, there has to be a reason for the project for it to obtain funding. Although in some cases, Wiki implementations do not cost money, people will have to spend time getting it up and running. Before a project is kicked off in terms of its implementation, it does need a senior management buy in. Sometimes once this buy in is given, communication with management only takes place sporadically and at the end, with a review taking place to see if it was a success. Management need to be involved more often than this as users needs to feel its importance of sharing.

Within a knowledge management implementation, there is normally a Wiki champion, knowledge manager or owner of some sort. There need to be more roles involved than this one person. A knowledge management project should be seen as an endless project. This project needs to be constantly marketed; it needs to be sold like a product. Users need to take responsibility of ownership of reviewing certain sections to make sure information that has been created is correct. A knowledge manager needs to create and manage a communication plan, highlighting each user's responsibilities and the impacts of non use.

Contribution Process

Users sometimes feel that they do not know where to start. It is important, not just to tell users how to use a Wiki, but what type of knowledge they should be contributing and what goes where. It is a good suggestion to give users a template so that they can at least have a basis to start from. It is important that whoever owns this section, that they take ownership of it and give users feedback and formally create a level of quality so that users can learn from this feedback in the future.

6.9.2 People

Environment

One of the factors of success of a knowledge management project is the environment for which it is set up. If there is a previous learning environment, it is easier to build on this environment and introduce new processes and procedures to share more and talk

more. Even if there is not going to be a tool to carry out this communication and instead communities of practice are going to be organised, then this is a step forward in sharing. No matter how strong a plan is put in place for the knowledge management project, it is hugely difficult if there is a culture that stems from senior management of a non-sharing environment.

Ownership Issues

It is important to address issues of knowledge ownership early at the start of the project. Sometimes users feel that they are not receiving anything back from sharing their knowledge. Rewards and recognition need to be given to those who share. Knowledge needs to be viewed as organisational knowledge and not just individual knowledge. Knowledgeable users feel that power will be removed from them and their value will be lowered if they share the knowledge.

Trust

It is hard to hand something over to someone with whom one does not have a relationship. Before implementation of a knowledge management project, the senders and receivers of knowledge need to be recognised. Relationships need to be reviewed to highlight and potential future break down of communication.

Face-to-Face Communication

It is hard to replicate face-to-face communication in terms of facial expressions and verification of an understanding of something. Communication carried out via technology has the obvious advantage of global viewing by its users and knowledge can be seen to grow and evolve. You can also tell who has the knowledge by the most contributions and this knowledge remains in place until the company no longer exists. Face-to-face communication is always going to be faster, it is not going to be interpreted incorrectly, and outcomes and decisions are come to quicker.

More communities need to be encouraged to discuss issues face-to-face, but ensure there is an owner of knowledge that will take ownership of contributing to this knowledge management tool.

6.9.3 Technology

User Interface

From a user interface perspective, users need to know exactly where to navigate to find what they are looking for. There needs to be an overall index of where a user can find something. The same principles of any website apply to an internal Wiki or knowledge management tool. If a user cannot find something on a webpage within a few seconds, they will more than likely not spend any time looking for it. The searching capabilities need to be as good as a Google Search Engine. Minimum effort needs to be invested in trying to find something.

Semantic and Knowledge Wikis

One of the main issues that exists and is still being investigated is the duplication of data on Wikis. Much work carried has been carried out in the line of Semantic Wikis and knowledge Wikis to represent the relationship of all of the knowledge that appears on the Wiki. The lesson to be learned from this is that you need to have the technical staff to install such a system, you need to have a high level knowledgeable worker to overview all of the knowledge contribution, set up the semantic rules and representation. From a user perspective, there needs to be little effort in adding semantic annotation to content.

Review Methods

There also needs to be some technical way to review knowledge that has been contributed and ensure that it achieves a certain level of quality before other users start to use this knowledge as their own. There has to be communication and understanding that knowledge is ever changing and that, at one point, knowledge in fact could be incorrect.

7 EXPERIMENTATION & EVALUATION

7.1 Introduction

Upon evaluation of points discussed in the literature review section, lessons learned from previous implementations of projects with the same requirements, a certain approach was taken based on the knowledge acquired.

This chapter will focus on the approach taken, in terms of gathering the project knowledge transfer requirements, and also discuss how the project was rolled out to the team. Pre and post analysis in terms of requirements and a comparison of the end product with the initial issue was reviewed.
In order to prove if the overall goal of the research was achieved, analysis was carried out. Interviews with members of the project team and users outside of the project team were carried out in order to conclude if knowledge was successfully transferred outside of the project team.

Usability tests in terms of asking non-project members to partake in exercises to do something that required reviewing tacit knowledge that was previously made explicit was carried out. Results during the evaluation and review of other similar implementations formed an overall conclusion in terms of knowledge transfer in a software team. These conclusions are discussed in the conclusion chapter.

Investigation into where the biggest loss in terms of knowledge transfer within the software lifecycle was analysed. Discussion as to whether a tool or process in terms of requirements management could improve on the knowledge transfer at these stages of the software lifecycle was carried out.

7.2 The approach

Like any project, a methodology and plan was set out at the start to highlight the steps to take on how the knowledge management project would be delivered. The requirements analysis phase was not refined to a standalone stage at the start of the project that ended once development began. The approach taken was more of a

prototype approach, where requirements were realised as users used the system more and more. A prototype approach was used in terms of rapid development to get the knowledge management system up and running.

The initial high level requirements were defined through documentation at the start of the project and requirements and design documentation were reviewed by senior management to ensure that opinions were not biased or one-sided. Requirements were carried out by using lessons learned from feedback for previous projects, face to face meetings, questionnaires and feedback after a small prototype at the early stages of the project.

7.2.1 The organisational culture

The analysis was completed on a small project team in Company E-Learn that was working on a customer project. The project was a small project that was starting requirements gathering when this knowledge management implementation started. Users were made aware of the project at the start of the implementation and were aware of the goals of the initiative. In terms of scheduling, the resources within the organisation were never resourced or scheduled to use the system or give feedback on the system. This may be seen as a downfall but if the system was to be seen as effective, it should not take a large enough chunk out of resources time to use the knowledge management tool.

The project team consisted of many team members, ranging from a Project Manager to a solution architect to a developer. All users within the project team had different roles within the team and so needed a varied understanding of project knowledge and needed to pose and ask questions about the project regularly.

The project involved building software for an extremely important client who expected tremendously high standards. Most of the client communication was fed through the Project Manager, with some technical correspondence being carried out from the Solution Architect to the client technical appointee. The client was not aware of any new internal system for the project in question and the new system was purely to improve upon internal client customer knowledge.

7.2.2 Requirements of the Organisation

Company E-Learn had already implemented a company-wide high level Wiki. This was not deemed as a knowledge management tool, but just a Wiki to store documents on an ad hoc basis. The Wiki was in operation for approximately four years but according to employees, the goal of the overall Wiki was never clearly defined. Users who had set up the Wiki originally had moved on and the overall ownership of updates was unclear.

Upon investigation into why certain tasks were being duplicated and why certain requirements were being lost, investigation commenced into where the knowledge transfer problem was. Actual project information was stored in a project folder on a server and communication was carried out via email, face-to-face or instant messaging. A large amount of the documentation was only documented at the start of the project where requirements were defined and signed off by the customer. Communication became looser when the project progressed into development stage. When questions were asked and changes were required in the project, they were committed to over email and never reflected in any formal documentation.

Issues arose when users outside of communication were not included, needed to know an answer and it was not documented in any form.
As time moved on, knowledge became lost and it was not formally documented.
Even if someone was in the office, if something needed to be referred back to, it took time to search through emails or scraps of paper from quick conversations. The knowledge gained from customer interaction at the user acceptance stage and the maintenance phases was rarely fed back into any requirements document which made it hard for new members or members outside the team to understand what was agreed outside of the normal documentation.

It was realised that there needed to be a better way to manage the valuable and diverse knowledge across the team so it could be condensed into a useable format and retrieved for later use by others outside of the team. There needed to be a way that all project knowledge, including documented explicit knowledge and conversational tacit knowledge, was in one place. This knowledge needed to be accessible to those working inside the project team at the time of occurrence and most importantly to those outside of the project team. The scope of the project and research was to provide

a clear project record for Company B-2-B and to ensure that this record was always up to date.

Below are the reasons the initiative of implementing a new way to manage knowledge transfer in the organisation were taken on board.

- The core requirement was that there needed to be a place where project knowledge was presented in such a way that anyone outside the actual project team would be able to understand what happened or is happening on the project.
- There needed to be somewhere to view all project documentation that contained both explicit and tacit knowledge.
- There needed to be somewhere to store all knowledge gained outside documentation, e.g. invaluable informal conversational knowledge.
- There needed to be somewhere to discuss this knowledge.
- There needed somewhere to keep this knowledge up to date.
- There needed to be somewhere for someone to see all knowledge and for one to one knowledge sharing to be reduced.
- There needed to be an environment where someone could upload information without asking permission.
- The users of the system varied and so the user interface and technologies implemented had to be easily comprehendible by all level of users.
- If mistakes were made, there had to be undo features to undo any mistakes that were made.
- Information stored in the knowledge management tool only had to be made public to internal staff members within the companies and not available to other external to the organisation.
- Use of the system in terms of time needed to either match the time it took for searching as it was, or reduce the amount of time. The new knowledge management tool should not have taken any longer to use than any other processes in place.

7.2.3 Management Buy In

Before the actual project could be kicked off, a high level overview of the goals of the project needed to be presented to senior management. A short presentation was put together to firstly highlight the overall scope of the project and try to sell the need for the project.

After an explanation and overview of the need, management had several questions regarding impact on other projects and time users had to dedicate to the project. The concern was that this new project would eat into billable time and have an effect on deadlines and quality of other projects. This is nothing new or something that was not foreseen or planned for. It was important at this stage that the amount of work and time required from the team members was quantified into a schedule. With backing from the services technical director, the project was given the go ahead.

7.2.4 Team Presentation and Brainstorming

In order to start with the implementation of the system, it was important to define the requirements from the project team. An overview of the thoughts of the needs for such a system was given and the expectations of what was needed from each individual were highlighted. Conversations on ideas for the implementation were run through but the initial reactions and opinions were divided. Some saw the need to extract the tacit knowledge from users to gain benefit and share this knowledge with others. Others had the opinion of when they heard the word Wiki for a possible solution they have the opinion of "it would never work”. The team was varied with different levels of seniority, some individuals not feeling comfortable talking about issues in terms of the project with others.

It was felt that an anonymous survey would be of more benefit in attaining truthful feedback in terms of failure and success of current procedures.

A one page document was distributed to explain what knowledge management was, current processes, issues with current processes and ideas about new processes moving forward.

7.2.5 Lessons Learned Documentation and User Feedback

Just before commencement of the knowledge management project, lessons learned workshops were carried out internally and externally with the project team and client from a previously implemented phase for Company B-2-B. There were series of

questionnaires filled out about the implementation about the project and the thoughts of the overall success/failures of the project. From a client perspective, the main issues arose regarding missed requirements and lack of communication or misunderstanding of what they deemed a requirement was. From an internal perspective, the feedback was generally good, except for some missed requirements from the customer and finding of project knowledge and the testing phase was not at the standard it should have been. The testing was not at the required standard as the developers were unsure of the requirements as they felt they were not clearly communicated. This feedback was used as the basis for the need for a new process or a new system to improve the transfer of knowledge from the requirements stage to the development and through to testing.

7.2.6 Surveying the Team

Before documentation of the requirements commenced, a questionnaire was put together so that all team members could voice their opinion. The questionnaire posed both open and closed questions, allowing users to fill in answers on some questions and just rate from agree to strongly disagree on others.

The survey contained 20 questions (see appendix A), and was intended to not be anymore to keep users interested in the questionnaire. The survey was optional as if the users did not fill out the survey, it did not have any impact on their existing roles.

All users did fill out the survey, which highlighted that either they had something to say or they did not want to look bad to senior management.

The opinions on proposed suggestions in terms of a new knowledge management solution varied.

Those who would have to contribute their valuable knowledge highlighted that a tool such as a Wiki would not be a success and no one would use it.

Others highlighted that such a tool would be of great benefit and is definitely needed but could see the barriers; for example getting the knowledgeable people to contribute.

7.2.7 Requirements Definition and Solution Design Document

The knowledge management project was treated as a normal project with the standard lifecycle methodology principles used. Before any work could commence on the knowledge management project, the requirements document had to be accepted by senior management. The requirements document was formed with feedback from the lessons learned, surveys and face-to-face feedback. See appendix B for requirements

documented accepted management. This requirement document was broken down into various headings, which addressed each of the issues raised. A follow on from the requirements document was a design document. The design document aimed to transfer the requirements of the knowledge management system into a proposed workable solution. As the project was more of a prototype project, with feedback being received about the project as it was being built and progressed, the design document did not translate exactly into the solutions that were proposed in the document.

7.2.8 Communication Strategy

In terms of ensuring the system was being moved along and being built in line with what was needed, weekly communication in terms of its progress was delivered. Feedback was delivered via email and then posted to the Project Wiki. Feedback was provided on the system by some users and not by others. Reminders were sent out but some users did not participate in the feedback. Reasons for this are discussed later in the post survey section.

7.2.9 Implementation Approach

As highlighted, earlier knowledge was recorded in documents, communicated on via email and via instant messaging. As email was one of the most used communication media, a Wiki email address was set up where users could email all the information they wanted to upload to the Wiki. Users also had the choice to upload information to the Wiki directly. Some users felt that the need to verify knowledge and ensure that it was checked by someone before uploading.

Microsoft Outlook Rules were created in outlook to catch all information that was tagged with a certain word. For example, rules were set up with key words such as Wiki article, Company B-2-B, missed requirements etc. This allowed for easier sorting of knowledge as it removed all of the unwanted day-to-day mails that were not relevant to the project. This information was then fed into the new knowledge management tool, which was Company's B-2-B Wiki. This task was only completed by one person, i.e. all users were not forced to sort emails into usable Wiki articles.

A large amount of conversations took place regarding Company's B-2-B project over instant messaging. There was investigation carried out into the possibilities of

integrating instant messaging tools such as Skype or MSN into a Wiki to try to overcome the issues of users having to use many tools. The scope of the project was not to investigate the technological capabilities of integrating Skype with a Wiki but to try to sort the knowledge that had been generated and merge it and convert this knowledge with other knowledge into quality wiki articles. Users were given the choice to extract conversations from Skype and post onto the Wiki. This was not taken on board as users liked to keep conversations private. Users did not want to run the risk of a private conversation being transferred to a Wiki and the entire idea was not taken on board.

Face-to-face conversations existed in two formats; formal and informal.
In most cases, informal conversations were not documented or recorded in any way.
Formal documentation of face-to-face proceedings came in the form of minutes of meetings, presentations etc. Documentation that was extracted and placed as pages on the Wiki was requirements and design documents. It was realised that information regarding requirements were easily found through easy browsing on the Wiki.

7.2.10Rollout All Tools

As there were technologies in place for email, instant messaging and document storage, the approach was not to disturb the current day-to-day tasks of the project team but integrate what was there and merge all knowledge together.

It was expressed by Senior Management, that the existing folder structure on the company's server was not to be replaced and only key documentation to be taken from here to be stored and shared somewhere else.
As a result, users were not forced to move to the new Wiki to look for documentation, as they still had the option to move to the old resource.

It was decided to use the company Wiki that was installed some years ago as a basis for the knowledge sharing process. This was decided on because there was no cost associated with the implementation of the tool, there was minimal time on set up compared to setting up a new Wiki and there were no training costs as users were already aware of how to use the Wiki. Investigation was completed into the capabilities around the existing requirements and it was realised that the Wiki more or less suited the needs of the team. Investigation into other tools was completed as seen by figure

Chapter 5. Media Wiki was picked as the tool of chosen for as it matched all of the requirements below and it was already installed in the organisation.

7.2.11Match of requirements against Existing Wiki

Requirement	**MediaWiki**
The knowledge needed to be presented in such a way that anyone outside the actual project team would be able to understand the project knowledge without prior explanation.	This was not a requirement of the Wiki, it was up the knowledge manager and project team to ensure that the knowledge was easily represented in a way that was understandable to all within and outside the project team
There needed to be someone to view all project documentation.	MediaWiki allows for all users view any types of text and images
There needed to be somewhere to store all knowledge gained outside documentation.	MediaWiki allows for users to easy change/update/delete existing documentation on the Wiki.
There needed to be somewhere to discuss this knowledge.	MediaWiki allows users discuss existing pages, using the discussion capability and the watch capability. The watch functionality is like a discussion forum where users are copied on communication that they have added to their watch list.
There needed somewhere to keep this knowledge up to date.	MediaWiki has the capability to update existing pages and record this update in a history. It is easy to see who updated what page on what time and undo the delete if necessary. MediaWiki does not have the capability to sync with server folder and tell when an update has occurred, therefore updating the Wiki with new information.
There needed to be somewhere for someone to overall see all knowledge and	No page within the Wiki will be locked down in terms of viewing content. There

for one to one knowledge sharing to be reduced.	are different administrative levels that allow users to protect pages against editing. Therefore the Wiki is a great place for all users to obtain knowledge and the one to one communication is removed.
There needed to be somewhere that anyone could upload information without asking a person.	There is no verification of an administrator required from the MediaWiki. All users can upload, change and delete content.
The users of the system varied and so the user interface and technologies implemented had to be easily comprehendible by all level of users.	MediaWiki allows for easy editing without any prior web developer experience. The WYSIWYG editing box allows for easily editing in terms of linking and text formatting.
If mistakes were made, there had to be undo features to undo any mistakes that were made.	The undo features of MediaWiki allows for easily undoing updates and changes
Information stored in the knowledge management tool only had to be made public to internal staff members within the companies and not available external to the organisation.	The link to the company Wiki is only available to internal staff and is not available outside the company firewall.
Use of the system in terms of time needed to either match the time it took for searching as it was, or reduce the amount of time. The new knowledge management tool should not have taken any longer to use than any other processes in place.	This would only be seen when analysis was done on use of the MediaWiki functionality

Table 7-1 Company Requirements V's MediaWiki Functionality

7.3 Approach on Upload of existing Information

An analysis in terms of what type of information would be used was carried out and reviewed through the questions posed in the survey.

It was realised that the project knowledge could be sub-divided into 2 main categories, this was project process information and project technical information.

Project process information was information such as project schedules, how to do certain things etc. Project Technical information was not blocks of how but more information from the solution architect on labeling in clear case, the need to knows in Firefox development etc. It was decided that the most important element of the Wiki was the table of content, homepage page or yellow pages of where everything was stored.

The structure of the homepage for the project is seen below

Figure 7-1 Company B-2-B Wiki Homepage

On top of the project process and project technical information, two additional areas or "corners" as they were called were also added. One was a latest news section that allowed easy publication of new news. The other corner was a communication and collaboration corner that allowed the team highlight their thoughts on what they thought was good about the project and add any beneficial pieces of their own personal

information. This was created so that the users realised that this Wiki was not just another information system and that they were included in the creating and expanding of the knowledge management system.

Once all of the project information was updated, users were given the chance to provide feedback on the system and highlight what they would have liked to have done differently. The main feedback from users was that it looked good and they just wanted to start using it. There were comments from a user interface perspective on small mistakes, which highlighted what users are concerned about, i.e. UI issues regarding rather than the functionality. One good point was highlighted about the navigation in that it was frustrating there was no forward and back navigation on the pages and that this would put a user off using the system. These links were added, which then improved the overall usability of the system. All of the feedback and constant changes were made in a test system. Users were allowed play with the system and give feedback in a certain user acceptance period and then were given a date to have all feedback submitted.

The system was then transferred to the live Wiki, where it was to be used as a normal knowledge management tool. The next steps were to extract all of the existing project knowledge using the tools and processes reviewed and highlighted in the literature review stage. This involved setting up certain tools within the email packages, instant messaging and making sure all knowledge was kept up to date. These tools were highlighted to the project team, but as you will see later the project team's view was to only pull information from the Wiki and not push it.

7.4 Approach to ongoing Knowledge

The focus of the case study was to try and create usable Wiki articles from the knowledge gained through communication through the project lifecycle.
The first approach was to create rules in Outlook to catch emails of a certain subject and section this in a separate folder. Once key emails were trapped in a separate area, a tag cloud could be run on the folder to see the most commonly used words

7.4.1 Tagging

Articles were created after the most common words were analysed. Tag cloud software generates the most frequent word in a piece of text. It is important to realise that the most frequent words such as someone's signature or the customer name should be removed. The tag cloud software will remove words like "and", and "the" but will not remove common words only related to a specific organisation.

Please see below for the results of the tag cloud run on the Wiki folder in the outlook inbox.

7.4.2 Tag Cloud results

Figure 7-2 Results of Tag Clouds Process on Email Folder

As a result of this tag cloud procedure, the following Wiki articles were created.

Figure 7-3 View of Wiki articles created

If you analyse the most frequent words in the tag cloud, you will see skills, requirements, download document etc. These all formed an article called “change to phase1 requirements for Company B-2-B”. You will also see the word SCORM, which was a large issue with the customer from the start of the project. The customer required a certain level of tracking with question types in the software for which the organisation did not currently support. There were emails from the client, internally and to the architect regarding this issue. This article highlighted exactly where the issues were and what needed to be done.

Categories allowed for the creating of new articles and linking what the existing information was related to. For example, if a user went to the phase 1 requirements document in the project process page, they would see a direct link to the change to scope for phase 1 article, meaning that there would never be any misunderstanding of the requirements.

7.4.3 Categories in Media Wiki

When all of the knowledge was uploaded onto the Wiki, categories were created within the Wiki. Categories in Wiki are like real life categories, such as in a catalog. By adding a page to a certain category, it groups its content with the associated category.
One of the main concerns regarding the successful implementation of the Wiki was that users needed to be able to search for the information and find the information they were looking for very easily. Without adding any page to a category, sometimes the search would not return the results of all pages containing the word; for example if it was not in the title of the page or added to the category.

This is where the Wiki would fail its users in terms of the capabilities of its searching. Unlike any other search engine, it did not understand the meaning of incorrectly spelt words. It left the categorizing of pages to the user. As users do not like to do any more than they should, most pages would not be categorized if left up to the normal user.

7.4.4 Semantic Wikis

Users were not made to add pages to categories or even link to other pages at all. Many users could have had the potential to upload the same content, several times without knowing that someone else had already uploaded the same thing. Investigation was

carried out into the possibilities of adding semantic extensions to the Media Wiki. Semantic Wikis add descriptions of data to certain WebPages. Semantic Wikis add content about content, thus giving an overall overview of the page. Semantic Wikis create mini abstracts for each page. Semantic Wikis try to create ontology's within the Wiki itself and try to link concepts together. As a result of this more detailed page, searching capabilities become enhanced as more concepts are linked together and better results returned to the user.

Time was spent trying to implement a Semantic Wiki with the existing Wiki but a large amount of time was spent trying to resolve database errors that were presenting themselves upon installation. After much investigation and comparison of the functionality against the requirements of the project, the Semantic Wiki idea was abandoned. It was soon realised that a small amount of users even contributing and using the Wiki, asking users to add context to their contribution would not work. What The auto-completion functionality may have worked in Semantic Wikis as it provides users with the descriptions of the content, based on what they are typing. It may not always be the correct prediction but allows users to see the options of what are already there in terms of categories.

7.4.5 Document Templates

Assumptions were made at the start of the knowledge management project that all users had used the company Wiki, and were familiar with how to contribute new knowledge to the Wiki. This was an incorrect assumption to make. The impression received from some, but not all, users was that they didn't even know how to add a page to the Wiki. There was a fear to contribute to the Wiki in case it was incorrect. Users did not know how to link pages or even care about linking pages if they did know how. In terms of a consistent look and feel, users did not look at other pages to make sure they were linked or created with the same look ad feel.

A document template was created which was placed on each section of the Wiki. This added in automatic links to other pages and added in the navigation for the user. It also created a table of contents and allowed users to easily add images with little know-how. This document template functionality did not encourage any users to contribute to the Wiki as, in the end, no one used the document template.

Also, there is a feature in Media Wiki called reviewer, editor. This allows certain users to upload content and then ask certain knowledgeable users to rate their contributions. This allows other users who are reading the article to see if the content is of a high standard. The technical solution was not implemented for this case study but a more manual approach was taken. In the document template, there was a section where the user had to sign their article, and also a section where they had to assign their article for someone to review. If this was reviewed, the reviewer would have to give the article a rating of 1-3 stars. As no one contributed blocks of knowledge, this functionality was not used.

7.4.6 Communication

A Wiki user manual was put in place that was broken down into sub sections in terms of what the users had to do, their role within the Wiki and the to do's on creating a page, editing a page, adding to a discussion forum etc. This was to ensure that users, who do not have much experience of using a Wiki, could easily add content to it. Even though this document was circulated over email and published on the Wiki, I do not think anyone actually read the document.

7.4.7 Integration of other systems

Something which is very new and is only a beta version in MediaWiki is Wikimindmap. You can see an example of this on www.wikemindmap.org.
It is an extremely clever tool that allows a user to type in a word and simply press enter and a mind map is created which gives an overall concept of the term in the Wiki.
With this functionality, the user could type in a word such as "Java" and all words associated with Java could be displayed in a usable format below. Users would then be able to link their subject matter to the correct concept as they will have seen the overall picture.

Please see below for an example mind map of how the Wiki could work if this functionality was implemented. Unfortunately, to represent the categories of knowledge on the Wiki was not possible as the technology was actually only available for certain Wikis and a project to build mind maps for this Wiki would have taken too long. The scope of the research was to transfer the knowledge of a project onto others and,

although the Wiki mindmap would have assisted users in the search of knowledge, this was not the focus of the research.

Figure 7-4 Wiki Mind Map

7.4.8 Functionality of the Wiki

The intention was also to extract the knowledge discussed on the watch and discussion pages into articles and pages and integrate this knowledge with existing articles.

The issue was that is was easier to walk to someone's desk or to send a Skype instant message rather than to login into the Wiki and add a comment to the discussion page. There were still the issues that there were too many clicks and ease of use was highlighted as an issue. As a result only two people used the discussion pages. The discussions were started but no one responded to the original questions posted and so the issues remained swept under the carpet.

7.5 Overall Evaluation Approach

When all tools and procedures had been put in place, an approach on the analysis of the results had to be defined. It was decided that a questionnaire on the project team opinions would be carried out, interviews on selected team members and a test cycle for users outside the project team (similar to a user acceptance testing). Both a questionnaire and a face-to-face approach were taken as sometimes users feel they can not truly express their thoughts on a subject if they are being asked face-to-face. Carrying out both techniques covered a large amount of ways to analyse the results on use and thoughts of the Wiki.

7.5.1 Questionnaire Analysis

The first approach to obtaining the results of the success or failures of the case study was to carry out a questionnaire on the project team. In order to classify the questions, a clear goal needed to be set out in order to pose the questions in such a way that would extract the thoughts from the individuals without guiding the individuals towards a certain conclusion.

The goal of the questionnaire was to:

- Evaluate if the Knowledge Management project was successful or a failure in terms of knowledge transfer of project knowledge?
- What were the barriers to success?
- What could have been or can be changed to make this knowledge transfer a success?

7.5.2 Questionnaire Goals

After the goals of the survey were clearly defined, decisions regarding the target population had to be set out. As users who did not use the system were being analysed in the UAT Testing section, the focus for the questionnaire was to just review the thoughts from the project team who used the Wiki. In terms of an audience, the goal of the tool was to try to get users to contribute both explicit and tacit knowledge to the Wiki, with a focus on the tacit knowledge as this was something that was not codified or made as a format that could be easily transferred.

7.5.3 Target Audience

The analysis needed to take place from a knowledge contribution and extraction perspective. Investigation into why users did or did not add or take knowledge from the Wiki was analysed. As a large amount of the knowledge in the company came from long-term employees and more senior employees, the focus was on these users to contribute. These users were the solution architects and senior developers on the team. The knowledge that was reused and pulled from the Wiki was aimed at all team members but largely on those who did not have previous project knowledge.

An example of these users were new developers to the team, the support team etc. As the team and organisation were quite small, it was decided to carry out the

questionnaire on all project team members. The project team was made up of 2 solution architects, 2 business analysts, 1 Project Manager, 4 Developers/Testers.
All users answered the questionnaire. Based on the amount of opinions in the questionnaire, people really did have an opinion on the knowledge management tool.

7.5.4 Project Team Questionnaire

It was a challenge to follow the KISS rules, which were Keep It Short and Simple and extract a large amount of opinions and thoughts. (Creative Research Systems) Kiss is an acronym survey specialist term for keeping it short and simple, i.e. not crowding the questionnaire with lots of questions, thus putting the user off answering any of the questions.

A cover page highlighting the goals of the questionnaire, the audience and what will be done with the outcomes of the survey were clearly stated on the introduction page. The ordering of questions was clearly defined and thought out as the users of the survey had to feel there was a beginning, middle and end to the questionnaire. A mix of questions types were placed in the questionnaire which were questions that allowed users to answer a scale between strongly disagree to strongly agree. This did not limit the question to a yes or no when a user did not feel inclined to either side. Some open questions, which allowed users to share opinions, were placed in the survey and a section at the end of the survey also allowed users to write as much as they wished on the overall knowledge management project.

The sections of the survey that were decided on were the Knowledge management project, knowledge management itself and changes for the Wiki. It was difficult to not pose questions in such a way that conveyed the opinions of the questionnaire designer and pushed individuals in a certain direction. Before the survey was distributed to the team, it was carried out by one person to make sure the same question was not posed on several occasions and the questions were not being pushed down a certain direction.

It is important to analyse the feedback in the pre-evaluation survey and the post evaluation survey to ensure the requirements were met and issues from the start of the project were addressed.

7.5.5 Pre Project Survey Results

Current Wiki	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
Is the Wiki a place for technical information, process information or both?	1		7		
	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
The Wiki assists in my current role.	1	2	1	4	
	Strongly Disagree	Disagree	Neutral	Agree	No Answer
How often would you use the current Wiki a week?	4	3			1
Knowledge Management	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
I see benefit in sharing my knowledge with others?			1	1	6
	Email	Face-2-face	Skype	Wiki	Other
How do you currently share knowledge, email, face-2-face, Skype, Wiki, other.	6	7	7	6	2
	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
I find it easy to share existing knowledge, enquire about other existing knowledge, ask questions etc.		2		5	1
	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
I feel rewarded when I learn something new?			2	5	1
	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
I know where to go when I need to find something that has been documented?		6		2	
	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
I know who to ask questions about certain things that I am uncertain about.				7	1
	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
There is duplication of work as I can not access existing information?			3	1	4

Potential Changes to the Wiki	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
A Wiki can become a central area for storage, communication and collaboration?	1			5	2
	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
I would be happy to make discussions regarding technical queries public to other team members	1		1	5	1
	Technical	Process (Release, clear case)	Project (Schedules etc)	Documentation	Other
Please rate the information that would be of most importance to you in order from 1-5, 1 being the most important.	19	22	25	18	14
Can you suggest a name for the new sub Wiki Project, such as share, 1place etc?					

Table 7-2 Pre Project Implementation Questionnaire Results

Below are some comments regarding thoughts on Knowledge Management and a Wiki as a knowledge management tool?

"It is a useful tool which we just do not use often enough, but it is also misperceived. I do not think it can replace one-2-one communication. If everything has to go through a Wiki it's too formal. Balance is needed."

I find that for technical resources the Wiki can be good as people in WBT tend to put stuff up there and t is then easy to find. However, for non-technical stuff I don't think the Wiki is utilized enough.

Obvious good point is the ability for Wiki to allow all, both non-technical and technical to develop a resource library of information.

The only barrier which exists is people's willingness.

I find that at the moment, people rely on Skype and face to face transfer of information rather than a more formal method that can be tracked – i.e. email and Wiki.

It should continue to be used for it's intended purpose, i.e. knowledge sharing but we need more people to contribute to it over and above the handful of people who currently do.

I would like the Wiki to be used to store all information on a project and links to all documents needed for a project. I would like a page to be available which would show the project schedule.

Without wishing to sound cynical, the question implies we have a process, which we do not. We have a WIKI, which is used only to a degree, but there is no process, except for with the Next Gen project.
I think that at the moment a lot of knowledge is stored in people's heads and is not written down anywhere.

Reviewing the results of the questionnaire, a vast amount of knowledge about existing processes, lack of processes and knowledge management tools was realised. Users highlighted that they used the Wiki but the vast majority used this less than 10%. The majority of users choose the opinion that the Wiki should both be a place to store and process technical information. It was highlighted that the majority of information was shared over email and face-to-face communication. Users did feel at ease asking questions but felt that they could not locate certain documentation, as they did not know where to look. Users did want to share knowledge and were happy to share knowledge according to the questionnaire.

There were some strong opinions when it came to what the role of the Wiki was. Users did not think that the Wiki had the capabilities to replace email or instant messaging tools. The Wiki was an area to store and update knowledge that was gained from these tools. The current Wiki that was in place was not organised and put users off using the system. Users felt if there were a clear link to documentation and project information, the Wiki would be of more use. It was learned that the current Wiki might work if there was structure placed, if it was introduced and managed by someone and also if people had the time to use it. One opinion was that currently the Wiki just stored explicit knowledge and no tacit knowledge was being saved.

7.5.6 Post Implementation Survey

Questionnaires were sent out to the project team post implementation of the new project Wiki. 5 out of 9 questionnaires were filled out. The questionnaire was broken down into questions about the new Wiki and questions about the knowledge management. Some questions had already been posed in the pre-evaluation questionnaire, but were asked again to see if anyone's thoughts had changed on knowledge management since the knowledge management tool implementation.

The results of the questionnaire can be seen below.

I think sharing knowledge within the Project team is of benefit to all members?	**Strongly Disagree**	**Disagree**	**Neutral**	**Agree**	**Strongly Agree**
			1	1	3
Knowledge sharing for this customer has increased since the implementation of this Wiki	**Strongly Disagree**	**Disagree**	**Neutral**	**Agree**	**Strongly Agree**
			5		
Rate from 1-5 (5being the lowest) the way that you share knowledge?	**Email**	**Face-2-face**	**Skype**	**Wiki**	**Other**
	12	9	11	6	10
I now know where to go when I need to find something for this project?	**Strongly Disagree**	**Disagree**	**Neutral**	**Agree**	**Strongly Agree**
			1	3	1
There is still duplication of work on the new Wiki	**Strongly Disagree**	**Disagree**	**Neutral**	**Agree**	**Strongly Agree**
			3	2	
I would be happy to share knowledge in the future if had time to do this?					
	Strongly Disagree	**Disagree**	**Neutral**	**Agree**	**Strongly Agree**
		1		2	2
I knew how to use the Wiki before this project					
	Strongly Disagree	**Disagree**	**Neutral**	**Agree**	**Strongly Agree**
			2		3
I know how to add					

content to the Wiki?	**Strongly Disagree**	**Disagree**	**Neutral**	**Agree**	**Strongly Agree**
		2		2	1
The new Wiki has assisted in my current role?					
	Strongly Disagree	**Disagree**	**Neutral**	**Agree**	**Strongly Agree**
		2	2		1
I can find documentation easily on the Wiki?					
	Strongly Disagree	**Disagree**	**Neutral**	**Agree**	**Strongly Agree**
				5	
I can easily discuss new articles, communication and carry out the discussions on the Wiki?					
	Strongly Disagree	**Disagree**	**Neutral**	**Agree**	**Strongly Agree**
			3	1	1
It was easy to navigate through the Wiki?					
	Strongly Disagree	**Disagree**	**Neutral**	**Agree**	**Strongly Agree**
				2	3
The Wiki is the way forward to share knowledge?					
	Strongly Disagree	**Disagree**	**Neutral**	**Agree**	**Strongly Agree**
	1		1	3	

Table 7-3 Post Project Implementation Questionnaire Results

From evaluation of the comments in the surveys the following conclusions were made.

Time

Some users felt that time needed to be dedicated to the using and input into the Wiki

"No time allocated in the schedule for this."

Wiki is not the solution

Some Users felt that Wikis were not the answer. It was felt that Wikis are more suitable for larger project teams and that the gain is not returned from the effort put in.

"I think the whole concept of a project Wiki is just a doubling-up of effort. It's just harder-to-maintain"

Integration with other applications

One user highlighted that it would be nice to integrate existing applications with the Wiki, for ease of use. It is important to have as little starting of applications and logging into applications as possible. There was a Skype-me functionality added which users seemed to like, even though it only removed a small number of clicks.

"New Wiki has a lot more relevant project info in it and it's all in one place which is good. Would be nice if you could get to the actual information directly from the Wiki, e.g. clicking on Bugzilla would bring up (or auto log you into) Bugzilla or clicking on the project schedule would open the latest schedule. I like the "Chat with me" idea which allows you to start a Skype conversation straight away."

Communities of Practice

It was highlighted that maybe a good knowledge management process outside a tool would be coming together as a team to learn from each other. As the person phrased it, "lunch and learns" would be a good way to learn from each other.

"I always find "Lunch and Learns" a good way to share project knowledge with team members and even more so with non-team members, e.g. providing sales people an overview of what we are doing on a project for a particular customer."

Integration with the existing Project Lifecycle

One user made a valid point that the lessons learned documentation should be fed into the Wiki. From a technical perspective, actions should be decided on such as documenting up how to code something etc and then highlighted on the Wiki.

"There is no step in the project lifecycle for this. We do sometimes have a lessons learnt meeting, but this is purely for the benefit of the PM… from a technical perspective we do not update any KM system."

A Wiki does not manage group collaboration

It was highlighted that a Wiki does not manage group collaboration effectively and it is more of a one to one communication visible to all.

"No, or not the way we use it anyhow. It is a 1-way system of communication, listing people's opinion of information. Does not allow for discussion… I would have thought a bulletin board would be better, where people can post questions might would be better?"

Document Management

There still appeared to be duplication in project folders and on the Wiki. There appears to be the possible issue of changing a document in one location and it becoming out of date in another location.

"If you have to change the solution design you are going to have to update it here too"

Interview with Project Team Members

A face-to-face interview was carried out with 2 project team members selected at random. A series of 12 questions were posed (please see appendix E). The interviews were taped and the interviewer did not engage in any feedback when questions were answered in order to remain subjective to the issues. Below are the conclusions made from the interview with the team members.

Time

Here it was also noted that there was no time scheduled for project to share knowledge. It was not seen as something that was part of a project schedule or plan.

Areas for Improvement

It was noted that the searching capabilities were not considered of a high enough standard. Improvements in this area were suggested. It was also noted that the technical writing within the team would have to be reviewed by others as some people would not be of a sufficient level to contribute. It was highlighted that the Wiki would be used if it was used for an area of reuse in terms of technical reuse. It was felt that the Wiki was not being used as a Wiki and that there needed to be more linking of topics together to avoid duplication. Message board functionality or a white board was suggested as a way to communicate.

Culture

It was felt that there was no-one in the organisation pushing this project from a management perspective. Users needed to be told that they had to use it, rather than leaving it optional. The project Wiki needed to be integrated into the organisation at an organisational level and not just at a project level. Management needed to agree to the time invested in knowledge management as time spent at present was minimal.

UAT Testing

As the focus of the research was to prove if both tacit and explicit project knowledge could be transferred to a person outside the project team, tests were set up with users outside of the project team. 3 users were selected to carry out these exercises.

Users were employees of the organisation and one user was not associated with the organisation. There were three exercises carried out which involved the following.

1. 12 questions which ranged from high level knowledge to deeper knowledge such as Company B-2-B functionality.

2. Upon review of the questions, the approach was altered in order to provide the user with some background knowledge. This involved a user reading both knowledge that was once tacit and that had been formalised by a project team member.

3. After the result of the second round of user testing, the exercise was altered based on the amount of knowledge the user could gain from the Wiki.

Round 1 Testing Results

One tester was from a development background and the other was from a customer support background, and the third person was from outside of the organisation but had an IT background. 12 questions were asked in a 15-20 minute period and were recording for referencing and analysis. The support individual got 3 out of the 12 questions incorrect and the developer got 4 out of the 12 questions incorrect and the person outside of the organisation got 4 out of 12 questions incorrect.

Questions	Tester 1	Tester 2	Tester 3
Question 1	Correct	Correct	Correct
Question 2	Correct	Correct	Correct
Question 3	Incorrect	Correct	Incorrect
Question 4	Incorrect	Incorrect	Incorrect
Question 5	Correct	Correct	Correct
Question 6	Correct	Correct	Correct
Question 7	Correct	Correct	Correct
Question 8	Correct	Correct	Correct
Question 9	Incorrect	Correct	Incorrect
Question 10	Correct	Incorrect	Incorrect
Question 11	Correct	Correct	Correct
Question 12	Correct	Correct	Correct

Table 7-4UAT Testing Results Round 1

Round 1 Testing Conclusions

Training

It was realised that all users did not know how to create a page in a Wiki, edit a page or discuss on a page. There was a user manual on the main page that showed a user exactly how to create a page, create a discussion forum but both users did not open this document.

Spell Checker

When searching for key words, both users typed in words incorrectly. The Wiki did not understand these incorrectly spelt words and so returned no results to the user.

Users were frustrated when they could not find this information and as a result got some questions incorrect.

Layout

When users had to find information that was not on the main page, they really struggled with linking concepts with main categories. For example, one question was “what type and version of database does Company B-2-B have?” There was a main menu of database information on the screen, which led users straight to the type and

version. Bother users got this answer correct. There was a question asking a user about a certain piece of functionality that was under the documentation main heading. Both users got this answer incorrect, as they could not link customisation documented in a document with the documentation functionality.

Regression Testing Phase 1

It was realised that asking users to find knowledge about a certain subject in a short space of time was difficult. Users were not aware of the layout and were not given time to familiarise themselves with the system. Questions that were posed were out of context to the environment that they were used to so it was like asking a developer to fix a car; they were not familiar with the environment.

The approach was altered in terms of setting the scene for which tacit knowledge existed. A knowledgeable person within the team created a Wiki article that asked a user to create what was called a learning plan. See Appendix H.
The user was then given an exercise on the Wiki and asked to set up this plan. The user was observed during the period of testing.

The user was asked to create courses and assign skills to certain courses. The user was then asked to assign these courses to learning plans. The user was then asked to assign a learning plan to a user, log in as that user and ensure that they could see the learning plan. This knowledge was tacit knowledge that had been documented by a person on the project team. This knowledge did not exist in a document anywhere, only details on how the system was built was highlighted in design documents.

The results showed that all three users could create a learning plan and assign this plan to a user. It was realised that, when users were not familiar with the system, they followed steps provided on the Wiki very closely. Certain steps were missing on purpose to see if users noticed or what their reaction would be. As suspected, users did not fill out sections that they were not informed to and were not aware that they had to. An example of this is where a step was left out to make certain courses mandatory or optional on a learning plan. All users did not enquire about this option on the learning plan and just left this blank.

Regression Testing Phase 2

A final test was carried out which involved fixing the mistakes that had been placed in the exercise. The 3 users managed to complete the learning plan successfully without any issues. A 4th user was asked to complete the fully formatted exercise and also completed this exercise successfully without any issues. This proved that the 3 users, who completed the test 3 times, did not eventually create the learning plan. This experiment proved that by repetition or learning from incorrect behaviour knowledge is built up over time which allows users do a task and allowing the task improve over time.

Discussion of UAT Findings

The results of the findings show that knowledge can be transferred from a tacit format to an explicit format and be understood by someone outside of the project team. As the knowledge was improved over time through additions of missing pieces and providing context in terms of the overall picture of where the knowledge sat, users' activities in terms of taking explicit knowledge that was once tacit and converting this into a practical tacit form again improved.

The overall exercise proved that it is possible to take a small piece of tacit knowledge, expand on this over time with feedback through the socialisation process. Lessons learned from this exercise show us that you have to put this tacit knowledge in context of where it sits in the overall project. Users struggled to comprehend the knowledge where it was not presented to them with some meaning. As the knowledge grew and as assumptions were made clear, the knowledge transfer process improved over time. This also proves that knowledge needs to be moved around the spiral of knowledge several times before it is at a high level. This exercise showed us that there were certain steps missing that the person within the project team omitted on purpose and also by mistake. When this knowledge is taken out of context, it is hard for someone else to comprehend as they do not have the requisite background knowledge.

7.6 Conclusion

This chapter sought to review if it was possible to transfer project knowledge of both an explicit and a tacit format to users outside of a project team. A review of how the project was approached was discussed, which led onto why the knowledge management tool was rolled out in the way that it was. Analysis of users inside the project team before and after the project was carried out to compare the knowledge management and knowledge transfer. Results from the questionnaires showed that all users saw benefit from sharing knowledge and the majority of users wanted to share knowledge. Results in terms of adding benefit to the users' roles after the Wiki was put in place were not so successful. Results showed that only two users posted to the Wiki and these posts were never responded to or followed up by anyone else.

It was shown that large amounts of knowledge, both explicit and tacit, were contributed to the Wiki but this was only by one person, who was the knowledge manager. This resulted in knowledge potentially being posted in a biased way.

As all of the project documentation was placed on the wiki, the initial project requirements were available to those outside of the project team. As a small amount of team members use the wiki, no updates to documentation in terms of changing requirements was carried out. The project manager tried to continually update the wiki with new information, but no other team members took part in this exercise. As seen in Chapter 3, the highest loss of customer requirements takes place in the user acceptance and maintenance phases when customers are using the system in real life scenarios. As none of this knowledge was captured within the company wiki, undocumented requirements and knowledge will remain with the individuals who discussed it directly with the customer.

As a small amount of users made contributions to the Wiki, user testing in terms of knowledge transfer had to be conducted not in a real life scenario. Through repetitive refining of knowledge over time and asking the tester to carry out a task that involved using the tacit knowledge codified by others, it was proven that it is possible to transfer tacit knowledge via a Wiki to people not involved with the project team. Although it had proven that this knowledge transfer is possible, it also highlighted the fact that the knowledge needs to be put in context for the receiver of the knowledge. Knowledge is

of little value to someone if they cannot fully comprehend the context for which it was intended.

As highlighted previously, knowledge transfer involves a pusher and a puller of knowledge. This research has highlighted that if tacit knowledge is made explicit and codified, it is possible to transmit to another person. The research has also proven that it is impossible to carry out knowledge transfer without the holder of the knowledge contributing the knowledge to start with.

8 CONCLUSION

8.1 Introduction

Upon review of what literature suggests, reviewing what other organisations and individuals have done and analysing tried and tested solutions within Company E-learn, one can now attempt to make a conclusion on the subject of knowledge transfer in a software project. This chapter attempts to prove if suggestions or hypotheses in literature are correct in their thinking by reviewing the practical implementations of what was suggested. Results of the experiment in Company E-Learn are reviewed and a conclusion in terms of knowledge transfer is reached. Future work in terms of ideas that could not be implemented due to the scope of the project is highlighted in the future work section. An overall conclusion on knowledge management in a software project is concluded based on literature reviewed and the experiences in Company E-Learn.

8.2 Research Definition & Research Overview

The goal of the research sought to overcome an issue of knowledge transfer of a tacit nature in a software project to users within the project team but focusing on users outside of the project team that did not have any experience with the project. Within the overall scope of this question, the following questions were asked in order to review if it was possible to transfer knowledge to users outside of a project and what were the barriers to this transfer.

What are the goals and aims of knowledge transfer within a software team?

The goals of the knowledge transfer process was to ensure that knowledge that existed in both an explicit and tacit format was in a central place for users inside and outside of the project team. The aim of successful transfer is to ensure that all users are up to date with information. Users, who are in the project team, need a central view of what has happened on a project, what is happening and what future plans are. If knowledge transfer does not take place successfully, valuable pieces of project knowledge will be missing, which will mean team members make decisions on incorrect knowledge,

incorrect software is built based on incorrect knowledge and the customer is frustrated as they are not getting what they want (Cortada & Woods 2000).

This results in time redeveloping projects and cost to the company, which cannot be passed onto the client. If the users outside of the project team are operating on out of date knowledge they will build incorrect customisations for this client if they were not aware of all of the knowledge. From a support perspective, if a user is trying to resolve an issue and they do not have access to all of the information, they again will make incorrect decisions based on out of date knowledge. The goals of the knowledge transfer is to ensure that everyone inside and outside of the organisation is operating on up to date information and has access to both the tacit and explicit knowledge that existed for that customer.

Why tacit knowledge is difficult to transfer to an explicit format?

Tacit knowledge is knowledge that is not explicit and that does not exist in an explicit format for others to reuse. As highlighted b Polanyi (1967) it is a combination of experiences and past implementations. The goal of this research was to investigate if this knowledge could be translated into a format that could be reused by others. In the literature review section, knowledge is translated to others in a spiral motion. Knowledge exists in a tacit form and is translated to others in what is called the socialisation process. Knowledge is made explicit when it is put in format that is visible to others and can be comprehended by others at what is called the externalisation process. This knowledge is combined with existing documented knowledge at the combination process and then translated to others by the practical implementation of what was once explicit in what was called the internalisation process (Nonaka & Takeuchi 1995).

From the experiment in Company B-2-B, the transfer of tacit knowledge into an explicit format back to a tacit format again was achievable.

There were two main challenges that existed in the transfer of the tacit knowledge which were context of knowledge in which the knowledge was being transferred and the contributors and receivers of this knowledge.

It was shown from the experiment on Company E-learn that as knowledge was refined and updated over time with deeper knowledge about the subject; users who were trying to understand this knowledge were made feel at ease as the knowledge improved. It was shown that at the first round of testing, where users were asked to find blocks of knowledge, users could only find and understand high-level knowledge as they did not understand the context of deeper knowledge.

Secondly as the test to prove the transfer of knowledge was simulated by asking a project team to document a piece of knowledge, no real life examples or proof of knowledge transfer occurred. This highlighted another point that knowledge transfer is about givers and receivers. As only one person contributed to the Wiki to share knowledge from the project team, it was a challenge to ensure that all tacit and explicit knowledge was being updated. In order to successfully create a record of a project, all users who worked on the team have to contribute; it cannot be just some selected members.

Can a Wiki tool manage the tacit knowledge of a project?

As highlighted from the literature review and case study section, before tacit knowledge can be transferred to a tool, it has to be firstly made explicit and then classified so that others can easily find this knowledge in the future (Sanchez 2000). As highlighted by Polanyi (1967) tacit knowledge exists in many formats; it exists in face to face conversations, email and instant messaging software. Before making this knowledge accessible to users, it needs to be translated into a format that makes sense to others who are going to use it. There are technologies to classify knowledge depending on the knowledge. It was shown that tag clouds classify words depending on their occurrence; Wikis classify knowledge depending on their category. A Wiki tool can manage knowledge transfer if the knowledge is classified and if the knowledge can be classified in a way that does not ask a large amount of time from the user. Knowledge changes all of the time and as a result the quality of Wiki articles may vary depending on when it was created and who is contributing to it. In the User testing for Company B-2-B, articles improved over time and users found it easier to retrieve knowledge once context and substance was added to it. Depending on the commitment of the team, requirements could be discussed and agreed without being reflected on the company wiki.

The Wiki tools needs to be able to

- allow for group collaboration and sharing
- to manage changeable tacit knowledge
- An area to discuss knowledge and for these discussions to be directly reflected back into the Wiki article.

The Wiki tool for Company B-2-B did prove to cater for all of these requirements. The experiments did show that there were certain barriers to the knowledge transfer process, which were foreseen at the start of the project.

What are the main barriers to knowledge management project and knowledge transfer?

As highlighted in the evaluation section there were certain barriers to the knowledge transfer process. These barriers were not really technical barriers but barriers against the knowledge management project in general. It could be noted that, no matter the scope of the knowledge management project, these barriers would have existed.

Barriers to the knowledge transfer and knowledge management project were

- Users not wanting to share knowledge
- Users not knowing they had to share knowledge
- Users not being shown how to share knowledge
- Users not being made share knowledge
- Knowledge Manager not selling the project
- Knowledge Manager not scheduling enough time to share knowledge

It was proven that tacit knowledge could be translated into an explicit format and made available to others outside of the project team. It was proven that a Wiki tool is a sufficient tool to manage knowledge in a small software organisation. Although these goals were achieved, there existed the barriers that exist on all knowledge management projects, which are people, process and technology barriers.

8.3 Contributions to the Body of Knowledge

Knowledge Management

Knowledge management consists of the management of knowledge of both an explicit and tacit form (Nonaka & Takeuchi 1995). This knowledge takes shape as it moves through the spiral of knowledge and grows and morphs into new knowledge. Knowledge moves from tacit to explicit knowledge back to tacit knowledge from where the process of re-use starts again until this knowledge can be brought no further or is never ending. Within a software development project, it is crucial to make sure that this knowledge is passed on correctly as it moves through the cycle, allowing requirements to be clarified, developed and delivered as per discussed at the start of the lifecycle.

It is important that this knowledge is not lost in translation or lost in a crack between two phases. There exists a real challenge with passing on tacit knowledge in particular. As proven from the experiment in Company B-2-B, users do not pass on tacit knowledge easily for many reasons. Some users feel that whatever is not documented and remains in their head is their own personal knowledge. Time also seems to have a large amount to do with the issues of reusing tacit knowledge. Time is not allocated to users to share, and so it becomes something that is at the bottom of a “to do” list. As we know, time is money so if something needs to be cut and a project needs to be delivered, a knowledge management project is not going to be put in front of a billable customer facing project.

As Nonaka (1997) highlighted, knowledge management is about vision, strategy, structure, system and staff. Vision is making a picture of what you want to see at want to see on a long terms from a knowledge management system, and truly knowing your goals. Knowing your goals is something that is often taken for granted and is often a failure of many software projects.

A strategy is not just a project plan; it is a communication strategy, a selling strategy and way to coerce and push people into using it. Users use many tools in their working life, which they complain about but they just do it because this is what they are told. Maybe if they were told that a Wiki was the only form of communication and knowledge management, then the Wiki would have been used. Structure is something that is not just specific to a knowledge management project but to all software projects. Users need structure from a point of view that they need to know what they are doing

or will become frustrated and leave. The system is the link between the knowledge and the users and it needs to be built in such as way that it does not take away or add time to their daily tasks but assists users.

Lastly, but most importantly are the people. People need to cooperate in terms of pushing knowledge, pulling knowledge, management need to support it through the project and not just at the start. People need to manage a knowledge management project from a project management and a technical point of view. A project manager needs to have weekly meetings to share knowledge, share issues about the knowledge management tool and constantly plan for change.

As highlighted in Company E-learn, management buy in was provided, but a strategy in terms of a continual internal marketing and forcing of system usage was never planned for. As a result, people felt no need to use it, and only browsed it on a few occasions but never uploaded any knowledge to it. A lot had to do with the organisational culture in terms of learning. Learning was carried out in a fire fighting approach, obtaining snippets of knowledge from brief conversations. No communities were set up or just devoted to learning. If the project was to start again with all of the lessons learned, a plan in terms of selling and expectations of users would be put in place. Weekly Wiki meetings, where users were expected to add at least one thing a week and discuss at a meeting should have been enforced. Rewards and recognition should have been given for contribution as users perform more when they know they are being monitored.

Knowledge Transferral in the software development lifecycle

As highlighted in the spiral of knowledge, there is an input and output into every stage. For example, in the socialisation stage when two people come together, the input is tacit knowledge and the output is tacit knowledge. If there is resistance or non conformity on each side, the knowledge stays stagnant. This is the same at every stage and is not just isolated to a human perspective. If the receiver of the knowledge at the externalisation stage is a tool but is not well equipped enough to absorb the knowledge, the transfer is stalled at this stage also.

Every stage of the spiral needs to be reviewed and roles and responsibilities need to be defined in terms of the pusher and the puller. Davenport and Prusak (1998) highlighted that there are buyers and sellers of knowledge. Unlike in a real world situation, sometimes the sellers are reluctant to sell and the buyers are extremely keen.

The main point to make about knowledge transfer in a software lifecycle is that it is a continuous circle and not a process with a start, middle and an end. At some point in the entire cycle, knowledge is going to be at a more mature state than before. There needs to be real management of when knowledge is used; at a young age or mature age. There has to be means of highlighting levels of accuracy, such as reviewing functionality of signatures on a page when someone has reviewed it. There is still an issue of Chinese whispers in terms of users talking to each other, and nothing being translated into a media for review from knowledgeable users. These knowledge users need to be clearly identified in a knowledge management project, as a project manager is not the right person to validate knowledge.

The process of knowledge transfer also needs to be managed throughout the entire software lifecycle. As the solution is delivered to the customer, key pieces of knowledge regarding changes to requirements are lost due to the focus on documentation only occurring at the earlier stages of the project. If a wiki is to be used as a central storage area for changing requirements, all individuals have to partake in usage of the tool, not only at the requirements analysis phase of the project but for the remainder of the project lifecycle, until the product becomes stagnant and does not change any more.

This research has highlighted that the main barriers to knowledge transfer are the converting of the tacit knowledge into an explicit format that is understandable by users who were not part of the original project team. Due to our human make up, people find it hard to step out of a scenario and include all of the knowledge that they were not even aware that they had or the knowledge that someone else would need. As it was highlighted in the case study, “the ace Wiki” tried to overcome the issues of users do not always know what they know by asking a series of questions on the user interface when using were inputting content onto the Wiki.

The second issues that were highlighted in Company E-learn and in McMahon's research were regarding the sharing environment within the organisation.
It is difficult to succeed in a knowledge transfer project if the person who has to share their knowledge is not willing to share. The sharing culture and the environment, which the knowledge management project exists in, needs to be addressed and users, need to be monitored and pushed into sharing the project knowledge.

8.4 Experimentation, Evaluation and Limitation

At the externalisation stage of the spiral of knowledge, knowledge is transferred into an explicit form from a tacit form (Nonaka & Takeuchi 1995). In order for this knowledge to be made reusable it has to be classified into a format that can be easily referenced and merged with other knowledge. Within a software project, there exists a real challenge as there is not one communication media for sharing. As highlighted from the experiment in Company E-learn, various technologies were tried and tested in terms of classification. As most of the knowledge existed in a conversational nature, it was a real challenge to classify this in terms of creating an introduction, a body and a conclusion. Classification was also carried out by the knowledge manager, which resulted in knowledge only being filtered from one source, which leads to the potential of losses in other areas. After knowledge is classified in different media, it needs to be brought together in one place and, in a sense, merged again in the explicit to explicit way. Privacy became an issue as only the knowledge manager made conversations public.

The goal of the Wiki implemented was to then translate the tacit knowledge back into tacit knowledge by reuse by others. The experiment involved classification of knowledge from different media such as email and instant messaging. This knowledge was contributed to a Wiki over a 3-month period and users were asked to also share and add value during this time. Review of the use of the Wiki was completed by using questionnaires, interviews and user testing. Users from inside and outside of the project team were interviewed to ensure that both the pushers and pullers of knowledge were questioned.

It was realised that only the knowledge manager was pushing knowledge onto the Wiki. All knowledge regarding the project was being taken from email etc. and was

published to the Wiki. This resulted in a very one-sided approach to the Wiki. Users were asked on many occasions to add content and review the Wiki. It was a challenge to prove and investigate the issue of knowledge transfer within the project as those with the knowledge were not willing to or did not have the time to share.

A test was carried out which involved asking several users outside of the project team to carry out several exercises to try to obtain tacit knowledge from someone else. Tests proved that over a series of updates to the Wiki article to add more background information, knowledge transfer was achieved to users outside of the project team.

Based on feedback from the team, review of testing and research undertaken, it proved that knowledge transfer is successful only if the pusher of the knowledge is willing to commit and the knowledge is in a format that is understandable by the receiver.

As the wiki tool was not used to capture changing requirements, it was hard to prove and test if this non-use had any repercussions. In order to test this scenario, changes to the system would have to have taken place without anyone capturing this knowledge. A new project member would then have had to change this functionality in the system without knowing what the original requirement was. As no development changes could take place within the scope of the experiment, this test could not be carried out or the theory could be proven. The knowledge transfer process was proven by the article test scenario, set up to prove that the knowledge transfer process improves over time with more tacit knowledge and contextualisation.

8.5 Future Work & Research

As the two main issues identified as barriers to the successful implementation of a knowledge management project were contextual knowledge management and user contribution, it would be beneficial to carry out future work in this area.

Semantic and Knowledge Wikis

As highlighted with the experiment for Company B-2-B, knowledge improved over time as more and more knowledge was added to the Wiki. As highlighted, knowledge Wikis ask a series of questions from the user. This issue that normal Wikis try to overcome is the issue of linking different concepts of categories of knowledge together

to form an overall picture of a subject. Within standard Wiki technology, there is the option to add what are called internal links to other pages. There is also the option to add these pages to categories in order to provide for easier retrieval at a later stage. Semantic Wikis ask users to add annotation to blocks of text in order to create ontology's of information in the background.

End users of most Wiki technology are made up of team of users. Users that vary in technical acumen such as Project Managers, testers developers etc
If these levels of users are asked to add annotation or do anything that involves stressful activities they are not sure about, they will not use the system.
As highlighted in the ACE Wiki example, the user interface needs to be in tune with what the user is trying to achieve. If more time was assigned to the research for Company B-2-B, investigation into the capabilities of knowledge Wikis should be investigated. A knowledge Wiki could remove the need for the steps that was highlighted for Company B-2-B, where additional knowledge had to be added to the Wiki on several occasions as the user did not know what the knowledge meant as it was out of context of their situation.

Agile Methodologies combined with Knowledge Management Initiative

As highlighted with Company B-2-B, the second biggest barrier to knowledge sharing was with users contributing their knowledge. From questionnaires and interviews, users felt that knowledge management projects take a large amount of time.
Normally, knowledge and lessons learned are only reviewed at the end of a project and most of the tacit knowledge that is gained at the time when it was at its best quality is now lost or of a lesser quality. As highlighted earlier in Company B-2-B, the Waterfall Methodology was used as the methodology of choice for software development.

Agile methodologies such as SCRUM and XP allow for a faster pace quick implementation of a software project. If a methodology such as SCRUM were taken on board, daily meetings with clients and developers would mean that knowledge is passed straight to the source of where the functionality is going to be developed. If the Wiki that managed the knowledge was the communication media between the client and the developer, then all of the project knowledge would be shared in this central

location. There would be a challenge to keep this knowledge sorted and classified and also to ensure that there was a clear line between internal and external communication.

If an agile methodology and a Wiki for project communication with the client were used as a project record, it would vastly improve the quality of tacit knowledge that is being recorded on a project. Users outside of the project team would then be able to see what exactly occurred, without it passing through several other media or users and thus losing its quality.

If this wiki was used as the medium of communication between the project team and the client, any discussions regarding requirements that were normally lost via tacit conversations could now be recorded for later reference for new project team members. This would be that the wiki tool is not just used as the tool for project record and communication at the earlier requirements and design stage, but as a tool for project record until the end of the life of the system.

8.6 Conclusion

Most knowledge Management Projects is rarely core to one's business; they are seen as nice to have and are often cut from a Company's budget. One needs to represent at the start of the project, in terms of costs, how this project will be of benefit to the team and company. If this cannot be proven at the end of the implementation, your project is at risk of discontinuing. If internal satisfaction and happiness is going to be the measure of success, it needs to be clearly stated at the start of the project.

All knowledge management projects require the following tasks to be completed:

- Obtain management buy in
- Create a list of requirements in terms of sharing, collaboration etc
- Create a mind map of the organisations knowledge including both explicit and tacit knowledge.
- Review each subcategory of knowledge and assign business owners.
- At this point, highlight the amount of time on a high level on how long this project will take.
- Categorise each sub section of the company and create an index of knowledge.
- Come together and compare all data to make sure there are no duplicates.

- At this point, review the tools that can manage all your requirements.
- Keep a business owner for each block of knowledge; keep a KM marketing person and KM project or, if you're lucky, someone who can do both.
- Ensure that the tools are effective enough to cater for fast searching and easy collaboration with the possibilities to integrate with existing tools.
- Never walk away from it or assume it will manage itself.

Every project needs to have a methodology for which it is going to be carried out but concluding a new approach on knowledge management projects in general was not within the scope of this research. It was seen in the Internet-Speed Software methodology, where agile principles were employed, that risks regarding requirements loss and redevelopment were reduced because of a closer communication between the client and the customer. The scope of this research was narrower in that the analysis was just on the transfer of tacit knowledge to a place that could be used for users outside of a project team. As research and experimentation proved, knowledge transfer was possible if users were pushed into sharing. Background work in terms of culture of sharing, buy in and creating an environment for sharing had to be addressed before users were asked to share. In Company E-Learn, users were asked to share and they were not told to share. If the knowledge management project became a core part of their daily role, then the knowledge management project would have been more successful.

Knowledge changes continually over time and varies in its change, depending on the methodology applied to the project. It was seen that a wiki could capture any type of project knowledge but it needed to be used at every stage of the project and capture all knowledge, including knowledge normally lost at the testing and implementation phase. Wikis have the potential to become out of date very quickly if they are not used so they have to be integrated into the project team as the place to store and communicate all project knowledge. The main barriers to this being used as the place for project record would be the people using the system.

If users were shown how to share and possibly the scheduling of mini workshops in terms of users coming together to highlight key valuable knowledge on a project the actual knowledge sharing process could have been improved. In Company E-Learn,

when users contribute tacit knowledge to those who were not in context of the original situation, they missed parts of the core knowledge. Without any technical assistance such as the introduction of knowledge Wikis, the only way to overcome this issue was to use the system over time and allow for the collaborative refining of knowledge from the Wiki users. Knowledge only improves over time.

BIBLIOGRAPHY

Abrahamsson, P, Salo., O, Ronkainen, Warsta, J., (2002), *Agile software development methods*, 2002, p.21, Available from http://www.vtt.fi/inf/pdf/publications/2002/P478.pdf, [Accessed in June 2009]

Aery, M. & Chakravarthy, S., (2005), *eMailSift: Email Classification Based on Structure and Content*, Proceedings of the Fifth IEEE International Conference on Data Mining, 2005, p.8, Available from http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1565657 [Accessed in April 2009]

Avram G., (2005) Knowledge Work Practices in Global Software Development, Martins, B, Remenyi, D, *8th European Conference on Knowledge Management 2007*, Academic Conferences Limited, 2007

Basili, V,R., & Calderia, G., (1995), *Improving software Quality by reusing knowledge and experience*, Available from http://www.cs.umd.edu/users/basili/publications/journals/J55.pdf [Accessed May 2009]

Baskerville, R., Ramesh, B., (2003), *Is Internet-Speed Software Development Different*?
Available from http://www.it-c.dk/courses/VIPS/E2004/Baskerville.pdf [Accessed in June 2009

Batista Dias, M, G, Anquetil, N., de Oliveira, K, M, (2003) Organizing the Knowledge Used in Software Maintenance, *Journal of Universal Computer Science*, 2003, 9, (7), p. 641-658, Available from http://www.jucs.org/jucs_9_7/organizing_the_knowledge_used/Dias_M_G_B.pdf, [Accessed June 2009]

Beck, K., (1999) *Embracing change with extreme programming*, Volume 32(10), p. 70-77, Available from http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=796139 [Accessed in June 2009]

Benjamin, F., 1748 Available on http://www.angelfire.com/biz3/eserve/ayt.html [Accessed in May 2009]

Boehm, B., (2006) A View of 20th and 21st Century Software Engineering, International Conference on Software Engineering, 2006

Boehm, B., Hansen, W, J., (2001), *The Spiral Model as a Tool for Evolutionary Acquisition*, Available from http://www.seecs.edu.pk/~naeem/SE/software%20p.%20models%20readings/presentation%201.pdf [Accessed in June 2009]

Bontis, N., (2001), Assessing Knowledge Assets: A review of the models used to measure intellectual capital, *International Journal of Management Reviews*, 3, (1), p. 41-60.

Bower, M., Woo, K., Roberts, M., Watters, P., (2006), Wiki Pedagogy – A Tale of Two Wikis, Available from http://web.science.mq.edu.au/~mbower/research/papers/ITHET06_wiki_pedagogy_mbower.pdf [Accessed May 2009]

Brännström, M., Mårtenson, C., 2006), *Enhancing situational awareness by exploiting Wiki technology*, Available from http://www.foi.se/infofusion/bilder/CIMI_2006_S3_2.pdf [Accessed in May 2009]

Buffa, M, (n.d) *Intranet Wikis, Available*, from http://www-sop.inria.fr/acacia/WORKSHOPS/IntraWebs2006/Buffa_Intrawebs2006.pdf, [Accessed in May 2009]

Bush, V., (1945), Available from http://www.answers.com/topic/hypertext [Accessed in April 2009]

Catrambone R., Holyoak, K, J. (1989) Overcoming Contextual Limitations on Problem-Solving Transfer, *Journal of Experimental Psychology:Learning, Memory, and Cognition*, 15, (6), p. 1147-1156
Available from
http://reasoninglab.psych.ucla.edu/KH%20pdfs/Catrambone_Holyoak1989.pdf
[Accessed April, 2009]

Cloutier, R, J, (2004), Migrating from a Waterfall Systems Engineering Approach to an Object Oriented Approach – Lessons Learned [Accessed in May 2009]

Charette, R, N., (n.d) *Why Software Fails*, Available from http://lrv.fri.uni-lj.si/~franc/COURSES/VP/Fails.pdf [Accessed in May 2009]

Cockburn, Alister, (2000), *The Agile Software Development Series*, Available from http://www.imamu.edu.sa/Scientific_selections/Documents/IT/AgileSwDevDraft3.pdf, [Accessed June 2009]

Collison C, Parcel, G., (2007) Learning to Fly*: Practical Knowledge Management from Leading and Learning Organizations*, Wiley-India

Cortada, J, W., Woods, J, A., (2000), *The Knowledge Management Yearbook*, Butterworth-Heinemann

Coughlan J., Macredie (2002), R.D Effective Communication in Requirements Elicitation: *A Comparison of Methodologies*, 7, (2), p. 47-60
Available from http://www.springerlink.com/content/y1pmvw1nu2gtb92g/ [Accessed April 2009]

Davenport, T. H., & Prusak, L. (1998), *Working knowledge: how organisations manage what they know*, Boston, MA: Harvard Business

Decker, B, Ras, E, Rech, J, Jaubert, P Rieth, (2007) *Wiki-Based Stakeholder, Participation in Requirements Engineering*, Available from

http://portal.computer.org/portal/cms_docs_software/software/content/promo/s2028_07.pdf [Accessed in May 2009]

Enrico R., (2005), Executive Chairman, DreamWorks Animation SKG and former Chairman and Chief Executive Officer, PepsiCo, Inc. Available on http://www.12manage.com/quotes_ki.html [Accessed April 2009]

Foley, M. (2002), instant messaging reference in an academic library: a case study, College & Research Libraries, Vol. 63 No.1, pp.36-45

Fowler, M., Highsmith, J., (2001), *The Agile Manifesto*,
Available from
http://hristov.com/andrey/fhtstuttgart/The_Agile_Manifesto_SDMagazine.pdf
[Accessed in May 2009]

Glass, Robert, L, (2001) Agile *v's Traditional, Make Love Not War*,
Available from http://www.itu.dk/courses/VIPS/E2005/Glass.pdf [Accessed June 2009]

Giles, J., (2005), *Internet encyclopedias go head to head*,
Available on http://www.nature.com/nature/journal/v438/n7070/extref/438900a-s1.doc
[Accessed April 2005]

Girard, J, P., (2006,) Journal of Knowledge Management, *Where is the knowledge we have lost in managers?* 10, (6), p. 22-38

Goodman J. L., (2003), Satellite Navigation Systems: Policy, *Commercial and Technical Interaction*, The Netherlands, M. Rycroft, Kluwer Academic Publishers

Gotel, Orlena, C. Z C. W, Finkelstein, Anthony (n.d) *An Analysis of the Requirements Traceability Problem*, Available from
http://csis.pace.edu/~ogotel/research/GOTEL93%20An%20Analysis%20of%20the%20Requirements%20Traceability%20Problem.pdf
[Accessed in June 2009]

Gruger, T, (1992), Available from http://www-ksl.stanford.edu/kst/what-is-an-ontology.html [Accessed in April 2009]

Grünbacher, P,Briggs, R,O, (2001), *Surfacing Tacit Knowledge in Requirements Negotiation:*
Experiences using EasyWinWin, Proceedings of the 34th Hawaii International Conference on System Sciences

Gupta, S., (1997), *Decimal Classification System*: A Bibliography for the Period 1876-1994, M.D. Publications Pvt.

Hansen, M. (1999), *The search-transfer problem: The role of weak ties in sharing knowledge across organisation subunits. Administrative Science Quarterly,* 44 (1) p. 82-111. Available from http://www.jstor.org/pss/30047959 [Accessed April 2009]

Hayes, C, Avesani, P., (2007), *Using Tags and Clustering to Identify Topic Relevant Blogs*, International Conference on Weblogs and Social Media (ICWSM) Available from http://www.icwsm.org/papers/2--Hayes-Avesani.pdf [Accessed May 2009]

Higgins, G., (2008) *Knowledge Swap Shop - Implementing a small but scaleable knowledge management initiative in a software development team,* Available from http://arrow.dit.ie/cgi/viewcontent.cgi?article=1010&context=scschcomdis [Accessed May 2009]

Hunter, E,. (2002), "Do we still need classification?" *Future of Classification*, Available from http://www.sims.monash.edu.au/subjects/ims2603/resources/week6/6.4.pdf [Accessed April 2009

Ichijo, K, Nonaka I, V. K George (2000), *Enabling Knowledge Creation, How to Unlock the Mystery of Tacit Knowledge and Release the Power of Innovation,* Oxford University Press US

Ivanovic, E, (2005), *Dialogue Act Tagging for Instant Messaging Chat Sessions, Department of Computer Science and Software Engineering,*
Available from http://www.cs.mu.oz.au/~edwardi/papers/da_modelling.pdf [Accessed April 2009]

Jones, C, *Applied Software Measurement*, (2001) New York, McGraw-Hill

Kasser, J, 2002, the Systems Engineering, Test & Evaluation Conference, *Cataract Methodology for Systems and Software Acquisition*, Sydney, Australia

Kassim, A,M, M, R. A. Rahman, Y, N. Cheah (2008), *A Knowledge-Based E-mail System Using Semantic Categorization and Rating Mechanisms, proceedings of world academy of science, engineering and technology*, 35, Available from http://www.waset.org/pwaset/v35/v35-19.pdf [Accessed in April 2009]

Khalifa, M, Verner, J, M., (2000), Engineering Management, *Drivers for software development method usage* 47(3), pp360-369

Kittur, A, Pendelton, B,A., (2007), *Power of Few v's Wisdom of a Crowd, Wikipedia and the Rise of the Bourgeoisie,* Available from http://www.parc.com/research/publications/files/5904.pdf [Accessed May 2009]

Kuhn, T, (n.d), AceWiki: A Natural and Expressive Semantic Wiki, Available from http://attempto.ifi.uzh.ch/site/pubs/papers/kuhn08aceWiki.pdf, [Accessed in May 2009]

Leuf, B, Cunningham, W., (2001), *The Wiki way: Quick collaboration on the web.* Boston: Addison Wesley, Available from http://66.102.1.104/scholar?hl=en&lr=&q=cache:BQyFjxX3quoJ:www.gbv.de/du/services/toc/bs/392217767+The+Wiki+way:+Quick+collaboration+on+the+web
[Accessed April 2009]

Locke, C. Et al. (2000). *The Cluetrain Manifesto: The End of Business As Usual*, Cambridge, MA: Perseus.

Malhotra, Y, (2000), Knowledge management and virtual organizations, *United States of America, Idea Group Publishing*

McMahon, C., (2008), A *synthesised knowledge mapping framework to embed a KM strategy using topic maps and Wikis: the Tao of Wiki*, Available from http://arrow.dit.ie/scschcomdis/3/ [Accessed in May 2009]

Montero' Y.H, Solana, V, H, (2006), *Improving Tag-Clouds as Visual Information Retrieval Interfaces*, Available on http://www.nosolousabilidad.com/hassan/improving_tagclouds.pdf [Accessed in May 2009]

Neef, D, Siesfeld, G,A, Cefola, J,(1998) The economic impact of knowledge, *Butterworth-Heinemann*

Nelson, K, M, Cooprider, J, G, (2001) *The Relationship Of Software System Flexibility To Software System And Team Performance*, available on http://dvda.cqu.edu.au/david-jones/Reading/ICIS_2001/01CRP03.pdf [Accessed May 2009]

Nonaka I, (2005), *Knowledge management: critical perspectives on business and management*, Routledge, Abigdon, Oxon

Nonaka, I, (1997) *Organizational Knowledge Creation, Available on http://www.knowledge-nurture.com/downloads/NONAKA.pdf [Accessed in April 2009]*

Nonaka, I. & Takeuchi, H., (1995) The Knowledge-Creating Company. Oxford University Press US *New York*,

Oren, E, Del, R, (n.d) *Annotation and Navigation in Semantic Wikis*
Available on http://www.eyaloren.org/pubs/semwiki2006.pdf [Accessed in May 2009]

Oren, E, (n,d) *SemperWiki: a semantic personal Wiki,* Available on http://www.eyaloren.org/pubs/semdesk2005.pdf [Accessed May 2009]

Oberg, R, Probasco, L, Ericsson, M, (n.d) *Applying Requirements Management with Use Cases,* Available from http://www.sos-portals.com/Interesting%20Articles/Manage%20Requirements/RUP_AppReqMgmt.pdf [Accessed June 2009]

Paetsch, F, Eberlein, A, Maurer, F, (2003) *Requirements Engineering and Agile Software Development,* Available from http://www2.enel.ucalgary.ca/People/eberlein/publications/WETICE_03.pdf Accessed [June 2009]

Pelfrey, L, (2002) The Explosive growth of the Internet, Available from http://atlas.kennesaw.edu/~lpelfrey/Internet%20Growth.doc [Accessed April 2009]

Polanyi, M., (1967), *The Tacit Dimension,* Anchor Books, New York.

Pyra, K, (2003) *Knowledge Translation:A Review of the Literature Prepared for the Nova Scotia Health Research Foundation, Available from* http://209.85.129.132/search?q=cache:3jEPGqGUAzEJ:www.nshrf.ca/Download.aspx%3Fserverfn%3D./files/drm/f3668d2d-af7c-4271-bf43-705d4d886464.pdf%26downloadfn%3Dktweb.pdf%26contenttype%3D+Knowledge+transfer+is+seen+as+a+process+(not+a+one-time+act).&cd=1&hl=en&ct=clnk&gl=ie [Accessed April 2009]

Journal

Raman, M, (2006) *Wiki Technology as a "Free" Collaborative Tool within an organisation setting,* Information Systems Management, Available from http://www.ism-journal.com/ITToday/wiki.pdf, [Accessed in June 2009]

Reinhart J, G, (2005) Wiki and the Wiki Way, Beyond Knowledge Management Solution, *Information Systems Research Center*, Available on

http://www.uhisrc.com/FTB/Wiki/Wiki_way_brief%5B1%5D-Jennifer%2005.pdf [Accessed in May 2009]

Ruhe, G, *Learning Software Organizations*, (2000), pp 2- 3 German, Springer-Verlag Available from http://66.102.1.104/scholar?hl=en&lr=&q=cache:PCyBO79kvagJ:citeseerx.ist.psu.edu/viewdoc/download%3Fdoi%3D10.1.1.24.1576%26rep%3Drep1%26type%3Dpdf+gunter+ruhe+Learning+Software+Organizations [Accessed May 2009]

Sanchez, R, (2000), *"Tacit Knowledge" versus "Explicit Knowledge" Approaches to Knowledge Management Practice*, Available on http://www.knowledgeboard.com/download/3512/Tacit-vs-Explicit.pdf, [Accessed April 2009]

Schaffert, S., (2004) Gruber,A, Westenthaler ,R ,*A Semantic Wiki For Collaborative Knowledge Formation*, Available from http://www.newmedia.at/research/gfx/SemWikiForCollKnowForm_20060120.pdf [Accessed in May 2009]

Schwaber,k, Beedle, M, (2002), *Agile Software Development with SCRUM*, Upper Saddle River, NJ, Prentice Hall.

Smith, D, E, (2000), *Knowledge, groupware, and the Internet*, Butterworth-Heinemann

Stvilia, B, Twidale, MB, Smith LC, Gasser L, (2005), *Assessing Information Quality Of A Community-Based Encyclopedia*, p. 442-454,
Available on http://www.annenbergonlinecommunities.com/node/445
[Accessed in April 2009]

http://www.surveysystem.com/sdesign.htm

Szulanski, G, (1999) *The Process of Knowledge Transfer: a Diachronic Analysis of Stickiness*, Available from

http://bbs.cenet.org.cn/uploadImages/200372613125848731.pdf, Accessed in May 2009]

Talisayon (2008) Available on http://thegiraffe.wordpress.com/2008/11/26/converting-tacit-to-explicit-knowledge-and-vice-versa/ [Accessed in May 2009]

Tan, D., Wong, S., (2008), Eureka: web based tool for collaborative problem solving and project work documentation of knowledge creation and discovery Available from http://smartech.gatech.edu/dspace/handle/1853/24418 [Accessed in May 2009]

The Standish Group, the Standish Report, 1995

Thoeny, P., (2002), http://tWiki.org/cgi-bin/view/Codev/TWikiTestimonials [Accessed in May 2009]

Turk, W, (2008), *Common Sense Project Management*, Milwaukee, Quality Press

Wagner, C., (2004), Communications of the Association for Information Systems, *Wiki: a technology for conversational knowledge management and group collaboration*, 13, p. 265-289, Available from http://researchWiki.peacocktech.com/images/9/96/Wiki_A_Technology_for_Conversational_Knowledge_Management_and_Group_Collaboration.pdf, [Accessed April, 2009]

Whitehead, A, N, (1938) *Lecture one in Modes of Thought*, New York, Macmillan

Young, R., (2005) *CEO*/CKO Knowledge Associates International, Available from http://www.knowledge-management-online.com/Definition-of-Knowledge-Management.html [Accessed April 2009]

Quan-Haase, A., Cothrel, J.,Wellman, B, (2005). Instant messaging for collaboration: A case study of a high-tech firm. *Journal of Computer-Mediated Communication*, 10,

*(*4), p. 13. Available from http://jcmc.indiana.edu/vol10/issue4/quan-haase.html [Accessed in May 2009]

Zeng, H, Alhossaini, M, A, Ding,L, Fikes, R, and McGuinness, D,L (n,d)Computing Trust from Revision History, Available from ftp://ksl.stanford.edu/pub/KSL_Reports/KSL-06-12.pdf [Accessed May 2009]

APPENDIX A

Senior Management Presentation

Improving the reuse of project knowledge within the organisation

Thesis idea

Knowledge Management in the Workplace

- I am completing a thesis in Computing and Knowledge Management starting Jan 2009.
- I have to try to think of an idea where I can perform an experiment in my workplace that will examine how a new KM technique or new KM process can improve work productivity.
- For example other students will introduce a Wiki into the workplace or a social software tool.

The Challenge

- We seem to loose information from the requirements workshop to the solution design to the detailed design to the developer to the support contact.
- Why ?

My Thesis Idea

- We don't capture all points raised during the requirements meeting.
- The solution design does not incorporate all points from the detailed design.
- The detailed design does not incorporate all points from the detailed design.
- The developer was not on site and does not understand the requirement.
- We can not think of every scenario to ask until we move into the design.
- We need to spread the knowledge to all resources not just the developers.

What can we do to Change? 1/4

- Templates for requirements analysis? All customer have different requirements but every customer needs to be asked certain questions? Would it be possible to record walkthroughs at the requirements to refer back to?
- At the end of the requirements we should document in some way a way to bring all requirements together and test all functionality together?

What can we do to Change? 2/4

- Review our Approach to the requirements design template. Does this need to be reformatted to consider all scenarios?
- We need to get a better feeling of high priorities such as Firefox testing. If we carry out a matrix of low to high priorities.
- If we have test case review workshop before the solution design is signed off to incorporate all new scenarios.

What can we do to Change? 3/4

- We should have a handover from the requirements person to all of the developers.
- Could we break the development tasks down per high priority and deliver in phases instead.
- Can we add a Wiki Site for every customer with the top 20 questions such as roles spreadsheet, languages, browser sizes and type.

What can we do to Change? 4/4

- From a support point of view, can we have decision trees on if this happens, then the result is this.
- Would it be possible to ask the developer to present the functionality to the solution architect to prove that they understand the idea?
- As a part of the close down and lessons learning, can we review the bugs that were not bugs but misses in the requirements and get feedback from the developers on why? i.e. they have to tell me why it went wrong

Why?

- Improve Client Relationship
- Reduce the amount of rework on requirements before and after delivery.
- Spread the knowledge from the Business Analyst to all team Members in the organisation.

Who with?

- To goal of the experiment is to introduce new processes to increase the customer satisfaction.
- We do not want to frustrate the customer so we need to introduce small changes.

APPENDIX B

Knowledge Transfer Requirements Document

Dissertation Requirements Document

Knowledge transferral within the Project Team throughout the s/w development lifecycle

Purpose of Document

The purpose of this document is to highlight the changes that need to be made to the current processes within the software development lifecycle. The requirements document will detail the requirements for the new process and technologies that will be put in place.

The requirements will be analysed and documented per phase of the software development lifecycle. The design document will follow on from the requirements document and will look at the overall requirements of the project to harness knowledge and build on knowledge within the software development lifecycle. As an output of this document a review of the requirements will be carried out and which will result in an overall solution being proposes.

Goal of the new case study

The company for which the case study is going to be completed is a small to medium sized software company producing an off the shelf piece of software with the majority of customers carrying out a large amount of customisations depending on their business needs. Clients vary from Oil refineries to car manufactures to Alcohol companies. A large amount of clients have similar customisations but most projects are started from scratch starting with the requirements gathering phase within the project lifecycle. Within the project lifecycle a different person is responsible for the tasks set out at every phase. It is the responsibility of the Project Manager to ensure that the correct process and procedures are followed. The challenge is to ensure that the knowledge gathered at the initial stage of the project is kept at the state it was captured and that all project resources are enriched with this information. The challenge is to capture the knowledge that is both documented and non explicit in a way that can be used to all resources within the company to also in way that can be used moving forward with the knowledge becoming obsolete.

Current Process

Requirements Analysis

Onsite requirements Capture Process

The current requirements gathering technique for the majority of project involves on site interaction with the client. As a result of this interaction, a business analyst documents the requirements in a document that is reviewed by the client.
This business analyst takes notes on site.
The challenge at this point is ensuring that all information discussed on site is captured in such a way that can be referenced again when the workshop is over.

Requirements Document

After documentation of all of the requirements, a requirements document is delivered to the client where the client will review the proposed requirements.
After several reviews the client will sign off on the requirements document.
This requirements document is mainly text based with some expansion of possible alternatives.
The challenge at this point is to ensure that all scenario's are were fully thought through by the business analyst and verified by the client and other internal project members.

Design

Solution Design Document

At the design phase the requirements document is used as a basis to write the solution design document. The solution design document proposes the solution to the requirements and suggests a way that the overall project solution will be delivered.
Depending on the client, the level of technical detail is varied depending on its audience. Some clients do not understand this document and only can come to terms with their requirements and expect the provider to deliver the solution without little consultation. Other clients require and review every piece of technical detail within the system.

The challenge here is creating more of a solution proposal than the requirements document and drawing a clear line between both documents. The challenge here is not loosing any information from the requirements and just to build on this information. The challenge is also in portraying this information in a way that is understandable by the client and covers all scenarios in the functionality.

Detailed Design Document

The detailed design document is an internal document that is a technical explanation of how to deliver the solution.

The challenge is to ensure that the detailed captured at the requirements and solution design is transferred into a technical explanation of how to do something.

There should be no need for a developer to look for information that in a previous document as the information is not detailed in a subsequent document. The challenge here is to ensure that the knowledge gained through interaction with the clients is clearly reflected at the every point as passed on internally to the project team.

Development

At the development stage depending on the project, the developer will read the detailed design document and start development.

The challenge here is to ensure that a full handover has taken place and that there have been no room for misinterpretation.

There is also the challenge to ensure that the proposed design is the correct design, there needs to be validation in place.

There also needs to be communication within the team of the overall functionality as several people could be working on the same thing.

More preparation and think through of the solution and discussion needs to take place.

It is important to ensure that the requirements document and the solution design document have been translated into the detailed the design document and no important parts have been lost.

It is important to ensure that the way something was developed can be clearly used in other projects and reduce work in the future.

The challenge is to highlight the functionality in a central way so everyone knows what was developed, when and by whom.

Testing

The current testing process involves in some cases using test cases to test the functionality that has been developed. This will involve the business analyst writing up test scenarios for the tester to run through when the development is complete.

The aim is to have different person test the functionality that has been developed. As a result of running the test cases bugs will be found with the system. The test cases are to test the functional scenarios. When the bugs are logged they should be reviewed by the developer who carried out the original work.

The challenge is to ensure that the test cases cover all scenarios. There needs to be an additional step of system testing where certain test plans are written to test issues such as validation of fields or scenarios that would break the system in system testing and not in functional testing. Performance testing is also something that needs to be focused on as a task. Most UAT phases concentrate on testing with a small amount of users and a small amount of scenarios. The results of this need to be reflected in an internal central area so other projects can be prepared and put in extra validation for performance testing. Fixes that have been used in other projects need to be easily found when the same issue occurs.

Implementation

Currently the implementation process involves the testing of the system on a UAT environment where after sign off a release will take place to what is called a production environment. With the current project the release process was streamlines and a new system was put in place which reduced the amount of hours hugely.

The challenge is to ensure that this release is reused is in other projects and the process is clear and any tips or lessons learned are documented in a central area. The challenge is to ensure that the majority of users can carry out this process and make sure that this responsibly is spread out over several people.

Details about the actual releases themselves do not have to be added to the central area as the source control system.

Maintenance

Currently when a system goes live it is supported by a support help desk when any issues arise.

These issues are reviewed by a support person who has not worked on the project but has received a handover of the project in terms of documentation and a handover process.

The handover documentation needs to clearly reflect the current state of the system.

At the moment a large amount of the issues are resolved by the original developers who developed the work. It should be clear by the documentation that was completed how something was done.
Any person who has not worked on the project should be able to resolve issues on any project from reviewing the clearly commented code and documentation.

Any lessons learned from the project should be documented in a central area place, used and actually learnt from.

The challenge is to get people to not instantly ask questions of the person who did the original development.
The goal is to try and get people to learn instead of asking questions and not taking in the knowledge that they could gain.
The challenge is saving the correct information and for people to not use out of date information.

Next Steps
The solution design document will detail how the challenges in each phase of the project lifecycle will be approached in terms of a new technical solution to manage knowledge and also the new process changes that will be put in place.

APPENDIX C

Knowledge Transfer Design Document

High Level Design

Dissertation

Knowledge transferral and loss within the software development lifecycle

Purpose of the document

The purpose of this document is to take the requirements captured in the requirements document and questionnaires filled out by the project team and translate these into a viable solution to problems raised.
The dissertation is focused around knowledge loss and knowledge transferral processes within the software development lifecycle.

The details regarding proposed solutions are regarded as possible solutions to several problems as opposed to a definitive answers to every issue raised.
The scope of the case study is merely to address and propose solutions to issues from the requirements to User Acceptance Testing as any work completed post this time due to the particular timeline can not be documented.

The proposed solution will be broken down into 2 parts proposed technical solution to deal with the issues and proposed process changes.
Upon reflection of the issues raised at the requirements stage, 3 main categories of knowledge management issues were realised, these were communication and collaboration, searching and document management.
The approach to try and deal with these current issues is to try and use an existing Wiki and build on the infrastructure that already exists.

Other factors and means of improvement will also be discussed within the document as a way to address and marry all suggestions together.

Current Wiki

The current Wiki is powered by MediaWiki.

The main focus of this Wiki is to store technical information, points to note etc.

There are existing sections for Project Information, People Information etc.

The responsibility of updating and managing the knowledge on the Wiki is not clearly defined with some project members contributing more than others. Communication of an update is done by the individual updating the Wiki sending an ad hoc email. There is little classification of this information within the Wiki.

The project that will be used as an example does not yet exist within the Wiki.

Searching

All phases

At the moment if someone is looking for a piece of information they will either search via the normal means of searching using the search functionality in Windows, searching via folders, or via the search mechanism in Outlook. Alternatively, an individual could ask someone on the project team.

Proposed Solution

Key of Information

In order to reduce the amount of time searching, it would be beneficial firstly to remove all the folders and have all information in one place.

Without introducing any technology, if there is a information key (similar to a table of contents) to tell someone exactly where they need to go, the amount of changes in terms of technology do not need to be large.

Key Word Searching

There is already the functionality of key word search within the current MediaWiki.

Additional investigation is required to be carried out into the additional attributes that can be added to this search page.

Move Search Feature

Also, the search functionality is quite invisible to the user and if its features and role was explained, then it might become more useful to the user.

The search feature should be made visible to the user at all times.

Technical Investigation

Page titles in MediaWiki are composed of two parts: an optional namespace name, and the remainder of the title. For example, this page has the title `Help:Namespace`, so it is in the *Help namespace*. A title without a colon, for example Goings-on, is in the *main namespace*.

http://meta.Wikimedia.org/Wiki/Help:Namespace

Below details how you can create extra name spaces.

http://www.mediaWiki.org/Wiki/Manual:Using_custom_namespaces

Further Research

1) Is it possible to change the left navigation pane?
2) Yes, see UI section.
3) Is it possible to move the search button?
4) Is it possible to add additional features to the Wiki for Searching?
   http://www.mediaWiki.org/Wiki/Manual:Using_custom_namespaces
5) If it possible to add Google Type Searching to the Page
   http://www.mediaWiki.org/Wiki/Extension:Google_Custom_Search_Engine

Documentation Storage and versioning

All phases

Documentation of requirements, solution design and detailed design normally takes place with a series of iterations as review meetings normal produce a series of changes that require documentation.

A clear definition of revision history needs to be seen in terms of updates, i.e. a main table of contents with all documentation that shows what is covered in each document and what has changed in terms of updates.

There needs to be one place for all documentation for all phases.

Proposed Solution

One place

All templates should be outlined in this “one place”.

Results of documentation such as questionnaires, test documents, minutes of meetings (internal and external) should be stored in this one place.

All other means of storage should be removed.

Information should be stored in a logical way and users should know where to go when looking for something rather than searching through lots of folders as they would have done previously. This will also tie into the searching section.

Online collaboration

There should be also a means of online collaboration in terms of discussing a document and updating as you go.

Communication and Collaboration

Analysis Stage

Documentation of the requirements will be improved by closer communication with the client and also a more story telling of the requirements rather than text based document.

Review Meetings were put in place to ensure that all on site requirements were captured, by adding additional reviews it ensured that there was a back up of information in case the requirements person lost any valid piece of information.

Design Stage

It is important at this stage to ensure that the design documents clearly reflect and translate what was captured on site and that it is turned into a workable solution for the client.

Development stage

It is important at the development stage to be provided an overview of coding standards, review lessons learned from past issues and communicated effectively so that it does not happen again.

Testing

It is important at the testing stage that the solution is well testing using the documented test cases provided by the business analyst. It is important the test cases reflect the solution after the solution architect has proposed the solution in the detailed design document. The test cases should clearly cover off all scenarios in order to be considered full tested.

Release

In order for the release phase to be successful, careful labelling of files as per the highlighted procedure should be followed. Clear communication of the procedure needs to be carried out at the start and throughout the project by the solution architect.

Proposed suggestions

As all items above can be carried out via clear communication we just need to find a way that all team members can communicate with each other so that all of the knowledge that is gained from these conversations can be reused all of the way through the lifecycle. There needs to be means to ask questions and get answers quickly and ensure that any updates are clearly reflected in documentation. (Please refer to documentation section).

Current means of communication is via meetings, face to face conversations, Skype, telephone and email.

Issues with this are that the majority of conversations are 1-2-1 conversations, only being made public if someone chooses to be made public.

Possible solutions could be:

IM within MediaWIKI

Integration of Skype with MediaWiki

As sometimes it is quite frustrating to copy people on email and keep a track of an email of questions and answers, a possible solution would be to have an option to integrate. The results of a question and answer section could be posed to the Wiki in a sub section depending on the category of the question.

Yes,http://Wiki.syncleus.com/index.php/SyncleusWiki:Extensions#Skype_Online_status

.

Message Boards Forums

An output of the conversations could feed into a message board in some way.

Review conversations regarding documentation could be discussed on a message board and in this way clear suggestions could be documented.

Coding standards and release process could be posted on a bulletin board.
The possibility to post emails to bulletin boards could also be investigated as users may find it frustrating logging into another system.

Yes, http://www.mediaWiki.org/Wiki/Extension:MediaWiki_Bulletin_Board

RSS Feeds

Can I integrate RSS feeds into Wiki?
http://www.packtpub.com/article/integrating-twitter-and-youtube-with-mediaWiki

UI Changes

Changing the left navigation bar might increase the use of the system if it was more informative.

http://www.mediaWiki.org/Wiki/Manual:Navigation_bar

Wiki Design

There will be 2 main sections

- Project Technical Information
- Project Process Information

UI Principles – It is important that it is recognised as a Company B2B project but not use the logo due to issues with Copyright.

Research User Preferences Link under MediaWiki

Ownership

It is important to make people feel involved and provide ownership. It may be an idea to add a developer's corner and post things that order people could use. It is important to make sure that it is categorised in such a way that it will be found again.

Roles and Responsibilities in terms of who creates updates and deletes documentation should be clearly highlighted.

This responsibility will be clearly detailed in the project section of the Wiki.

APPENDIX D

Pre-Implementation Questionnaire

Questionnaire
Sharing customer knowledge within the Project Team throughout the Project Lifecycle

Goal of the Questionnaire
The goal of this questionnaire is to understand the project team's opinions on the current knowledge sharing process and tools available within the Project Team. *"**Knowledge** is defined in the Oxford English Dictionary as expertise, and skills acquired by a person through experience or education; the theoretical or practical understanding of a subject"*

Audience
This questionnaire is aimed at all project team members including implementation consultants, business analysts, solution architects and Project Managers.

How to use the Questionnaire
The questionnaire contains a list of questions regarding the current knowledge sharing techniques and tool, your opinion on knowledge management and what you would like to see for future changes. If there are any opinions that you would like to express, please put these in the additional comments section.

Results of the Questionnaire
All of the results of the questionnaire will be reviewed and the results will feed into the requirements of the new Wiki system and knowledge management initiative.

CUSTOMER KNOWLEDGE SHARING QUESTIONNAIRE					
PLEASE STATE YOUR CURRENT ROLE IN THE PROJECT TEAM			I.C ☐ S.A ☐ P.M ☐ B.A ☐		
ALL QUESTIONS CONTAINED IN THIS QUESTIONNAIRE ARE KEPT STRICTLY CONFIDENTIAL.					
Current Wiki					
	Is the Wiki a place for technical information, process information or both?		☐ Technical	☐ Process	☐ Both
	The Wiki assists in my current role.	☐ Strongly Disagree	☐ Disagree	☐ Neutral	☐ Agree	☐ Strongly Agree
	How often would you use the current Wiki a week?		☐ 0-10%,	☐ 10-30%,	☐ 30%-60%	☐ 60%-100%
	Do you think there are any issues/good points about the Wiki that you could highlight? Please discuss?				
Knowledge Management	I see benefit in sharing my knowledge with others?	☐ Strongly Disagree	☐ Disagree	☐ Neutral	☐ Agree	☐ Strongly Agree
	How do you currently share knowledge, email, face-2-face, Skype, wiki, other.	Email ☐	Face-2-face ☐	Skype ☐	☐ Wiki	☐ Other
	I find it easy to share existing knowledge, enquire about other existing knowledge, ask questions etc.	☐ Strongly Disagree	☐ Disagree	☐ Neutral	☐ Agree	☐ Strongly Agree
	Please discuss details on ways to current aids or barriers to knowledge sharing?				
	I feel rewarded when I learn something new	☐ Strongly Disagree	☐ Disagree	☐ Neutral	☐ Agree	☐ Strongly Agree
	I know where to go when I need to find something that has been documented?	☐ Strongly Disagree	☐ Disagree	☐ Neutral	☐ Agree	☐ Strongly Agree
	I know who to ask questions about certain things that I am uncertain about	☐ Strongly Disagree	☐ Disagree	☐ Neutral	☐ Agree	☐ Strongly Agree
	There is duplication of work as I can not access existing information?	☐ Strongly Disagree	☐ Disagree	☐Neutral	☐ Agree	☐ Strongly Agree
Potential Changes to the Wiki	What would you like to see the Wiki used for? Please discuss					
	A Wiki can become a central area for storage, communication and collaboration?	☐ Strongly Disagree	☐ Disagree	Neutral	☐ Agree	☐ Strongly Agree
	I would be happy to make discussions regarding technical queries public to other team members	☐ Strongly Disagree	☐ Disagree	Neutral	☐ Agree	☐ Strongly Agree
	Please rate the information that would be of most importance to you in order from 1-5, 1 being the most important.	Technical	Process (Release, clearcase)	Project (Schedules etc)	Documentation	Other
	Can you suggest a name for the new sub Wiki Project, such as share, 1place etc?					
Additional Comments Please include any additional comments on the current KM Process. Please detail what you what like to see changed in the future						

APPENDIX E

Post Implementation Questionnaire

Post Implementation Questionnaire
Sharing customer knowledge within the Project Team throughout the Project Lifecycle via a customer Wiki

Goal of the Questionnaire
The goal of this questionnaire is to understand the project team's opinions on the current knowledge sharing process and tools post implementation of the KM tool. *"**Knowledge** is defined in the Oxford English Dictionary as expertise, and skills acquired by a person through experience or education; the theoretical or practical understanding of a subject"*

Audience
This questionnaire is aimed at all project team members including implementation consultants, business analysts, solution architects and Project Managers.

How to use the Questionnaire
The questionnaire contains a list of questions regarding the implementation of the customer wiki and knowledge management processes. If there are any opinions that you would like to express, please put these in the additional comments section.

Results of the Questionnaire
All of the results of the questionnaire will be fed into the results of research being carried out for the thesis in question. Results and opinions in the questionnaire, will form the overall conclusion of the successes and failures of the knowledge management initiative.

	PLEASE STATE YOUR CURRENT ROLE IN THE PROJECT TEAM	I.C ☐ S.A ☐ P.M ☐ B.A ☐				
	ALL QUESTIONS CONTAINED IN THIS QUESTIONNAIRE ARE KEPT STRICTLY CONFIDENTIAL.					
	I think sharing knowledge within the Project team is of benefit to all members?	☐ Strongly Disagree	☐ Disagree	☐ Neutral	☐ Agree	☐ Strongly Agree
	Knowledge sharing for this customer has increased since the implementation of this Wiki	☐ Strongly Disagree	☐ Disagree	☐ Neutral	☐ Agree	☐ Strongly Agree
	Rate from 1-5 (5being the lowest) that you share knowledge?	Email ☐	Face-2-face ☐	Skype ☐	☐ Wiki	☐ Other
Knowledge Management	Please discuss details on current aids or barriers to knowledge sharing pre and post implementation of the wiki?					
	I now know where to go when I need to find something for this project?	☐ Strongly Disagree	☐ Disagree	☐ Neutral	☐ Agree	☐ Strongly Agree
	There is still duplication of work on the new wiki	☐ Strongly Disagree	☐ Disagree	☐ Neutral	☐ Agree	☐ Strongly Agree
	Do you think the cost on investing time in sharing knowledge is worth the investment?					

	I would be happy to share knowledge in the future if had time to do this?	☐ Strongly Disagree	☐ Disagree	Neutral	☐ Agree	☐ Strongly Agree
	What other tools or processes would you suggest for sharing project knowledge?					
	If you were the KM Project Manager what would you have done differently?					
Additional Comments Please add your comments on the overall goals of the project, the results of the project and future changes you would like to see?						
Wiki Tool	I knew how to use the Wiki before this project	☐ Strongly Disagree	☐ Disagree	Neutral	☐ Agree	☐ Strongly Agree
	I know how to add content to the Wiki?	☐ Strongly Disagree	☐ Disagree	Neutral	☐ Agree	☐ Strongly Agree
	The new Wiki has assisted in my current role?	☐ Strongly Disagree	☐ Disagree	Neutral	☐ Agree	☐ Strongly Agree
	How often have you used the wiki? (Weekly)	☐ Strongly Disagree	☐ Disagree	Neutral	☐ Agree	☐ Strongly Agree
	I can find documentation easily on the Wiki	☐ Strongly Disagree	☐ Disagree	Neutral	☐ Agree	☐ Strongly Agree
	Do you think there are any issues about the new Wiki that you could highlight? Please discuss?					
	What are the new good things about the Wiki?					
	I can easily discuss new articles, communication and carry out the discussions on the wiki?	☐ Strongly Disagree	☐ Disagree	Neutral	☐ Agree	☐ Strongly Agree
	It was easy to navigate through the wiki	☐ Strongly Disagree	☐ Disagree	Neutral	☐ Agree	☐ Strongly Agree
	The wiki is the way forward to share knowledge	☐ Strongly Disagree	☐ Disagree	Neutral	☐	☐

APPENDIX F

Post Wiki Implementation

Interview Questions

1. Please state your role within the Project Team?

2. What is knowledge management to you?

3. What are the benefits/disadvantages of knowledge management in a project team?

4. What is the main communication media for knowledge for you?

5. Since the Wiki implementation, on average how many times would you have searched for information?

6. Since the implementation of the Wiki, on average how many times would you have posted information on the Wiki?

7. What are the main barriers to knowledge sharing within the team?

8. Do you think a Wiki is a good tool for sharing knowledge?

9. Do you think a Wiki is a good tool for collaborating on knowledge?

10. Do you think knowledge can be kept up to date on a Wiki?

11. What do you think a Wiki should be used for?

12. What improvements would you suggest to the Wiki?

13. What improvements would you suggest to the knowledge sharing processes?

APPENDIX G

Purpose: The purpose of this testing is to ensure that the knowledge that is saved within the Wiki is easy to find for users outside the Project Team.

Audience: These steps are for users outside the project team to ensure that the new Wiki is easily to navigate.

***** Wiki User Testing

1. Please write down how you navigate to the Company B2B Wiki?

2. Please state who carried out the customization category maintenance?

3. What article would Frank like to expand and discuss?

4. Please highlight the deliminater used for the HR user import?

5. Please provide a screenshot on the Training History Screen for Company B2B.

6. Who was the Business Analyst for Phase 2?

7. What role in Company B2B uploads the electronic attendance list which contains the physical signatures of those learners who attended an ILT Class?

8. What role does Michael play in the Company B2B organisation?

9. How do I get copied on communication on a discussion regarding a certain page?

10. When creating a page, what should you always do before submitting a page?

11. What must you do before you log into UAT Company B2B environment from a remote access point of view?

12. What version of a database does Company B2B use?

APPENDIX H

Regression Test 1

How to Create a Learning Plan in Company B2B

-Overview – A Learning Plan for Company B2B is a Plan that consists of several courses that a user has to complete before being awarded a learning plan. There can be several optional and mandatory courses on a Learning Plan.

-Users only have to complete all mandatory courses on a learning plan for it be fully attained.

-Learning Plans are set up by Managers, VG Training Coordinators, VG Curriculum Administrators, VG Admins, LMS admins, LMS IT admins

-You can assign as many courses as you like to a learning plan.

-You can not enrol in a learning plan, you have to be assigned to it by one of the varies roles above.

-Learning Plans are associated with VG's which are like departments within the organisation.

-There is also an option to associate one learning plan with all VG's.

-Before you set up a learning plan you should set up courses that you would like to associate with your learning plan.

-When you are logged in as your VG Curriculum Admin, Course Designer, VG Admin, LMS Admin, LMS IT Admin you can create courses.

'''How to create a course'''

-Create a course by navigating to "My courses" under the "My Administration Functions" menu.

-Go to the first tab on the courses screen.

-Call the course something that is suitable for that course.

-All fields marked with - have to be filled in

-Give the course a version, for example version 1 or 2.

-Give the course a long name.

-If you want the course to be of download document type, which would be like a word document, pdf etc, choose the icon that is of download type. If you are unsure of what the course type is hover over both images.

-If you would like to set up the course as type WBT which would be an online course that users would take, select the icon that is of WBT type.
-Give the course a description, for example, this is to assist users in building kitchen cookers.
-Objectives, are the objectives of the course, for example to be able to install and fit a washing machine.
-You then need to choose which category in the catalog that this course will be assigned to.
-You will then be asked what subcategory this course should be assigned to.
-The extended details tab within the course is optional. This is information should as location and instruments you might need for training.
-You need to go to the content tab to add content to your course.
-From this tab, you can add roll up rules and also associate skills with this course.
-The catalog tab allows you create the catalog information.
-By default the “hide in catalog” flag is checked.
-Uncheck this if you want the course to appear in the catalog.
-Each course can have other course pre-requisites before taking the course in question.
-The access tab allows you associate a course with a certain group.
-Finally the Upload tab is a tab where you can upload a word or pdf etc document and associate this with a course of -type “download document” already selected on the first tab.

Once your course has been set up, they then need to be associated with a learning plan.

""To create a learning plan, browse to “My Administrative functions”.""

-A short name needs to be given to the Learning Plan.
-Again all fields marked with - are mandatory fields.
-A learning Plan needs to be associated with a VG and associated with a learner types.
-In Company B2B Learning Types are of type, Employee, partner and Customer.
-If you want to associate a learning plan to an employee you have to make sure the Learning Plan is of type Employee.
-You can again only associate a learning plan with a certain VG, as in this VG will be the owner of a VG...

-You can have a status of open or closed or you can choose to expire a learning plan, if it is no longer available.
-You then can associate this learning plan with many VG's by choosing the VG Groups option.
-You can select courses by selecting what courses you would like to assign to the learning plan.
-Once your courses are associated with a learning plan, you can choose if these courses are optional or mandatory and you can choose the order of these courses by typing 1, 2, 3 etc in the LP Level option.

-If you would like to assign a learning plan to a user, you choose the "My administration Functions" and "assign learning Plan" option.
-You simply select the learning plan you would like to choose and then select the users you would like to assign this learning plan to.
-The next time the user's logs in they will see the learning plan assigned to them by navigating to the "My learning Plans" screen.

Exercise
-Log in as the LMSITADMIN user.
-Create a course of type download document.
-Call this course something that you will remember.
-Assign any 3 skills to this course.
-Create a blank word document and call it test.doc.
-Associate this document with the course.
-Make sure that the course is visible in the catalog.
-Make sure that the course is associated with VG 1.
-Create a new learning plan and call it something that you will remember.
-Associate the course already created with the learning plan already created.
-Give this Learning Plan to the normal employee user.
-Log out as the LMSITADMIN users and log in as the normal Employee.
-Navigate to my Learning Plans and make sure that you can see the learning plan on this screen.