

# Tools and Techniques for Effective Distributed Requirements Engineering: An Empirical Study

Wes J. Lloyd

Thesis submitted to the Faculty of the  
Virginia Polytechnic Institute and State University  
in partial fulfillment of the requirements for the degree of

Master of Science  
in  
Computer Science

Stephen Edwards, Co-chair  
Mary Beth Rosson, Co-chair  
James D. Arthur  
Doug A. Bowman

July 11, 2001  
Blacksburg, Virginia Tech

Keywords: Distributed Requirements Engineering, Groupware, Computer Supported Cooperative Work, Requirements Analysis

Copyright 2001, Wes J. Lloyd

# Tools and Techniques for Effective Distributed Requirements Engineering: An Empirical Study

Wes J. Lloyd

(ABSTRACT)

Software development teams are often geographically distributed from their customers and end users. This creates significant communication and coordination challenges that impact the effectiveness of requirements engineering. Travel costs, and the local availability of quality technical staff increase the demand for effective distributed software development teams. In this research an empirical study was conducted on the application of groupware to facilitate the requirements phase of the software engineering life cycle. The study considered the impact of the distributed work environment on requirements engineering process quality, as well as the effectiveness of requirements elicitation techniques when used in the distributed setting. Through the examination of the effectiveness of groupware tools, and requirements elicitation techniques, customer participation is found to be paramount for distributed requirements engineering. As requirements negotiation becomes more asynchronous in nature, it appears that customer participation in the elicitation process becomes very important for process effectiveness. Verbal communication barriers negatively effect customer participation. Such barriers cause customers to rely upon mediums such as email and text chat that are very good at reducing social distance, but are less rich in informational content.

# Acknowledgements

---

I would like to thank my advisor co-chairs, Dr. Mary Beth Rosson and Dr. Stephen Edwards. Your enthusiasm for research has inspired me through this whole process. I would like to thank Dr. Rosson and Dr. Edwards for helping me identify and clarify this research topic. I would like to thank Dr. Rosson for allowing access to her graduate level CSCW class the source for role-playing customers in the study. Dr. Rosson also should be acknowledged for her tremendous suggestions for writing style and experiment design. I would like to thank Dr. James D. Arthur who cooperatively worked with me to allow access to his graduate level software engineering class, a vital source for study participants. I would also like to thank him for initially encouraging my attendance at Virginia Tech! I would like to thank Dr. Bowman for his enthusiasm, direction with statistical concepts, and study feedback.

I would like to thank everyone at the Faculty Development Institute and New Media Center. Without their help the computing facilities to conduct this research would have been unavailable. Specifically needing to be mentioned are: Ed McPherson, Mark Raby, Eric Gilmore, M. Shannon Craig, and New Media Center director Ed Schwartz.

I would like to thank the students of CS 5704 Software Engineering and CS 5734 Computer Supported Cooperative Work. In all 46-study participants worked diligently on this distributed requirements engineering project. Your enthusiasm for research truly helped make this empirical study possible. I would like to thank Pete Schoenhoff who volunteered his free time to be a customer participant. He was the only participant in the study not receiving course credit for participation, and his contribution as a customer was outstanding.

I would like to thank Philip Isenhour who also helped develop the research ideas of this study and was an indispensable resource for MOOsburg knowledge. I would also like to thank the members of the Center for HCI, and specifically Wendy Schafer and Craig Ganoë who helped inspire this research. I would like to thank Dr. Henry for her knowledge of Belbin roles and team formation that helped me in developing the methods to form teams in the study. Finally I would like to thank my fiancé Dana for always being a constant source of support.

# Table of Contents

---

<b>Chapter 1. Introduction .....</b>	<b>vi</b>
1.1. Problem Statement.....	9
1.2. Research Goals.....	12
1.3. Empirical Study .....	14
1.4. Research Contributions.....	16
1.5. Thesis Overview .....	16
<b>Chapter 2. Background and Related Work.....</b>	<b>17</b>
2.1. Distributed Software Engineering.....	17
2.2. Reasons for Distributed Software Development.....	17
2.3. Advantages of Distributed Software Development .....	19
<b>Chapter 3. Experiment Design and Methods.....</b>	<b>29</b>
3.1. Project Specification .....	29
3.2. Groupware Tools .....	35
3.3. Requirements Elicitation Techniques .....	41
3.4. Customer Roles.....	44
3.4.1. Secretary Role.....	45
3.4.2. Engineer Role.....	46
3.4.3. Administrator (Vice President) Role .....	46
3.5. Requirements Engineer Roles.....	47
3.6. Team Formation.....	48
3.7. Participant Training and Instruction .....	52
3.8. Experiment Facilities .....	53
3.9. Survey and Observation Methods.....	55
<b>Chapter 4. Results and Discussion .....</b>	<b>58</b>
4.1. Assessing Team Effectiveness.....	58
4.1.1. Software Requirements Specification Grade .....	59
4.1.2. Measurement of Requirements Evolution .....	62
4.1.3. Requirements Identification and Evaluation Metric .....	64
4.1.4. Original Requirements Supported Metric.....	65
4.1.5. Calculating the Overall SRS Document Quality .....	67
4.2. Results of Team Performance: Overall SRS Quality.....	67
4.2.1. Customer Participation versus SRS Quality .....	69
4.2.2. Engineer Experience versus SRS Quality.....	72
4.2.3. Peer Participation versus SRS Quality.....	76
4.2.4. Requirements Elicitation Methods versus SRS Quality .....	78
4.3. Groupware Tools .....	80
4.3.1. MOOsburg Effectiveness.....	81
4.3.2. Tool Effectiveness and Subject Participation .....	82
4.3.3. Factors Influencing Participation.....	86
4.3.4. Tool Effectiveness and SRS Quality.....	87
4.3.5. Participant Groupware Feedback.....	90

4.4.	Requirements Elicitation Methods.....	94
4.4.1.	SRS Quality and Requirements Elicitation Method Effectiveness.....	95
4.4.2.	Participation and Requirements Elicitation Methods Effectiveness.....	97
4.4.3.	Engineering Experience and Requirements Elicitation Method Effectiveness.....	100
4.5.	Customer Participation.....	102
<b>Chapter 5.</b>	<b>Conclusions and Future Work .....</b>	<b>105</b>
5.1.	SRS Quality .....	105
5.2.	Groupware Tools .....	108
5.3.	Requirements Elicitation Techniques .....	111
5.4.	Contributions.....	114
5.5.	Future Directions .....	115
<b>Bibliography .....</b>		<b>118</b>
<b>Appendix A. Groupware System Function List.....</b>		<b>122</b>
<b>Appendix B. System Overview Document.....</b>		<b>125</b>
<b>Appendix C. Customer Role Descriptions.....</b>		<b>127</b>
<b>Appendix D. DRA Final Survey .....</b>		<b>132</b>
<b>Appendix E. Requirements Analysis Slides.....</b>		<b>142</b>
<b>Appendix F. Groupware Training Slides .....</b>		<b>147</b>
<b>Appendix G. Post Meeting Survey #1 .....</b>		<b>152</b>
<b>Appendix H. Post Meeting Engineer Survey.....</b>		<b>154</b>
<b>Appendix I. Requirements Analysis Experience Survey.....</b>		<b>158</b>
<b>Appendix J. Software Engineering Experience Survey .....</b>		<b>159</b>
<b>Appendix K. The Belbin Self-Perception Inventory.....</b>		<b>160</b>
<b>Appendix L. Informed Consent Form .....</b>		<b>162</b>
<b>Vita .....</b>		<b>165</b>

# List of Figures

---

Figure 1-1 - Relative cost to repair a defect at different software lifecycle phases .....	10
Figure 2-1 - Time space groupware taxonomy .....	24
Figure 2-2 - Two dimensions of the application level groupware taxonomy .....	24
Figure 3-1 - General Requirements Analysis Process .....	32
Figure 3-2 - Sample Use Diagram .....	43
Figure 4-1 - Contribution of Individual Metrics on SRS Quality for Groups.....	69
Figure 4-2- Relative Size of Bubble Indicates Magnitude of Experience .....	75
Figure 4-3 - Team Software Engineering Experience vs. Team Requirements Analysis Experience .....	76
Figure 4-4 - Individual Correlations with Requirements Elicitation Experience.....	77
Figure 4-5 - Several Factors Effecting Overall SRS Quality.....	79
Figure 4-6 - Requirements Engineers Reported Usefulness of Groupware Tools.....	80
Figure 4-7 - Reported Effectiveness of Requirements Elicitation Techniques.....	94

# List of Pictures

---

Picture 2-1 - The planner Room in TeamWave .....	27
Picture 3-1 - A participant's view of a Centra Symposium Session.....	39
Picture 3-2 - A MOOsburg Project Space.....	40
Picture 3-3 - Torgersen computer lab room 3060.....	54

# List of Tables

---

Table 3-1 - Suggested meeting schedule for virtual meetings .....	33
Table 3-2 - Required Functions for Prototype Groupware System .....	35
Table 3-3 - Optional Functions for Prototype Groupware System .....	36
Table 3-4 - Summary of Applications considered for Prototype Groupware System .....	37
Table 3-5 - Initial Secretary Role Conflicts.....	45
Table 3-6 - Initial Engineer Role Conflicts .....	46
Table 3-7 - Initial Management Role Conflicts .....	47
Table 3-8 – Self-Reported Software Engineering and Programming Experience of Requirements Engineering Teams. Points indicate relative experience level .....	50
Table 3-9 - Group sizes of Project Teams .....	51
Table 3-10 - Desired Belbin Roles for Customers.....	51
Table 3-11 - Description of Belbin Roles .....	52
Table 4-1 - Identification and Evaluation of Requirements Table.....	60
Table 4-2 - Requirement Types.....	60
Table 4-3 - Questions for Assessing Requirement Defects .....	61
Table 4-4 - Requirement Types for Evolution Metric .....	62
Table 4-5 - Original System Requirements from System Overview Document.....	66
Table 4-6- Overall SRS Quality Metric Scores .....	68
Table 4-7 - Group Scores for Software Engineering and Requirements Engineering Experience .....	72
Table 4-8 - Correlations with Perceived Customer Participation .....	83
Table 4-9 - Groupware Effectiveness on Individual Participation .....	84
Table 4-10 - Tool Usefulness vs. Individual Participation .....	84
Table 4-11 - Reported Tool Usefulness vs. Perceived Customer Participation.....	85
Table 4-12 - Correlations of Influencing Factors on Reported Subject Participation.....	87
Table 4-13 - Reported Groupware Tool Usefulness vs. Overall SRS Quality.....	88
Table 4-14 - Groupware Reported Effectiveness vs. Overall SRS Quality .....	89
Table 4-15 - MOOsburg Positive/Negative Features .....	90
Table 4-16 - Centra Symposium Positive/Negative Features .....	90
Table 4-17 - Group Email List Server Positive/Negative Features .....	91
Table 4-18 - Reported Requirements Elicitation Method Effectiveness vs. Overall SRS Quality .....	95
Table 4-19 - Requirements Elicitation Technique effectiveness vs. Self Reported Participation.....	99
Table 4-20 - Perceived Customer Participation vs Elicitation Method Effectiveness .....	99
Table 4-21 - Correlation between reported method effectiveness and reported experience using method. ....	101
Table 4-22 - Overall Simulation Quality vs. Perceived Customer Participation .....	103
Table 4-23 - Positive Comments supporting Simulation Quality .....	103
Table 4-24 - Negative Comments about Simulation Quality.....	104



## 1.1. Problem Statement

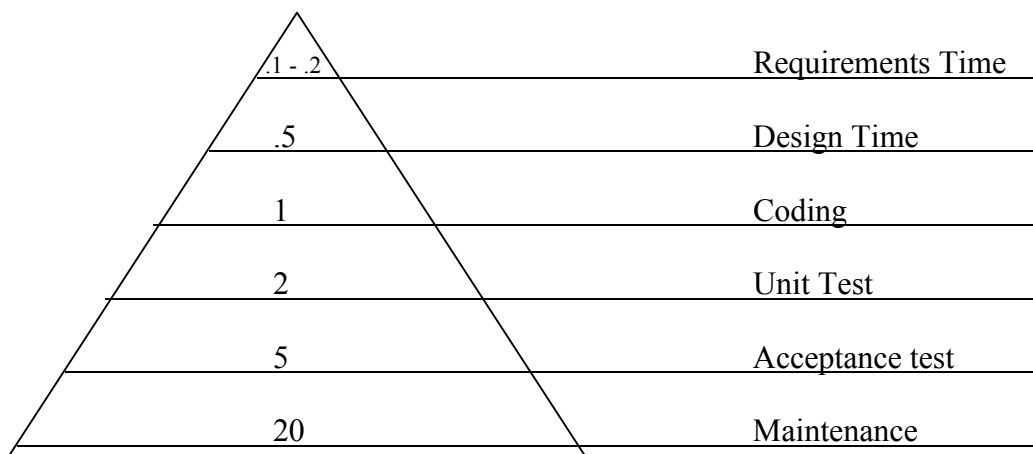
In today's world of distributed software and communication, software development organizations are often geographically distributed from their customers and end users. This creates significant communication and coordination challenges that impact the effectiveness of software engineering. Travel costs, and the local availability of quality technical staff increase the demand for effective distributed software development efforts. Communication technology is continually improving with the advent of the Internet, high-bandwidth fiber optic networks, cell phones, and mobile computing. With the aforementioned improvements in communication technology the feasibility of performing distributed work has increased.

Success in software development occurs when we can deliver a product of high quality to the customer and end users. Software quality is often reflective of the quality and maturity of the software development process applied during development. Pedagogical software process models such as the Waterfall, Spiral, and rapid prototyping models can be applied with varying degrees of success to different software projects. We note that each of these pedagogical software process models includes requirements analysis activities. It is generally accepted that the ultimate quality of delivered software depends upon the requirements upon which the system has been built. [1] [13] To quote Fred Brooks [1, pg. 11]:

“The hardest single part of building a software system is deciding precisely what to build. No other part of the conceptual work is as difficult as establishing the detailed technical requirements, including all the interfaces to people, to machines, and to other software systems. No other part of the work so cripples the resulting system if done wrong. No other part is more difficult to rectify later.”

Software development typically follows a software development process or life cycle. Common to many software processes are the basic activities of requirements analysis, specification, design, implementation, testing, integration, and maintenance.

Correct and complete requirements can assure us that we are building the “right system” and not just a system that is error-free but lacking the required functionality. Many costly errors in software development can be traced to misunderstandings and ambiguities from poor requirements analysis. Studies performed at several companies have measured and assigned costs to errors at various phases of the project lifecycle. In [7], Davis summarized a number of these studies and computed the general cost of fixing errors at different phases of the software lifecycle. As shown in Figure 1-1, if a unit cost of one is assigned to the effort required to detect and repair an error during the coding phase, we can see that it takes somewhere between 100-200 times more effort to correct an error during the maintenance phase as compared to the requirements phase. Correcting an error during unit testing requires 10-20 times more effort than if the error were detected at the requirements phase. A well-conducted requirements analysis study carried out prior to the design and implementation of software can significantly reduce cost throughout the software life cycle [6].



**Figure 1-1 – Relative cost to repair a defect at different software lifecycle phases [6]**

Performing requirements engineering typically involves gathering information about the requirements for the desired software system using an array of techniques that rely on interaction with stakeholders. Group requirements meetings discuss required software functions and identify critical needs. User studies and questionnaires are used to help identify the habits and needs of users. System prototypes are developed and presented to users to gain valuable feedback. All of the information gathering techniques we use rely on interaction with the customer in order to be effective. In distributed requirements engineering we are unable to meet with the customer on a face-to-face basis. Our ability to obtain the breadth and depth of knowledge about product requirements is reduced. Without the informational rich medium of face-to-face personal communication of how can we cope? How do we perform a high quality requirements analysis without easy access to customers?

Each software development activity presents its own unique challenges with distributed development teams. Commercial Computer Aided Software Engineering (CASE) tools such as version management and defect tracking applications have added web-based client interfaces to support distributed use via the internet or corporate WANs. Examples include: Rational's ClearCase / ClearQuest, Intersolv's PVCS and Tracker, Microsoft's Source Safe and companion 3rd party product Visual Intercept.

Requirements Analysis has not benefited as greatly from the development and application of CASE tools as the other phases of the software life cycle. This may be the result of the unstructured nature of requirements analysis. Software engineers gather requirements using a variety of techniques based on the type of application. Requirements elicitation techniques such as brainstorming, prototyping, storyboarding and use cases are very popular techniques. [6] A single monolithic CASE tool may not easily support the plethora of requirements analysis methods.

Considerable work has been directed at supporting distributed tasks in general. Many varieties of groupware systems have been developed and evaluated by researchers and practitioners in the study of computer supported cooperative work. Groupware Systems are computer based software systems that facilitate organizational work processes. Groupware systems enable groups to work collaboratively together on common tasks by providing an interface to a shared environment. [2] Some common

groupware applications include: email, a shared drawing tool (whiteboard), a virtual meeting application, and a shared text editor.

There are few tools specifically designed for supporting distributed requirements engineering. Several process specific requirements engineering groupware systems have been built and tested but none of these systems supported all of the distributed work related to requirements engineering. [11] [9] [16] [5] Most systems of these system were requirements repositories. Comprehensive groupware tools that support all of the distributed interaction of requirements engineering need to be developed and tested in realistic project scenarios. Such groupware tools offer great promise for supporting distributed requirements engineering. [5]

## **1.2. Research Goals**

This research project conducts an empirical study on the effectiveness of requirements analysis in a distributed setting using a suite of groupware tools to facilitate distributed communication and work. Subjects from a graduate-level software engineering class role-play as software engineers. They interact with customers from a virtual corporation using the groupware tools. Customers representing the virtual corporation are graduate students from a graduate-level Computer Supported Cooperative Work class. (CSCW)

From this empirical study we seek to address several research goals:

### **(RG1) SRS Document Quality**

- Examine the output from the requirements engineering teams. Perform quantitative and qualitative review using metrics to assess the quality of the software requirements specification documents produced. Correlate what factors were responsible for the positive/negative performance of the requirements engineering teams.

We begin by assessing the quality of the software requirements specification (SRS) documents produced by each group. We then consider the relative performance of each group with respect to the experimental observations and data. Can we say anything about what caused a group to produce a better specification document? What types of

requirements activities did groups with high quality specifications documents use? What was unique about their collaboration with the customer, which allowed them to produce high quality documents? Did their experience level in software engineering influence their performance? Did customer participation influence document quality?

### **(RG2) Groupware Tools**

- Based on observations and survey analysis, determine which groupware tools best support distributed requirements engineering. Develop a summary of the basic functionality for a groupware system that could support distributed requirements engineering.

From our experimental observations we will determine which tools are used most frequently in the empirical study. We will survey users for their opinions on the available tools and what functionality seems missing from the overall suite of groupware tools. Based on the data collected we hope to define needed functions for a groupware system to support distributed requirements engineering. Once we determine the frequently used tools, we will also examine the relationship between tool use and group performance. Will high performance groups use the groupware tools differently than reduced performance groups? Will specific aspects of the groupware system improve or degrade group performance?

### **(RG3) Requirements Analysis Techniques**

- Based on observations and survey analysis, determine what requirements elicitation techniques work best for distributed software engineering. Develop initial recommendations for techniques that are most useful for distributed requirements engineering.

The very nature of distributed interaction in requirements analysis has different positive and negative consequences when compared with the interaction of non-distributed requirements analysis. Some requirements elicitation techniques may work very well in a distributed setting, where others may fail due to the limited informational bandwidth of distributed interaction. From this empirical study we will observe a variety of requirements elicitation techniques applied in the distributed setting. We will also observe each group execute a full iteration of the requirements engineering lifecycle. We

hope to make suggestions about which elicitation techniques to use in a distributed setting. We may begin to define a new process for doing distributed requirements engineering. We will pay particular attention to the requirements elicitation methods used by high performance and reduced performance groups.

#### **(RG4) Future Work**

- Identify future research areas in distributed requirements engineering.

At the conclusion of the empirical study we expect to identify areas for further research. Some areas could be studied with further empirical studies of a similar nature, and others areas may be completely new and require entirely new investigative and empirical research. We acknowledge that research in distributed software engineering is in its infancy.

### **1.3. Empirical Study**

This research conducts an empirical study on the application of virtual meeting software and other groupware tools used to facilitate the requirements analysis phase of a software project. This study seeks to obtain a better understanding of how well requirements engineering activities are supported using existing groupware tools. We seek to learn what software functionality is required to support a distributed requirements engineering process. As with the support of any remote work, certain activities may be well supported remotely, while other activities may not be well supported in the virtual setting. Through observations in the experiment we intend to obtain an understanding of the effectiveness of supporting requirements methods remotely such that we can begin to specify which activities should be selected for use in a distributed requirements engineering process.

For this study we realize that there is no one currently available CASE tool sufficient to replace the rich communication experience of face-to-face meetings. Any real world software group needing to conduct a requirements analysis in a remote setting would not rely on a single tool for all communication and collaboration. Such a real

world project would use all available communication outlets without hesitation to help best define the product and satisfy the customer. Email, telephone, and CASE tools are all likely to be used frequently during a remote requirements analysis.

This empirical study uses commercial off the shelf (COTS) products to provide the required functionality of a groupware system to support distributed requirements engineering. The emphasis of this research is not to develop a tool, but rather to study the problem of supporting distributed requirements analysis, which can lead to a better understanding of the features an effective tool requires. We have chosen a suite of tools that supports both asynchronous and synchronous communication, which enables group collaboration to support both the individual and collective work of a distributed requirements engineering project. In this study the suite of tools collectively provides support for remote collaboration, which takes the place of the typical face-to-face contact of requirements engineering activities.

In the empirical study we provide training on the use of the groupware tools to all research participants, and then direct the requirements engineers to conduct a requirements analysis with a remote customer. The end results will be a software requirements specification (SRS) document describing the software product desired by the remote customer. Computer science graduate students studying software engineering will role-play as the requirements engineers who conduct the distributed requirements engineering process and write the SRS document. Graduate students studying computer-supported cooperative work (CSCW) role-play as customers of a virtual corporation requesting a requirements analysis to be performed and a SRS document to be written. The customers are given role descriptions and a system overview identifying the major function points of the system. The students never meet face to face to discuss the project during the study. The requirements engineers develop and write a traditional SRS document about the hypothetical system using only the knowledge gained through the distributed collaboration. From this empirical study we observe the effectiveness of requirements analysis in the distributed setting.

## **1.4. Research Contributions**

This thesis investigates using groupware to support requirements engineering in the distributed setting. We perform an empirical study encompassing a complete iteration of a requirements engineering process. From observations and participant surveys we proceed to identify what groupware tools and requirements analysis techniques are best suited for distributed requirements engineering. We seek to understand how the distributed nature of the work affects the quality of the finished product, the software requirements specification document. This understanding can lead to the creation of better software systems to support distributed requirements engineering.

In the empirical study we assess the effectiveness of a suite of groupware tools to enable distributed requirements engineering. Through the examination of the effectiveness of various groupware tools, and requirements elicitation techniques, we conclude that customer participation is paramount for distributed requirements engineering. Asynchronous collaboration techniques seem less effective for performing requirements analysis. It appears that as requirements negotiation becomes more asynchronous, customer participation in the elicitation process becomes increasingly more important for effectiveness. Customer participation is negatively effected by the existence of verbal communication barriers. These barriers cause stakeholders to use less rich communication mediums such as email and text chat for requirements negotiation. [44]

## **1.5. Thesis Overview**

In Chapter 2 we present background and related work for our research. This includes overviews of previous studies in distributed requirements engineering as well as other related work of interest. Chapter 3 describes the empirical study. We detail the project specification, the groupware system, participant roles, participant education/instruction, experimental facilities, and finally survey and observational methods. Chapter 4 provides a presentation and discussion about the results of the empirical study and Chapter 5 provides conclusion and suggests directions for future investigation.



## Chapter 2. Background and Related Work

---

### 2.1. Distributed Software Engineering

Technological improvements in the fields of communication and networking are making distributed group work more feasible. [3] The increased quality and availability of this technology makes distributed software engineering efforts a more viable option for organizations. Software development projects, which utilize distributed teams, where members of the project group are geographically distributed among different physical sites, are known as virtual projects. [8] The teams that work on these virtual projects are known as virtual teams. [24]

In this chapter we begin in Section 2.2 by looking at the motivations behind distributed software projects. We enumerate a number of common business factors that motivate the desire for distributed development. In Section 2.3 we discuss advantages of distributed software development. We present studies that suggest that distributed development leads to higher efficiency and creativity. In Section 2.4 we discuss some common problems of distributed software development. Section 2.5 follows with a discussion regarding distributed software processes. We present a distributed software development process as well as two processes for distributed requirements engineering. Section 2.6 begins with an overview of groupware systems. We conclude Section 2.6 with the presentation of several research projects related to groupware systems that have been developed or applied to the task of supporting distributed requirements engineering.

### 2.2. Reasons for Distributed Software Development

Virtual software projects have become practical because of the technological improvements in communications structure, bandwidth, and performance. Capitalism and global business ventures have created the demand for distributed development. Strategic partnerships, joint ventures, and global companies all share a need in supporting distributed software development efforts. [38] Strategic partnerships have evolved so that companies can develop and promote software products. A product may be

developed by one company but require the market channels through an established vendor in the new market. Existing products may need to be modified and supported by companies other than the original developer. A common example is database interface products. Many relational database management systems (DBMS) are available from vendors such as Oracle, IBM, and Microsoft. A vendor developing a database interface product will need to consult with many vendors to develop custom interface software for each platform. The technical interchange required for the development of this type of software requires distributed collaboration among companies. Joint ventures result in a separate company formed from venture partners. Distributed collaboration may be required between separate partners and the joint venture company. Global companies with offices in multiple countries may conduct joint product development between divisions.

Although performing distributed work in software development may not be desirable, there are often many organization factors that require distribution of software development efforts. In [4], five commercial software development and maintenance projects were studied to see the advantages and disadvantages of the distributed work model. Several common organizational factors contributing to the distribution of software development efforts were identified. (Summarized from [4]):

- o Client may request or require on-site support. It may be necessary to have some project members located on or near a client site. (Affected 2 projects)
- o Project members are unwilling or unable to travel or relocate to other sites. (Affected 2 projects)
- o Skilled workers that are necessary to the success of a project are based at different sites. (Affected all projects)
- o Travel or relocation costs of moving a large number of project members to a different site could be exorbitant. (Affected 2 projects)
- o Technical resources such as specialized hardware are only available at certain locations. (Affected 2 projects)

- o Organizations felt uneasy having staff from another company work at their site, or the software development organization did not want to have their staff working at a customer site. (Affected 2 projects)
- o There was a shortage of office space at a given location. (Affected 1 project)

### **2.3. Advantages of Distributed Software Development**

Despite the many reasons necessitating a distributed development approach, distribution alone does not automatically constitute in gains in productivity and quality. Microsoft considers doing new product development on site as an advantage. The ability to meet informally face-to-face to discuss interdependencies among products is advantageous. [4] However software engineering literature has identified several advantages in distributed work. [3] [19] [22] [24]

Telework, also know as tele-commuting, is the process of working from one's home, satellite office, or mobile computer station. The many benefits of telework, also know as tele-commuting, have been noted. The enhancement of organizational productivity, delivery of customer services, reduction of commute time are frequently mentioned benefits. In [3], electronic mail (email) is noted to be an important technology enabling telework. Broadcasting capability, management of communications, access to information resources, low communication cost, file transfers, and temporal and spatial flexibility are all communication benefits indigenous to email which help to enable distributed work. It should be noted that World Wide Web offers many of the same communication benefits as email.

In [19], a software design experiment was conducted to compare the quality and creativity of system designs for groups with two different meeting environments. Forty-one groups of about five participants each participated in the study. One set of groups held traditional face-to-face meetings, while the other set used distributed asynchronous computer conferencing meetings. The study showed that the distributed asynchronous groups produced more creative designs than the designs produced by the face to face groups ( $p=.003$ ) Although only marginally significant in the study another observed outcome was that the quality of the designs produced by the distributed asynchronous

groups were of higher quality than face-to-face groups. ( $p=.07$ ) This research suggests that the more creative and higher quality results of the distributed groups may be attributed to several factors. The use of a collaborative writing tool, the ability to work in parallel to combine ideas, the production of a written memory reducing the number of lost ideas, and the increased connectivity using computer support are all factors which may have led to better results from the distributed groups in the study.

Virtual teams can be more productive than their co-located counterparts. In [22], a distributed concurrent development process replaced a sequential centralized process for a large-scale telecommunications software project. At Fujitsu, multiple functions for a family of software systems were developed concurrently, by different development teams working at geographically distributed sites. In this particular example the development cycle-time was reduced 75 percent, from one year to only three months. As discussed in [24] the use of the parallel development groups can improve the overall productivity output on a software project. However the coordination effort required for parallel development groups is challenging. [23] [25]

## **2.4. Distributed Software Development Problems**

Performing software development with virtual teams is a complex undertaking. Many coordination and communication problems arise with virtual teams. In [23] and [25], an embedded system product at Lucent Technologies was developed in two locations: Germany and the UK. This project revealed many problems resulting from the distributed development process. Unit testing and development was plagued by incomplete specifications. Implementation diverged from the design document, and the design document was not updated. Bug reports were generated from out-of-date design documents that identified ill-functioning components. While the components were working as implemented, the tests generated at other sites used design documentation that was never updated after designs changed.

Many of the coordination problems seen from this project can be attributed to communications losses incurred because of distributed development. These are the same communications losses which can occur with all distributed work; Requirements

engineering is not unique in this regard. Handling unanticipated events requires flexible ad hoc communication. Unplanned face-to-face contact, where social chance meetings occur, quickly helps to resolve issues. For distributed groups knowing whom to contact about particular issues and the sheer difficulty of making contacting complicates distributed work. Email and voicemail is not answered promptly because the importance of the messages is not well understood. Messages sent by unknown people are treated as unimportant. Communication between sites is also complicated by differences in environments. It is hard to see what is happening at the other site, thus obscuring problems. Language differences complicate communication when people at separate sites do not share the same native language and accent. By nature distributed groups come from different cultures and do not know each other. They can have conflicting values and work habits. These problems are common to distributed requirements engineering because often distributed requirements engineering relies on the same means of remote collaboration.

We believe that with the development and customization of a groupware system we can reduce some of the inherent communication losses incurred with distributed work. By modifying current work processes we can aim to optimize these processes for better application in distributed situations. We believe that a study of the distributed work to be supported, in this case requirements engineering, will assist us in understanding the requirements for a groupware system. This knowledge may then be applied towards developing a customized groupware system for our purpose.

## **2.5. Distributed Development Processes**

Research in distributed software engineering has led to the development of software process models that help coordinate distribution. The Agile Software Process discussed in [22] is a distributed concurrent software process. It defines a method to distribute the workload to multiple distributed teams and defines a process of performing frequent merges and builds of the system. This generally consists of assigning different distributed teams to be responsible for the coding and testing of modules which are later integrated together in frequent merges and builds.

As seen in the literature [22] [23] [25] well organized, and formally defined processes and techniques help improve the success of distributed software development projects. Many software-engineering professionals believe that requirements engineering has its own life cycle. [18] Many requirements engineering life cycles are presented in the literature. [20] [18] [17] [6] We conclude that since distributed software engineering has benefited from processes and coordination, distributed requirements engineering will also benefit [13] [9] [10] from the use of organized processes.

In [13], Digital Corporation was challenged to perform distributed requirements engineering for the specification of several telecommunications products. The products were being developed for the global marketplace and involved busy people distributed across international sites. Digital created a distributed version of the Quality Functional Deployment (QFD) process for requirements engineering. Experiences gained from iterations of the process were used to refine the distributed version of the Quality Functional Deployment process. Using video conferencing and teleconferencing, same time, different place meetings were conducted to replicate the familiar QFD process in the virtual setting. Digital was able to effectively implement the QFD process successfully and they report on the lessons learned from its use. They reported several needs for successful deployment of the Distributed QFD process: trained facilitators and participants, preparation and planning, and a team willing to work actively toward solutions using brainstorming and consensus building.

In [9] and [10], the Stakeholder WinWin success process and the USC WinWin negotiation process were employed with a groupware support system in a requirements negotiation experiment. Fifteen 6-member teams participated in the requirements analysis study. Some teams included remote members, necessitating distributed work. Several positive effects were observed between the stakeholders participating in the study. Benefits included increasing cooperativeness, increased attention to key issues, reduced friction, and facilitation of distributed collaboration.

In this empirical study as Research Goal #3 (RG3) we seek to determine which requirements elicitation techniques and requirements engineering activities that will work well in the distributed setting. These activities when collectively used together may approximate a new process for performing distributed requirements engineering.

## 2.6. Groupware for Distributed Software Development

Concurrent and continuous software development can be accomplished using virtual teams. By distributing the work on a project among geographically separate groups it is possible to perform 24-hour software development. Distributing the work among teams at separate sites can be accomplished using three possible models: cooperation, delegation, and consultation. For more information about these three methods of splitting and distributing the work of software development see [42]. The challenge in coordinating successful virtual software projects identifies the need for groupware support for this sort of collaborative activity. In [24] the Global Working in Software Engineering (GWSE) system is proposed to coordinate the delegation of work among development sites. In [22], the PRIME, (PRocess Information ManagEr) was developed to help coordinate distributed development.

The needs identified for software systems and groupware to support distributed software development suggests that having groupware to support any distributed work including requirements analysis is desirable. Requirements analysis, the act of defining the software product, is typically the first step of all software processes. Distributing the work of requirements analysis can benefit from the application of groupware and tools. [11] [5] [14] [15] [16] [9] [10]

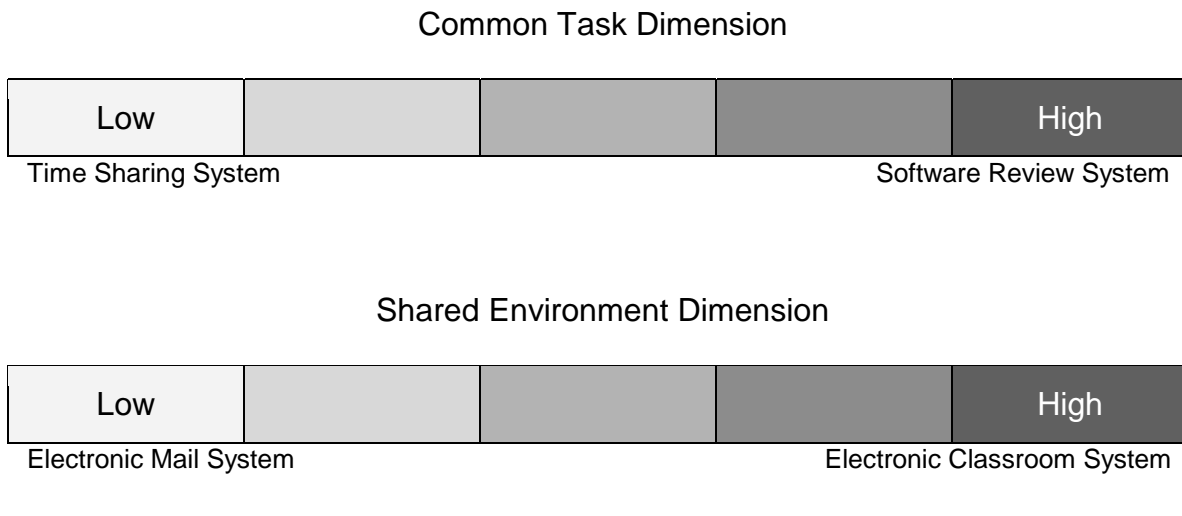
Groupware is defined as “computer-based systems that support groups of people engaged in a common task (or goal) and that provide an interface to a shared environment.” [2] Groupware systems can be classified by two separate taxonomies: time space, and application level.

---

		Same Time	Different Times
Same-Place	Same-Place	Face-to-face Interaction	Asynchronous Interaction
	Different Places	Synchronous Distributed Interaction	Asynchronous Distributed Interaction

---

**Figure 2-1 - Time space groupware taxonomy [2]**



**Figure 2-2 - Two dimensions of the application level groupware taxonomy [2]**

The time-space taxonomy classifies groupware systems based on the type of distributed work they support. In one direction groupware applications are classified based on whether they support co-located versus distributed collaboration. (same place or different place) In the other dimension groupware applications are classified based on whether they support synchronous (same time) or asynchronous (different time) collaboration. Ideally for a groupware system to support distributed requirements analysis both distributed asynchronous and distributed synchronous work would be supported. (different place with either same time or different time).



The application level taxonomy describes the level of integration the software has with the group. This taxonomy has two dimensions: common task, and shared environment. A system with a high value for the shared environment dimension will enable a large common workspace with the support for many tasks. A system with a low value for the shared environment dimension would support only a single task or mode of communication. A system with a high value for the common task dimension will support collaboration on tasks such as document editing. A system with a low value for the common task dimension will not support such collaborative work.

For Distributed Requirements Analysis we desire a system to support both synchronous and asynchronous distributed interaction. The groupware system we desire could be considered a coordination system. Coordination systems enable the integration of individual work efforts toward the application of a larger goal. They allow users to view their actions and track other relevant actions of other users within the context of the overall goal. In distributed requirements engineering we desire a groupware system that integrates the work efforts of all the project stakeholders. We desire an integrated system that supports customer and engineering collaboration to enable requirements elicitation and development throughout the requirements engineering life cycle.

Based on the definition of groupware, many existing distributed requirements engineering tools based on the definition of groupware can be considered as groupware systems. None of these systems can be considered comprehensive coordination systems. Most of these tools are examples of computer aided software engineering (CASE) tools, because they facilitate some part of the requirements engineering process, but not the entire life cycle. In this sense (CASE) tools can be considered as groupware systems with a low level of application integration. Research Goal #2 (RG2) of this empirical study is to determine the necessary functionality for a groupware system for distributed requirements engineering. Our goal in this is to consider a comprehensive approach to coordination.

In [11], the Goal-Based Requirements Analysis Tool (GBRAT) is presented. GBRAT uses the internet as the application interface to enable collaborative distributed requirements analysis. GBRAT enables the Goal Based Requirements Analysis Method (GBRAM) process. Using a web based interface users enter requirements known as

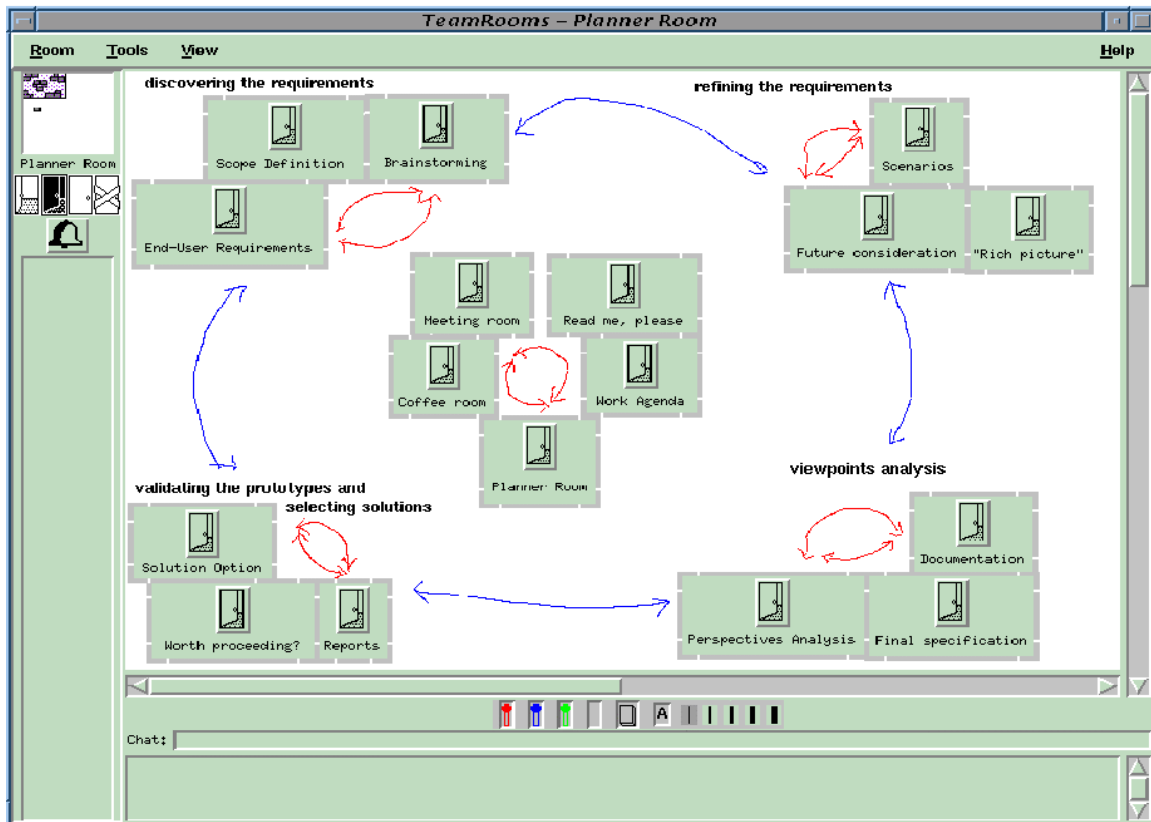
goals into a web based repository. Goals can be classified and organized with hyperlinks to enable traceability to various artifacts. Relationships can be specified between goals and subgoals. GBRAT harnesses the power of the internet for collaborative requirements engineering by providing a shared requirements repository for classification and mediation of system requirements. As a distributed requirements analysis tool GBRAT could be considered a CASE tool. It supports the collection of software requirements, but none of the elicitation techniques necessary to discover them.

Front Loaded Accurate Requirements Engineering (FLARE) is a distributed requirements engineering environment which uses a set of applications and tools to provide an environment to enable distributed requirements analysis. [16] The system is a prototype developed by the United States naval postgraduate school. The system can be considered a comprehensive coordination groupware system because it aims to coordinate and facilitate the entire distributed requirements analysis life cycle, including requirements discovery. However to our knowledge no empirical studies were conducted to assess the effectiveness of this system.

The WinWin System is a collaboration tool which supports the WinWin negotiation model, which is essentially a requirements analysis process. The tool uses web-based support to enable distributed synchronous and asynchronous collaboration. The system supports a distributed repository of WinWin components. The WinWin system stores a description of each requirement with links for artifacts and related information necessary to track and assist in negotiation, and execution of the WinWin negotiation model. A number of external support tools are integrated with the WinWin system to assist in analysis of cost, schedule, performance, modeling, and simulation. The WinWin system is a comprehensive coordination groupware system which supports the requirements negotiation process. It is groupware that enables the WinWin negotiation model in a rigid way. The WinWin system is not as flexible to dynamic process changes as other groupware systems reviewed. This may benefit users by forcing adherence to the process that could otherwise result in deviation and incomplete requirements analysis.

Herlea in [5] [14] [15] used the virtual environment groupware system TeamWave to develop a requirements engineering collaboration space. TeamWave with

a room-based metaphor allows collaborative spaces to be configured in a virtual community environment to facilitate interaction and collaboration among users. Rooms in the space can contain any number of collaborative tools that can broadly support the various tasks of distributed requirements analysis. The requirements engineering space in TeamWave has a set of rooms to support various activities at different steps of the requirements engineering life cycle. Rooms are configured for various activities such as scope definition, brainstorming, and elicitation of end user requirements. A navigation diagram allows the navigation through the life cycle by representing each activity space as an icon that represents a doorway to that space.



Picture 2-1 - The planner Room in TeamWave [15]

TeamWave offers a variety of collaborative tools including: a shared whiteboard, post-it notes, bulletin boards, note organizers, brainstorming tools, voting tools, file

viewers, databases, action item organizer, text-based chat, list of other people present, and telepointers. Alho notes in [8], that groupware often fails because of the limitation to support ad-hoc dynamic processes. We believe that the diverse array of collaborative tools in TeamWave is well fitted to support distributed work because of the great flexibility of task support with such a wide variety of tools. This should be noted as an important requirement for any groupware system supporting distributed requirements analysis.

Herela's requirements engineering space in TeamWave presents a comprehensive coordination groupware system to support all of the activities of distributed requirements analysis. Herela presents several research issues that we seek to elaborate on through our empirical study. These issues include: Using the groupware can users learn enough about requirements from the customer to perform distributed requirements analysis? What requirements analysis activities still need to be supported? Is a requirements analysis facilitator needed? Are there problems with deploying the system into use? Does the groupware scale for larger projects? Can the groupware's effectiveness at supporting distributed requirements analysis be evaluated?

In Chapter 3 we present the empirical study and describe how it was conducted. Section 3.1 provides an overview of the project specification supplied to participants. Section 3.2 discusses our search and selection of available groupware tools for supporting the distributed collaboration in our study. Section 3.3 presents an overview of the team formation methods. Section 3.4 presents the customer roles which were role-played by the customer participants in the study. Section 3.5 discuss the requirements engineer roles and responsibilities for participating in the study. Section 3.6 describes our education and instruction provided to study participants prior to the start of the experiment. Section 3.7 presents an overview of the experimental facilities used in conducting the experiment. Section 3.8 concludes with a presentation of the survey and observation methods used throughout the study.

## 3. Experiment Design and Methods

---

### 3.1. Project Specification

This empirical study simulates a distributed requirements specification project. The software engineers are students from the graduate Software Engineering course at Virginia Tech CS-5704. The customers are students from the graduate Computer Supported Cooperative Work (CSCW) course CS-5734. The CSCW students role-play as customers of a virtual corporation. They are given role descriptions that describe their role in the virtual company. (See Appendix C) They select a name for their company and effectively act on behalf of that company for the duration of the experiment. Each company has the same demographical structure, “a multi-discipline engineering firm with approximately 100 employees”. Each firm employs engineers, project managers, secretaries, accountants, and administrative personnel and is housed in a modest 2-story office building.

Requirements engineering is supported using a set of collaborative tools. These tools are referred to as the groupware tools and will be discussed in detail in Section 3.2. The tools consist of: Centra Symposium, to support real time virtual meetings, MOOsburg, to facilitate file sharing and informal impromptu meetings, and email, for file sharing and asynchronous discussions.

The system being specified is a personal planner and scheduling system for the virtual corporation. The virtual companies have a fixed number of meeting rooms and audio/visual equipment in their 2-story office building. The desire is to develop a software system to help schedule shared company resources as well to aid the personal planning of all company employees. Being a small company both their manpower and physical resources are limited, and an automated scheduling system has been identified as a way to efficiently optimize finite resources to increase the productivity of the company.

A meeting scheduler system is the benchmark problem that the research community has adopted for requirements engineering studies. [20] There is an existing two-page requirements document describing the scheduling problem. [21] Because determining and negotiating requirements for a meeting scheduling system is challenging

and problematic, this benchmark problem has enough complexity that it is a good sample problem, with issues similar to real systems. A positive note for the customers is that the meeting scheduler system requires no special domain knowledge. Since we could not assume the customers would have special knowledge that would allow for the use of unique requirements problems, the scheduling system seemed like an ideal sample system for the study. We agree with Colin Potts' statements about the meeting scheduler problem.

“Some would argue that the meeting scheduler is not a real system and in fact is a trivial exercise because the requirements document is only two pages of text. To this we answer that many significant real-world problems should be described as briefly as this initially. The real challenge for requirements analysis at this point is not to make specifications the size of *War and Peace* more readable or formal. Rather, it is to turn a very vague and high-level mission statement into a detailed specification as early as possible.” [20]

Although the above mentioned two-page requirements document describing the scheduling problem written by Axel van Lamsweerde [21] was available, we wrote our own problem statement, and then extended it with details about customer roles and specific requirements. Rewriting the specification allowed us to modernize the problem by requesting a current technological solution. We also were able to define customer roles and customize initial conflicts between customers over system requirements.

Each software engineering group conducted a distributed requirements analysis to develop and write a SRS document that specified the requirements of the corporate scheduling system. There were a number of constraints placed on the engineers that helped to structure the project.

Each requirements engineering team conducted a series of four planned virtual meetings with the customer groups. Each meeting was allowed to take up to 90 minutes. The software engineers and customers met using Centra Symposium, a point-to-point audio conferencing and meeting support tool. (See Section 3.2) The software engineers needed to plan an effective agenda to make optimal use of the limited time in virtual

meetings. The Requirements engineers were also allowed to conduct additional meetings using MOOsburg, a room based collaborative environment. Because MOOsburg does not support point-to-point audio conferencing these meetings were limited to text chat. Group-based Email was available to complement virtual meetings. Unclear issues and questions could be clarified by sending email using a project list server. Email sent to the project list server would be automatically copied to all customers and software engineers. The email list server emulates an ongoing forum for discussing project issues.

To assist the software engineers in requirements analysis we provided guidelines for planning the activities of each formal virtual meeting. (see Table 3-1) This provided a benchmark so software engineers could understand the types of activities that should occur at different points within the requirements engineering lifecycle. We also described a general requirements engineering process that could be followed during the project. This process is described in Figure 3-1.

### **Step 1 - Capture User Requirements**

General requirements statements are written. Example: "In a public organization a new system of documentation access management is going to be built. This system organizes the documents within the institution and manages the access to them by authorized people."

From these general statements user requirements can be identified, use-cases envisioned, and prototypes built.

### **Step 2 - Analyze requirements, Construct and demonstrate prototypes (interface mockups), Use Case Modeling**

From the general requirements statements, use cases (usage-scenarios) can be generated to capture all information relevant to the problem, concepts, characteristics, and relationships. Vocabulary can be clarified.

Prototypes should be developed and presented to customers. Customers walk through usage scenarios using the prototypes to identify issues and requirements.

Relations among requirements should be identified. Requirements should be classified by common attributes such as: users, security, and implementation priority. Common attributes for a set of requirements are unique based on the problem-domain of the system.

### **Step 3 - Write the requirements specification document**

The requirements specification document should be written to follow a standardized format. It should give a clear picture as to what the software system is to do in terms of the problem domain. The requirements specification document should not discuss how the application is implemented or how implementation problems are solved.

### **Step 4 - Verification and revision of requirements specification document**

Customers and engineers will work together to verify the requirements specification document. Use cases should be associable to functional requirements. Every function identified though use cases should be addressed in a functional requirement. Functional requirements should be clear, concise, unambiguous statements that clearly communicate the intent of the developers to the customers.

**Figure 3-1 – General Requirements Analysis Process**



Table 3-1 - Suggested meeting schedule for virtual meetings

Meeting	Topic	Timeframe
Meeting 1  Capturing User Requirements	<p><b>BEFORE</b> Before the meeting the software engineering group should meet to appoint a leader who will serve as the moderator for virtual meetings. The group should also appoint a primary designer/maintainer for the project's virtual meeting space in MOOsburg.</p> <p><b>AT MEETING #1</b> <b>Capture user requirements.</b> Topic Introduction, obtain basic information about requirements. Engineers should be able to write basic statements about the system requirements.</p>	February 7 - 16
Meeting 2  Analysis, Prototyping, Modeling	<p><b>BEFORE</b> Using information from meeting #1, engineers should write and revise basic statements about system requirements. Basic prototypes should be developed which will be used to drive the discussion and identification of use-cases in meeting #2. The virtual meeting space in MOOsburg should be setup after meeting #1 to facilitate any offline communication between meetings.</p> <p><b>AT MEETING #2</b> <b>Analysis, Prototyping, Use-Case Modeling.</b> Generate use-cases (scenarios) based on statements about system requirements. Analyze use-cases to identify information about the problem-domain and system. Present prototypes that demonstrate and facilitate use-cases. (scenarios)</p>	February 19 - March 2
Meeting 3  Further Analysis, Prototyping, Modeling	<p><b>BEFORE</b> After meeting #2 the software engineers are expected to enhance prototypes and further develop use-cases that will be used in the requirements discussions in meeting #3.</p> <p><b>AT MEETING #3</b> <b>Analysis, Prototyping, Use-Case Modeling.</b> Meeting #3 will continue the activities from meeting #2. At the conclusion of meeting #3 the software engineers should have enough system detail to write the requirements specification document.</p>	March 12 - 23
Requirements	Requirements Specification Document due in class	March 29

Specification	on Thursday March 29th.	
Meeting 4 Walkthrough, verification of specification	<p><b>BEFORE</b> The software engineers will take information from the Analysis activities in meetings #2 and #3 and write a formal software requirements specification document. The document should be completed for review by the customers during meeting #4. This document will be collected in class on Thursday March 29th. The client will be provided a copy of the software requirements document prior to the meeting #4.</p> <p><b>AT MEETING #4</b> <b>Walkthrough, and verification of specification.</b> At meeting #4 the engineers will present their document to the customer. The customer will comment on necessary revisions to meet their expectations.</p>	April 2 - 13
Requirements Specification	The final revised requirements specification document is due in class on Thursday April 19th.	April 19

All virtual meetings took place outside of class time. We scheduled twenty-four virtual meetings: four meetings for six groups, of approximately 90 minutes each for the forty-six participants in the project. Given that all participants were graduate students with unique schedules, it was difficult to schedule meetings so that everyone could attend. A complicating factor was that the teams were assigned before meetings were scheduled. Each participant specified what times they could meet from a set of 67 possible meeting times. The scheduling was biased for the engineering groups because the work was defined as a required class project. Thus, meeting times were chosen to allow the maximum number of customers to be present but only if a majority of the engineering team members were available. If a meeting was scheduled for a conflicting time for some participants, they were asked to try to attend if at all possible. Since all meetings for the entire project were scheduled at the beginning, participants had a significant advanced warning so they could adjust their schedules. If meeting times were still not satisfactory customers and engineers were permitted to negotiate a better meeting time for everyone.

## 3.2. Groupware Tools

This empirical study examines the use of groupware tools in support of distributed requirements engineering. We choose not to develop a groupware system for the study since the time and efforts required to develop such a system would be an excessive undertaking out of scope of the time and manpower constraints for this research effort. Instead, we have studied the available set of collaborative COTS, commercial off the shelf, tools and selected the appropriate tools to form a suite of tools that together provide a combined suite of tools for distributed requirements analysis. This tool set supports our distributed requirements engineering effort throughout the empirical study.

Table 3-2 shows the minimum set of requirements that we identified for our prototype groupware system. R1 is the minimum communication required to conduct a synchronous virtual meeting. Synchronous meetings are an integral part of requirements engineering lifecycles, and therefore this was a system requirement.

R2 was identified as a requirement because we wanted our engineers to be able to use storyboarding as a requirements elicitation method. The ability to have a shared display of images would support the showing screen shots, for read-only prototypes. R2 was also identified as a requirement because we wanted the engineers and customers to be able to conduct a review session of requirements statements and documents. The ability to have a shared display of documents is necessary for conducting an effective review in the virtual setting.

**Table 3-2 - Required Functions for Prototype Groupware System**

<b>Groupware Requirement</b>	<b>Purpose</b>
R1- Text Chat	Synchronous Distributed Communication
R2- Shared Display of artifacts: images, documents	Asynchronous/Synchronous Distributed Collaboration
R3- A shared telepointer for user awareness during display of images and documents	Synchronous Distributed Collaboration
R4- Email	Distributed asynchronous communication
R5- Recording capability	Archiving meetings, emails, document repositories for historical reference by study participants and observational statistical data

R3 was identified as a requirement because we thought that it would augment the shared view of images and documents. Having a telepointer, also known as a shared cursor, or mouse pointer, allows a distributed user to bring attention to parts of an image or document that are relevant to the present discussion.

R4 was identified as a requirement because we thought that in a typical distributed project, customers and engineers would have access to each other either through the use of a telephone or internet email. Since monitoring of all participants' use of telephone during the empirical study was impractical we elected to only allow the use of email for remote collaboration for the study.

R5 was necessary to facilitate the concept of a group memory for the project. Archives of emails, document repositories, and meeting recordings are all artifacts that help support the group's memory. Enabling a group memory is a design goal of groupware systems. The group needs a record of part activities and decisions so that future work may proceed. [27] R5 assists us in making observations and collecting statistical data throughout the study.

**Table 3-3 - Optional Functions for Prototype Groupware System**

<b>Groupware Requirement</b>	<b>Purpose</b>
OR1- Multipoint Voice Teleconferencing	Synchronous Distributed Communication
OR2- Multipoint Audio/Video Teleconferencing	Synchronous Distributed Communication
OR3- Shared Image Editing	Asynchronous/Synchronous Distributed work
OR4- Shared Document Editing	Asynchronous/Synchronous Distributed work

Several optional requirements were identified as possible enhancements to the suite of groupware tools. (OR1-OR4). The use of multipoint audio conferencing (OR1) in addition to text chat increases the richness of content and communication in virtual meetings. Audio conferencing through groupware software substitutes for the use of telephone conference calls. Although not required, we believe that multipoint audio conferencing will help increase the productivity during synchronous meetings.

Multipoint Video/Audio teleconferencing is considered the closest approximation to face-to-face communication. However, productivity gains experienced from the use of video teleconferencing technology for remote collaboration may not be excessive. Little

advantage has been discovered when comparing video versus audio-only communication. [43] The use of video has been shown to encourage participants to participate actively in remote collaboration, and video has been shown to add value to certain kinds of remote negotiation tasks. For conducting this empirical study we had only limited access to video conferencing equipment. We chose not to supplement our suite of groupware tools with a video conferencing tool because of both limited accessibility to the technology, and the non-deterministic benefits of using video.

For that reason we have not mandated it as a requirement, and have left it as an optional feature for our prototype groupware system. Shared image and document editing are desirable features for collaborative distributed work (OR3-4). Although these features are nice, and will enhance collaborative work in distributed meetings, collaborative editing is not absolutely required for our purposes because it is typically not an activity explicitly mandated in requirements engineering life cycles.

Several existing groupware applications were considered for the use in the prototype groupware system for this empirical study. (see Table 3-4) Early in the study we identified the fact that it was unlikely that a single available software package would exist that would fulfill all of the minimum requirements. Ultimately the packages selected were those that fulfilled the requirements and were also available within the computing infrastructure at Virginia Tech.

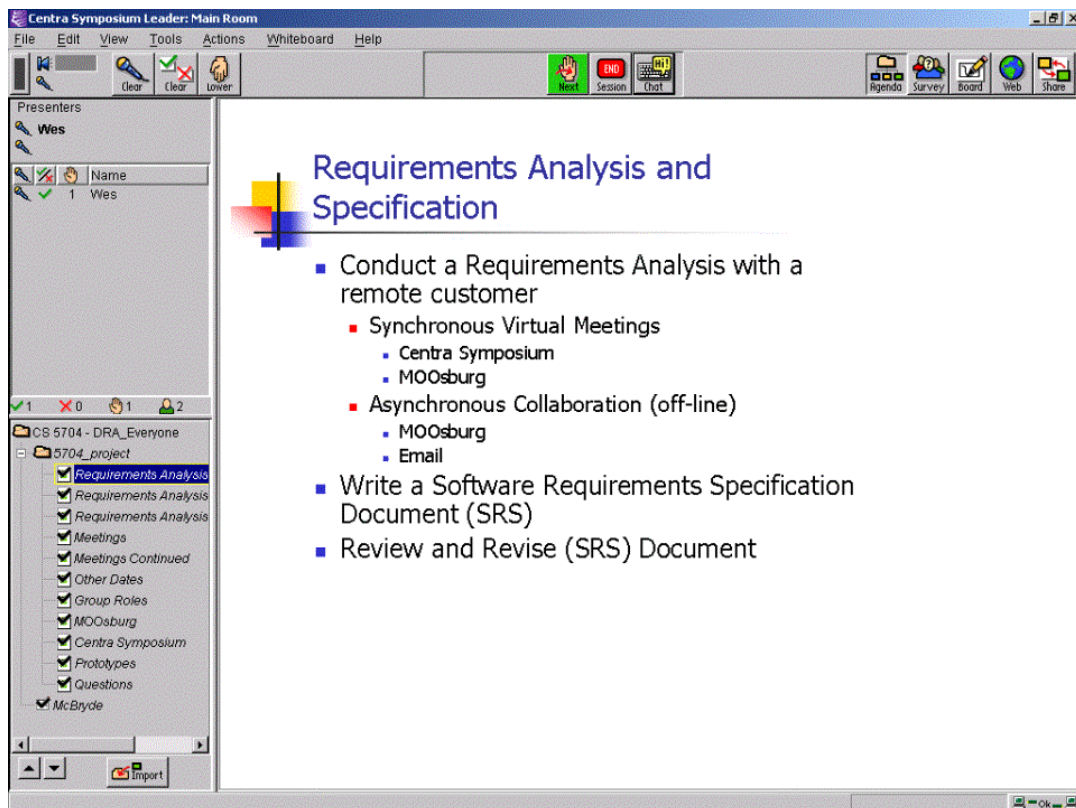
**Table 3-4 - Summary of Applications considered for Prototype Groupware System**

Microsoft Netmeeting [34]	1:1 video/audio conferencing Application sharing: users share a single thread of control per application (/w telepointer) Multi-page shared whiteboard Synchronous Text Chat Freely available
Centra Symposium 99 [32]	1:many audio conferencing Application sharing: users share a single thread of control per application Shared whiteboard Synchronous Text Chat (public and private modes) Available at Virginia Tech Slideshow: imports Powerpoint slides for static visuals (/w pointer) Voting (/w telepointer)

	<p>Shared web browsing (/w telepointer)  Records meetings</p>
MOOsburg [31]	<p>Room based collaborative environment  Shared whiteboard  Asynchronous and Synchronous Text Chat  File Sharing: provides a graphical shared file repository for uploading and downloading files  Freely available</p>
Teamwave [35]	<p>Room based collaborative environment, used in [5,14,15].  Shared whiteboard  Asynchronous and Synchronous Chat  File viewer  Concept Map  Threaded Discussions  Calendar  Web Browser  Stick Notes  To-do list  Doorway  Brainstorming tools  Voting tools  Slide viewer</p>
Meeting Works [36]	<p>A meeting aide tool, designed to augment audio/video conferencing  Brainstorming tools  Idea organization  Voting tools  Cross impact / multi-criteria analysis tools  Report generation</p>
Centra eMeeting [33]	<p>1:many audio conferencing  Application sharing: users share a single thread of control per application  Synchronous Text Chat (public and private modes)  Slideshow: imports Powerpoint slides for static visuals (/w pointer)  Voting (/w telepointer)  Shared web browsing (/w telepointer)</p>
CuSeeMe [37]	<p>1:many audio/video conferencing  Shared Whiteboard  Synchronous Text Chat</p>
IVisit [41]	<p>1:many audio/video conferencing  Small video images for faster performance  Synchronous Text Chat  Freely Available</p>

We selected Centra Symposium 99, MOOsburg, and email as the set of software tools to implement our prototype groupware system.

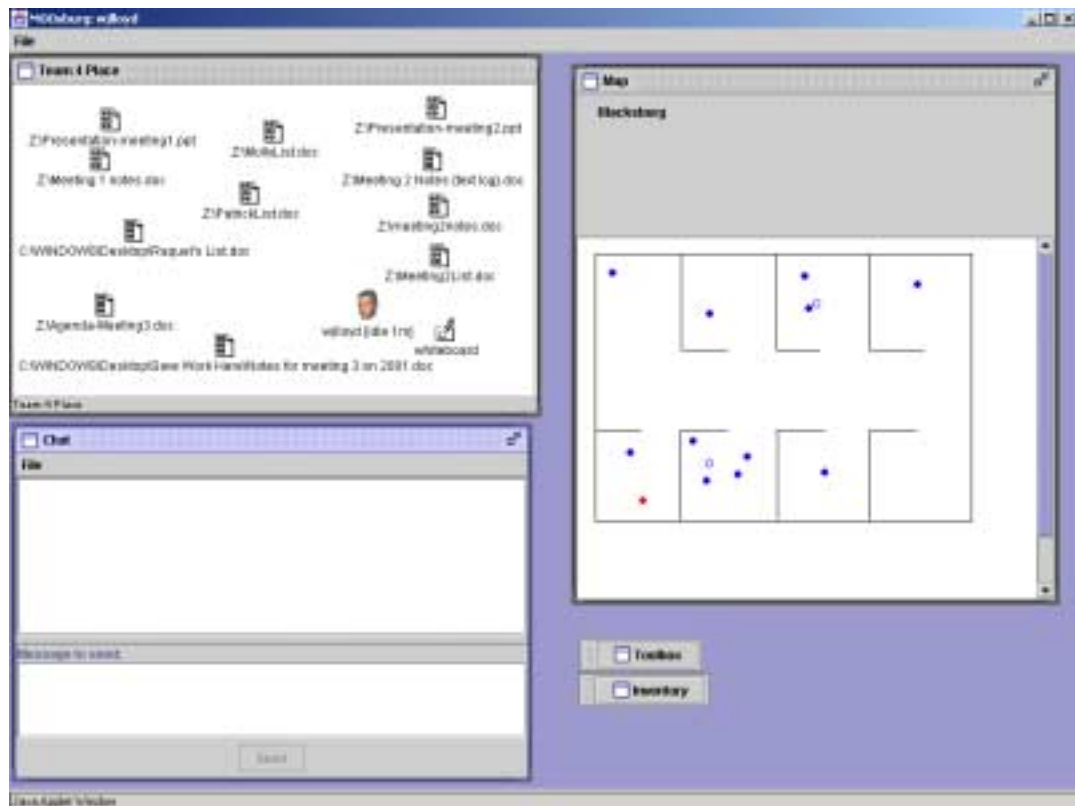
Centra Symposium supports distributed synchronous meetings with 1:many voice conferencing. It offers several meeting aides including: shared whiteboard, shared web browser, slide show for agenda display, application sharing, and voting. Centra Symposium is designed for distributed learning. It is designed to have a session leader with several students watching and requesting permission to participate in discussion. For facilitating distributed meetings many of the floor control mechanisms built into Centra Symposium help the session leader control the flow of the meeting to conduct the most productive meeting possible. Many of the features intended to support an academic instructional setting in Centra Symposium are just as useful for distributed meetings. For our study Centra Symposium provided functionality for requirements: R1, R2, R3, R5, OR1, OR3, and OR4.



Picture 3-1 - A participant's view of a Centra Symposium Session

MOOsburg was selected for support of unplanned impromptu collaboration in the requirements analysis project. It can be used for both asynchronous and synchronous communication and meetings. For our study MOOsburg provides functional support for requirements: R1, and OR3.

Each group of software engineers created a project workspace in MOOsburg. This was a virtual space where project documents were made available. Collaborative tools such as a shared whiteboard, asynchronous chat, and a list editor were also available. This space could support both impromptu meetings between customers and engineers, and serve as a shared file repository for each project group.



**Picture 3-2** - A MOOsburg Project Space

For email support we configured a set of group list server accounts. These list servers accounts automatically distribute all email to a set list of recipients. The list



server account was configured so that any email sent to the address would automatically be copied to the entire group. By requiring all mail messages be sent to the list server account we are able to archive all mail sent to list server accounts, to monitor the use of email, and generate statistics. Our email configuration supported requirements: R4, and R5.

Microsoft Netmeeting offers audio and video conferencing, but only a 1:1 conference not 1:many. Teamwave was the community-based groupware system used in [5, 14, 15]. MOOsburg was chosen over Teamwave for several reasons; 1: the packages have similar functionality, 2: MOOsburg is a locally developed product, and therefore is easily enhanced and modified for research purposes, 3: MOOsburg is freely available for use whereas Teamwave was not. Meeting Works is a meeting support tool that provides meeting tools to enhance both distributed and non-distributed meetings. This package was not available due to cost limitations. Centra eMeeting offers much of the same functionality, as Centra Symposium except it was not available because of cost limitations. CuSeeMe offered 1:many video conferencing but did not support requirements R2 and R5. iVisit offered 1:many video conferencing but lacked support for several other functions including R2 and R5.

### **3.3. Requirements Elicitation Techniques**

Research Goal #3 of this thesis is to analyze how well requirements elicitation techniques work in the distributed setting. We hypothesize that some requirements elicitation methods are superior for distributed requirements engineering due to their unique characteristics. The requirements engineer participants in the study were introduced to a number of popular requirements elicitation methods as part of their participant training for the experiment. The particular details of the training are discussed in Section 3.7.

The requirements engineers were not required to use any particular elicitation techniques for requirements engineering. Instead, we instructed them on the use of several techniques and allowed them to choose techniques most appropriate for the situation. Participants had varying degrees of experience using the elicitation methods.

Ultimately selection of methods was influenced by method experience and instruction in the course. A few of the techniques were particularly emphasized as part of the instruction in the software engineering course.

Here is a summary of the elicitation techniques presented and evaluated in this empirical study [6]:

- Requirements Workshops

A Requirement workshop is a 1-2 day intense meeting with all project stakeholders where the focus is on requirements elicitation through brainstorming.

- Role playing

The developers study the user's work process so they can take on the role of the user. This allows the developers to experience the user's perspective towards the software system being developed.

- Brainstorming and Idea Reduction

A brainstorming session consists of the brainstorming phase and the idea reduction phase. The session begins with brainstorming to generate as many ideas as possible without criticism or debate over the details. The generated ideas are then grouped by similar categories, and ideas not worth further investigation are dismissed. The end result is a collection of useful ideas and feedback to drive the requirements elicitation process.

- Question and Answer

The Question and Answer technique of requirements elicitation is similar to a customer interview, except that it is more informal. Questionnaires are not formally written out before the interview. The path of the dialogue during the session varies based on user feedback and participation.

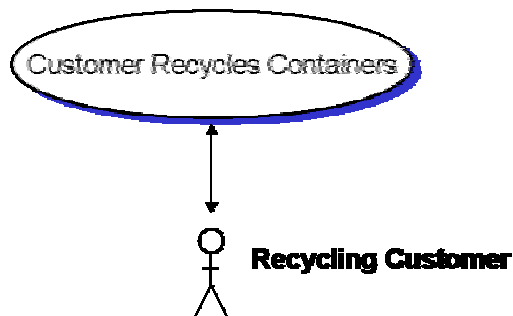
- Storyboards

Storyboards are collections of screen shots and pictures which display how the end system may look. Passive storyboards are simply collections of pictures which the

requirements engineering shows the users in sequence. Active storyboards are closer to a system prototype. An active prototype can be navigated, but it is still just a collection of pictures showing an end system. Storyboards are not interactive as they do not expect any data.

- Use Cases

A use case describes a set of actions and variant actions that a system performs producing viewable results to the actors. Actors are someone or something that interfaces with the system. A use case diagram shows the relationships between the use case and the actors. Arrows show the flow of interaction between actors and the use case. A use case defines a common usage scenario and includes possible exceptions to the normal flow of events. Use cases can be used in software testing to validate that original functionality has been implemented in the software system.



**Figure 3-2 - Sample Use Diagram**

- Prototyping

Requirements engineers build software prototypes to help identify system requirements. A prototype is a partial implementation of the software system, built to help developers, users, and customers better understand system requirements. Throwaway prototypes are built quickly using shortcuts and alternate technologies to gain knowledge and demonstrate ideas. Evolutionary prototypes are built using the architecture of the end system with the intent that they will eventually be developed into the real system.

Horizontal prototypes include a wide range of the system functionality, while vertical prototypes exhibit only a few requirements of the system.

- Requirements Management

Requirements Management is the process of collecting, categorizing and prioritizing the requirements that are identified from requirements elicitation. Organization of requirements allows the software developers to concentrate on the most crucial requirements during system implementation. This conserves the limited resources of time and money throughout the software development lifecycle.

### **3.4. Customer Roles**

Three distinct roles were developed for the customer role-playing in the project. These roles included: the secretary, administrator (vice president), and engineer. The original team size was to be three customers per team. Due to enrollment changes in both classes however, we used 3 teams with four members, 2 teams with 3 members, and 1 team with 2 members. The vice president and engineer were considered the most important roles for contributing to the requirements meetings. The original design was that any teams with extra members, would be assigned as engineers. However Team #3 ultimately had 2 members role-playing as secretaries due to uncontrollable enrollment changes.

To elicit discussion and negotiation among customers and engineers over system functionality we included in the initial role descriptions, predefined requirements conflicts between different customer roles. In [12] requirements negotiation is identified as an essential part of requirements specification. We wanted our empirical study on requirements analysis to require some requirements negotiation between group members to increase the realism of the study. If the system description given to the role-playing customers was complete and specific at the beginning of the study, the customers could simply read the requirements to the engineers, and a robust and dynamic requirements gathering process would be less unlikely to occur. Since several system details were in

conflict or left only partially specified this allowed for both requirements negotiation and evolution throughout the empirical study.

### 3.4.1. Secretary Role

The secretary role was that of a non-technical employee of the virtual company. The secretary was interested in an easy to use system, without complex features. The secretaries were previously in charge of maintaining the schedule for meeting rooms and multimedia devices at the company. The secretaries see the development of an automated scheduling system as a threat to their job security. Because of the perceived threat, the secretaries are interested in having a large role in maintaining the system. The secretaries do not believe that an automated system will work. Because they have an integral role in the current scheduling system, they see that current system problems are largely due to the engineering staff's irresponsibility in scheduling rooms and devices properly. The secretaries do not see how a computer program would make the engineers more responsible for reserving meeting rooms and devices correctly. Summaries of the conflicts that involve the secretary role appear in Table 3-5.

**Table 3-5 - Initial Secretary Role Conflicts**

Secretary Conflict 1. Believe engineers and company management are irresponsible for reserving meeting rooms and equipment, and are at fault for the failure of the existing scheduling system.
Secretary Conflict 2. Feel that an automated software scheduling system will eliminate secretarial job positions. Want to become the system administrator of the system to maintain job security, yet have limited computer knowledge.
Secretary Conflict 3. Interested in overseeing the scheduling for meeting rooms and multimedia devices. Feel that others should not be able to see the schedule for rooms and devices, but should go through a secretary, as a central agent for reservations. (basically the existing system)
Secretary Conflict 4. Disagree that automatic conflict resolution will work. There are too many practical issues that will make it impossible to implement in real life.
Secretary Conflict 5. Would like to have some control over engineers' schedules, since engineers are too busy to maintain their own schedules.

### 3.4.2. Engineer Role

The engineer in the company is a technical person working at the virtual company. They are not restricted to any particular engineering domain such as software or hardware. The engineer is assigned the role of the idea person in the requirements meetings. Engineers are to introduce as many ideas as possible for system features. The engineer can be thought as a person with a constant brainstorm. They are to help with the evolution of system requirements through the constant identification of novel features for the scheduling system. This of course, will challenge the software engineers to find a way to filter out wish list features, and implement the needed functionality within the fixed project budget. The engineers are frustrated that the secretaries can't keep up with their scheduling requests. This may explain why they often bypass making the proper reservations for meeting rooms and multimedia equipment. The engineers desire an automated system that they can use as needed to schedule company resources. The engineer is a self-sufficient planner and thinker who wants automation just as he designs and develops automated engineering solutions for his clients. Summaries of the conflicts that involve the engineer role appear in Table 3-6.

**Table 3-6 - Initial Engineer Role Conflicts**

Engineer Conflict 1. You think the secretarial staff makes mistakes and loses reservations for meeting rooms and equipment.
Engineer Conflict 2. You don't want management to see or override your schedule. You have had management override your schedule with clients to attend company business meetings in the past, and you want the new scheduling system to prevent this.
Engineer Conflict 3. You want to be able to view the schedules of your peers and subordinates.
Engineer Conflict 4. You favor many interface shortcuts, and usability enhancements to the scheduling system that will cause cost overruns. The management wants to keep development costs low for the system, and therefore limit non-required product features.

### 3.4.3. Administrator (Vice President) Role

The Administrator, also called referred to as a company vice president, represents the member from corporate management who wrote the original scheduling system overview. This person defined the goals for the scheduling system, and recruited the software engineering firm to further specify and develop it. The administrator is

primarily concerned with solving company scheduling and resource problems while staying within a fixed budget. The scheduling system budget, after requirements analysis is 4 full time developers working for 6 months, for a total of 4,160 man hours. The actual dollar figure is not significant for our purposes, but if a cost of \$100 an hour is assumed, then an actual budgetary figure would be \$416,000.

The administrator is not a technical person, although he or she is familiar with technical buzzwords. Administrators do not understand software development, but they like asking for features for the system that they don't fully understand. For example they throw out words such as "Java", "Platform Independent", and others just because they generally think they are good. They don't fully understand the cost implications of these technologies they suggest, but expect the software engineers to develop a system with the latest and greatest features for the agreed upon budget. Summaries of the conflicts that involve the administrator role appear in Table 3-7.

**Table 3-7 - Initial Management Role Conflicts**

Management Conflict 1. Against removal and change of features as defined in the system overview.
Management Conflict 2. Like features that will enhance system functionality and usability, but against anything which will incur increased development costs.
Management Conflict 3. No intention to have secretarial staff serve as system administrators, also no intention to eliminate secretaries, just reassign them to more productive and interesting company tasks.

### **3.5. Requirements Engineer Roles**

Requirements engineers in the study were grouped in teams varying in size of three to five members. Each team appointed a member to serve as the session leader. This person was in charge of floor control during the virtual meetings. With the Centra Symposium system this gave the session leader the power to grant meeting participants permission to talk and control the telepointer. Participants with permission to talk were able to control the web browser during the use of the Web Safari shared web browser. These participants could also control the shared application during application sharing.

Each team of requirements engineers also appointed a note taker for each virtual meeting. The note taker was responsible for recording the significant decisions during virtual meetings. They were to write and distribute notes that would be used for developing the SRS document and for remembering decisions from previous meetings

Finally, the requirements engineering teams also appointed a MOOsburg site developer. This team member was responsible for maintaining the group's collaborative MOOsburg space. They were responsible for posting documents and organizing the space for collaborative activities with the customer.

### **3.6. Team Formation**

Our empirical study has two significant groups of participants: customers and software engineers. The software engineers in the study were students from the graduate level software engineering course. The customers in the study were students from the graduate level computer supported cooperative work (CSCW) course. For the semester during which the study was conducted some students overlapped between both courses. In this case the students participated as software engineers.

As an educational benefit the course project for the graduate software engineering course was to conduct a requirements analysis and produce a Software Requirements Specification document. This is a typical course project for a graduate software engineering class, but in this case the students conducted a *distributed* requirements analysis using the groupware tools provided. They conducted the entire requirements analysis without meeting the customer face to face to discuss and collaborate about the proposed product. All communication between customers and engineers was conducted using the prototype groupware system. Approximately 25% of the course grade was derived from work on the semester long requirements analysis project. We believe the motivation to obtain a high course grade positively effected the participation level of students role playing as software engineers. The influence to achieve a high grade is comparable to the influence of monetary reimbursement that software engineers receive in the workplace.



The students who role-played as customers from the CSCW course were awarded class participation points for their participation in the study. The students received approximately 10% of their course grade based on participation in the study. We believe that since the use of groupware systems was the topic of this study, students studying CSCW would be interested and motivated in participating. In both cases alternative projects were available to students choosing not to participate in the study to obtain similar course credit.

We formed requirements engineering teams by balancing the perceived software engineering and programming experience scores on each team. Students took a demographics survey that asked questions regarding experience with programming and software engineering/ requirements analysis experience. We collected and used data on programming experience because we felt programming experience would influence the creation of quality software prototypes. Each person received a separate score for software engineering and programming experience. Each team was balanced so that each team would have as close as possible to the average group score for programming and software engineering experience; the deviation from the average was minimized for each team.

Three iterations were required to balance the software teams due to students dropping the course. The original course enrollment of thirty-six students was reduced to twenty-six by the third iteration. By the third iteration it was impossible to move most students since they had already committed to a virtual meeting schedule for the semester. This resulted in experience scores of the final teams that were not as closely matched as intended but it is interesting to report the data so we can compare the difference between final group performance and self-perceived engineering/programming experience. The deviation range for self-reported software engineering experience was  $-10.67$  to  $9.83$ . The deviation range for self-reported programming experience was  $-13.75$  to  $8.25$ . Final requirements engineering teams ranged in size from 3 to 5 members. (Table 3-9 shows team size data.)

**Table 3-8 – Self-Reported Software Engineering and Programming Experience of Requirements Engineering Teams. Points indicate relative experience level.**

<b>Student</b>	<b>Engineering Experience 0 minimum, 39 maximum Points indicate relative experience</b>	<b>Programming Experience 0 minimum, no maximum Points indicate relative experience</b>
<b>TEAM 1</b>		
Student 1	14	7
Student 2	17	22
Student 3	6	7
Student 4	0	2
<b>Total Experience</b>	<b>37</b>	<b>38</b>
<b>Deviation</b>	<b>-8.66667</b>	<b>-13.75</b>
<b>TEAM 2</b>		
Student 5	12	8
Student 6	14	15
Student 7	9	8
<b>Total Experience</b>	<b>35</b>	<b>31</b>
<b>Deviation</b>	<b>-10.6667</b>	<b>1.25</b>
<b>TEAM 3</b>		
Student 8	9	7
Student 9	4	4
Student 10	26	6
Student 11	13	8
Student 12	0	1
<b>Total Experience</b>	<b>52</b>	<b>26</b>
<b>Deviation</b>	<b>6.333333</b>	<b>-3.75</b>
<b>TEAM 4</b>		
Student 13	9	11
Student 14	7	6
Student 15	16	4
Student 16	15.5	6
Student 17	8	11
<b>Total Experience</b>	<b>55.5</b>	<b>38</b>
<b>Deviation</b>	<b>9.833333</b>	<b>8.25</b>
<b>TEAM 5</b>		
Student 18	11	7
Student 19	10	9
Student 20	10	10
Student 21	9	7
<b>Total Experience</b>	<b>40</b>	<b>33</b>
<b>Deviation</b>	<b>-5.66667</b>	<b>3.25</b>
<b>TEAM 6</b>		
Student 22	13.5	8.5
Student 23	12	7
Student 24	16.5	11
Student 25	10	5.5
Student 26	2.5	2.5
<b>Total Experience</b>	<b>54.5</b>	<b>34.5</b>
<b>Deviation</b>	<b>8.833333</b>	<b>4.75</b>
<b>Total Points</b>	<b>274</b>	<b>178.5</b>
<b>Average Team Scores</b>	<b>45.66667</b>	<b>29.75</b>

**Table 3-9 - Group sizes of Project Teams**

<b>Teams</b>	<b>Number of Engineers</b>	<b>Number of Customers</b>
Group 1 – R&H Associates	4	4
Group 2 – Centra Systems	3	4
Group 3 – Harmony Technologies	5	4
Group 4 – Snafoo Inc.	5	3
Group 5 – Omnilink.com	4	3
Group 6 – Farrago Studios	5	2

For the role of company Administrator (vice president) we sought a participant with “chairman” or “shaper” qualities. (see Table 3-11) The Chairman role is a guiding and controlling leader. The Shaper is a confrontational leader who will be more aggressive and vocal than a chairman.

For the role of engineer we desired a member with plant or resource investigator qualities. The Plant is an innovative problem solver, the kind of team member likely to bring to requirements meetings, wish lists of special features and add-ons to any product. A Resource Investigator is a researcher who brings external information to the team to aide in problem solving.

For the secretary we desired a member with Company Worker, or Completer-Finisher qualities. The Company Worker is a dutiful team member who can be viewed as a cog-in-the-wheel. Company Workers are hard working people who quietly go about their jobs without the aspiration for leadership or the creativity of those desiring long lists of product features. Completer-Finishers are people who try to push jobs through to completion. They are anxious to achieve results through organized painstaking work.

**Table 3-10 - Desired Belbin Roles for Customers**

<b>Customer Role</b>	<b>Desired Belbin Roles</b>
Administrator / Vice President	Chairman, Shaper
Engineer	Plant, Resource Investigator
Secretary	Company Worker, Team Worker, Completer Finisher

**Table 3-11 - Description of Belbin Roles [29]**

<i>Name</i>	<i>Symbol</i>	<i>Behavioral Description</i>	<i>Typical Features</i>	<i>Positive Qualities</i>	<i>Allowable Weaknesses</i>
Chairman	CH	Guiding and controlling leader, knows the members' abilities well	Calm, self-confident, controlled	A capacity for treating and welcoming all potential contributors on their merits and without prejudice. Strong sense of objectiveness	No more than ordinary in terms of intellect or creative ability
Shaper	SH	Demanding, coercing, confrontational leader, pushes for members to excel	Highly strung	Drive and a readiness to challenge inertia, ineffectiveness, complacency or self-deception	Proneness to provocation, irritation and impatience
Plant	PL	Innovator and problem solver, the "idea" member	Individualistic, serious-minded, unorthodox	Genius, imagination, intellect, knowledge	Up in the clouds, inclined to disregard practical details or protocol
Resource Investigator	RI	Contact person for resources external to the team, brings resources into the team	Extroverted, enthusiastic, curious, communicative	A capacity for contacting people and exploring anything new. An ability to respond to challenge	Liable to lose interest once the initial fascination has passed
Monitor-Evaluator	ME	Analyzes, evaluates proposed solutions & choices	Sober, unemotional, prudent	Judgement, discretion, hard-headedness	Lacks inspiration or the ability to motivate others
Company Worker	CW	Implements agreed upon plans	Conservative, dutiful, predictable	Organizing ability, practical common sense, hard-working, self-discipline	Lack of flexibility, unresponsiveness to unproven ideas
Team Worker	TW	Facilitates team functions, mediates issues within the team	Socially oriented, mild, sensitive	Ability to respond to people and to situations, & to promote team spirit	Indecisiveness at moments of crisis
Completer-Finisher	CF	Focuses on details and meeting deadlines	Painstaking, orderly, conscientious, anxious	A capacity for follow-through, perfectionism	A tendency to worry about small things, a reluctance to "let go"

### 3.7. Participant Training and Instruction

Before the start of the experiment participants were given a tutorial on the use of Centra Symposium and MOOsburg. The tutorial consisted of an hour-long demonstration/lecture that demonstrated how to create user accounts and use the applications. Participants were asked to contact the experimenter through email or other means to ask questions or resolve any issues using the system. All participants spent about the first thirty minutes of the first virtual meeting practicing their use of the Centra Symposium system.

Requirements engineers were given an hour-long lecture on requirements analysis techniques, based on the book: "Managing Software Requirements" by Leffingwell and Widrig. [6] Several requirements analysis techniques were described in detail during the lecture. These techniques included: Question/Answer method, interviews, brainstorming/idea reduction, storyboards, use case analysis, and requirements management techniques such as ranking and prioritization of requirements.

Requirements engineering participants were also given a sample requirements engineering lifecycle. (See Figure 3-1, Pg.32) and a guideline for planning the activities of each planned virtual meeting in the experiment. (See Table 3-1, Pg. 33) In addition to the hour-long lecture on requirements elicitation techniques, much of the course content of the graduate level software engineering course covered requirements engineering principles.

To promote communication and instruction a web page was configured to act as a center of information and project documents for experiment participants. All project related documentation, meeting schedules, and group information was available on this website.

Customers were given a system overview and a role description. The system overview was a two-page summary of the functionality of the scheduling system that was proposed. The role description described the role of the participant as a customer of the virtual company. Role descriptions described conflicts among different viewpoints, and general suggestions on how to role-play. To aid the customers in role-playing, and to elicit active participation during virtual meetings the experimenter used private text chat messages to remind customers to bring up points related to their customer role. (See Appendix C)

### **3.8. Experiment Facilities**

Since this empirical study studied distributed work, we knew participants would work on the project from a variety of locations. MOOsburg was provided to support distributed work outside of planned meetings. Participants created MOOsburg user accounts and were able to access the system from any computer with an internet web browser having the Java JDK 1.3 plug in. MOOsburg was intended to support both planned and impromptu synchronous and asynchronous meetings throughout the project. Participants could log on to MOOsburg from their home computer as needed for collaborative work.

Use of the virtual meeting system Centra Symposium was restricted to on-campus computer labs during assigned meeting times. Although there were no technical issues

preventing students from accessing the software from remote locations, they were required to use the provided computer labs for all virtual meetings. This allowed for a controlled environment and for convenient observation by the experimenter. The software engineers used a large computer lab (room 3060) with approximately thirty networked PCs on the third floor of Torgersen Hall at Virginia Tech. The customers used a small computer lab that was connected to the New Media Center on the first floor of Torgersen Hall at Virginia Tech. The experimenter had access to both sides of the meeting by simply going between the labs during the session.



**Picture 3-3 - Torgersen computer lab room 3060**

Due to the physical layout of Torgersen Hall, the customers and engineers would typically use different doors to enter the building and therefore they did not typically encounter one another. Customers were generally allowed and expected to leave the lab facility immediately following the completion of the virtual meeting. This was because the lab, is staffed by university employees, and closes upon the end of the day's sessions. The software engineers were typically detained 5-10 minutes after the completion of each

meeting so that they could ask questions of the experimenter and so they could complete post-meeting surveys.

### **3.9. Survey and Observation Methods**

Several methods of both observation and data collection were used to collect data and measure the performance of groups throughout the empirical study. Surveys collected participant opinions and feedback. Meeting observations recorded software usage trends during virtual meetings. Quantitative data was gathered from analysis of email archives, and the MOOsburg project spaces.

The empirical study used surveys to determine the level of experience on each software engineering team. Measuring the prior experience level will allow us to note how it affects group performance. Software engineers were given two demographics surveys to assess software engineering experience. The first survey focused on programming experience, and software engineering experience. The second survey focused exclusively on experience with requirements engineering.

Throughout the empirical study data were collected from the virtual meetings by using two methods: participant surveys, and observational note taking. At the conclusion of virtual meeting #1, everyone, both customers and engineers, individually completed a common post-meeting questionnaire. This questionnaire surveyed the use of functions of Centra Symposium and MOOsburg, inquired about the subject's participation in the meeting, and queried about factors influencing participation. At the conclusion of virtual meetings #2, and #3 the software engineers as a group completed a post-meeting questionnaire. This survey inquired about interaction between virtual meetings, requirements analysis techniques used, perceived effectiveness of techniques, and how requirements techniques were implemented with the groupware software. There was no post-meeting questionnaire after meeting #4. At the conclusion of the project, an extensive online survey was conducted. The survey asked many questions gathering opinions and feedback covering the following topics: virtual meeting format, participation level, factors influencing participation, effectiveness of groupware tools, effectiveness of

requirements analysis methods, face-to-face contact, Centra Symposium, MOOsburg, and email issues, customer satisfaction, realism, and customer participation.

Virtual meetings were recorded when possible using the recording mechanism within Centra Symposium. These meeting records were made available on the World Wide Web. Recordings allowed the customers and engineers to replay the occurrences of the meetings to clarify issues that were perhaps misunderstood. These recordings allow us to review the proceedings and make additional observations on an as needed basis. Centra Symposium recordings record the voice track and all user interaction with the meeting interface. This includes all user interface activity such as stepping through the slide show, and presentation of documents using application sharing, and shared web-browsing activity. Observational notes were taken for all virtual meeting sessions. The primary observational objective was to note how Centra Symposium was used in the sessions. In addition the software engineers recorded notes and wrote meeting minutes that summarized the requirements decisions for each meeting. These artifacts were collected from the email archives and from each group's MOOsburg project space.

Each project group created a MOOsburg project space to share documents and serve as a place for impromptu meetings. All MOOsburg tools support asynchronous collaboration. This means that all tools retain their previous state after modification. At the conclusion the all MOOsburg project spaces were locked to prevent further modification and to allow the analysis of the state of the spaces.

All email communication between customers and software engineers was facilitated using group list server accounts. Each project group had a list server account, so that when an email message was sent it was copied to all project members both customers and software engineers. We monitored the messages exchanged on each group's list server and created a permanent message archive for each account. Analyzing the email archives can generate quantitative data about the quantity of usage, as well as statistics about the type of messages exchanged.

We collected both a working draft copy, and a final copy of the SRS document. By assessing the SRS documents this provides a source of quantitative data for the empirical study. We can attempt to rate the quality of the documents by applying a series of metrics to measure various aspects. By comparing function points from the



original system overview document given to customers, to the final specification developed by the software engineers we can assess the maturity of the requirements evolution on a group by group basis. We can identify original requirements that are missing and note where requirements have been added, or enhanced throughout the requirements engineering process.

In chapter 4 we present more formally the metrics used to assess SRS quality. In Section 4.3 we shall use SRS quality rankings to correlate with various data observations in an attempt to determine what factors influenced the production of high quality SRS documents. In Section 4.4 we identify which groupware tools were most effective and attempt to assess their impact on the project. In Section 4.5 we identify the successful requirements elicitation techniques and assess their effectiveness for use in the study.

In this chapter, data and results from the empirical study are presented. Section 4.1 presents the metrics used to evaluate SRS quality. In Section 4.2 Research Goal #1 is considered. The performance of the requirements engineering teams is presented and data is examined to determine which factors affected team performance. Section 4.3 presents findings related to Research Goal #2. Data is examined to determine the effectiveness of the groupware tools used in the empirical study. Of interest is how the groupware tools affected the requirements elicitation process. Did the available groupware tools enhance requirements elicitation? Section 4.4 presents findings related to Research Goal #3. Data is presented and examined to determine the effectiveness of requirements elicitation methods for a virtual requirements engineering project. Do particular elicitation methods perform better in a distributed setting? The chapter concludes with a brief look at the importance of customer participation for distributed requirements analysis. A summary of the findings appears in Chapter 5 of this thesis.

### **4.1. Assessing Team Effectiveness**

Six requirements engineering teams conducted a distributed requirements analysis project over the course of about three months during the spring semester 2001. Each group produced both a draft and final version of the Software Requirements Specification document for their respective customer. To measure the effectiveness of each group's efforts an assessment was performed of the final requirements documents to determine their overall quality.

Four different metrics were used to measure the quality of each SRS document. Each metric produced a decimal value from 0 to 1. The decimal value could be considered as a percentage of the overall quality relative to that metric. Once computed, the metrics were combined to produce a composite value that represents the overall quality of the SRS document. This composite value of SRS quality was used to classify the performance of each requirements engineering team.

The remainder of this section presents the four metrics used to measure the quality of the SRS documents. Each metric is described and the process of how to apply it to SRS documents is explained.

#### **4.1.1. Software Requirements Specification Grade**

##### MOTIVATION

Because the development of the SRS document was a course project for the graduate level software engineering course at Virginia Tech, each SRS document received a grade from the course instructor. We believe that grade is an interesting and worthwhile data point in determining overall SRS document quality.

##### APPLICATION

The grade was based on the following criteria:

- o **Student Assessment**

At the conclusion of writing the SRS documents, the students in the Software Engineering class were assigned a document from another group on which they performed a SRS evaluation. Students wrote a one-page summary that described their overall impression of the document. They described: presentation style, organization, effectiveness of the presentation format, sufficiency for guiding a design process, and their overall impression of the SRS. As a secondary task the students identified the requirements and proceeded to classify each of them. They were given a template spreadsheet in Microsoft Excel and asked to populate rows of data describing each requirement in the SRS. The template headers are shown in . Each requirement was to be classified as: Functional, Performance, Interface, Design/Implementation, or Inverse. The requirement types are described in Table 4-2. From the identified requirements students were to identify the defect category and provide a description of the problem. The defect categories included: Ambiguous, Incomplete, Inconsistent, Not Traceable, and

Not Verifiable. Questions for assessing if a requirement exhibits defects are seen in Table 4-3.

**Table 4-1 - Identification and Evaluation of Requirements Table**

Identify And Evaluate All Requirements					
Functional Performance Interface Design/Impl Inverse	List ALL Requirements			Ambiguous Incomplete Inconsistent Not Traceable Not Verifiable	Note Only Those Requirements that have a Problem
Type	Req ID #	Requirement	Defect Category	Problem Description	

**Table 4-2 - Requirement Types**

Requirement Category	Description
Functional	Specifies the functions the software is to perform, and how inputs are transformed into outputs
Performance / Reliability	Specifies each performance requirement in testable and quantitative terms. <ul style="list-style-type: none"> <li>• timing and sizing requirements</li> <li>• sequence and timing of events, including user interaction tolerances</li> <li>• throughput and capacity requirements</li> </ul>
Interface	Specifies the communication protocol in terms of: <ul style="list-style-type: none"> <li>• data transmitted</li> <li>• sequencing of data</li> <li>• (send, response) action pairs</li> </ul>
Design Constraints	Conditions imposed due to resource constraints, environment impositions and design preferences
Inverse	Describes constraints on allowable behavior in terms of what must never happen <ul style="list-style-type: none"> <li>• safety and security requirements</li> </ul>
“ilities”	Non-time related, non-functional requirements that software must meet (project specific) <ul style="list-style-type: none"> <li>• maintainability</li> <li>• portability</li> <li>• usability, etc...</li> </ul>

**Table 4-3 - Questions for Assessing Requirement Defects**

<b>Requirement Defect Type</b>	<b>Questions for assessment</b>
Ambiguous	<ul style="list-style-type: none"> <li>• Does every requirement have only one interpretation?</li> <li>• Are requirements clear and specific to be basis for detailed design specs and functional test cases?</li> <li>• Does SRS contain only necessary implementation details and no unnecessary details? Is it over specified?</li> </ul>
Incomplete	<ul style="list-style-type: none"> <li>• Does SRS include all user requirements (as defined in concept phase)?</li> <li>• Do functional requirements cover all abnormal situations?</li> <li>• Have temporal aspects of all functions been considered?</li> </ul>
Inconsistent	<ul style="list-style-type: none"> <li>• Is there internal consistency between software requirements?</li> <li>• Is SRS free of contradictions?</li> <li>• Are specified models, algorithms, and numerical techniques compatible?</li> <li>• Does SRS use standard terminology and definitions throughout?</li> </ul>
Not Traceable	<ul style="list-style-type: none"> <li>• Is there traceability back to previous specification level?               <ul style="list-style-type: none"> <li>○ system concept/requirements and user needs as defined in concept phase, and system design</li> </ul> </li> <li>• Is SRS traceable forward through successive development phases?               <ul style="list-style-type: none"> <li>○ into the design, code, and test documentation</li> </ul> </li> </ul>
Not Verifiable	<ul style="list-style-type: none"> <li>• Are requirements verifiable?               <ul style="list-style-type: none"> <li>○ can software be checked to see if requirements have been fulfilled?</li> </ul> </li> <li>• Are mathematical functions defined using notation with precisely defined syntax and semantics?</li> <li>• Is there a verification procedure defined for each requirement in SRS?</li> </ul>

o **Professor's Impression of SRS document Quality**

The course professor assigned an SRS grade after reading the student's assessment of SRS documents, and after reading and reviewing the SRS documents themselves. The professor considered all input and assigned percentage scores that were reflective of the quality of each SRS document.

#### 4.1.2. Measurement of Requirements Evolution

##### MOTIVATION

The identification and development of new requirements is an indicator of SRS document maturity. A document that has a high degree of maturity should show evidence of evolution from the initial requirements. The term “evolved requirement” refers to a software requirement that was generated during the requirements elicitation phase of requirements analysis. These are requirements that were not in the original system overview document provided to customer participants. As the definition of a software product matures, stakeholders in the requirements elicitation process begin to visualize the system in new ways. They are thinking out of the box. Stakeholders realize many new features for the new system. These features range in significance from showstopper requirements crucial for system deployment, to interface shortcuts to satisfy advanced users. One challenge with conducting requirements elicitation is determining when all of the requirements have been identified [6]. Mature SRS documents should have a higher quantity of evolved requirements because they are more fully developed such that nearly all-necessary system requirements have been identified.

##### APPLICATION

In determining the quantity of evolved requirements in the SRS documents we assessed each document on a bullet-by-bullet basis. Each bulleted or numbered item in the document was read and classified to be one of the following types: Original Requirement, Original Requirement with Evolution, Evolved Requirement, or Vague Requirement. Enumerating requirements on a bullet-by-bullet basis helped to assure a consistent measurement of requirements evolution that did not produce biased results.

**Table 4-4 - Requirement Types for Evolution Metric**

<b>Requirement Type</b>	<b>Description</b>
O: Original Requirement	A bulleted or numbered item is labeled as “O” if it is either: <ul style="list-style-type: none"><li>• Complete Original Requirement from Customer’s System Overview document</li><li>• Partial Original Requirement from Customer’s</li></ul>

	System Overview document. A original requirement can contained modified details from the requirement as stated in the Customer’s System Overview document.
OE: Original Requirement with Evolution	A bulleted or numbered item is labeled as “OE” if it is: <ul style="list-style-type: none"> <li>• Original Requirement with evolved additions</li> <li>• Partial Original Requirement with evolved additions</li> <li>• Original Requirement with changes</li> </ul>
E: Evolved Requirement	A bulleted or numbered item is labeled as “E” if it is a completely original requirement that was not mentioned in the Customer’s System Overview document.
V: Vague, Unclassifiable	A bulleted or numbered item is labeled as “V” if it is so ambiguous that it can’t be classified by one of the other types.

After each requirement in a document was classified the total requirements of each type was computed. The total number of requirements with evolution was calculated by adding the scores for OE and E requirement counts. The document parsing method performs differently based on document style because it affects the total number of identified requirements. If the authors of a particular document did not divide sections frequently this could result in fewer enumerated requirements than a group who used many document divisions. To compensate for this effect we divide the raw count of OE plus E by the total number of identified requirements. This results in the percentage of total requirements with evolution. This number represents a maturity rating of an SRS document, and it is the value obtained by applying the metric described here.

Document Evolution =	$\left[ \frac{OE_{TOTAL} + E_{TOTAL}}{REQ_{TOTAL}} \right]$
	OE = Original Requirement with Evolution
	E = Evolved Requirement
	REQ <sub>TOTAL</sub> = Total Requirements identified

### **4.1.3. Requirements Identification and Evaluation Metric**

#### **MOTIVATION**

The requirements identification and classification metric discussed in Section 4.1.1 was used as a third metric to assess SRS quality. This metric classifies requirements based on defect type. This identifies the clarity and completeness of requirements in the document. A metric that attempts to measure the clarity and completeness of requirements is important when assessing overall SRS document quality.

#### **APPLICATION**

To apply this metric the document was parsed on a bullet-by-bullet basis. Each bulleted or numbered item in the document was read and enumerated as a requirement. When numbered or bulleted items contained multiple requirements the statements were separated and counted as unique requirements. This is different than the parsing method used for measuring document evolution. Here text below each document division was considered as individual requirements when necessary. The requirements engineers would frequently supply several requirements under a single document bullet. In many cases these requirements were unique features that should have been presented as different requirements in the document. This document parsing method produced a total higher count of requirements per document. Each identified requirement was classified by type as described in Table 4-2. Next the requirements were classified as either defect free or as containing one or more of the following defects: Ambiguous, Incomplete, Inconsistent, Not Traceable, or Not Verifiable. (See Table 4-3) To calculate a meaningful numeric value to describe SRS document quality the percentage of requirements with defects was calculated. If a single requirement had multiple defects it was not counted twice. By subtracting the percentage from 100 the percentage of requirements without defects was obtained. This quantity, the percentage of requirements without defects, reflects SRS quality.



$$\text{Percentage of requirements without defects} = 1 - \left[ \frac{\text{REQ}_{\text{WITH DEFECTS}}}{\text{REQ}_{\text{TOTAL}}} \right]$$

$\text{REQ}_{\text{WITH DEFECTS}}$  = Count of Requirements with 1 or more defects  
 $\text{REQ}_{\text{TOTAL}}$  = Total Requirements identified

#### 4.1.4. Original Requirements Supported Metric

##### MOTIVATION

The final metric used in determining overall SRS quality measures how many of the original system requirements introduced to the customer participants in the System Overview document at the start of the empirical study were included in the final SRS document. The System Overview document identified the major system functions that were required for the scheduling system (See Appendix B). It was considered to be “the original system proposal.” The customers consulted this document frequently in guiding the requirements analysis. These requirements should have been both identified and supported as originally stated, or identified and modified through evolution. If groups completely missed identification of these original requirements this was considered as a communications failure between customers and engineers. Although the causes of the failure may vary the argument here is that communication failures reflect poor quality requirements elicitation.

##### APPLICATION

To simplify the recognition of the original requirements when reading the SRS documents the function points of the System Overview document were first enumerated. The enumerations are shown in the table below.

**Table 4-5 - Original System Requirements from System Overview Document**

<b>Detail / Requirement</b>	<b>Description</b>
D1	System is for small company, 100 people
D2	Company is a multi-discipline engineering firm
D3	Company roles are: engineers, project managers, secretaries, accountants, admin personnel
D4	Company has a 2 story office building, fixed number of meeting rooms and a/v equipment
D5	Scheduling system will be a web based system with a client interface that runs on any computer over the corporate intranet.
D6	Users will have different permission levels.
D7	All users can view the schedule of a meeting room or device.
D8	A User can view an individual person's schedule if they are a higher ranking employee on the corporate management flow chart.
D9	All users can schedule rooms, devices, and request employees to attend meetings.
D10	Administrative users can schedule meetings and override all fellow employees schedules so they must attend. High Power administrative users can override conflicts as needed.
D11	Events are scheduled by the month, day, and year.
D12	A user scheduling a meeting can request that an employee attend the meeting.
D13	A user scheduling a meeting can request a meeting room(s) for the meeting.
D14	A user scheduling a meeting can request audio/visual devices and equipment for the meeting.
D15	The system reports scheduling conflicts between events concerning people, rooms, and devices.
D16	The system performs automatic conflict resolution on request.
D17	System has a facility for adding/editing information about employees, meeting rooms, and devices.

The SRS documents were then parsed to identify these original requirements in any form. Statements in the document were labeled as either being one of the original requirements, marked by number such as “D5” or they were marked as an evolved requirement. Evolved requirements were labeled with an “E”. After a complete search of the document the number of original requirements supported was tallied. For the total requirements count partial support of a requirement was counted as ½ of a requirement regardless of the magnitude of partial support. To determine a final value for this metric

the total number of requirements supported was divided, by the total number of requirements in the System Overview document. There were 17 original requirements.

$$\text{Percentage of Original Requirements Supported} = \left[ \frac{\text{REQ}_{\text{TOTAL ORIG SUPPORTED}}}{\text{REQ}_{\text{ORIGINAL}}} \right]$$

$$\text{REQ}_{\text{ORIGINAL}} = 17$$

#### 4.1.5. Calculating the Overall SRS Document Quality

Once a value was determined for each metric a value that represents the overall quality of a SRS document can be computed by adding the value of all of the metrics together to obtain a total quality value. To compute the value as a percentage the total is divided by four. The value obtained represents our measurement of SRS quality, which is used for the presentation, and analysis of the results.

$$\text{Overall SRS Quality} = \frac{\text{SRS Grade} + \text{Document Evolution} + \text{\% of requirements without defects} + \text{\% of Original Requirements Supported}}{4}$$

## 4.2. Results of Team Performance: Overall SRS Quality

This section presents the results describing the overall SRS quality for groups. Next, correlations are presented that suggest what factors may have led to higher quality SRS documents for some requirements engineering teams, and lower quality SRS documents for other teams. Research Goal #1 of this thesis is to determine what factors in the experiment were responsible for impacting the quality of the SRS documents

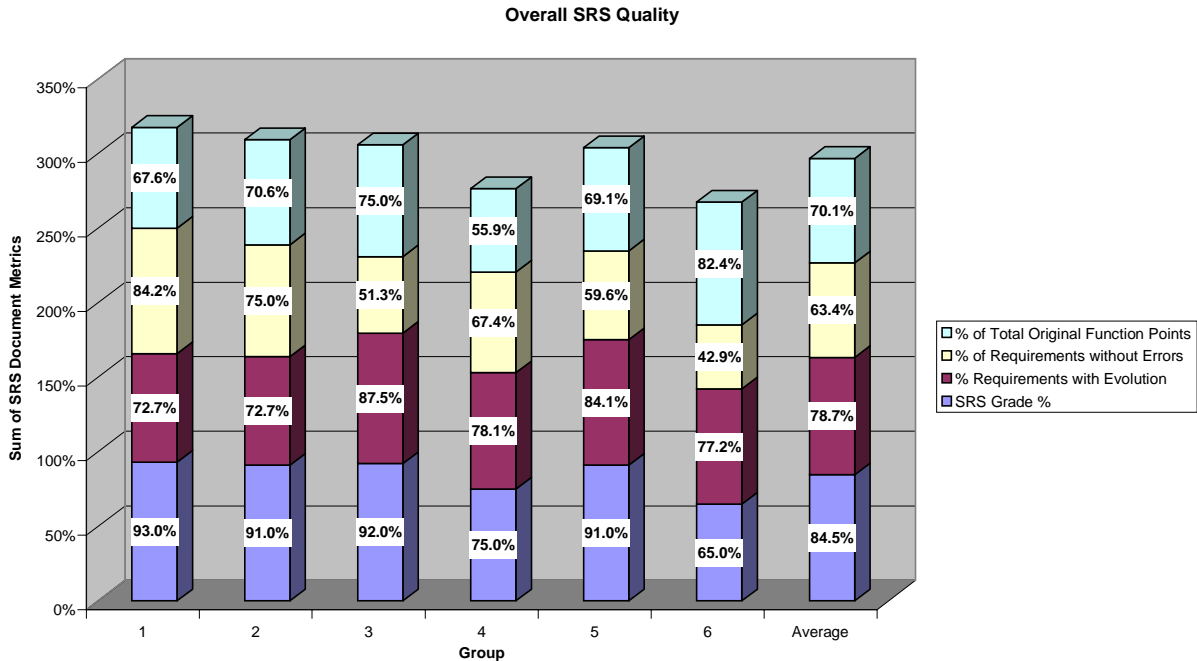
produced. We shall present and consider factors such as: customer participation, software engineering experience of teams, requirements analysis experience, elicitation method effectiveness, experimental observations, and requirements engineer team perceived contributions of teammates.

Each Requirements Engineering team can be categorized as either exhibiting high performance or reduced performance. Group performance is rated by applying the metrics described in Section 4.2 to rate the SRS documents produced by each group. Four groups were classified as high performance groups with SRS document quality ratings having a range of 3.4%. Two groups were classified as reduced performance groups with SRS document quality ratings having a range of 2.3%. The high and reduced categories have a 6.9% separation between their ranges. The total spread between all groups for overall SRS quality is 12.5%. We believe that a 6.9% separation between groups when the total spread is only 12.5% is large enough to make our classifications distinct. (See Table 4-6)

Groups 1, 2, 3, and 5 are classified in the high performance group. Groups 4 and 6 are classified in the reduced performance group.

**Table 4-6- Overall SRS Quality Metric Scores**

<b>High Performance Group</b>		<b>Reduced Performance Group</b>	
Group 1	79.34%	Group 4	69.11 %
Group 2	77.32%	Group 6	66.85 %
Group 3	76.44%		
Group 5	75.96 %		



**Figure 4-1 – Contribution of Individual Metrics on SRS Quality for Groups**

Figure 4-1 – Contribution of Individual Metrics on SRS Quality for Groups depicts the contribution of each metric to the overall quality rating. What factors caused the high performance groups to perform well? What factors caused the high performance groups to out perform the reduced performance groups? What factors were responsible for reduced SRS Quality in the reduced performance groups? The remainder of this section considers these questions.

For exploring the factors contributing to overall SRS quality the analysis is limited to a sample size of six, since only six groups participated in the study. Correlations based on a sample size of six have only four degrees of freedom (df). With  $df=4$  a correlation must be quite high, ( $r=\pm 0.811$ ) to be statistically significant at  $p<.05$ .

#### **4.2.1. Customer Participation versus SRS Quality**

Analysis starts by looking at customer participation to see if there is a correlation between customer participation and Overall SRS Quality. The requirements engineers were asked about their perception of customer participation.

(See final survey in Appendix D) Requirements engineers were queried in the final survey to determine their perception of customer participation during the planned virtual meetings, outside of the virtual meetings and overall for the entire project. For the survey a standardized likert scale was used with values from 1=low participation to 5=high participation. N/A was an available choice if the question was not applicable which resulted in a participation value of 0.

Did customer participation play a role in the quality of SRS documents produced? This seems possible because requirements engineers are dependent upon informational interchange with their customers. In the distributed setting, the quantity and quality of this interchange should be important. There was a weak positive relationship between group perceived customer participation and overall SRS quality. There was a weak but interesting negative relationship between average self-reported requirements engineer member participation by group versus overall SRS quality. This trend may mean that the engineers perceived themselves as working more diligently when the overall SRS quality was lower. This suggests that group members in the reduced performance groups perceived less customer involvement in the requirements engineering process. They also perceived themselves as working harder than their counterparts in the high performing groups.

Requirements Engineers were asked “What Factors made the simulation unrealistic?” Reduced performance groups responded with 5 negative comments concerning lack of customer participation from 9 possible respondents. High performance groups responded with only 2 negative comments from 17 possible respondents. Here are some sample comments regarding customers participation from reduced performance groups:

"Lots of arguing amongst customers. I expect there would be some in real life but somehow I think especially at the beginning, ours went overboard. ... Also, I wonder about the lack of response to our e-mail. If we were building this software for a company in real life, you think they'd be more interested about responding to our questions in a timely manner." *Group 6 participant*

"They didn't care about the agenda, and they clearly had no communication about anything among themselves. I would

assume that things like what users should be able to override who is something you decide on your own time instead of us being given conflicting requirements and then the VP just sits idly by without commenting. " *Group 4 participant*

"the customer group didn't care as much as real customers would have, our group was not as pro-active and didn't put as much effort into their feedback." *Group 4 participant*

This suggests that the reduced performance groups were not obtaining desired results from their customer interaction. They stated they were not satisfied with customer involvement and they believed that they were working harder than their high performance counterparts in the class, possibly to counteract the lack of customer/engineer interaction. One requirements engineer from a reduced performance group asked this question prior to the SRS draft review meeting: "Will the customer now be able to say 'yes' or 'no' without (us performing) the same level of detailed interrogation we have been doing up until now?" This statement suggests that this engineer was expecting the customer group to simply tell them what the system was to do. This is the same naïve view of requirements engineering that is described in [5]. It seems that this engineering team member expected the requirements to be "immutable facts" which were preexisting, and that the process of requirements analysis should simply be to mediate a one-way transfer of information from the customers to the engineers. The requirements engineering participants were aware that there existed a System Overview document, so they may have believed that a quality SRS document was a document that merely echoed this original document.

A number of questions from requirements engineers inquired whether all of the relevant system functionality had been captured. These questions could not be answered because requirements are a negotiated product that results through collaborative requirements analysis. These engineers were experiencing the "Undiscovered Ruins" Syndrome [6]. They couldn't determine when they had captured all system requirements. This suggests that the engineer's expectations of customer interaction may have influenced their view of customer participation. This in turn suggests that overall SRS quality may not be correlated with perceived customer participation because this measure doesn't accurately reflect the customer's actual participation. When customer's self-

assessment of participation is correlated with overall SRS quality there is a weak negative correlation. This correlation is not significant, however it is consistent with the notion that for customer groups who reported less participation the overall quality of SRS documents from these groups was reduced.

#### 4.2.2. Engineer Experience versus SRS Quality

Another interesting trend can be seen between perceived software engineering experience and team performance for each of the SRS metrics. As described in Section 3.3 a self-survey of each participant’s software engineering experience and programming experience was conducted. This data was used to attempt to balance requirements engineering teams to have similar levels of experience.

**Table 4-7 - Group Scores for Software Engineering and Requirements Engineering Experience**

Groups ordered by performance	Requirements Eng. Experience Exp score, diff. from mean	Software Eng. Experience Exp score, diff. from mean
Group 1	37, -8.7 (-19%)	2.4, -.2 (-6%)
Group 2	35, -10.7 (-23%)	3.1, +.5 (+21%)
Group 3	52, +5.33 (+14%)	2.4, -.2 (-6%)
Group 5	40, -5.7 (-12%)	3.0, +.4 (+16%)
Group 4	55.5, +10.17 (+22%)	2.3, -.3 (-12%)
Group 6	54.5, +9.17 (+19%)	2.2, -.4 (-13%)

The reduced performing groups had an average software engineering experience (SEE) score of 55 with a standard deviation of 0.5. The high performing had an average SEE score of 41 with a standard deviation of 6.6. There is a marginally significant negative correlation between the SRS error metric and software engineering experience. ( $r[df=4]=-0.70, p < .12$ ). Another marginally significant negative correlation is between SRS document grade and software engineering experience. ( $r[df=4]=-0.73, p < .10$ ). A significant negative correlation is between group Overall SRS quality and software engineering experience. ( $r[df=4]=-0.81, p < .05$ ) Why does software engineering experience seem to negatively affect SRS document quality? There was only a weak negative relationship between programming experience and overall SRS quality. A display of the magnitude of experience in Software Engineering and Requirements



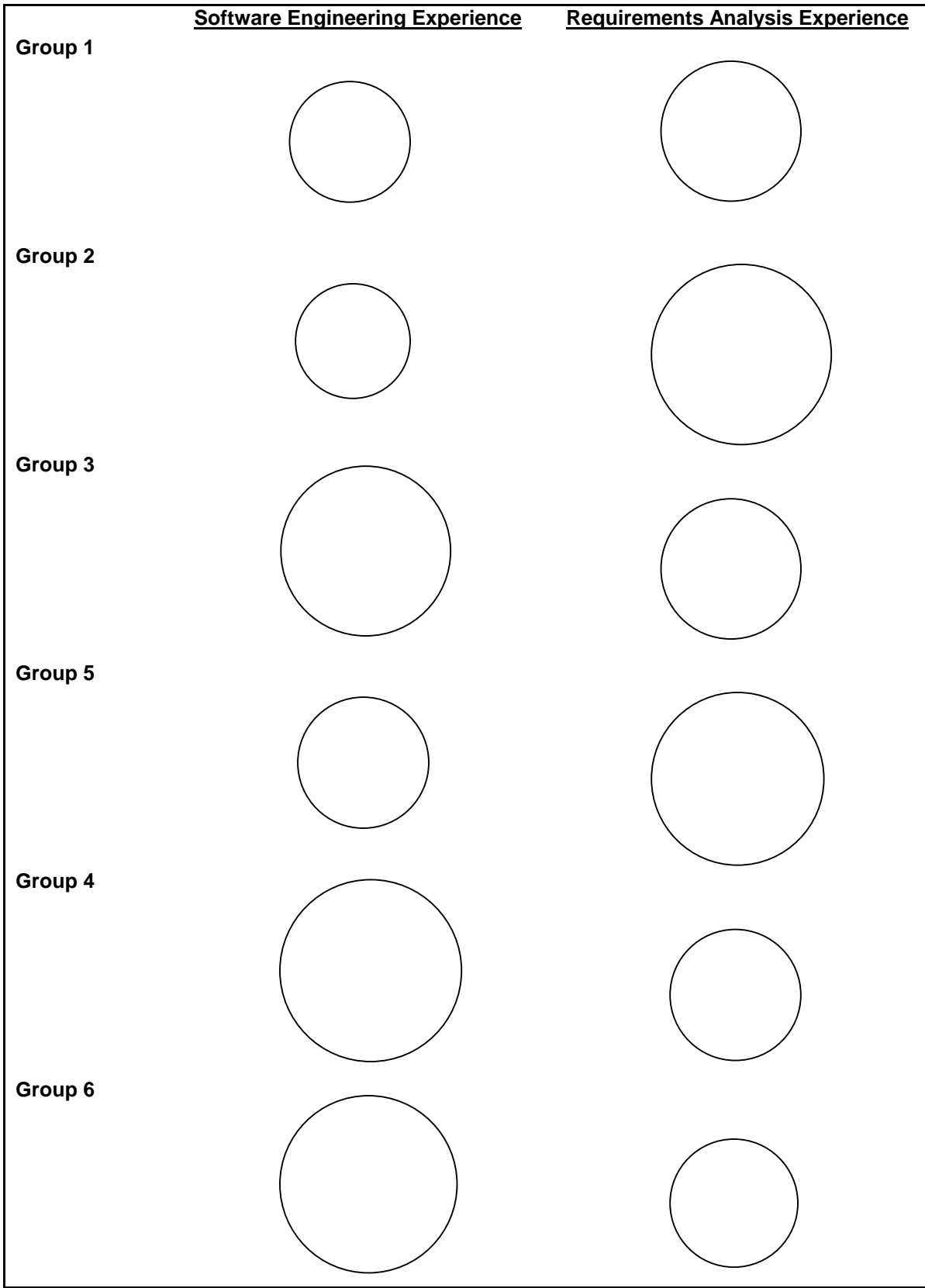
Engineering for all groups is displayed visually in **Error! Reference source not found.** In this figure a requirements engineering group with an average experience value is shown using a circle with a 1-inch diameter. Starting at one inch the percentage difference from the average is subtracted/added to determine the diameter of the circle.

Other data suggests a possible explanation for the negative correlation between SRS quality and team software engineering experience. If team requirements elicitation experience (REE) scores are compared with SEE scores there is a marginally significant negative correlation. ( $r[df=4]=-0.77, p<0.08$ ) This indicates that the teams that had more software engineering experience had less requirements analysis experience. When comparing REE scores with overall SRS quality there is a weak positive relationship. ( $r[df=4]=0.56, p < .25$ )

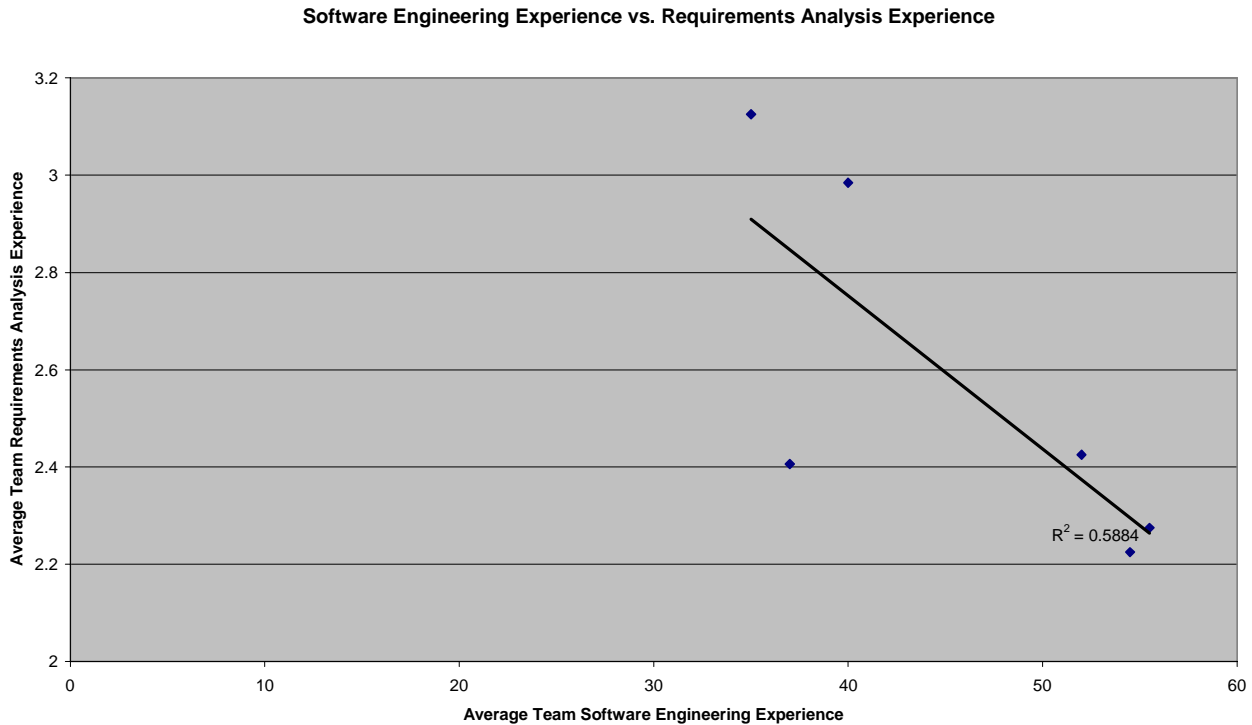
This suggests an explanation for why teams with high software engineering experience performed poorly on overall SRS quality. These teams may have had software engineering experience, but they lacked experience with requirements elicitation. As a result the teams formed in the experiment may have exhibited the “User and the developer” syndrome [6]. This syndrome is defined as the communication gap between the user and the developer. This is caused because users and developers speak from different worlds of thought, have different backgrounds, motivations and objectives. This syndrome applied to this observation suggests that teams of experienced software engineers had less experience with requirements engineering and therefore they exhibited this communication gap more profoundly. The effect of this syndrome carries through to overall SRS quality ratings, because groups fitting this description produced lower quality SRS documents.

If an individual's REE score is correlated with their SEE score there is a marginally significant positive correlation. ( $r[df=24]=0.37, p < .06$ ) This indicates that participants with requirements elicitation experience also tended to be experienced in software engineering. On the SEE survey approximately four questions were related to requirements elicitation and the remaining 9 questions were about general software engineering activities. So most experienced software engineers had dealt with requirements analysis to some degree,  $r=0.37$ . There was a significant positive correlation

between individual SEE score and programming experience (PE) score. ( $r[df=24]=.40, p < .04$ ) The relationships between SEE and REE and PE and REE are displayed in .



**Figure 4-2-** Relative Size of Bubble Indicates Magnitude of Experience

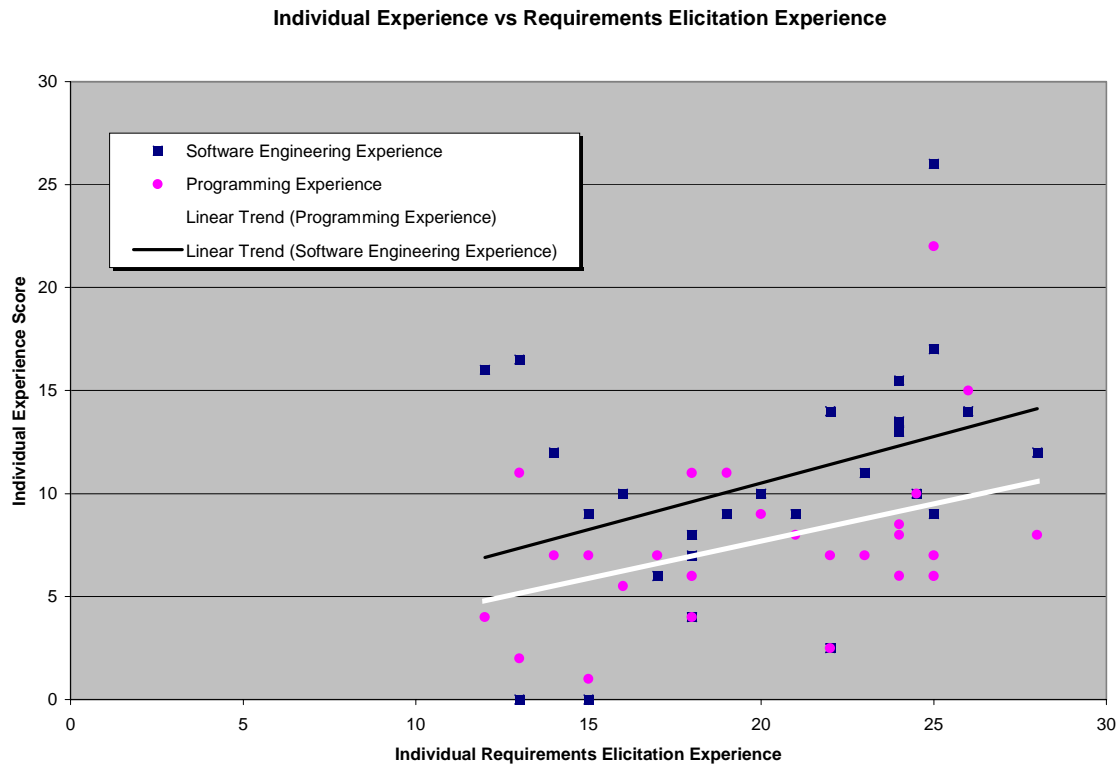


**Figure 4-3 - Team Software Engineering Experience vs. Team Requirements Analysis Experience**

There was a marginally significant negative correlation that teams formed in this experiment lacked REE with respect to SEE. However for requirements engineering participants there was a significant positive correlation between REE and SEE. Considering both correlations the data suggests that typically participants with reported SEE had some experience with REE. It is unusual that the teams formed were able to exhibit a negative correlation between SEE and REE. I attribute this to random chance.

**4.2.3. Peer Participation versus SRS Quality**

One final remaining variable seems to be associated with overall SRS quality. After the conclusion of each virtual meeting the software engineering team members were asked to rate the contribution and participation of their peers for both the virtual meeting and all project related work leading up to the meeting. First all available scores for perceived peer participation per meeting were averaged. Then these scores were



**Figure 4-4** - Individual Correlations with Requirements Elicitation Experience

averaged yielding the average perceived peer participation per group (APPPG). Thus the final set of numbers describes everyone’s perception of how much contribution and participation was occurring from the group members on each requirements engineering team. This measure has a weak positive relationship with overall SRS quality ( $r[df=4]=.60, p < .21$ ) and a marginally significant positive relationship with SRS correctness ( $r[df=4]=.71, p < .11$ ) and with SRS Grade ( $r[df=4]=.65, p < .16$ ). This suggests that participants had some perception of how well the group was performing. Participant’s perceptions of their peers’ efforts throughout the project closely matched the grade they received on the SRS document. Groups who believed that peer participation was good performed well. Groups who believed perceived their peers were not participating as much as they expected performed more poorly. One possibility is that there were conflicts within the reduced performance groups that were not present in the high performance groups. The two reduced performance groups were the only groups who reported intra-group conflicts. Group 6, the lowest performing group, was the only

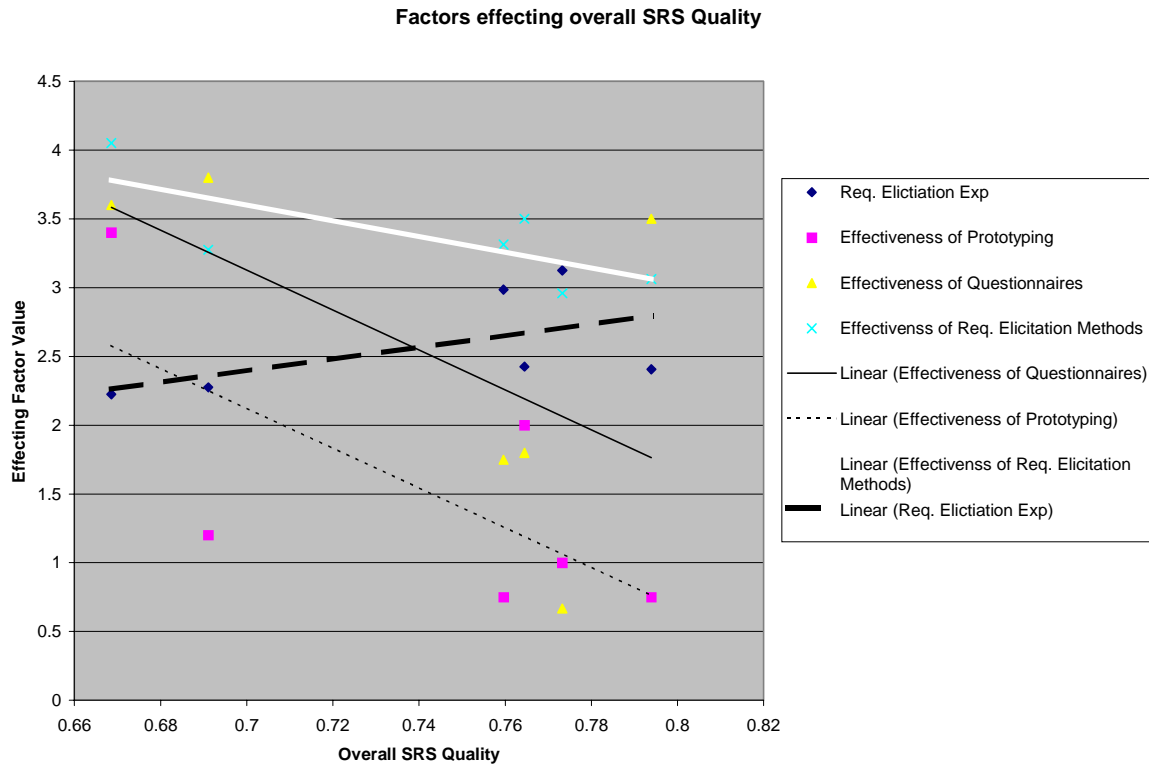
group with requirements engineers who missed a scheduled virtual meeting with the customer. Observations and discussions with group members, and with the course professor indicated that both of the reduced performance groups exhibited leadership differences from the high performance groups. These groups faced clear leadership challenges due to their personality and experience of the team members. These factors created conflicts between group members. Observations suggest that intra-group challenges contributed to reduced performance. These findings help to explain some of the problems experienced by the reduced performance groups, and at the same time encourage analysis of the methods and groupware that might alleviate these problems.

#### **4.2.4. Requirements Elicitation Methods versus SRS Quality**

There was a marginally significant negative correlation between requirements elicitation technique effectiveness (RETE) and overall SRS quality. ( $r[df=4]=-0.74$ ,  $p < .09$ ) This correlation will be discussed in detail in Section 4.4 on requirements elicitation methods.

In this case RETE refers to the average reported effectiveness score for all requirements elicitation methods. This correlation implies that groups who developed a high quality SRS document perceived that the requirements methods were ineffective. When scores for individual methods are examined only two elicitation methods have significant correlations with SRS quality: Prototyping and Questionnaires. From observations of the virtual meetings, prototyping was never truly used as an elicitation technique. Storyboarding, a similar technique to prototyping was used instead by groups in virtual meetings. As for the negative correlation with Questionnaires and overall SRS quality, there is only a weak negative relationship present. ( $r(df=4)=-0.56$ ,  $p < .24$ ) This suggests groups who used questionnaires as an elicitation technique produced lower quality SRS documents. This claim will be examined further in Section 4.4.

In summary in this distributed requirements engineering experiment, groups produced higher quality SRS documents if they had more experience with requirements analysis and requirements elicitation techniques. Groups who perceived their teammates as contributing to the group seemed to produce better quality SRS documents. Groups produced lower quality SRS documents if they used questionnaires as a requirements



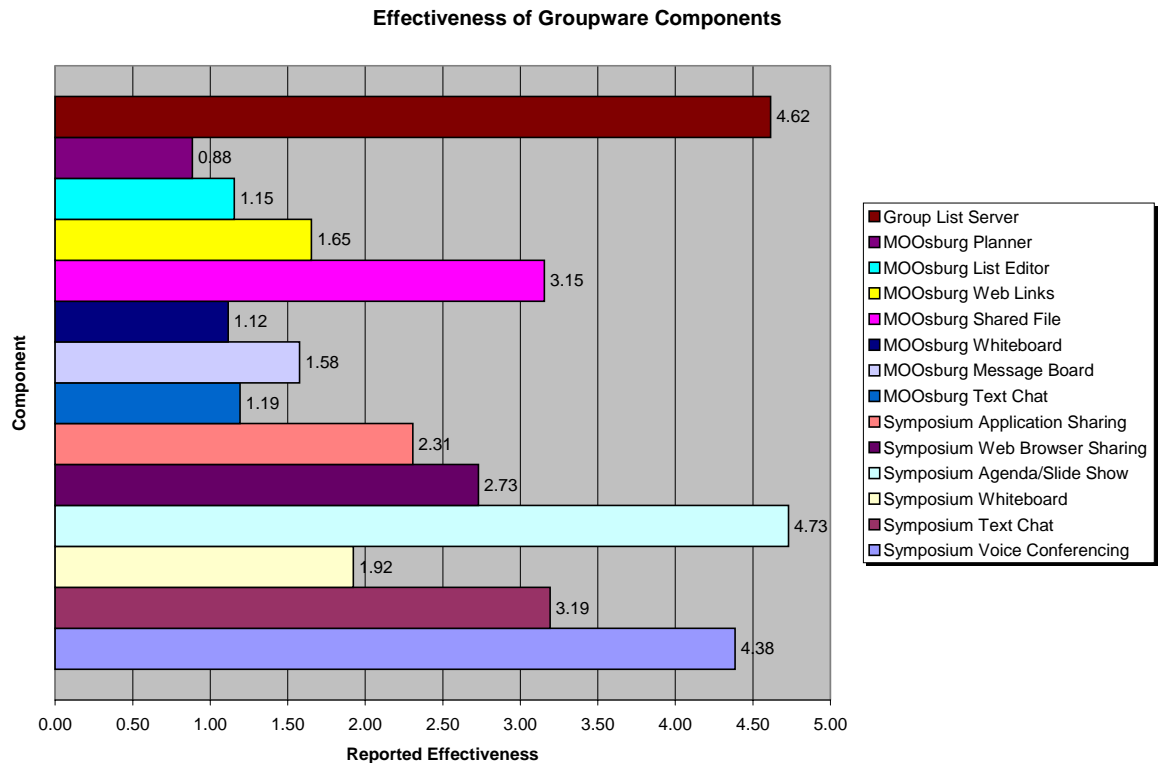
**Figure 4-5 – Several Factors Effecting Overall SRS Quality**

elicitation technique, a trend that is discussed further in Section 4.4. Analysis of the available data and observations fails to describe any interesting relationship between customer participation and overall SRS quality.

Finally, the negative correlation between SEE and overall SRS quality can be dismissed. This is because there was a marginally significant negative correlation between team average SEE score and REE score, and there was a weak positive relationship between team average REE score and overall SRS quality. This suggests that teams with higher REE scores performed better, a reasonable assumption. Teams with higher SEE scores performed worse, an unreasonable assumption. Since there was a marginally significant positive correlation between individual reported SEE score and REE score the conclusion that teams with higher SEE scores produce lower quality SRS documents does not hold. An unexpected distribution of SEE and REE across the teams produced the observed result. (See **Error! Reference source not found.**)

### 4.3. Groupware Tools

Research Goal #2 of this empirical study was to identify groupware tools which best support distributed requirements engineering. The intent is to determine what functionality is needed for a groupware system to support distributed requirements engineering. This section presents the results of the empirical study that suggests the performance of groupware tools used in this study. The impact of groupware tool usage on SRS quality is considered as is the usefulness of specific tools, how participants in the experiment utilized tools, and what participant feedback was provided.



**Figure 4-6 - Requirements Engineers Reported Usefulness of Groupware Tools**

Figure 4-6 presents bar graphs that describe which of the groupware tools were rated as most useful, if used in the study. As seen in Figure 4-6, the top four groupware tools when rated for usefulness by the requirements engineers were: Symposium



Agenda/Slide Show, Group List Server, Symposium Voice Conferencing, and Symposium Text Chat. The customers choose the same four groupware tools as being the most useful, except they ranked them in the order of: Symposium Voice Conferencing, Symposium Agenda/Slide Show, Symposium Text Chat, and the Group List Server. Tool usefulness was rated with the following likert scale: 1= Disruptive, 2= Not Useful, 3= Possibly Useful, 4= Useful, 5= Very Useful, 0- Not Used.

#### **4.3.1. MOOsburg Effectiveness**

MOOsburg, a geographically oriented room based collaboration tool was used very little by participants in the experiment. Most participants reported that they primarily used MOOsburg as a file repository for sharing documents. In fact all of the experiment participants reported that no synchronous virtual meetings were conducted in MOOsburg. (even though this was repeatedly suggested as a method to supplement the planned virtual meetings) The experiment participants were asked whether they would use MOOsburg if they needed to perform a requirements analysis in the future, their response was 37 to 9 against using MOOsburg for requirements engineering. Email appears to have been the most frequently used asynchronous tool. It may be that the familiarity of email was so great that participants had no desire to learn how to use the novel MOOsburg environment. The MOOsburg application supports the same room-based navigational metaphor as Team Wave, which was used to enable distributed requirements analysis in [5]. Both of these systems allow the user to create rooms with collections of shared collaborative objects that can be used by visitors to the rooms.

Other factors that may have discouraged MOOsburg use were the relative maturity of the application, and its lack of integration with the user's working GUI environment. To access MOOsburg users need to explicitly login and launch the Java application each time they wanted to use the system. Several performance issues cause the MOOsburg client to consume memory resources making it prohibitive to keep the application running at all times.

### 4.3.2. Tool Effectiveness and Subject Participation

More detailed analyses examined the relationships between reported tool effectiveness and other variables. Tool effectiveness was calculated by combining the tool effectiveness ratings for each individual groupware tool assigned by all of the requirements engineers in the study. Customers' ratings of the tools were not included in the calculation because the customers had a smaller stake in the success of the project. Because less of the customers' course grade was based on the project, the requirements engineers' reports can be taken more seriously in analyses involving overall SRS quality. When considering all 14 groupware tools there was a strong positive correlation between the customers' and engineers' reported tool effectiveness. ( $r[df=12]=.89$ ,  $p < .00002$ ) This correlation shows that the customers and engineers generally agreed on tool effectiveness, so it seems safe to discount customer feedback for correlations with SRS quality.

There was no observed relationship between reported tool effectiveness per group and overall SRS quality. This implies that requirements engineers did not report groupware tool effectiveness differently based on the quality of the SRS document they developed.

There is a positive correlation between perceived customer online participation and average reported tool effectiveness. ( $r[df=24]=.49$ ,  $p < .025$ ) This finding suggests that the requirements engineers may have believed that the groupware tools were more effective when they perceived more active participation by their customers in the virtual meetings. Were the tools causing the customers to be more active? Or were engineers merely happier with the tools because they believed that the customers were actively participating? This trend also implied that the requirements engineers perceived the groupware tools as being less effective when they perceived that the customers were less active in the virtual meetings. Table 4-8 shows the correlations between the requirements engineers' perception of customer participation and the effectiveness of groupware tools. For the table, shaded  $r$  values have significance of  $p < .05$ . The sample size is 26, with  $df=24$ .

**Table 4-8 - Correlations with Perceived Customer Participation**

<b>Perceived Customer Participation Correlations</b>	<b>r value</b>
Online Customer Participation vs. Reported Groupware Tool Effectiveness	.49
Online Customer Participation vs. MOOsburg Effectiveness	.22
Online Customer Participation vs. Symposium Effectiveness	.39
Online Customer Participation vs. Combined (MOOsburg, Symposium, Email) Effectiveness	.44
Offline Customer Participation vs. Reported Groupware Tool Effectiveness	.07
Offline Customer Participation vs. MOOsburg Effectiveness	.12
Offline Customer Participation vs. Email Effectiveness	.10
Offline Customer Participation vs. Combined (MOOsburg, Symposium, Email) Effectiveness	.33
Overall Customer Participation vs. Reported Groupware Tool Effectiveness	.22
Overall Customer Participation vs. MOOsburg Effectiveness	.09
Overall Customer Participation vs. Symposium Effectiveness	.43
Overall Customer Participation vs. Email Effectiveness	.16
Overall Customer Participation vs. Combined (MOOsburg, Symposium, Email) Effectiveness	.32
r  value required for significance (p<.05)	.388

These data suggest that Symposium played a significant role in how the engineers perceived customer participation. This indicates that when the requirements engineers perceived the customers as actively participating they were more satisfied with the effectiveness of the Centra Symposium tool. An alternative interpretation is that when requirements engineering groups were more effective in using symposium, their customers were able to participate more.

Another analysis examines the correlations between perceived effectiveness of the groupware versus each individual participant's self reported participation level in the project. Did the groupware tool effectiveness impact an individual's participation? Table 4-9 shows the correlations between individual participation and groupware effectiveness. The only significant correlation in the data is between the Customer's rated effectiveness of MOOsburg and self-reported participation. ( $r[df=18]=-0.44, p < .05$ ) This indicates that MOOsburg may have actually negatively impacted customer participation. This can be explained by several reports from customers, that they were unable to get MOOsburg running on their home computers. This suggests that if customers had better access to the MOOsburg application outside of the planned virtual meetings, then they may have participated more in the project.

**Table 4-9 - Groupware Effectiveness on Individual Participation**

Correlations	Everyone r value	Engineers r value	Customers r value
MOOsburg Effectiveness vs. Self-Reported Participation	-.15	-.05	-.44
Email Effectiveness vs. Self-Reported Participation	-.05	-.09	-.09
Symposium Effectiveness vs. Self-Reported Participation	.07	.17	.09
Sample Size	46	26	20
r  value required for significance (p<.05)	.304	.388	.444

Another set of correlations examines the effect of the individual tools on participation. Was there a correlation between reported usefulness of specific tools and individual participation level? Is there a correlation between usefulness of specific tools and perceived customer participation?

**Table 4-10 - Tool Usefulness vs. Individual Participation**

	Everyone Tool Usefulness vs. Self-Reported Project participation r value	Requirement Engineers Tool Usefulness vs. Self-Reported Project participation r value	Customers Tool Usefulness vs. Self-Reported Project participation r value
Symposium Voice Conferencing	0.01	0.29	-0.15
Symposium Text Chat	0.15	0.32	0.09
Symposium Whiteboard	0.17	0.30	0.02
Symposium Agenda/Slide Show	-0.22	-0.14	-0.45
Symposium Browser Sharing	0.18	-0.07	0.18
Symposium Application Sharing	0.27	0.20	0.20
MOOsburg Text Chat	0.04	0.01	-0.01
MOOsburg Message Board	0.08	0.02	-0.09
MOOsburg Whiteboard	0.11	0.19	-0.02
MOOsburg Shared File	0.06	0.14	-0.19
MOOsburg Web Links	-0.02	-0.24	-0.11
MOOsburg List Editor	0.04	-0.14	0.01
MOOsburg Planner	-0.03	-0.19	-0.03
Group List Server	0.26	0.20	0.04
Samples	46	26	20
r  value for significance (p<.05)	.3	.388	.444

Table 4-10 shows correlations between reported tool usefulness and individual participation. There is a significant negative correlation between the customer reported usefulness of the Centra Symposium Agenda/Whiteboard (CRU-AW) and customer reported participation level in the project. ( $r[df=18]=-0.45, p < .05$ ) This correlation indicates that there was a relationship between how useful the Agenda/Slide Show feature seemed versus how much customers participated in the project. When customers

participated less in the study, they felt the agenda/slide show was more useful. One possible explanation was that the Agenda/Slide Show tool was very useful but was causing customers to be more passive with their participation.

**Table 4-11 - Reported Tool Usefulness vs. Perceived Customer Participation**

	Perceived Customer online participation	Perceived Customer offline participation	Perceived Customer Cooperation in Project
Symposium Voice Conferencing	0.32	-0.13	0.14
Symposium Text Chat	-0.17	-0.27	-0.39
Symposium Whiteboard	0.52	0.10	0.26
Symposium Agenda/Slide Show	0.44	0.25	0.14
Symposium Web Browser Sharing	0.62	0.21	0.50
Symposium Application Sharing	0.24	-0.04	0.03
MOOsburg Text Chat	0.06	-0.16	-0.01
MOOsburg Message Board	0.30	0.18	0.22
MOOsburg Whiteboard	0.19	-0.25	0.03
MOOsburg Shared File	0.25	0.20	0.03
MOOsburg Web Links	0.34	0.19	0.08
MOOsburg List Editor	0.06	0.05	0.09
MOOsburg Planner	0.34	0.14	0.26
Group List Server	0.35	0.12	0.28
Samples	26	26	26
r  value for significance (p<.05)	.388	.388	.388

Table 4-11 shows correlations between reported tool usefulness and the requirements engineers' perception of customer participation. The Centra Symposium Whiteboard, Agenda, and Web Browser Sharing were all identified as useful tools in cases where there was an increased perception of customer participation. This means that in virtual meetings these tools were identified as useful when the customers were actively participating. These trends also reflect the engineer's perspective that they were in control. This suggests that each of these tools enhances the quality of the distributed interaction when participants are active. They effectively increase the informational bandwidth of the distributed meeting. There were no significant trends in the relationship of groupware tools usefulness to the perception of offline customer participation.

Centra Symposium text chat was rated a more useful tool when customer participation was lower. (NS) This suggests that when customers were less vocal in meetings, they may have seemed less active. In these cases the engineers needed to rely on text chat. Therefore when the perceived participation of customers decreased the reported usefulness of text chat increased because engineers used it to get information

from the otherwise silent customers. In the virtual meetings this phenomena was observed frequently, particularly in cases where the customer was uncomfortable with the English language. Often in these cases the customers barely spoke in meetings and contributed almost exclusively using the Symposium text chat tool.

In cases where the requirements engineers perceived high overall customer participation, Centra Symposium Web Browser Sharing was identified as a useful tool. There is a significant positive correlation between customer online participation and reported usefulness of Web Browser Sharing (RU-WBS). ( $r[df=24]=.62, p < .01$ ) Web Browser Sharing can be said to increase the informational bandwidth during virtual meetings. Another significant correlation is between RU-WBS and overall customer cooperation in the project. ( $r[df=24]=.50, p < .01$ ) This finding strongly suggests that Web Browser Sharing is a useful tool when customers actively participate in virtual meetings. With Web Browser Sharing customers could control the shared browser, and point at items of interest being displayed. Web Browser Sharing was usually done for the presentation of storyboards or documents. These activities are interactive requirements elicitation methods that encourage customer participation.

### **4.3.3. Factors Influencing Participation**

Experiment participants were asked about which factors influenced their individual participation. The final survey asked participants to identify what factors were responsible for influencing their participation in the project. (see Appendix D) Subjects rated the effect of 16 factors using a standard likert scale from 1 to 5. Table 4-12 shows the correlations between influencing factors and self reported participation level in the project. Values with a probability  $p=.05$  or better are shown in gray. When an influencing factor has a positive correlation with participation then this suggests that as participation level increased, the influencing factor was seen as contributing more to participation. When a negative correlation is seen, it seems that as participation level decreased the influencing factor was seen as contributing less to participation.

**Table 4-12 - Correlations of Influencing Factors on Reported Subject Participation**

Correlations	Everyone r value	ENGRS r value	CUSTOMERS r value
Subject Knowledge	0.18	-0.11	0.20
Preparation Time	0.50	0.13	0.49
Assigned Role	0.38	0.40	0.47
Physical Energy Level	0.30	0.40	0.16
MOOsburg Usability	0.19	-0.07	0.17
Symposium Usability	0.21	-0.27	0.32
Workstation Quality	0.38	0.12	0.35
Arrival Time / Attendance	0.26	0.12	0.25
Quality of Net Bandwidth	0.24	0.00	0.27
My Personality Attributes	-0.01	0.54	-0.29
English Comfort	-0.05	-0.07	-0.19
Well Coordinated Meetings/Agenda	0.40	0.07	0.29
Software Eng	0.19	0.03	0.36
Customers	0.21	-0.18	0.35
Course Professor	0.28	-0.37	0.31
Experimenter	0.13	-0.29	0.33
Sample size	46	26	20
r  value required for significance (p<.05)	.304	.388	.444

There was an interesting trend between customer participation and Symposium usability. However this result is not significant with the sample size of 20 customers. This slight trend suggests that the usability of Symposium may have impacted the participation of the customers. In contrast there was an interesting negative trend between symposium usability and requirement engineer subject participation. This suggests that Symposium usability did not increase requirements engineer participation, and in fact, these engineers may have participated more when they felt the groupware was inadequate. One possible explanation is that some engineers contributed extra effort to compensate for perceived difficulties collaboration with the customer.

**4.3.4. Tool Effectiveness and SRS Quality**

Table 4-13 reports correlations between each group’s combined score for tool usefulness versus overall SRS quality. Although we have significantly less power in analyzing trends in these data, are there correlations between developing a high quality SRS document and the rated usefulness of groupware tools?

**Table 4-13 - Reported Groupware Tool Usefulness vs. Overall SRS Quality**

<b>Groupware Tool</b>	<b>r value</b>
Symposium Voice Conferencing	-0.26
Symposium Text Chat	-0.73
Symposium Whiteboard	-0.49
Symposium Agenda/Slide Show	0.07
Symposium Web Browser Sharing	0.38
Symposium Application Sharing	-0.13
MOOsburg Text Chat	0.38
MOOsburg Message Board	0.34
MOOsburg Whiteboard	0.38
MOOsburg Shared File	-0.57
MOOsburg Web Links	0.17
MOOsburg List Editor	0.44
MOOsburg Planner	0.80
Group List Server	-0.32
Sample Size	6
r  value required for significance (p<.05)	.811

There are no correlations between reported groupware tool usefulness and overall SRS Quality at  $p=.05$ . However there is a marginally significant negative correlation between Symposium Text Chat usefulness and overall SRS Quality. ( $r[df=4]=-0.73$ ,  $p < .10$ ) This indicates that groups who produced a lower quality SRS document perceived text chat to be more useful, while groups producing higher quality SRS documents perceived the reverse. This is interesting. It may mean that the groups who produced lower quality SRS documents relied more heavily on Text Chat during virtual meetings for information exchange. It is reasonable to say that Text Chat offers lower informational bandwidth with respect to other groupware tools available during the virtual meetings. If groups were relying on a lower informational bandwidth tool during meetings then perhaps a smaller quantity of information was obtained from the virtual meetings which lead to lower quality SRS documents. Group four reported the highest usefulness for Symposium text chat = 4.4. During the observations of the virtual meetings, group four customers were often intimidated either by the use of the English language or by members of the requirements engineering group 4. This explains why text chat was rated so useful for this group, and since group 4 produced a lower quality SRS document, they were influential in establishing this correlation. At the same time, the experiences of this group may be unique, making it difficult to generalize the relationship.



With a positive correlation of  $r=.80$  the MOOsburg planner appears to be a significant tool which may have effected overall SRS quality. However, both the customers and engineers consistently rated the MOOsburg planner as the least useful tool in the groupware tool suite at 14th place. Thus, this positive correlation between SRS quality and MOOsburg planner usefulness can be attributed to simply noise in the data or the subliminal response of high performance groups members rating a tool with the word “Planner” in the title as slightly higher in usefulness than their reduced performance group counterparts. Examining the survey data more closely reveals that this score was an effect caused by respondent tendency to assign a usefulness value to a tool even if it was not used. The instructions for the survey explicitly say to select “N/A” when a tool was not used, giving it a 0 value for tool usefulness. When a tool was not used occasionally one group member would assign a usefulness value anyways. The reduced performance groups reported usefulness for the MOOsburg planner in 20% of the cases, where high performing groups reported usefulness for the MOOsburg planner in 50% of the cases.

**Table 4-14 - Groupware Reported Effectiveness vs. Overall SRS Quality**

<b>Correlation with Overall SRS Quality</b>	<b>r value</b>
Reported MOOsburg effectiveness	0.06
Reported Symposium effectiveness	-0.04
Reported Email effectiveness	-0.29
Sample Size	6
r  value required for significance (p<.05)	.811

As seen in Table 4-14 the lack of correlations in the data between overall SRS quality and groupware tool effectiveness suggest that the groupware tools themselves did not impact the requirements engineering project in a positive or negative way. This suggests that for the most part the groupware merely enabled the distributed collaboration that took place during the project. Factors such as requirements engineering experience and customer participation more critically affected overall SRS document quality as opposed to groupware. It may be that the groupware tools were transparent to the task. They merely enabled the distributed collaboration throughout the project, which in general is a positive result. Although it was disappointing to find that the use of

groupware tools was not associated with higher SRS quality, the data does not suggest the reverse case, that the use of groupware is associated with lower SRS quality. The data is inconclusive, and further experimentation is required to determine how groupware influences SRS quality.

#### 4.3.5. Participant Groupware Feedback

The final survey of the participants of the empirical study included a number of questions that solicited suggestions for the groupware system. (See Appendix D) The summarized positive and negative comments obtained from the participants in the final survey are presented in the following three tables. Comments are presented concerning: MOOsburg, Centra Symposium, and the Group email list server.

**Table 4-15 - MOOsburg Positive/Negative Features**

<b>Positive Feature</b>	<b>Negative Feature</b>
Shared Files (document repository)	No voice communication
Group Spaces	Poor Usability
Message Board	Navigation to find project spaces
	Remote Access was non-optimal
	Slow Performance
	No shared document editing
	No slide presentation feature
	No Desktop Integration
	System Errors
	Security Configuration
	Macintosh Support unstable
	Navigation is complex
	Maturity of System is lacking
	No Shared Folders to put shared files in
	NO textual mode, forced to use graphical mode for everything.
	No automatic notification to users about changes within spaces.
	No guarantee that customer will check space to see updates.

**Table 4-16 - Centra Symposium Positive/Negative Features**

<b>Positive Feature</b>	<b>Negative Feature</b>
Voice Conferencing	Delayed Voice Responses
Text Chat	Poor Usability of Floor Control features
Agenda / Slide Show	No Virtual Avatars (pictures of users)
Easy to use	No Videoconferencing
Audio playback of meetings	Slow performance with document sharing

Application Sharing	No Shared Editor with user awareness
Shared Whiteboard	Web and presentation tools should be integrated.
Shared Web Browser	No private audio chat among participants
Hand Raising	Hard to request floor control (Difficult to get the opportunity to speak in meetings)
Voting Features	Permissions were not useful.
Better ability to detect misunderstandings that text chat	No "human touch"
Enables a single facilitator	Increases "US vs. THEM" mentality
	Sometimes there were too many tasks to complete at once. (talking, writing, listening)
	Steep Learning Curve initially
	Audio was only half-duplex.
	Those uncomfortable with English were at a disadvantage with audio chat.
	Poor voice quality
	Difficulties getting Audio to work on some workstations
	Sudden connection drops
	Requires MS Windows to operate.
	No file sharing.
	Determining the context of Yes/No reply by meeting participants is difficult.
	Sometimes need to playback entire recording to find information. Need a way to index meeting recordings.
	No awareness of text chat message changes unless user is watching the chat window.
	The text chat window is a separate window and should be integrated with all other windows.
	No Instant playback of previous audio to catch lost points.
	Slide navigation was forced to a linear sequence.

**Table 4-17 - Group Email List Server Positive/Negative Features**

<b>Positive Feature</b>	<b>Negative Feature</b>
Facilitate communication between Requirements Engineers	Customer rarely responds to messages.
Can be used to clarify questions in the meeting.	Not interactive
Messages were sent to entire group.	Possible to forget to respond to information requests.
Able to get feedback from customers quickly.	Takes too long to get responses back.
Provides communication between meetings.	No awareness if people were reading messages
Self documenting	Discussion over email was ineffective.
Easy access from anywhere	Since messages go to entire group, often no one replies because the message wasn't addressed to a particular individual.
Ability to transfer large amounts of data easily.	No defined standard for using email to prevent

	abuse.
Provided broadcast medium	Limitation on file sizes of attached documents.
	Have a threaded discussion board somewhere.

The positive features state what features of the prototype groupware system were beneficial in supporting distributed requirements engineering. The negative features state what aspects of the groupware system hindered the requirements engineering process. For these tables, the negative feature lists also includes customer feedback about desired changes and enhancements to the groupware. A complete list of the desired features appears in Appendix A.

One function requested by the participants' was not classifiable by subsystem (MOOsburg, Symposium, Email). This function was to integrate each of the groupware subsystems together as a single groupware application. There were a very large number of function requests from the customers. If a groupware system was made available with all of the functions requested it would be interesting to conduct a new empirical study to determine how the new groupware system would perform at enabling distributed requirements engineering.

Groups in the requirements analysis did not extensively use the MOOsburg application. Participant comments identified usability and access limitations as barriers to using the application for the requirements engineering project. Using MOOsburg required running an extensive slow Java application that required a large amount of system memory. MOOsburg could be simplified and integrated as part of the desktop workspace to increase user access to the environment. MOOsburg usability could be improved by simplifying navigation within spaces, and adding more tools for collaboration, such as a shared text editor.

One feature that participants reported as missing from Centra Symposium was the ability to see one another's faces. By adding avatars, small images of users, meeting participants could become more familiar with each other by associating a face with a voice. Video conferencing could be added to help participants assess immediate reaction to discussion in meetings. Adding full-duplex audio to Symposium could allow speakers to be interrupted during long orations in the meetings. Presently with half-duplex audio

once a speaker has control, only the session leader can interrupt the speaker. In addition, if the session leader begins talking there is no way to interrupt them.

Participants had fewer suggestions for improving email use for the project. A threaded archive of messages was suggested to help group members access prior discussions. One participant suggested defining a group-wide standard for how email should be used. A consistent policy for email use could increase the effectiveness of the collaboration. One problem with email use in the study was the recipients occasionally forgot to respond to messages. A reminder system could be added that would remind recipients to respond to urgent messages.

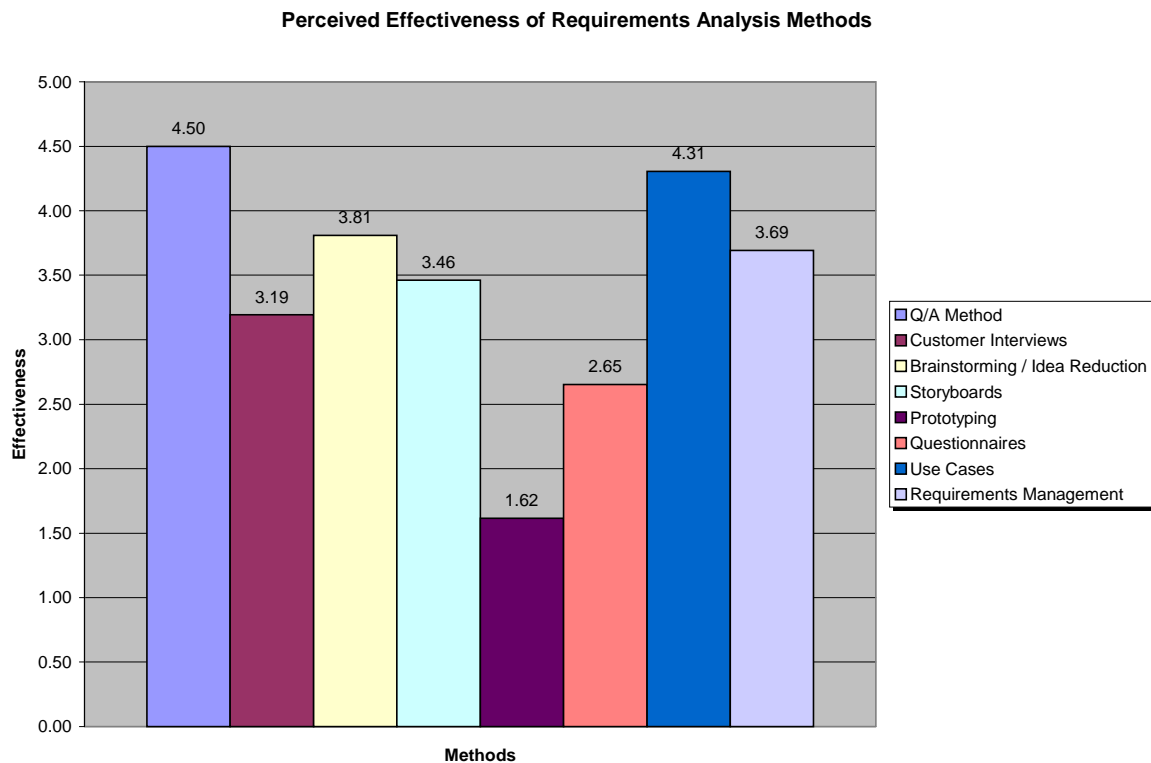
To draw generalizations from observations of groupware tool usage, text chat seemed to be an effective tool along with application sharing, and the shared web browser. They were used to interactively present documents and storyboards. Since each of the meetings had a different purpose, the tools chosen to facilitate tasks for each meeting varied. Tool usage did not converge because requirements engineers would use different tools to show a storyboard, a different tool to walkthrough the rough draft of their SRS, and a different tool to perform requirements management. Requirements engineers were not excessively creative in their use of tools. They did converge to using the basics, voice chat, and slide show, with occasional usage of the more complex tools like application sharing and web browser sharing. In general groups tended to avoid using complex tools, and stayed with simpler tools and tasks.

In summary the research results suggest that the Symposium Agenda/Slide Show, Group List Server, Symposium Voice Conferencing, and Symposium Text Chat were reported as the most useful groupware tools in our combined suite of tools. We can conclude that the Symposium Whiteboard, Agenda/Slide Show, and Web Browser Sharing are useful when customers participate actively in virtual meetings. We also note that Symposium Text Chat, is a useful groupware tool when customers participate less actively in virtual meetings.

## 4.4. Requirements Elicitation Methods

Research Goal #3 of this empirical study was to identify which requirements elicitation methods are most effective when used in distributed requirements engineering. By the nature of their inherent characteristics, some elicitation techniques may be more well suited to use in a distributed setting, while others may function poorly in the distributed mode. An understanding of the best elicitation techniques for distributed use may allow a generic requirements engineering process that is optimized for distributed use to be proposed. In this section the results of the empirical study and present findings that suggest which elicitation techniques worked well are presented. Then the impact of the application of various elicitation techniques on overall SRS quality is considered.

The assessment of requirements elicitation techniques does not take into account the survey values of the customers. Because the customers were not software



**Figure 4-7 - Reported Effectiveness of Requirements Elicitation Techniques**

engineering students, that they had an inadequate knowledge of the techniques to be able to assess their effectiveness accurately. For the eight elicitation techniques there is a strong positive correlation between average effectiveness ratings between customers and Engineers. ( $r[df=6]=.91, p < .002$ ).

In Figure 4-7 the average effectiveness ratings of the requirements elicitation techniques is shown. The top four elicitation techniques as rated by the requirements engineers were: Q\A Method, Use Cases, Brainstorming/Idea Reduction, and Requirements Management. Techniques were rated with the following likert scale: 1-Not Effective, 2- Possibly/Slightly Effective, 3- Effective, 4- Very Effective, 5- Outstanding Effectiveness, 0- Not Used. Did the use of these elicitation techniques impact overall SRS quality?

#### **4.4.1. SRS Quality and Requirements Elicitation Method Effectiveness**

Table 4-18 presents correlations between reported effectiveness of each requirements elicitation method and overall SRS quality. Surprisingly there is a marginally significant trend for SRS quality to have a negative relationship with reported elicitation method effectiveness. ( $r[df=4]=-.74, p < .09$ ) This implies that groups that produced high quality SRS documents reported that as a combined set the requirements methods were ineffective. The individual correlations in the table provide some indication of which techniques might have been most detrimental to producing a high quality SRS document. The overall negative correlation seems to be due to negative correlations between SRS quality and just two methods: Prototyping ( $r[df=4]=-.70, p < .12$ ) and Questionnaires ( $r[df=4]=-.56, p < .24$ ). But making a conclusion from these data is not so straightforward. The relation is between reported effectiveness and SRS quality, which doesn't say anything really about the actual effectiveness of requirements elicitation methods. It is quite possible that the positive and negative outcomes from each participant's experience influenced these judgments.

**Table 4-18 - Reported Requirements Elicitation Method Effectiveness vs. Overall SRS Quality**

Elicitation Method	r value
Q/A Method	-.06
Customer Interviews	-.38
Brainstorming / Idea Reduction	-.09
Storyboarding	-.10
Prototyping	-.70
Questionnaires	-.56
Use Cases	-.29
Requirements Management	-.33
Sample Size	6
r  value required for significance (p<.05)	~ +/- .811

Starting with the analysis of the negative correlation with SRS Quality and Prototyping, it may be that many of the participants in the study did not initially understand the difference between the Storyboarding technique and Prototyping. Before the requirements elicitation process started the customers claimed that Storyboarding was the second most unfamiliar technique. On the other hand at the start of the study the engineers identified Prototyping as the elicitation technique with which they were most familiar. Storyboarding was described to the participants as “a non-live prototype, just pictures or screen shots shown in example.” Prototyping was described as a “real live prototype that can be navigated.”

Prototyping was rated as the least effective technique in the study. It is interesting to note that even at the end of the study the participants still did not understand the difference between prototyping and storyboarding. In fact the instructions for the survey specified that if a method was not used that the respondent should mark “Not Used” for the effectiveness rating. In each requirements engineering group, there was disagreement whether prototyping was or was not actually used. This presents an alarming possibility that the data are invalid with respect to prototyping because there were disagreements on whether or not techniques were used. To investigate this possibility the average effectiveness of requirements elicitation techniques was correlated together with and without the “Not Used” values. There was a strong positive correlation between the tool effectiveness scores with and without “Not Used” values. ( $r[df=6]=.83, p < .01$ ) This suggests that the conflicts among the respondents interpretation as to whether or not specific techniques were used can be ignored.



The next elicitation technique to consider is Questionnaires. Were questionnaires less effective when SRS quality was high, and more effective when SRS quality was low? This weak negative relationship is seen in the data. ( $r[df=4]=-0.56, p < .24$ ) Questionnaires are generally considered an asynchronous elicitation technique. The engineer drafts a list of questions that is given to the user to answer offline from live meetings. For the average effectiveness rating for each group, there is a weak positive relationship between Questionnaires being rated as an effective technique and offline participation. This suggests that in order for the Questionnaire elicitation technique to be effective the customers need to be active participants outside of the live meetings. This fact was reiterated by several of the comments expressed by engineers.

"overview our customers are good, but sometimes the reply was very delay, that affects our work." *Group 6 participant*

"The arguing in the beginning was a little excessive and made it very frustrating. Also, Customer B was the only one who responded to our e-mails. I felt bad that he was really the only one majorly participating. Also, since they were the only one that responded quite often, this caused problems with disagreements in the meetings since we thought that what Customer B sent us was the final answer and then in the meetings we came to find out that this was not the case." *Group 6 participant*

"it affected the process to a great extent. it was great that they participated very actively during online meetings, but it would have helped if they'd not been so silent offline." *Group 3 participant*

The trend seen for questionnaire effectiveness requiring offline customer participation is also seen by examining individual requirements engineer responses. There is a marginally significant relationship between individual requirements engineer reported questionnaire effectiveness vs. perception of customer offline participation. ( $r[df=24]=0.30, p < .14$ ).

#### **4.4.2. Participation and Requirements Elicitation Methods Effectiveness**

Each email messages exchange on the group email accounts were counted and classified. The classification categories included: Information Requests, Requirements

Engineering team member interchanges, Scheduling issues, Customer Answers to questions, document posts, agenda distribution, MOOsburg information, General Announcements, Meeting Notes, and attachments. There is a weak positive trend between email message count of “Customer Answers to questions” and the effectiveness of the Questionnaire technique, rated by group. This suggests that the Questionnaire technique requires offline participation from the client. In this case offline participation is measured as the number of email messages where the customer is answering questions. Qualitatively it simply does not make sense that increased levels of offline customer participation would inversely affect SRS quality. However there is a marginally significant trend between Customer Answers to questions via email and Overall SRS quality. ( $r[df=4] = -.64, p < .17$ ) This suggests that in our study groups that elicited requirements from questionnaires over email had lower quality SRS documents.

Could the interaction in the planned virtual meetings be insufficient for low performance groups? Were email questionnaires needed to supplement information gathering? The data suggests that groups producing better quality SRS documents tended to avoid using email to resolve issues. This suggests that in our study groups that reverted to email questionnaires for requirements elicitation were not getting enough information during the scheduled virtual meetings. These groups may have been forced to use asynchronous collaboration to clarify issues because they did not gather enough information from virtual meetings. If customers failed to participate in offline discussion, this irritated these engineers who were already falling behind and relying on offline collaboration to complete the requirements assessment. Negative comments regarding customer participation were more abundant in groups with low SRS quality. (See Section 4.2.1)

In Table 4-19 there is one significant positive correlation between self-reported project participation level and perceived effectiveness of requirements elicitation method.

**Table 4-19 - Requirements Elicitation Technique effectiveness vs. Self Reported Participation**

<u>Correlation with</u>	<u>Everyone Self Reported Project Participation</u>	<u>Engineer Self Reported Project participation</u>	<u>Customer Self Reported Project Participation</u>
All Method Average	n/a	-0.01	n/a
Q/A Method	0.22	0.23	0.08
Customer Interviews	0.13	0.05	0.10
Brainstorming / Idea Reduction	0.30	-0.01	0.25
Storyboards	0.23	0.00	0.11
Prototyping	0.25	0.11	0.32
Questionnaires	0.02	-0.22	-0.01
Use Cases	0.25	0.04	0.11
Requirements Management	0.08	-0.09	0.03
Samples	46	26	20
r  value for significance (p<.05)	.3	.388	.444

There is a trend between self reported participation and reported effectiveness of the brainstorming technique. ( $r[df=44]=.30, p < .05$ ) It appears that if people perceived themselves as actively participating in the study, then they felt the brainstorming technique is more effective. Because the customers were the ones generating the ideas from brainstorming it is not surprising that the relationship is apparently larger in their data,  $r=.25$ . Although none of the other methods showed a significant relationship, several have a positive relationship with self-reported participation. (Q\A Method, Storyboards, Prototyping, Use Cases) When a requirements elicitation technique was reported as being more effective, the participant typically was participating more actively in the project. These elicitation techniques may have influenced participants to contribute more to the requirements analysis.

Table 4-20 shows the correlation between the requirement engineer's perception of customer participation and the different requirements elicitation techniques. There are only two significant trends in the data.

**Table 4-20 - Perceived Customer Participation vs. Elicitation Method Effectiveness**

<u>Elicitation Method Effectiveness</u>	<u>Online Participation</u>	<u>Offline Participation</u>	<u>Overall Participation</u>
All Method Average	0.42	0.18	0.09
Q/A Method	0.41	0.14	0.16
Customer Interviews	0.28	0.24	0.25
Brainstorming / Idea Reduction	0.27	-0.01	0.02
Storyboards	-0.05	-0.22	-0.16
Prototyping	0.38	0.10	0.04

Questionnaires	0.08	0.30	0.09
Use Cases	0.27	-0.01	-0.03
Requirements Management	0.32	0.10	-0.05
Samples	26	26	26
r  value for significance (p<.05)	.388	.388	.388

A trend exists between the reported effectiveness of the Q/A method and the perceived level of participation during virtual meetings. ( $r[df=24]=.41$ ,  $p < .05$ ) This implies that if the customers seemed to participate more actively in virtual meetings, then the Q/A method seemed more effective. Intuitively this makes sense. There was a correlation between perceived customer participation in virtual meetings and the average reported effectiveness of all tools. ( $r[df=24]=.42$ ,  $p < .05$ ) This implies that as the customers seemed to participate more in virtual meetings, the requirements engineers reported that the requirements elicitation techniques were more effective. Again this is a very understandable result.

There is a marginally significant trend that suggests that questionnaires are reported as more effective when there is more offline participation. ( $r[df=24]=.30$ ,  $p < .14$ ) The effectiveness of Prototypes seems to be related to the degree of online participation. ( $r[df=24]=.38$ ,  $r < .06$ ) This is also a finding as prototype evaluation requires that the customers actively test and use the prototype to contribute useful feedback.

#### **4.4.3. Engineering Experience and Requirements Elicitation Method Effectiveness**

Table 4-21 shows the relationships between individual reported experience with requirements elicitation methods prior to the empirical study, and the reported effectiveness of using those methods in the study. A positive correlation indicates that having prior experience using the method was associated with higher ratings of effectiveness. This could be attributed to the fact that experience using the method was needed to be effective.

**Table 4-21 - Correlation between reported method effectiveness and reported experience using method**

<b>Method Experience vs. Reported Effectiveness</b>	<b>r value</b>
Question/Answer Method	n/a
Customer Interviews	-0.09
Brainstorming / Idea Reduction	0.14
Storyboarding	-0.17
Prototyping	0.37
Questionnaires	-0.35
Use Cases	0.26
Requirements Management	0.43
Samples	26
r  value required for significance (p<.05)	.388

There is a significant positive correlation for the Requirements Management method between reported effectiveness and experience using it. ( $r[df=24]=.43$ ,  $p < .03$ ) This suggests that Requirements Management was more effective when the participants had prior experience with this technique. Prototyping was also reported as being more effective when the requirements engineers had previous experience with the method. ( $r[df=24]=.37$ ,  $p < .06$ ). There was a marginal trend suggesting that prior experience with use cases led to increased perception of effectiveness in the experiment.

There was a marginally significant negative correlation between reported questionnaire effectiveness and experience using questionnaires. ( $r[df=24]=-.35$ ,  $p < .08$ ). It appears that prior experience using questionnaires did not correlate with increased perceptions of effectiveness. In fact prior experience with questionnaires led to the perception that the technique was not effective in the experiment. Questionnaires are an asynchronous elicitation technique. Generally questionnaires were facilitated using the group list server. Requirements engineers sent emails asking questions with the hope that the customers would respond with adequate detail in a timely manner. Earlier it was suggested that questionnaires are more effective when customers actively respond to email and participate actively outside of the virtual meetings. It appears that reduced performance groups relied on questionnaires to gather requirements since during their real time virtual meetings they were unable to gain enough knowledge about customer requirements. This may have caused the negative trend that suggests questionnaires are not an effective elicitation technique.

In summary the Q/A Method, Use Cases, Brainstorming/Idea Reduction, and Requirements Management were reported as the most effective requirements elicitation

techniques of the methods attempted in the study. Overall the impact of specific requirements elicitation techniques on overall SRS quality was inconclusive. In this study, the Questionnaires was used more frequently by reduced performance groups in this study, but this does not necessarily indicate that Questionnaires are an ineffective requirements elicitation technique. This suggests that substantial asynchronous participation may be required for Questionnaires to be an effective elicitation technique. Prototyping and the Question and Answer method were determined to be more effective when the customer actively participates in virtual meetings. Requirements Management and Prototyping are techniques best used when the requirements engineers have prior experience using them.

## **4.5. Customer Participation**

This chapter concludes with a brief look at a significant factor that impacted the overall quality of this requirements analysis experiment. This experiment attempted to simulate a real world requirements engineering project. To obtain similar results to a real case study the experiment should be as realistic as possible. In order to have a realistic experiment the customer's participation and role-playing abilities was relied upon. Customers role-played as concerned stakeholders in the requirements definition of the company wide scheduling system. The engineers were surveyed as to their perception of the overall quality of this requirements engineering project simulation. Table 4-22 shows a significant correlation between the perceived online participation of customers and the requirements engineers' view of experiment realism. ( $r[df=24]=.48, p < .05$ ) The importance of customer participation in requirements engineering studies is conveyed by this relationship.

**Table 4-22 - Overall Simulation Quality vs. Perceived Customer Participation**

<b>Correlation with Reported Simulation Quality</b>	<b>r value</b>	<b>Significance p</b>
Online Participation	0.48	.01
Offline Participation	0.22	.28
Overall Participation	0.26	.19

The average simulation quality was 3.88 on the 1-5 likert scale. This indicates that the requirements engineers generally believed the simulation was of reasonable quality.

There were only seven cases of absence, and six cases of late arrival/early departure for the entire experiment. It is interesting to note that there was only one case of absence and one case of a late arrival for requirements engineer participants. The remaining six cases of absence and five cases of late arrival/early departure were by customer participants. This observation indicates that they requirements engineers were somewhat more committed to the success of the project than the customers.

Comments were solicited from the requirements engineer participants in order to obtain feedback about what made this requirements engineering project simulation both realistic and unrealistic. A handpicked set of these statements appears in the tables below. These comments represent a summary of the overall types of comments made.

**Table 4-23 - Positive Comments supporting Simulation Quality**

<b>What factors made the simulation realistic?</b>
"The fact that the customers had conflicts on issues made it the most realistic. You could get a feel that in a real company, the higher ups don't really understand what the secretaries go through and vice-versa. My mom's a secretary and now I understand."
"There were differences in opinions among the customers...each one requesting for different attributes...difficulty in satisfying all the customers, developing prototypes to clarify the issues, etc."
"I guess the fact that the customers did not have a clue about the requirements themselves, which is what happens in real life. ... Also, I didn't know any of the customers personally, which made it all the more realistic."
"Fighting between the developers and the client. Tough issues that had to be resolved with compromise. "

**Table 4-24 - Negative Comments about Simulation Quality**

<b>What factors made the simulation unrealistic?</b>
"the customer group didn't care as much as real customers would have, our group was not as pro-active and didn't put as much effort into their feedback."
"The customers didn't really seem to know what they wanted. I realize this can be the case in real life, but they didn't seem to be thinking as though they would really be using the system, or that they cared very much."
"the SRS will not be designed and implemented by the software engineering who created."
"Only thing is the absence of video features which didn't allow us to perceive the facial expressions of the customers."

Chapter 5 summarizes the observations seen in this empirical study. Conclusions on what factors impacted overall SRS quality are presented. Conclusions are made from observations in this study and the lessons learned are about groupware tools to support distributed requirements engineering are described. Conclusions regarding requirements elicitation methods for use in distributed requirements engineering projects are presented. Chapter 5 concludes with future directions for this research as well as several questions to ponder.



## 5.

# Conclusions and Future Work

---

This empirical study was an exploratory study that sought to identify how well groupware tools would support distributed requirements analysis, and also how well traditional requirements elicitation techniques work in a distributed environment. Findings related to each of the research goals outline from Chapter 1 are presented. These research goals were:

- o (RG1) measurement of the quality of the SRS documents produced and determination of causality
- o (RG2) effectiveness of groupware tools at supporting distributed requirements engineering
- o (RG3) effectiveness of requirements analysis techniques for distributed use
- o (RG4) identification of future research directions.

### 5.1. SRS Quality

Research Goal #1 of this empirical study was to first assess SRS document quality, and then to investigate factors related to the production of a high quality SRS document in this distributed requirements engineering project. Using a series of four metrics the quality of the SRS documents produced by each group was assessed and an overall quality rankings was assigned to each SRS document produced. Correlations and interesting trends with SRS document quality are studied and analyzed for relevance. Experimental observations and participant feedback is then used to help explain the results obtained. Six interesting and significant correlations were found (in order of statistical strength):

- Claim 1. Groups with less software engineering experience produced better quality SRS documents. ( $r[df=4]=-0.81, p < .05$ )
- Claim 2. Groups reporting lower effectiveness of requirements elicitation techniques produced better quality SRS documents. ( $r[df=4]=-0.74, p < .09$ )

- Claim 3. Groups reporting higher usefulness of Symposium Text Chat produced lower quality SRS documents. ( $r[df=4]=-0.73, p < .10$ )
- Claim 4. Groups obtaining more information from customers via email produced lower quality SRS documents. ( $r[df=4]= -0.64, p < .17$ )
- Claim 5. Groups perceiving their members as contributing more to the group effort produced better quality SRS documents. ( $r[df=4]=0.60, p < .21$ )
- Claim 6. Groups having more experience with requirements elicitation techniques produced better quality SRS documents. ( $r[df=4]=0.56, p < .25$ )

Of these claims, only Claim 1 is statistically significant for  $p=.05$ . All other claims are just general trends in the data with minimal statistical significance. Claim 1 can be refuted as the side effect of the unusual relation between the requirements engineering and software engineering experience level of groups.

Claim 2 states that groups who perceived the requirements elicitation techniques as less effective produced better quality SRS documents. This trend can be explained by effectiveness scores for Prototyping and Questionnaires. The usage of Questionnaires was associated with the need for asynchronous communication. Because reduced performance groups may have acquired less information from virtual meetings, they may have needed to augment information gathering with asynchronous tools. Claim 4 can be linked with this same discussion. Email was an asynchronous tool that was used to augment information gathering in reduced performance groups.

Claim 3 states that groups who relied more heavily on text chat in virtual meetings produced lower quality SRS documents. This could be because they were not making effective use of their time in the meetings. Text chat simply does not have the informational bandwidth of speech. As groups rely more on text chat for the interchange in meetings, they have less total information exchange. Groups with less information from virtual meetings relied more on asynchronous methods for requirements gathering. These groups produced reduced quality SRS documents. Groups that produced high quality SRS documents appeared to make the best use of time in the meetings using voice conferencing, and web browser sharing. They did not need to rely on asynchronous

information exchange because at the end of each virtual meeting they walked away with enough information to proceed with the requirements development.

Claim 5 is a logical assertion that simply states that the group members were somewhat aware of the state of their group. Based on the interactions with group members and the project progress group member peer perception reasonably predict the general trend in SRS document quality.

Through observations we believe that the reduced performance group problems were related to inner group dynamics. However, there is some suggestion here that asynchronous requirements elicitation is not as effective as synchronous requirements elicitation. This may be due to less customer use of asynchronous tools. It is not known whether distributed customers always participate less asynchronously, or if this trend is unique to our role-playing customers. Synchronous interchanges may be more effective at distributed requirements elicitation, but asynchronous collaboration needs to be studied in more detail. From this study we can suggest that a purely asynchronous distributed requirements engineering effort could be very ineffective.

Claim 6, requirements engineering experienced positively effects SRS quality is valid is a reasonable finding. This is an interesting trend in the data that simply makes sense. One would expect that groups more experienced with requirements analysis would perform better at requirements analysis. With the small sample size in the study factors such as group dynamics may have played a more significant role in the data than the actual measures of experience. From observation it can be noted that none of the groups had members who were experts in requirements analysis. The average score for requirements analysis experience was 2.5. On the survey a value of two indicated “familiar with, but never used”, and a value of three was “used 1-3 times before”. From this it can be implied that participants only used most requirements elicitation techniques perhaps once before. If the participants had more requirements engineering experience the overall impact on SRS quality may be greater. Further research trials with more experience requirements engineering participants could attempt to statistically validate this claim.

## 5.2. Groupware Tools

Research Goal #2 of this empirical study was to assess the effectiveness of the groupware tools used in the experiment to develop an understanding of what tools and features are needed for a integrated groupware system which supports distributed requirements engineering. Appendix A lists functional requirements for a groupware system to support distributed requirements engineering elicited from participant feedback in this study.

To directly determine which tools perform best, if compared against overall SRS Quality there are only six project groups as data points. This limits the correlation to (df) degrees of freedom of four, and requires very strong correlations in the data to find significant correlations. For studying the performance of groupware tools it was interesting to correlate against project variables other than SRS quality, both because these relationships are intriguing and for more statistical significance. The effect of tools on participation, and the need for participation when using particular tools was considered. From experimental observations, the following claims about groupware tools for distributed requirements engineering can be made (in order of statistical strength):

- Claim 1. Symposium Voice Conferencing, Agenda/Slide Show, Text Chat, and the Group Email List Server are effective tools for distributed requirements engineering.
- Claim 2. The usability and configuration issues associated with MOOsburg caused customers to not use it.
- Claim 3. Centra Symposium Whiteboard ( $r[df=24]=.52, p < .01$ ), Agenda/Slide Show ( $r[df=24]=.44, p < .025$ ), and Web Browser Sharing ( $r[df=24]=.62, p < .01$ ) are useful tools when customers actively participate in virtual meetings.
- Claim 4. Overall groupware tools are more effective when customers participate actively in virtual meetings. ( $r[df=24]=.49, p < .025$ )

- Claim 5. The Centra Symposium application is more effective at supporting distributed requirements engineering when customers participate actively in virtual meetings. ( $r[df=24]=.39, p < .05$ )
- Claim 6. Centra Symposium Text Chat is a useful tool when customers are not active during virtual meetings. (NS)
- Claim 7. A slight trend indicates that Asynchronous groupware tools such as the Group List Server, and MOOsburg Shared Files are more useful when virtual meetings do not produce enough information for requirements analysis. (NS)

These claims are derived from observations and correlations made based on the quantitative data and behavioral observations witnessed in the empirical study. They all suggest possibilities of the effectiveness of groupware tools for performing distributed requirements engineering. Further experimentation is required to study these observations to see if they are significant. With the claims stated above we can say that claims 2, 3, 4, and 5 are significant at  $p=.05$  using data from the 26 requirements engineers or 20 customer participants in the study.

Claim 1 enumerates the most useful groupware tools based on the average scores for groupware tool usefulness as reported project participants. These tools were rated the highest for usefulness and effectiveness in the experiment.

Claim 2 is derived from the relationship between self-reported participation level and MOOsburg usefulness. The trend indicates that customers participated less in the experiment if they believed MOOsburg was not useful. There were a few customers who reported difficulties running MOOsburg remotely, which explain this trend. It can be expected that these customers may have assumed that if they could have used MOOsburg remotely they would have been more active in participating in the requirements analysis.

Claim 3 was made from data that suggests that Centra Symposium Whiteboard, Agenda/Slide Show, and Web Browser Sharing were reported as more useful tools when customers participated actively in virtual meetings.

Claim 4 states that the tools seemed more useful when participation level was higher. This claim suggests the importance of customer participation both in this study

and for all requirements engineering projects. It is difficult to perform requirements analysis if the customer is uncooperative. Claim 5 states that Centra Symposium itself seems more useful when customers participate actively.

Claim 6 was derived from data trends that indicate that Text Chat seemed more useful in cases where the customer seemed to not participate actively. Customers that appear inactive may appear this way because of a language barrier. Using text chat for communication reduces some of the difficulties associated with the language barrier related to speech. This behavior was observed several times from silent non-native English speaking customers. The customers often used text chat actively in virtual meetings, but spoke very little.

Claim 7 is derived from the usefulness ratings for the two primary asynchronous tools: the group email list server and the MOOsbrug Shared File. These tools seemed more effective when groups produced reduced quality SRS documents. Groups that produced reduced quality SRS documents may have reverted to using more asynchronous methods for requirements elicitation because they did not obtain enough information from planned virtual meetings. This trend suggests that the Asynchronous methods were preferred when additional information was required outside of planned virtual meetings. Early it was suggested that because the reduced performance groups used asynchronous methods more than high performance groups, the asynchronous methods made be inferior for requirements gathering in comparison to the synchronous techniques that can be used during live virtual meetings.

Some of the problems in distributed requirements engineering are the same problems encountered in normal requirements engineering. Two common requirements analysis problems identified in [6] were observed in this study: the “Undiscovered Ruins” syndrome and the “User and Developer” syndrome. From general observations the groupware system enabled requirements engineering such that groups were able to conduct the requirements elicitation, and produce final drafts of SRS documents. Customers had several comments that support the fact that requirements engineering was enabled transparently by the groupware. The engineers were asked the question “How did the distributed software (Centra Symposium, MOOsburg, Email) effect elicitation and clarification of difficult requirements?”

"The distributed software did not effect the elicitation of these requirements. Since the customers did not know much about them, we had to do our own research, independent of these softwares."

"Not at all. Except for the lag in speech. "

"I don't believe that they negatively effected elicitation."

"Just by providing the facility to communicate"

"we could ask them (the customers) what details they wanted. The means of communication did not have much effect; the fact that there was communication available is what mattered."

Some engineers responded positively:

"allowed the issue to be iterated over many times until the customer was happy"

"Email allowed us to clarify issues as they arose, Centra Symposium allowed us to communicate with the client while keeping a full electronic log ... "

"They were easy to use and had lots of features"

And there were a few negative comments:

"E-mail caused part of the problem, in that the customers didn't understand our questions but we couldn't clarify them until we got their incorrect responses back... "

"The lack of a face-to-face meeting may have adversely affected our ability to clarify these requirements easily."

"Symposium made it easier to clarify some points but wasn't enough"

### **5.3. Requirements Elicitation Techniques**

Research Goal #3 of the empirical study was to identify which requirements elicitation techniques were effective for distributed requirements analysis. In this research an array of requirements elicitation techniques were tested to see if they were effective for distributed use. The requirements engineers were instructed on eight different techniques: Brainstorming / idea reduction, feature definition (Q&A method), Storyboarding, Use Cases, Prototyping, Requirements Management, Customer

Interviews, and Questionnaires. From observations in the study the following claims about requirements elicitation for distributed requirements engineering can be made (in order of statistical strength):

- Claim 1. The Q/A Method, Use Cases, Brainstorming / Idea Reduction, and Requirements Management are effective techniques for requirements elicitation in the distributed setting.
- Claim 2. Requirements Management is a more effective elicitation technique if the engineer applying the technique has prior experience using it. ( $r[df=24]=.43, p < .03$ )
- Claim 3. Brainstorming is an effective technique when used with active participants. ( $r[df=44]=.30, p < .05$ )
- Claim 4. Requirements analysis techniques are more effective when customers participate actively during virtual meetings. ( $r[df=24]=.42, p < .05$ ) The Q/A method is especially effective when customers actively participate in virtual meetings. ( $r[df=24]=.41, p < .05$ )
- Claim 5. Questionnaires are a more effective requirements analysis technique when customers participate offline outside of virtual meetings. ( $r[df=24]=.30, p < .14$ )

These claims are derived from observations and correlations made based on the quantitative data and behavioral observations from the empirical study. They all suggest possibilities of the effectiveness of requirements elicitation methods for use in distributed requirements engineering. Further experimentation is required to study these observations to see if they are significant. Claims 2, 3, and 4 are significant at  $p=.05$  using data from the 26 requirements engineers in the study.

Claim 1 was derived from the average effectiveness ratings, supplied by the requirements engineers, rating the requirements elicitation techniques. Requirements Management was reported as being a more effective technique when the requirements



engineer applying the technique had more experience using it. This was also the case with prototyping and use cases but the trend was not as significant.

Claim 2 was derived from the data correlation between requirements method effectiveness and method experience. Requirements engineers reported that the requirements management method for requirements analysis worked better when they had prior experience using it. This suggests that requirements management is a requirements analysis technique that requires experience to use. There were also interesting but not significant trends between method experience and method effectiveness for prototyping and use cases.

Claim 3 is made from the data correlation between self reported participation and effectiveness rating of brainstorming. Participants who reported brainstorming as an effective elicitation technique indicated that they participated more actively than participants who rated brainstorming as being less effective. Participants were not asked whether they were active in brainstorming. The significant correlation suggests that active participants seem to favor brainstorming as an effective elicitation technique.

Claim 4 was made from the data correlation between perceived customer participation in the virtual meeting versus the average rating of requirements technique effectiveness. The average effectiveness of all techniques combined was seen as having a positive correlation with the perception of customer participation during virtual meetings. Especially significant was the trend between Q/A method effectiveness and customer participation in virtual meetings. This claim supports the need for customer participation for the effective application of all requirements elicitation techniques.

Claim 5 shows a positive correlation between perceived customer offline participation and effectiveness of using questionnaires as a requirements elicitation technique. Since questionnaires were sent via email, customers needed to participate offline from the virtual meetings in order for this to be an effective elicitation technique.

## 5.4. Contributions

In this empirical study distributed a requirements engineering project was facilitated using a suite of groupware tools to support collaboration. Six requirements engineering teams followed a basic requirements engineering process to elicit, refine, and specify requirements for a company wide scheduling system. Each group produced a rough and final draft of an SRS that was later analyzed to determine the quality of performance of each group.

Factors effecting group performance were determined (Research Goal #1). Most notable is that groups relying on email and text chat, groupware tools with less informational bandwidth than tools such as voice chat and application sharing, produced lower quality SRS documents. Through data and observations we believe that groups with communication challenges were attracted to the email and text chat because these tools ease communication barriers associated with social distance [44]. However these use of tools with less informational bandwidth tended to produce lower quality SRS documents.

By studying the effectiveness of groupware tools (Research Goal #2) and requirements elicitation techniques (Research Goal #3), the importance of customer participation in distributed requirements analysis was identified. To be perceived as effective, synchronous groupware tools required customer participation in live sessions, and asynchronous tools required offline customer participation.

Elicitation technique effectiveness was also impacted by customer participation. Asynchronous elicitation techniques such as Questionnaires seemed less effective when customers did not participate actively outside of planned virtual meetings. Groups who

relied on Asynchronous elicitation techniques for requirements analysis tended to produce lower quality SRS documents. We believe that the asynchronous requirements elicitation techniques are less informationally rich than synchronous techniques. In this study, groups who relied upon on asynchronous elicitation techniques may have needed more asynchronous customer participation to help improve SRS quality.

## **5.5. Future Directions**

We believe that this empirical study represents perhaps the first study that focuses on the effectiveness of groupware tools for supporting distributed requirements engineering projects. Previous studies consider academic processes and models of requirements analysis that may involve some distributed collaboration. Other research has developed and presented prototype systems that are process specific. A study that simulates distributed requirements engineering projects for the purpose of studying the effectiveness of generalized tools and requirements elicitation methods is interesting. More iterations of this empirical study will serve to expose more information about significant problems in distributed requirements analysis. With only six groups our observations may be too insignificant to concretely identify the recurring problems with distributed requirements engineering. We can study the problems we have encountered in this study and suggest that these may be common problems needing to be addressed for groupware tool support. Ultimately additional trials of this experiment should be conducted to add significance to the findings and to identify which trends are truly representative of problems in distributed requirements engineering.

We believe that having control variables would have been interesting in this study. If there were a control group of requirements engineering teams that performed the requirements project with co-located face-to-face meetings with customers we could compare their effectiveness versus our distributed groups. Having a control group of co-located teams to compare against distributed teams would allow more contrasting views on the effects the distributed setting versus the traditional face-to-face setting. Perhaps a

future study could conduct the entire requirements analysis just with co-located teams and then those results could be compared to the results of the distributed teams found in the experimental trial.

The location of the groups is not the only interesting variable we could control in future studies. We could consider controlling the specific elicitation methods that groups were allowed to use. By fixing the methods we could have sharper contrasting views on the effectiveness of specific requirements elicitation methods. This study was an exploratory study where we presented an array of options and did not specify which elicitation methods groups were allowed to use. We wanted the groups to have some freedom to choose their approach merely for the sake of seeing what they would do, and how this would affect SRS quality in the end.

Another possible control variable would be the groupware tools. We could require groups to use different tools to compare and contrast their effectiveness. Perhaps a future experiment could contrast asynchronous and synchronous groupware tools to support distributed requirements engineering. This research loosely suggests that asynchronous tools and methods may not be as effective at supporting distributed requirements engineering. What if we had a control group of engineers that only could perform requirements analysis with asynchronous techniques? Imagine an Asian software development firm conducting a requirements analysis for a product with the customer based in the United States. Whenever a customer's workday does not have overlapping hours with the engineers this is a possibility. In this situation asynchronous collaboration may be the normal with few opportunities for synchronous meetings.

Another possible enhancement to this study would be to find participants with more requirements engineering experience. We observed that in many cases our requirements engineers were novices in requirements engineering. They were often applying elicitation techniques for the first or second time during their requirements analysis with the customer. More experienced engineers would eliminate the effect that a learning curve may have had on the overall effectiveness of the elicitation techniques.

Finally another possible direction for this research is to build a prototype groupware system that has the functionality described in Appendix A. If such a groupware system was available studies could be conducted to see what aspects of the

functionality are really effective. Many participants in this study asked for video conferencing. Would the use of video conferencing enhance the overall quality of SRS documents? Will the use of video conferencing increase or decrease speed and accuracy of requirements elicitation? Is video conferencing worth the high cost in additional system hardware and network bandwidth? With further studies the groupware system's functional requirements could be optimized to include only the most relevant and effective tools and features for distributed requirements engineering. The final product could be a simplistic intuitive groupware system with good usability for distributed requirements engineering. Often complex systems with unnecessary features are intimidating and ineffective because they are not easily learned in the short span of a requirements engineering project.

Distributed requirements engineering, and distributed software engineering in general are large research areas for the future. With the increasing quality of communication and the decrease in communication cost it only makes sense that more distributed collaboration will be the norm in the future. Eventually we may have trouble remembering a time when we didn't perform distributed work in software development. Research on groupware to support distributed work can help enable this future.

# Bibliography

---

- [1] Brooks, F.P., "No Silver Bullet: Essence and Accidents of Software Engineering," *IEEE Computer*, 20, 4 (April 1987), 10-19.
- [2] Ellis, C.A., Gibbs, S. J., and Rein, G. L. 1991. Groupware: Some issues and experiences. *Communications of the ACM*, 34, 1 (January 1991), 38-58.
- [3] Higa, K. Understanding Relationships Among Teleworkers' E-Mail Usage, Email Richness Perceptions and E-Mail Productivity Perceptions Under a Software Engineering Environment. *IEEE Transactions on Engineering Management* 47, 2 (May 2000), 163-173.
- [4] French, A. A Study of Communication and Cooperation in Distributed Software Project Teams. In *Proc. International Conference on Software Maintenance*. (1998), 146-154.
- [5] Herela, D., Greenberg, S. Using a Groupware Space for Distributed Requirements Engineering. In *Proc. Seventh IEEE International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises WET ICE '98*. (1998), 57-62.
- [6] Leffingwell, Dean and Don Widrig, *Managing Software Requirements: A Unified Approach*, Addison Wesley., Boston, MA, 2000.
- [7] Davis, Alan M. *Software Requirements: Objects, Functions, and States*. Englewood Cliffs, NJ: Prentice-Hall, 1993.
- [8] Alho, Kari and Reijo Sulonen. Supporting Virtual Software Projects on the Web. In *Proc. Seventh IEEE International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises WET ICE '98*. (1998), 10-14.
- [9] Boehm, Barry and Egyed, A, WinWin Requirements Negotiation Processes: A Multi-Project Analysis. In *Proceedings of the 5<sup>th</sup> International Conference on Software Processes (ICSP)*, Lisle, IL, June 1998, pp. 125-136.
- [10] Egyed, A and Boehm, Barry, Telecooperation Experience with the WinWin System. In *Proceedings of the 15<sup>th</sup> IFIP World Computer Congress (WCC)*, Vienna, Austria and Budapest, Hungary, September 1998, pp. 37-46.
- [11] Antón, A. and Liang, E, A Web-Based Requirements Analysis Tool. In *Proc. Fifth IEEE International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises WET ICE '96*. (1996), 238-243.
- [12] Herela, D. Eveblein, A and Shaw, M, Using Different Communication Media in Requirements Negotiation. *IEEE Software* 17, 3 (May/June 2000), 28-36.

- [13] Hrones, J., Jedrey, B., Zaaf, Driss. Defining Global Requirements with Distributed QFD. *Digital Technical Journal* 5, 4 (Fall 1993), 36-46.
- [14] Herela, D. "Users' Involvement in the Requirements Engineering Process" available from <http://spuds.cpc.ucalgary.ca/KAW/KAW96/gherlea/FINAL.html> Internet accessed: 7 November 2000.
- [15] Herela, D. "Computer supported collaborative requirements negotiation" available from <http://www.cpsc.ucalgary.ca/~danah/kaw98/Improved.html> Internet accessed: 11 October 2000.
- [16] Leonard, T. Berzins, V., LUQI, Holden, M. J., Gathering Requirements from Remote Users. In *Proceedings of the 9<sup>th</sup> IEEE International Conference on Tools with Artificial Intelligence*. (1997), 462-471.
- [17] Alcázar, E., Monzón, A., A Process Framework for Requirements Analysis and Specification. In *Proceedings of the 4<sup>th</sup> IEEE International Conference on Requirements Engineering*. (2000), 27-35.
- [18] Siddiqi, J., Shekaran, M. C., Requirements Engineering: The Emerging Wisdom. *IEEE Software* 12, 2 (March 1996), 15-19.
- [19] Ocker, R., Hiltz, S.R., Turoff, M. Fjermestad, J., Computer Support for Distributed Asynchronous Software Design Teams: Experimental Results on Creativity and Quality. In *Proceedings of the 28<sup>th</sup> IEEE International Conference on System Sciences*. (1995), 4-13.
- [20] Potts, C., Takahashi, K., Antón, Inquiry-Based Requirements Analysis. *IEEE Software* 11, 2 (March 1994), 21-32.
- [21] van Lamsweerde A, Darimont R, and Massonet Ph. The Meeting Scheduler System: A Problem Statement. Available via ftp anonymous: <ftp://ftp.info.ucl.ac.be/pub/public/92/MeetingScheduler.ps>
- [22] Aoyama, M. Agile Software Process Model. In *Proceedings of the 21<sup>st</sup> IEEE International Computer Software and Applications Conference COMPSAC '97* (1997), 454-459.
- [23] Herbsleb, J., and Grinter, R. Architectures, Coordination, and Distance: Conway's Law and Beyond. *IEEE Software* 16, 5 (September/October 1999), 63-70.
- [24] Gorton, I., Hawryszkiewicz, I., Ragoonaden, K., Chung, C., Lu, S., and Randhawa, G. Groupware Support Tools for Collaborative Software Engineering. In *Proceedings of the 30<sup>th</sup> IEEE International Conference on System Sciences*. (1997), 157-166.

- [25] Herbsleb, J., and Grinter, R., Splitting the Organization and Integrating the Code: Conway's Law Revisited. In Proceedings of the 1999 international conference on Software Engineering. (1999), 85-95.
- [26] Grudin, J. Groupware and social dynamics: Eight challenges for developers. *Communications of the ACM* 37, 1 (January 1994), 93-105.
- [27] Mandviwalla, M. and Olfman, L. What do groups need? A proposed set of generic groupware requirements. *ACM Transactions on Computer-Human Interaction* 1, 3 (September 1994), 245-268.
- [28] Henry, S. and Stevens, K.T. Using Leadership Roles to improve Team Effectiveness: An Empirical Investigation. In *Journal of Systems and Software* 44 (1999), 241-250.
- [29] Belbin, R. Meredith, *Management Teams*, John Wiley & Sons, New York, 1981.
- [30] Stevens, K. Todd. PhD dissertation, Department of Computer Science, Virginia Tech, March 1998.
- [31] Isenhour, P, "The MOOsburg Project" available from: <http://linc.cs.vt.edu/moosburg> Internet accessed 10 October 2000.
- [32] Centra, "Centra Products and Services – Centra Symposium" available from: <http://www.centra.com/products/symposium/info.asp> Internet accessed 10 October 2000.
- [33] Centra, "Centra Products and Services – Centra eMeeting" available from: <http://www.centra.com/products/emeeting/info.asp> Internet accessed 10 October 2000.
- [34] Microsoft, "NetMeeting Home" available from: <http://www.microsoft.com/windows/netmeeting/> Internet accessed 10 October 2000.
- [35] Teamwave Software, "TeamWave Software – Workplace Overview" available from: <http://www.teamwave.com/advantages/index.html> Internet accessed: 10 October 2000.
- [36] Meetingworks, "MeetingWorks for Windows" available from: [http://www.entsol.com/html/meetingworks\\_for\\_windows.html](http://www.entsol.com/html/meetingworks_for_windows.html) Internet accessed: 10 October 2000.
- [37] CUseeMe Networks, "CUseeMe Networks – CUseeMe Pro" available from: <http://www.cuseeme.com/products/cuseemepro.htm> Internet accessed: 28 March 2001.
- [38] Karolak, D. *Global Software Development: Managing Virtual Teams and Environments*. IEEE Computer Society, Los Alamitos, CA, 1998.



[39] Herela, D., Eberlein, A., Shaw, M., Gaines, B., Group Performance and distributed requirements negotiations. Networked Computer Science Technical Reports Library, University of Calgary, March 2000. Available from [http://pharos.cpsc.ucalgary.ca/Dienst/Repository/2.0/Body/ncstrl.ucalgary\\_cs/2000-650-02/pdf](http://pharos.cpsc.ucalgary.ca/Dienst/Repository/2.0/Body/ncstrl.ucalgary_cs/2000-650-02/pdf) Internet accessed: 10 October 2000.

[40] Guinta, L. R., and Praizler, N.C. The QFD Book, The Team Approach to Solving Problems and Satisfying Customers Through Quality Function Deployment. AMACOM Books. 1993.

[41] Eyematic Interfaces Inc., “iVisit > home” available from: <http://www.ivisit.com/> Internet accessed: 28 March 2001.

[42] Gorton, I., and Motwani, S., Towards a Methodology for 24 Hour Software Production Using Geographically Separated Teams, in Proceedings of the First IFIP International Conference on Software Quality and Productivity, Hong Kong, December 5-7<sup>th</sup>, Chapman and Hall, pages 50-55, 1994.

[43] Veinott, E., Olson, J., Olson, G., and Fu, X., Video Helps Remote Work: Speakers Who Need to Negotiate Common Group Benefit from Seeing Each Other, in Proceedings of the CHI 99 conference on Human factors in computing systems, Pittsburgh, PA, May 15-20, pages 302-309, 1999.

[44] Markus, M.L. 1994. Finding a happy medium: Explaining the negative effects of electronic communication on social life at work. ACM Transactions on Information Systems, 12, 2 (February 1994), 119-149.

## Appendix A. Groupware System Function List

Here we present a wish list of functions that an integrated group system could provide to support distributed requirements engineering. This list of functions is derived from participant feedback in the empirical study. Although this list is not a mature requirements specification, it provides general features that should be specified in a requirements specification of a groupware system. Fine details of these features could be studied in requirements analysis and polished by software design.

1. Integrated Groupware System
  - a. Support for synchronous virtual meetings and collaboration
  - b. Support for asynchronous meetings and collaboration
2. Shared Files (document repository)
3. Remote Access
4. Adequate Performance for all functions
  - a. No delays to due limited network bandwidth.
  - b. Actual specifications dependant upon implementation
5. Shared document editor with user awareness features
6. Meeting Agenda / slide presentation
7. Desktop Integration
8. Reliability: System errors should not reduce usability.
  - a. Error handling should prevent system crashes.
  - b. Audio subsystem should be reliable.
  - c. Actual specifications dependent upon implementation.

9. Intuitive Security Configuration,
  - a. Restrict and control access to shared artifacts.
10. Provide cross platform support
  - a. PCs
  - b. Macintosh
11. Intuitive Navigation through system, with easy to understand well organized GUI.
12. No Folders to organize shared files and artifacts.
13. Textual mode for navigating shared files and artifacts.
14. Automatic notification to users about changes within artifacts
  - a. Groupware System Integration with user's desktop
  - b. Change notifications through email or desktop messages.
15. Voice Conferencing with high quality audio
16. Voice no delays / echoes in audio conferencing.
17. Text Chat as an integrated window, not a popup in the GUI.
18. Intuitive Floor Control for synchronous meetings and artifact sharing.
19. Graphical Avatars (pictures of users) when not using video conferencing
20. Point to point Videoconferencing for virtual meetings.
21. Ability to record and playback of virtual meetings.
22. Application Sharing, with fast performance
23. Shared Whiteboard
24. Shared Web Browser
25. Private audio chat among participants in virtual meeting

- a. Could be extended to include private video conferencing chat
26. Easy intuitive method for requesting floor control in virtual meetings
- a. Hand raising
  - b. Possibly other methods
27. Voting Features
28. Support a single facilitator for virtual meetings.
29. Flat Learning Curve. System should be fast and easy to learn how to use
30. Full-duplex audio for voice conferencing.
31. Indexing method for meeting recordings.
32. Awareness of changes throughout system: text chat, whiteboard, shared documents regardless of what activities user is currently performing.
33. Instant playback of previous audio clip to catch lost points in virtual meetings
34. Ability to sort and edit slides, Non-linear navigation of slides
35. Integrated email capability
36. Notification and tracking of when user reads email messages.
37. Ability to query who has\has not read messages.
38. Ability to send email messages to entire project group or individual
39. Tagging of email messages to remind user to that a response is required.
40. Ability to attach or pass large files and artifacts via email
41. Threaded discussion board

## Appendix B. System Overview Document

---

The System Overview Document was presented to the customers at the beginning of the empirical study. The document describes the desired functionality for the company wide scheduling system that the customers want developed. This document was only available to the customers, and the necessity to not disclose this document to the requirements engineers was stressed.

### **System Overview**

The XYZ Company is a small company employing approximately 100 people. It is a multi-disciplinary engineering firm that employs engineers, project managers, secretaries, accountants, and administrative personnel. XYZ Company is housed in a modest 2-story office building with a fixed number of meeting rooms, and audio/video/multimedia equipment.

The active projects at XYZ Company often require meeting space and facilities to conduct various ongoing meetings at the company. The impromptu nature of meetings has created headaches for the secretarial staff who often becomes frustrated at the over abundance of requests for meeting rooms. Often the needs for meeting space cannot be met for impromptu gatherings.

Meetings at XYZ Company are further challenged by the unavailability of audio/visual equipment. Secretaries often loose track of which engineer or project manager last had possession of computer projection units. When such devices are required at meetings often they cannot be acquired because of poor scheduling and management of these limited resources. Several of the audio/visual devices used at XYZ company meetings are extremely expensive, and it is unfeasible to buy additional devices to solve the audio/visual equipment resource problem.

Of late XYZ company engineers have become very busy and are assigned to multiple projects at a time. The competitive business environment for the XYZ Company has forced the engineering staff to take on several development projects simultaneously. This has created scheduling problems for XYZ Company's engineers. They often have difficulties managing their schedule, and coordinating meetings with clients, project managers, and development engineers. The unavailability for XYZ Company engineers to meet with clients can lead to business losses for the XYZ Company.

Because the environment at the XYZ company recently has become nothing short of chaotic, the XYZ vice presidents and senior managers have decided that the XYZ Company should implement a company wide personal planner and meeting scheduling

system to help coordinate and manage meeting rooms, devices, and available engineers.

- The scheduling system should be web based with a client interface that runs on any system over the corporate intranet.
- Users of the system will have different permission levels when using the scheduling system. All users should be able to view the schedule of a meeting room, or meeting device. The ability to view an individual person's schedule should be restricted based on the corporate management flow chart.
- Different users should be allowed to schedule rooms, and devices, and request XYZ employees to be in attendance at meetings.
- Administrative users, which could be high-level management and company vice presidents, should be able to schedule meetings with any XYZ company employee's schedule, overriding scheduling conflicts as needed.
- Events are scheduled by the month, day, and year.
- For scheduling events the system can request particular XYZ employees to attend. The scheduler also requests meeting rooms, and any audio/visual/multimedia devices that are needed.
- The system will report scheduling conflicts with people, rooms, and devices.
- The system will perform automatic conflict resolution on request. "High power" administrative users can override conflicts as needed.
- The system should support a facility for adding and editing information about employees, meeting rooms, and devices within the system.
- It should be possible to view the schedule for a person, room, or device.

Examples of audio/visual/multimedia devices are items such as a slide projector, overhead, computer projection system, laptop computer, etc.

## Appendix C. Customer Role Descriptions

---

Here are the customer role descriptions that were given to the customers at the start of the empirical study. They describe the details of each employee's role in the company and set up a number of initial conflicts between the customers with respect to the functional requirements of the company wide scheduling system.

**WARNING:** Under no condition are you to disclose the details of your role description to anyone. You are to disclose and bring life to your assigned role, but you may not communicate the details provided here to anyone. NO EXCEPTIONS.

### **Role Description: XYZ Company Vice President of Development**

As a company vice president you will not think in terms of system implementation or coding techniques. You will however think in terms of technology choices and buzzwords. Like most upper level management you are vaguely familiar with recent computer technology, but you don't fully understand how things work. You are interested in things like "cross-platform", "intranet wide usage", "web client", "client/server", "Java". Although you don't fully understand the implications of these technology choices, you think they are good.

You and the other vice presidents at XYZ Company proposed the company wide personal planner and meeting scheduling system. The functions that are outlined in the system description are from your original proposal. You strongly favor implementing all features that are outlined in the proposal. You are against removing or changing the details about the original features. However, you are accepting of new features to the proposed system that will increase functionality and usability. You are concerned about too many features only if it increases the cost of purchasing the system.

Your primary motivation for proposing the company wide personal planner and meeting scheduling system is to reduce cost and increase efficiency with regards to scheduling and conducting business and client meetings at the XYZ company. You believe that the system will be a long-term solution that has many benefits over short-term solutions such as purchasing additional audio/visual/multimedia equipment.

You are aware that some employees of the XYZ Company see the implementation of a company wide personal planner and meeting scheduling system as precursor to eliminate excess secretarial staff at the company. You have no intention to reduce the secretarial headcount at the company, but rather see the implementation of the system as a way to free up the secretarial staff to more interesting and productive tasks. You do not think that the secretarial staff should serve as administrators of the system, as they should be not be overly involved with the system.

WARNING: Under no condition are you to disclose the details of your role description to anyone. You are to disclose and bring life to your assigned role, but you may not communicate the details provided here to anyone. NO EXCEPTIONS.

### **Role Description: XYZ Company Secretary**

As a secretary you will absolutely not think in terms of system implementation, technology choices, or coding techniques for the system. You are only concerned with what the product will do and what your role will be using it. You are also interested in simplistic systems that accomplish their desired tasks without overly complex and sophisticated user interfaces.

As a secretary of XYZ Company you have been in charge of scheduling meeting rooms and reserving audio/visual/multimedia equipment in the past. You have kept a schedule book for each meeting room and device and required that engineers and administrators contact you to sign up for meeting times and audio/visual/multimedia equipment. Occasionally people would not schedule a meeting room and would use a meeting room that was not reserved. This would cause problems when others tried to use a meeting room during their reserved time. You've also had problems with engineers and administrators not returning the audio/visual/multimedia equipment to the equipment room. A few computer projectors and laptops have been missing occasionally when people failed to return them. These employees would then attend business trips or vacations, often leaving the equipment locked in an office or at home.

As a secretary of the XYZ Company you have taken a great deal of blame for the mismanagement of rooms and equipment. You feel that one of the reasons that the XYZ Company is developing the personal planner and meeting-scheduling system is to reduce the required needs for secretarial staff at XYZ Company. Because you feel that this software system could eventually lead to the elimination of one or two secretarial positions, you are strongly interested in becoming the system administrator of the personal planner and meeting scheduling system. You also think that the company's administrative vice presidents may have considered developing the system because of the perceived poor job that the secretarial staff has done. You feel that although the system promises to better coordinate and manage resources within the company, you feel that it will not solve many of the problems that have led to the failure of the existing manual reservation system.

You are interested in maintaining and controlling the scheduling for meeting rooms and audio/video/multimedia devices. You feel that others should not be able to see the schedule for meeting rooms or devices, as they should go through a reservation process with a company secretary. You would also like the system to be extremely easy to use. Others should be able to request meeting rooms and equipment, but you as the system administrator and secretary will need to approve and manage these requests.

You are against implementing automatic conflict resolution for events. You believe that this feature just will not work in practice. You would rather have conflicting events go to a system administrator, and then the administrator will decide how to resolve conflicts.

You are even interested in having some control over engineer's schedules. You think that any outside meetings with customers and clients should be arranged through



secretarial staff. You feel that the engineers are too busy and don't have time to negotiate with customers and clients to arrange for meetings at XYZ company.

WARNING: Under no condition are you to disclose the details of your role description to anyone. You are to disclose and bring life to your assigned role, but you may not communicate the details provided here to anyone. NO EXCEPTIONS.

### **Role Description: XYZ Company Engineer**

As an engineer you are interested in feature rich, and highly customizable systems to help you accomplish your daily work. You frequently offer ideas for features and customizations that will make the scheduling system more flexible for all that use it. You are somewhat interested in the specific details of the implementation of the scheduling system, and will not hesitate to suggest solutions to the engineers who have been hired to develop the system.

As a XYZ company engineer you have in the past managed your schedule with clients and engineers using a day planner or palm pilot. You have occasionally become frustrated at the mismanagement and lack of meeting rooms and audio/visual/multimedia equipment for having meetings at XYZ Company. You feel that often meetings with customers are not productive because you lack good quality computer projection systems to demonstrate system prototypes.

Although the XYZ Company has several laptops and projection systems you feel that successfully reserving and acquiring one for a meeting is about as easy as winning the weekly lottery. You and your cohorts think that the XYZ Company should either buy many more laptops and projection systems, or they should replace the secretarial staff that seems to constantly lose track and befuddle reservations for equipment.

When the XYZ Company vice presidents proposed the personal planner and meeting scheduling system you were excited. Although you think that the system will not solve all of the resource problems for meeting rooms and equipment, you feel that the reduction on the dependency on secretarial staff to plan and arrange meetings may help reduce the resource conflicts. You also believe that being able to schedule and reserve meeting rooms and devices online on the corporate intranet will save time. You are excited about the conflict resolution features proposed for the scheduling system.

These are some of your interests regarding the scheduling system:

- You don't think that high-level management and company vice presidents should be able to override your schedule without permission. You think that this is disrespectful of management to simply disregard your schedule and force you to attend company business meetings without your approval. You also think meetings with customers should be a higher priority than company business meetings.
- You think that you should be able to view the personal schedule of any XYZ company employee except schedules of vice-presidents and high-level managers. You don't think that your ability to view schedules of others should be limited by your position on the company organization chart. You feel that in order to resolve scheduling conflicts you should be able to view other's schedules.

- You and your cohorts would like the personal planner and meeting scheduling system to interface with palm pilot organizers. Your cohorts would also like the system to print out schedules for Franklin daily planners.

## Distributed Requirements Analysis Final Survey



---

This is the final survey for the distributed requirements analysis project. This survey should take about 30 minutes or less to complete. You need to complete the survey in a single session. Thanks for taking this survey! Your feedback is extremely valuable and appreciated! Completed surveys will be entered into a drawing to win gift certificates to the restaurant of the winner's choice.

1. Please enter your name: (Required)
2. Select the role title that applies to you: (Required)
  - CS 5704 Requirements Engineer
  - CS 5734 Secretary
  - CS 5734 Vice President
  - CS 5734 Engineer
3. What group were you in? (Required)
  - Group 1 - RH & Associates
  - Group 2 - Centra Systems
  - Group 3 - Harmony Technologies
  - Group 4 - Snafoo Inc
  - Group 5 - Omnilink.com
  - Group 6 - Farrago Studios
4. Was the number and length of virtual meetings adequate? (Required)

- Yes
- No

5. If not, what improvements would be desirable with regards to the number of meetings and length of meetings?

---

How much did you contribute to the ENTIRE PROJECT relative to your peers?  
Choose from a scale of 1 to 5, where 1 = "low participation" and 5 = "high participation".

6. Please Rate

- low participation 1 2 3 4 5 high participation  n/a

---

For the next 16 questions: How much influence did these factors have in your overall participation level in the project?  
Choose from a scale of 1 to 5, where 1 = "low influence" and 5 = "high influence".

7. Subject Knowledge (+/-)

low influence 1 2 3 4 5 high influence  no influence

8. Preparation Time (+/-)

low influence 1 2 3 4 5 high influence  no influence

9. Assigned Role in Meetings

low influence 1 2 3 4 5 high influence  no influence

10. Physical Energy Level

low influence 1 2 3 4 5 high influence  no influence

11. Usability of MOOsburg

low influence 1 2 3 4 5 high influence  no influence

12. Usability of Centra Symposium

low influence 1 2 3 4 5 high influence  no influence

13. Quality of Computer Workstation for meetings

low influence 1 2 3 4 5 high influence  no influence

14. Arrival Time/Attendance at Meetings

low influence 1 2 3 4 5 high influence  no influence

15. Quality of Network Bandwidth During Meetings

low influence 1 2 3 4 5 high influence  no influence

16. My Own Personality Attributes (for example: assertive versus shy)  
low influence 1 2 3 4 5 high influence  no influence

17. Comfort level with English Language  
low influence 1 2 3 4 5 high influence  no influence

18. Well Coordinated Agenda / Meetings  
low influence 1 2 3 4 5 high influence  no influence

19. Influence of Software Engineers  
low influence 1 2 3 4 5 high influence  no influence

20. Influence of Customers  
low influence 1 2 3 4 5 high influence  no influence

21. Influence of Course Professor  
low influence 1 2 3 4 5 high influence  no influence

22. Influence of Experimenter Wes  
low influence 1 2 3 4 5 high influence  no influence

---

For the next 14 questions: Please indicate tools used in the project and your overall impression of the usefulness of each.

Choose from the following options:

1 = Disruptive

2 = Not Useful

3 = Possibly Useful

4 = Useful

5 = Very Useful

Not Used = Not Used

23. Centra Symposium Voice Conferencing  
1 2 3 4 5 Not Used

24. Centra Symposium Text Chat  
1 2 3 4 5 Not Used

25. Centra Symposium Whiteboard  
1 2 3 4 5 Not Used

26. Centra Symposium Agenda/Slide Show  
1 2 3 4 5 Not Used

27. Centra Symposium Web Browser Sharing  
1 2 3 4 5 Not Used

28. Centra Symposium Application Sharing

1 2 3 4 5 Not Used

29. MOOsburg Text Chat

1 2 3 4 5 Not Used

30. MOOsburg Message Board

1 2 3 4 5 Not Used

31. MOOsburg Whiteboard

1 2 3 4 5 Not Used

32. MOOsburg Shared File

1 2 3 4 5 Not Used

33. MOOsburg Web Links

1 2 3 4 5 Not Used

34. MOOsburg List Editor

1 2 3 4 5 Not Used

35. MOOsburg Planner

1 2 3 4 5 Not Used

36. Group List Server

1 2 3 4 5 Not Used

---

For the next 8 questions: How effective were various requirements analysis methods used in the project?

Choose from the following options:

1 = Not Effective

2 = Possibly/Slightly Effective

3 = Effective

4 = Very Effective

5 = Outstanding Effectiveness

Not Used = Not Used

37. Question/Answer method (Engineers ask Questions, Customer Answers)

1 2 3 4 5 Not Used

38. Customer Interviews

1 2 3 4 5 Not Used

39. Brainstorming / Idea Reduction (Group brainstorms, then the idea set is reduced through some method)

1 2 3 4 5 Not Used

40. Storyboards (non-live prototype, just pictures or screen shots shown in example)  
1 2 3 4 5 Not Used
41. Prototyping (real live prototype which can be navigated)  
1 2 3 4 5 Not Used
42. Questionnaires (list of questions that customer responds to usually with a written response)  
1 2 3 4 5 Not Used
43. Use Cases  
1 2 3 4 5 Not Used
44. Requirements Management (ranking and prioritization of requirements)  
1 2 3 4 5 Not Used
45. Have you ever met in person the virtual members of the project? (Required)  
 Yes  
 No
46. Did you ever discuss the requirements analysis project with the virtual project members face-to-face? (Required)  
 Yes  
 No
47. If yes, Please describe briefly the meetings or discussions that took place:
48. In your group, did meetings occur in MOOsburg? (Required)  
 Yes  
 No
49. Did you participate in these meetings? (Required)  
 Yes  
 No
50. Were MOOsburg meetings: (Required)  
 Planned/Scheduled  
 Impromptu/Random  
 Both  
 No Meeting Occurred
51. If there were meetings in MOOsburg, please summarize what was accomplished during these meetings?



52. If meetings did NOT occur in MOOsburg please explain why.

---

Please Rate

Choose from a scale of 1 to 5, where 1 = "Highly Ineffective" and 5 = "Highly Effective".

53. How Effective/Ineffective was MOOsburg at enhancing the requirements analysis process?  
Highly Ineffective  1  2  3  4  5 Highly Effective  n/a

54. What were some positive/negative features of MOOsburg that affected the requirements analysis?

---

Please rate

Choose from a scale of 1 to 5, where 1 = "Very narrow creativity" and 5 = "Very Extensive creativity".

55. How extensive was your group's exploration and use of the power and features of MOOsburg?

Very narrow creativity  1  2  3  4  5 Very Extensive creativity  n/a

56. In your group, was the list server email account used? (Required)

- Yes
- No

57. Did you, personally, send email to the list server email account? (Required)

- Yes
- No

58. If the list server email account was used, please summarize what was accomplished through its use. (Please list tasks performed)

59. If the list server email account was not used, please explain why.

60. Did you communicate with virtual project members using email sent directly to them and not through the listserv? (Required)

- Yes
  - No
- 

Please Rate

Choose from a scale of 1 to 5, where 1 = "Highly Ineffective" and 5 = "Highly Effective".

61. How Effective/Ineffective was email at enhancing the requirements analysis process?  
Highly Ineffective 1 2 3 4 5 Highly Effective  n/a

62. What were some positive/negative features of Email that affected the requirements analysis?

---

Please Rate

Choose from a scale of 1 to 5, where 1 = "Highly Ineffective" and 5 = "Highly Effective".

63. How Effective/Ineffective was Centra Symposium at enhancing the requirements analysis process?  
Highly Ineffective 1 2 3 4 5 Highly Effective  n/a

64. What were some positive/negative features of Centra Symposium that affected the requirements analysis?

---

Please rate

Choose from a scale of 1 to 5, where 1 = "Very narrow creativity" and 5 = "Very Extensive creativity".

65. How extensive was your group's exploration and use of the power and features of Centra Symposium?

Very narrow creativity 1 2 3 4 5 Very Extensive creativity  n/a

66. If you were contracted to perform a requirements analysis with a remote client would you use the tools provided to you in this experiment to facilitate your requirements engineering process? (Required)

- Yes
- No

67. If you were contracted to perform a requirements analysis with a remote client would you use Centra Symposium to facilitate your requirements engineering process? (Required)

- Yes
- No

68. Would you use MOOsburg to facilitate your requirements engineering process? (Required)

- Yes
- No

69. Would you use a EMail to facilitate your requirements engineering process? (Required)

- Yes
- No

70. What would you change about Centra Symposium if you really needed to conduct an effective requirements analysis with a remote customer?

71. What would you change about MOOsburg?

72. What would you change about Email usage?

73. Please explain why you would recommend these changes.

74. What additional functions and tasks need to be supported in the overall system?

---

QUESTION FOR CUSTOMERS ONLY: Please rate  
Choose from a scale of 1 to 5, where 1 = "At least one page" and 5 = "All Pages".

75. CUSTOMERS ONLY: Did you read the draft software requirements specification prior to Virtual meeting #4?

At least one page 1 2 3 4 5 All Pages  Did not read

76. CUSTOMERS ONLY: Did you mark errors or issues in the draft software requirements specification prior to virtual meeting #4?

- Yes
  - No
- 

For the next 2 questions: QUESTION FOR CUSTOMERS ONLY: Please rate  
Choose from a scale of 1 to 5, where 1 = "Low Satisfaction" and 5 = "High Satisfaction".

77. CUSTOMERS ONLY: Were you satisfied with the requirements document developed by the engineers?

Low Satisfaction 1 2 3 4 5 High Satisfaction  Did not read

78. CUSTOMERS ONLY: Were you satisfied with the overall performance of the engineering group?

Low Satisfaction 1 2 3 4 5 High Satisfaction  Did not read

79. CUSTOMERS ONLY: Please provide positive/negative feedback about the engineers that your virtual company hired:

80. SOFTWARE ENGINEERS ONLY: What were the 3 most difficult requirements to develop and clarify?

81. SOFTWARE ENGINEERS ONLY: What caused these 3 requirements to be difficult to develop?

82. SOFTWARE ENGINEERS ONLY: How did the distributed software (Centra Symposium, MOOsburg, EMAIL) effect elicitation and clarification of these difficult requirements?

---

QUESTION FOR SOFTWARE ENGINEERS ONLY: Please rate  
Choose from a scale of 1 to 5, where 1 = "Very Poor Simulation" and 5 = "Very Good Simulation".

83. SOFTWARE ENGINEERS ONLY: How well do you feel this experiment simulated a real requirements analysis project with a remote customer?

Very Poor Simulation 1 2 3 4 5 Very Good Simulation  n/a

84. SOFTWARE ENGINEERS ONLY: What factors made the simulation realistic:

85. SOFTWARE ENGINEERS ONLY: What factors made the simulation unrealistic:

---

For the next 2 questions: QUESTION FOR SOFTWARE ENGINEERS ONLY: Please rate  
Choose from a scale of 1 to 5, where 1 = "Low Participation" and 5 = "High Participation".

86. SOFTWARE ENGINEERS ONLY: Please classify the customers' participation level in  
Symposium virtual meetings.  
Low Participation 1 2 3 4 5 High Participation  n/a

87. SOFTWARE ENGINEERS ONLY: Please classify the customers' participation level off-line  
from the virtual meetings.  
Low Participation 1 2 3 4 5 High Participation  n/a

---

QUESTION FOR SOFTWARE ENGINEERS ONLY: Please rate  
Choose from a scale of 1 to 5, where 1 = "Low Cooperation" and 5 = "High Cooperation".

88. SOFTWARE ENGINEERS ONLY: Please classify the customers' overall cooperation level  
throughout the requirements analysis process.  
Low Cooperation 1 2 3 4 5 High Cooperation  n/a

89. SOFTWARE ENGINEERS ONLY: How did the customers' participation in the project effect  
(good or bad) the outcome of the requirements analysis? Please Comment!

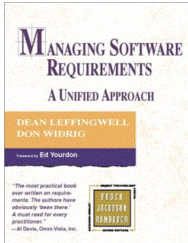

Thanks for taking this survey! Thanks for your participation and cooperation through this project!

-----  
Send comments about this questionnaire to Wes J. Lloyd  
-----

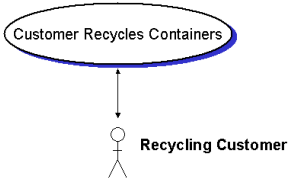
This page generated by WhizQuest v1.1 on 21-Jun-2001 at 00:44 am  
Developed at ISIS, Virginia Tech  
Contact Developers

Copyright © 1997-2001 Virginia Tech, All Rights Reserved.

# Appendix E. Requirements Analysis Slides

<h2>Requirements Engineering Process</h2>	<h2>Reference</h2>  <ul style="list-style-type: none"> <li>■ Managing Software Requirements</li> <li>■ Dean Leffingwell</li> <li>■ Don Widrig</li> <li>■ © 2000 Addison-Wesley</li> <li>■ Chapters 7-15,23-26,30</li> </ul>
<h2>Requirements Engineering Process</h2> <ul style="list-style-type: none"> <li>■ PHASE 1             <ul style="list-style-type: none"> <li>■ Capture High-Level User Requirements</li> </ul> </li> <li>■ PHASE 2             <ul style="list-style-type: none"> <li>■ Move from High Level Elicitation to detailed requirements understanding</li> <li>■ Analyze and Manage requirements</li> <li>■ Construct and demonstrate prototypes (interface mockups)</li> <li>■ Use Case Modeling</li> </ul> </li> </ul>	<h2>Requirements Engineering Process</h2> <ul style="list-style-type: none"> <li>■ PHASE 3             <ul style="list-style-type: none"> <li>■ Write the Requirements Specification Document</li> </ul> </li> <li>■ PHASE 4             <ul style="list-style-type: none"> <li>■ Verification and revision of specification document</li> </ul> </li> </ul>
<h2>Meeting Etiquette</h2> <ul style="list-style-type: none"> <li>■ Establish professional objective tone of meeting</li> <li>■ Start and stop meeting on time</li> <li>■ Establish/Enforce rules of the meeting</li> <li>■ Present goals and agenda for meeting</li> <li>■ Facilitate decision making</li> <li>■ Avoid participating in content</li> <li>■ Make stakeholders participate</li> <li>■ Eliminate disruptive behavior</li> <li>■ Make available meeting notes after each meeting             <ul style="list-style-type: none"> <li>■ Record what was decided</li> </ul> </li> </ul>	<h2>Capturing User Requirements</h2> 
<h2>PHASE 1: Capturing User Requirements</h2> <ul style="list-style-type: none"> <li>■ Requirements Elicitation             <ul style="list-style-type: none"> <li>■ Sit down with users and ask them what they need the system to do.</li> <li>■ No problem, right?</li> </ul> </li> <li>■ "Yes, But" Syndrome             <ul style="list-style-type: none"> <li>■ First time users see a system they truly understand what you meant, this brings to mind many new features and possibilities</li> </ul> </li> </ul>	<h2>Requirements Elicitation</h2> <ul style="list-style-type: none"> <li>■ "Undiscovered Ruins" Syndrome             <ul style="list-style-type: none"> <li>■ The more requirements found, the more that may remain.</li> <li>■ How do you know when you've found all of the requirements?</li> </ul> </li> <li>■ "User and Developer" Syndrome             <ul style="list-style-type: none"> <li>■ Communication gap between users and developers</li> </ul> </li> </ul>

<h2>Requirements Elicitation</h2> <ul style="list-style-type: none"> <li>■ Techniques <ul style="list-style-type: none"> <li>■ Interviewing and questionnaires <ul style="list-style-type: none"> <li>■ Use context free questions <ul style="list-style-type: none"> <li>■ Avoid Prejudicing user responses</li> </ul> </li> <li>■ Once problem domain is defined use questions within solution context <ul style="list-style-type: none"> <li>■ Gives users insights to what is possible</li> </ul> </li> <li>■ Use 1 on 1 interviews with customers</li> </ul> </li> </ul> </li> </ul>	<h2>Elicitation Techniques</h2> <ul style="list-style-type: none"> <li>■ Requirements Workshops <ul style="list-style-type: none"> <li>■ 1-2 day big meeting with all project stakeholders</li> </ul> </li> <li>■ Roleplaying <ul style="list-style-type: none"> <li>■ Study and learn enough to take on the role of the user</li> </ul> </li> <li>■ Brainstorming and idea reduction <ul style="list-style-type: none"> <li>■ No criticism or debate</li> <li>■ Generate as many ideas as possible</li> <li>■ Mutate and combine ideas</li> </ul> </li> </ul>
<h2>Brainstorming and idea reduction</h2> <ul style="list-style-type: none"> <li>■ Idea Reduction mediated by facilitator <ul style="list-style-type: none"> <li>■ Pruning <ul style="list-style-type: none"> <li>■ Trimming ideas not worth further investigation</li> </ul> </li> <li>■ Grouping <ul style="list-style-type: none"> <li>■ Group ideas by similar category</li> </ul> </li> </ul> </li> <li>■ Feature Definition <ul style="list-style-type: none"> <li>■ Take brainstormed ideas and define them more fully</li> </ul> </li> </ul>	<h2>Storyboards</h2> <ul style="list-style-type: none"> <li>■ Used for specification of Human Machine Interface</li> <li>■ Passive <ul style="list-style-type: none"> <li>■ Manual display of pictures, screen shots, slides</li> <li>■ Presents a walkthrough explanation of the system</li> <li>■ Shows how system responds under certain conditions</li> <li>■ Non-interactive, user can't propose what-if questions</li> </ul> </li> </ul>
<h2>Storyboards cont'd</h2> <ul style="list-style-type: none"> <li>■ Active <ul style="list-style-type: none"> <li>■ Automated display of media</li> <li>■ Presents a walkthrough of typical usage/operational scenarios of the system</li> <li>■ A non-interactive movie</li> </ul> </li> </ul>	<h2>Storyboards cont'd</h2> <ul style="list-style-type: none"> <li>■ Interactive <ul style="list-style-type: none"> <li>■ User participates</li> <li>■ Simulation or mockup of system</li> <li>■ Basically like a throwaway prototype</li> <li>■ User experiences end system operation in a realistic manner</li> </ul> </li> </ul>
<h2>Use Cases</h2> <ul style="list-style-type: none"> <li>■ A Use case describes a set of actions and variant actions that a system performs producing observable results to an actor(s).</li> <li>■ Symbols: <div style="display: flex; justify-content: space-around; align-items: center; margin-top: 10px;"> <div style="text-align: center;">  <p>Actor</p> </div> <div style="text-align: center;">  <p>Use Case</p> </div> </div> </li> </ul>	<h2>Use Cases</h2> <ul style="list-style-type: none"> <li>■ Actor <ul style="list-style-type: none"> <li>■ Someone or something that interacts with the system <ul style="list-style-type: none"> <li>■ Users</li> <li>■ Devices</li> <li>■ Other Systems</li> </ul> </li> </ul> </li> </ul>

<h3>Example Use Case</h3> <ul style="list-style-type: none"> <li>■ Name <ul style="list-style-type: none"> <li>■ Recycle containers</li> </ul> </li> <li>■ Brief Description <ul style="list-style-type: none"> <li>■ The customer inserts cans and bottles into the recycling machine, presses a button, and receives a printed receipt that can be exchanged for money.</li> </ul> </li> </ul>	<h3>Creating Use Cases</h3> <ul style="list-style-type: none"> <li>■ Give all actors unique names.</li> <li>■ Give the use case a unique name.</li> <li>■ Write a brief description <ul style="list-style-type: none"> <li>■ An informal overview of the functionality</li> </ul> </li> <li>■ Define the flow of events <ul style="list-style-type: none"> <li>■ Textual description of the operations by the actor, with the system's various responses.</li> <li>■ Only specifies what happens, not how the system performs the work</li> </ul> </li> </ul>
<h3>Creating Use Cases cont'd</h3> <ul style="list-style-type: none"> <li>■ Alternative flows <ul style="list-style-type: none"> <li>■ Describe system response to various system conditions Error Conditions</li> </ul> </li> <li>■ Identify Preconditions and Postconditions <ul style="list-style-type: none"> <li>■ Only necessary to clarify the system behavior</li> </ul> </li> <li>■ State special requirements if any</li> </ul>	<h3>Use Cases</h3> <ul style="list-style-type: none"> <li>■ Excellent for identifying functions for systems with many users and interfaces.</li> <li>■ Excellent for systems using UML and OO techniques for implementation</li> <li>■ Poor for systems with few users and interfaces.</li> <li>■ Poor for systems dominated by nonfunctional requirements</li> </ul>
<h3>Example: Use Case</h3>  <p>The diagram shows a use case named "Customer Recycles Containers" represented by an oval. Below it is an actor named "Recycling Customer" represented by a stick figure. A vertical line with an arrowhead at the top connects the actor to the use case, indicating that the actor is associated with the use case.</p>	<h3>Example: Use Case</h3> <ul style="list-style-type: none"> <li>■ Name <ul style="list-style-type: none"> <li>■ Customer recycles containers</li> </ul> </li> <li>■ Brief Description <ul style="list-style-type: none"> <li>■ The customer inserts cans and bottles into the recycling machine, presses a button, and receives a printed receipt that can be exchanged for money.</li> </ul> </li> </ul>
<h3>Example: Use Case</h3> <ul style="list-style-type: none"> <li>■ Flow of events <ol style="list-style-type: none"> <li>1. The customer inserts aluminum cans or glass bottles into the recycling machine.</li> <li>2. The customer presses the end batch button.</li> <li>3. The printer device prints a receipt detailing the value of the exchange.</li> <li>4. The customer removes the receipt from the machine and the transaction is complete.</li> </ol> </li> </ul>	<h3>Example: Use Case</h3> <ul style="list-style-type: none"> <li>■ Alternate Flow of Events <ul style="list-style-type: none"> <li>· If the printer device is out of receipt paper the following actions occur: <ol style="list-style-type: none"> <li>1. The customer inserts aluminum cans or glass bottles into the recycling machine.</li> <li>2. The recycling machine displays a message that it is out of receipt paper and it can't accept containers. The customer is advised to see the customer service manager.</li> <li>3. The recycling machine rejects glass bottles and aluminum cans into the return bin.</li> </ol> </li> </ul> </li> <li>■ Many more Alternate Flows are possible</li> </ul>


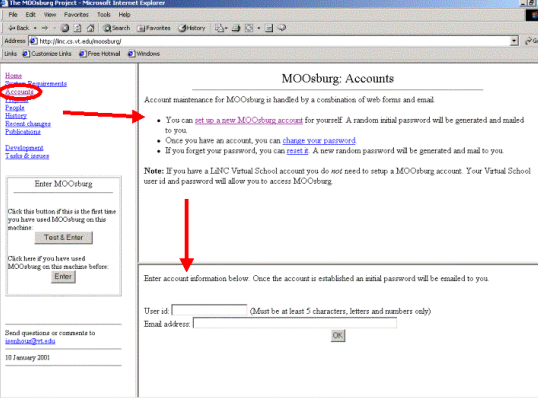


<h3>Example: Use Case</h3> <ul style="list-style-type: none"> <li>■ Preconditions <ul style="list-style-type: none"> <li>■ The recycling machine must be powered-on.</li> <li>■ The recycling machine must have empty space in the storage bin.</li> </ul> </li> <li>■ Postconditions <ul style="list-style-type: none"> <li>■ The recycling machine logs all transactions to memory.</li> </ul> </li> </ul>	<h3>Prototyping</h3> <ul style="list-style-type: none"> <li>■ Prototype the areas of the system that are most "fuzzy"</li> <li>■ Prototype construction should be fast, efficient, and require minimal effort.</li> <li>■ Types of Prototypes <ul style="list-style-type: none"> <li>■ Throwaway * <ul style="list-style-type: none"> <li>■ Built quickly using shortcuts, and alternate technologies, solely for the purpose of gaining knowledge</li> </ul> </li> <li>■ Evolutionary <ul style="list-style-type: none"> <li>■ Built using same architecture as end system</li> <li>■ The final system can be built from it.</li> </ul> </li> </ul> </li> </ul>
<h3>Prototypes</h3> <ul style="list-style-type: none"> <li>■ Types of Prototypes <ul style="list-style-type: none"> <li>■ Horizontal <ul style="list-style-type: none"> <li>■ Exhibit a wide range of the system's functionality</li> <li>■ Rough, Low Quality</li> </ul> </li> <li>■ Vertical <ul style="list-style-type: none"> <li>■ Exhibits a few requirements of the system</li> <li>■ High Quality</li> </ul> </li> </ul> </li> </ul>	<h3>Prototype Evaluation</h3> <ul style="list-style-type: none"> <li>■ Users should evaluate and exercise the prototype <ul style="list-style-type: none"> <li>■ Fuzzy requirements should be better understood.</li> <li>■ New requirements are often identified</li> </ul> </li> </ul>
<h3>Requirements Management</h3> <ul style="list-style-type: none"> <li>■ Problem: Not all features identified by elicitation can be implemented within the time constraints</li> <li>■ Categorize features <ul style="list-style-type: none"> <li>■ By Importance <ul style="list-style-type: none"> <li>■ Critical, Important, Useful, Bell/Whistle</li> </ul> </li> <li>■ By Effort Required</li> <li>■ By Risk <ul style="list-style-type: none"> <li>■ Can the feature cause cost overruns, schedule delays, due to misunderstandings and possible change</li> </ul> </li> </ul> </li> </ul>	<h3>Requirements Management</h3> <ul style="list-style-type: none"> <li>■ After categorization <ul style="list-style-type: none"> <li>■ Scope in the product definition to maximize what can be accomplished that is most important to the customer</li> </ul> </li> <li>➔ Remember the 6-month system design and development constraint</li> </ul>
<h3>PHASE 3: Software Requirements Specification</h3> <ul style="list-style-type: none"> <li>■ Traditional Method <ul style="list-style-type: none"> <li>■ List requirements in textual form sorted by functional area</li> </ul> </li> <li>■ Use-Case Method <ul style="list-style-type: none"> <li>■ List nonfunctional requirements, and requirements that do not map to use cases</li> <li>■ Provide use cases that describe action sequences with alternatives and exceptions</li> </ul> </li> </ul>	<h3>SRS Recommended Format</h3> <ol style="list-style-type: none"> <li>1. Project Overview and Objectives</li> <li>2. System Requirements</li> <li>3. Requirements</li> <li>4. Use Cases</li> <li>5. Prototypes</li> <li>6. Traceability</li> <li>7. Design and Development Estimate</li> <li>A. Data Dictionary</li> </ol> <div style="border: 1px solid black; padding: 5px; text-align: center; font-weight: bold; font-size: 1.2em;">       IN YOUR PROJECT HANDOUT     </div>

<h3>Requirements Are</h3> <ul style="list-style-type: none"> <li>■ Correct</li> <li>■ Unambiguous <ul style="list-style-type: none"> <li>■ If and only if it can be subject to only one interpretation</li> </ul> </li> <li>■ Complete <ul style="list-style-type: none"> <li>■ It describes all significant requirements of concern to the user, including requirements associated with functionality, performance, design constraints, attributes, external interfaces</li> </ul> </li> <li>■ Consistent <ul style="list-style-type: none"> <li>■ requirements are not in conflict with each other</li> </ul> </li> </ul>	<h3>Requirements are</h3> <ul style="list-style-type: none"> <li>■ Verifiable <ul style="list-style-type: none"> <li>■ Is there a process with which a person or machine can determine that the software system does indeed meet the requirement</li> </ul> </li> <li>■ Traceable</li> <li>■ Understandable</li> </ul>
<h3>Types of Requirements</h3> <ul style="list-style-type: none"> <li>■ Functional Requirements</li> <li>■ Non-Functional Requirements <ul style="list-style-type: none"> <li>■ Usability</li> <li>■ Reliability</li> <li>■ Performance</li> <li>■ Supportability</li> </ul> </li> <li>■ Design Constraints</li> </ul> <p><small>From Managing Software Requirements  • A Unified Approach  • Lettingwell, Wilkie, Addison Wesley 2000</small></p>	<h3>PHASE 4: Verification and Revision</h3> <ul style="list-style-type: none"> <li>■ SRS Document walkthrough performed with customer during final meeting <ul style="list-style-type: none"> <li>■ Correct errors</li> <li>■ Clarify Ambiguity</li> <li>■ Verify Completeness</li> </ul> </li> </ul>

### QUESTIONS ?

- Project Webpage
  - <http://dra.mainpage.net>
- Email Wes
  - [wlloyd@vt.edu](mailto:wlloyd@vt.edu)

	<h2>MOOsburg</h2> <ul style="list-style-type: none"> <li>■ <b>M</b>ulti-User-Domain <b>O</b>bject-Oriented <b>s</b>burg</li> <li>■ <b>C</b>ommunity Internet Resource for Blacksburg             <ul style="list-style-type: none"> <li>■ An information database</li> <li>■ Virtual meeting rooms</li> <li>■ Education Tools</li> </ul> </li> <li>■ Text/Graphical based Virtual reality environment</li> <li>■ <a href="http://linc.cs.vt.edu/moosburg">http://linc.cs.vt.edu/moosburg</a></li> </ul>
<h2>MOOsburg</h2> <ul style="list-style-type: none"> <li>■ Based on Blacksburg</li> <li>■ Virtual Places and Spaces to meet             <ul style="list-style-type: none"> <li>■ Tools</li> <li>■ Synchronous Collaboration                 <ul style="list-style-type: none"> <li>■ Same Time, Same Place</li> </ul> </li> <li>■ Asynchronous Collaboration                 <ul style="list-style-type: none"> <li>■ Different Time, Different Place</li> </ul> </li> </ul> </li> </ul>	<h2>Creating a MOOsburg account</h2> <ul style="list-style-type: none"> <li>■ Go to <a href="http://linc.cs.vt.edu/moosburg">http://linc.cs.vt.edu/moosburg</a></li> <li>■ Select "Accounts" on left side bar.</li> <li>■ Choose "set up a new MOOsburg account"</li> <li>■ Choose a user id and specify your email address.</li> <li>■ Account password is mailed to you.</li> <li>■ Change your password once you have an account.</li> </ul>
	<h2>System Requirements</h2> <ul style="list-style-type: none"> <li>■ MOOsburg is a client/server application</li> <li>■ The client runs as a Java applet</li> <li>■ Windows             <ul style="list-style-type: none"> <li>■ Netscape 6.0</li> <li>■ Internet Explorer or Netscape with Java plugin version 1.3</li> </ul> </li> </ul>


## System Requirements

- UNIX
  - Netscape with the Java plugin version 1.3
  - JDK 1.2 applet viewer

appletviewer <http://linc.cs.vt.edu/moosburg/applet/moosburg.html>

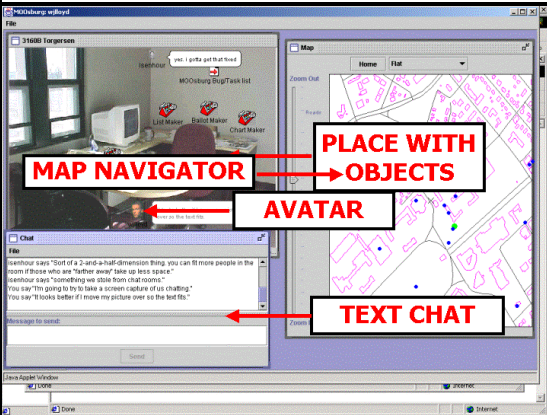
- JDK 1.1
  - See website for details
- MAC OS
  - See website for details

## Logging onto MOOsburg



## MOOsburg User Interface

- View of Current Location
- Personal Avatar
- Chat
- Toolbox
- Inventory
- Map



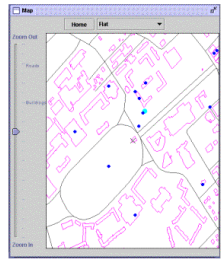
## Navigation

- Zoom In / Zoom Out
- Left click to enter a space
- Map Views
  - Flat
  - Parabolic
  - Hyperbolic
  - ArcTan
  - Parabola-Like

## Navigation

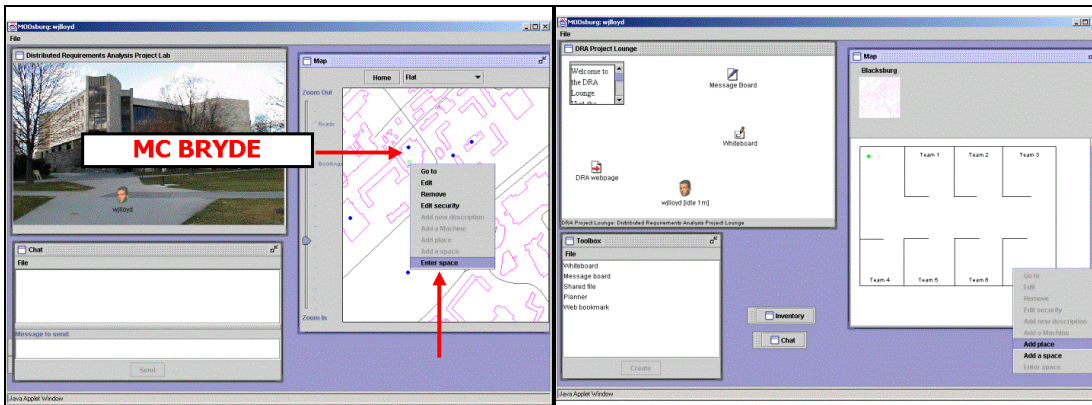
- Green indicates your location
- Light Blue indicates where other user(s) are located
- Blue Indicates a vacant place

## Navigation



## Creating your MOOsburg Home

- Spaces
  - Hollow Circles
  - Spaces contain places
  - A space is like a building
- Places
  - Filled Circles
  - A place is like a room.
- DRA Project Spaces/Places should be created inside your project cubicle in McBryde Hall



## Creating your MOOsburg Home

- Tools
  - Can add many instances of objects from toolbox
  - Rename instances of tools to describe content and usage
- Make Links for
  - Requirements documents
  - Meeting Notes, etc.

## MOOsburg Tools

- The Toolbox provides tools which can be added/removed from your places
  - Whiteboard
  - Message board
  - Shared File
  - Web Bookmark
  - List Editor
  - Planner
    - GANT chart editor
  - Machines
    - Tools under development

## Centra Symposium

## Centra Symposium

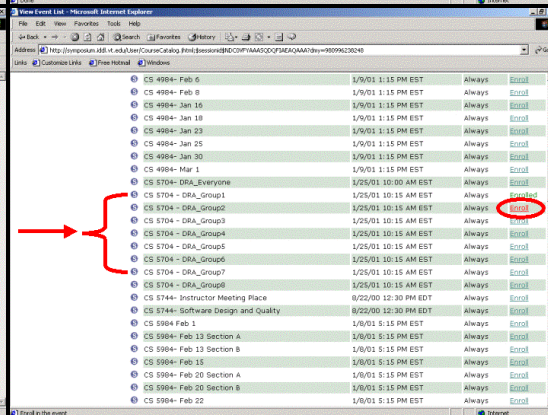
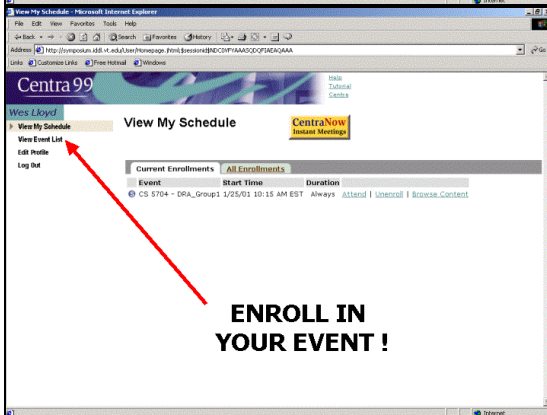
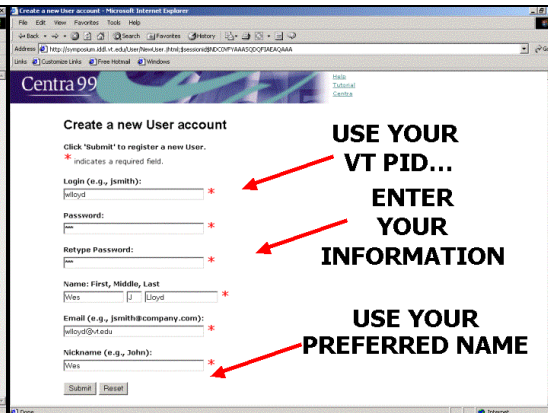
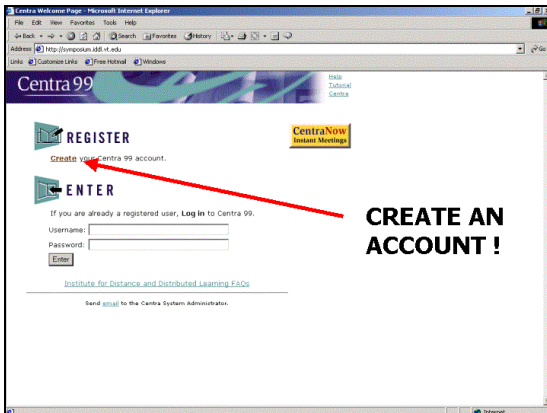
- Point-to-point audio conferencing
- Meeting Agenda
  - Imports Power Point Slides
- Survey Users
- Text Chat
- Whiteboard
- Used for Distance Learning at Virginia Tech

## Centra Symposium

- To be used only during scheduled meetings in the lab

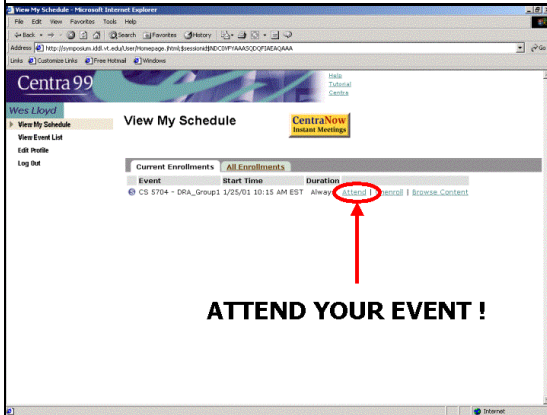
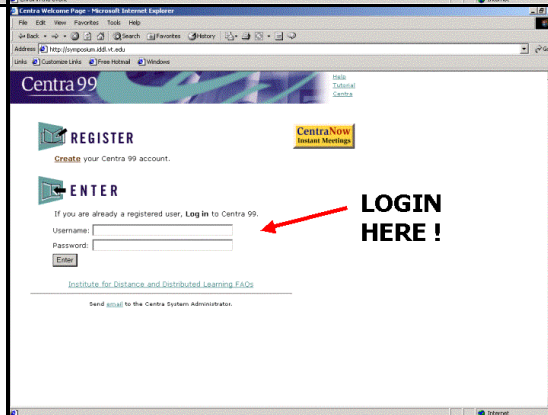
## Creating an account

- Go to: <http://symposium.iddl.vt.edu/>
- REGISTER, by creating your Centra 99 Account



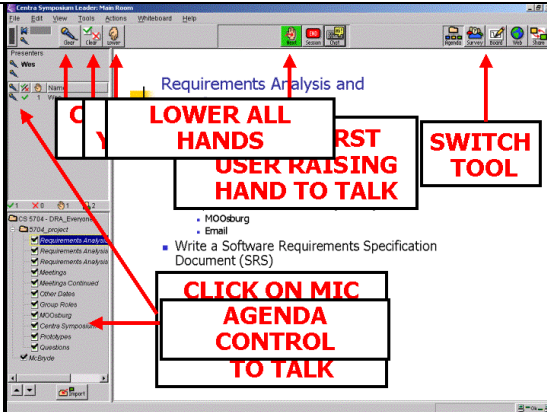
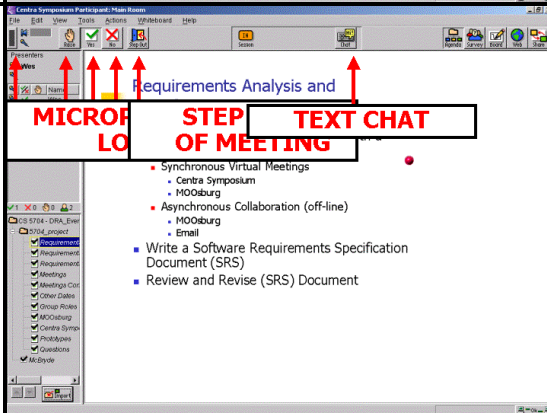
## Logging On

- Go to: <http://symposium.iddl.vt.edu/>
- ENTER, by supplying username and password
- On your schedule, select to attend the event



## Basic Use

- Only one person may talk at a time
- Hold down <CONTROL> Key to Talk
- Session Leader grants permission to talk.
- Raise Hand to request permission

<h3>Session Leader</h3> <ul style="list-style-type: none"> <li>Controls Agenda <ul style="list-style-type: none"> <li>Import Power Point Slides</li> <li>Controls Flow of Slides in meeting</li> </ul> </li> <li>Grants permission to Talk</li> <li>Controls which tool is active</li> </ul>	 <p>Requirements Analysis and</p> <p><b>LOWER ALL HANDS</b></p> <p><b>FIRST USER RAISING HAND TO TALK</b></p> <p><b>SWITCH TOOL</b></p> <p><b>CLICK ON MIC AGENDA CONTROL TO TALK</b></p>
<h3>Meeting Participant</h3> <ul style="list-style-type: none"> <li>All users can use text chat at all times</li> <li>All users can only watch until Session Leader grants permission</li> <li>With Permission <ul style="list-style-type: none"> <li>Can Talk</li> <li>Can Point and Draw on Slides &amp; Whiteboard</li> <li>Can control application</li> </ul> </li> </ul>	 <p>Requirements Analysis and</p> <p><b>MICROPHONE</b></p> <p><b>STEP OF MEETING</b></p> <p><b>TEXT CHAT</b></p>
<h3>Special Features</h3> <ul style="list-style-type: none"> <li>Web Safari <ul style="list-style-type: none"> <li>Shared Web Browsing</li> </ul> </li> <li>Application Sharing <ul style="list-style-type: none"> <li>Single Thread of Control</li> </ul> </li> </ul>	<h3>QUESTIONS ?</h3> <ul style="list-style-type: none"> <li>Project Webpage <ul style="list-style-type: none"> <li><a href="http://dra.mainpage.net">http://dra.mainpage.net</a></li> </ul> </li> <li>Email Wes <ul style="list-style-type: none"> <li><a href="mailto:wlloyd@vt.edu">wlloyd@vt.edu</a></li> </ul> </li> </ul>

# Appendix G.

# Post Meeting Survey #1

Name: \_\_\_\_\_ Group Number: \_\_\_\_\_  
 Group Role: \_\_\_\_\_ Meeting Number: \_\_\_\_\_

## Post-Meeting Questionnaire

NOTE: Information supplied **WILL NOT** be used to assign participation grades for CS 5704 or CS 5734

### --- PLEASE CHECK BOXES ---

Please indicate tools used in this virtual meeting, and the overall usefulness of each:

Software Component / Feature	Check Here if Used	5 Extremely Useful	4 Useful	3 Possibly Useful	2 Not Useful	1 Disruptive
Centra Symposium Voice Conferencing						
Centra Symposium Text Chat						
Centra Symposium Whiteboard						
Centra Symposium Agenda/Slide Show						
Centra Symposium Web Browser Sharing						
Centra Symposium Application Sharing						
MOOsburg Text Chat						
MOOsburg Message Board						
MOOsburg Whiteboard						
MOOsburg Shared File						
MOOsburg Web Links						
MOOsburg List Editor						
MOOsburg Planner						

What was your participation/contribution Level in today's meeting?

(check only **one**)

- \_\_\_ 5- Very Active
- \_\_\_ 4- Active
- \_\_\_ 3- Some Participation, Mostly Watched
- \_\_\_ 2- Minimal Participation
- \_\_\_ 1- Observer Only

What factors led to your amount of participation in today's meeting?



(check only those that apply)

- (+/-) Subject knowledge
- (+/-) Preparation Time
- Assigned Role in Meeting
- Physical Energy Level
- Usability of MOOsburg
- Usability of Centra Symposium
- Quality of Computer Workstation
- Arrival Time at Meeting
- Quality of Network Bandwidth
- Personality Attributes (for example: aggressive versus shy)
- Comfort Level with English Language
- Well coordinated agenda / meeting
- Influence of engineer team members (CS 5704)
- Influence of customer team members (CS 5734)
- List others factors:



## VIRTUAL MEETING #2

What requirements analysis techniques were used in today's meeting?

- |   |  |
|---|--|
| <input type="checkbox"/> Interviews                     | <input type="checkbox"/> Questionnaires          |
| <input type="checkbox"/> Brainstorming / Idea Reduction | <input type="checkbox"/> Feature Definition      |
| <input type="checkbox"/> Storyboards                    | <input type="checkbox"/> Use Cases               |
| <input type="checkbox"/> Prototyping                    | <input type="checkbox"/> Requirements Management |
| <input type="checkbox"/> Other:                         |  |

Why did you choose the requirements analysis techniques listed above?

Method #1 : \_\_\_\_\_ (use name from above)

Why:

Method #2 : \_\_\_\_\_ (use name from above)

Why:

Method #3 : \_\_\_\_\_ (use name from above)

Why:

Method #4 : \_\_\_\_\_ (use name from above)

Why:

Method #5 : \_\_\_\_\_ (use name from above)

Why:

How effective were the requirements methods chosen in the distributed setting?

Method #1:(circle one)

++ Much Better than expected	+ Better than expected
= Worked as expected	- Worse than expected
-- Much Worse than expected	

Method #2:(circle one)

++ Much Better than expected	+ Better than expected
= Worked as expected	- Worse than expected
-- Much Worse than expected	

Method #3:(circle one)

++ Much Better than expected	+ Better than expected
= Worked as expected	- Worse than expected
-- Much Worse than expected	

Method #4:(circle one)

++ Much Better than expected	+ Better than expected
= Worked as expected	- Worse than expected
-- Much Worse than expected	

Method #5:(circle one)

++ Much Better than expected	+ Better than expected
= Worked as expected	- Worse than expected
-- Much Worse than expected	

How were Symposium and MOOsburg used to implement the requirements analysis methods:

___	Centra Symposium Text Chat					
	For Requirements Method #(circle):	1	2	3	4	5
___	Centra Symposium Whiteboard					
	For Requirements Method # (circle):	1	2	3	4	5
___	Centra Symposium Agenda/Slide Show					
	For Requirements Method # (circle):	1	2	3	4	5
___	Centra Symposium Web Browser Sharing					
	For Requirements Method # (circle):	1	2	3	4	5
___	Centra Symposium Application Sharing					
	For Requirements Method # (circle):	1	2	3	4	5
___	MOOsburg Text Chat					
	For Requirements Method # (circle):	1	2	3	4	5
___	MOOsburg Message Board					
	For Requirements Method # (circle):	1	2	3	4	5
___	MOOsburg Whiteboard					
	For Requirements Method # (circle):	1	2	3	4	5
___	MOOsburg Shared File					
	For Requirements Method # (circle):	1	2	3	4	5
___	MOOsburg Web Links					
	For Requirements Method # (circle):	1	2	3	4	5
___	MOOsburg List Editor					

\_\_\_\_ For Requirements Method # (circle): 1 2 3 4 5  
MOOsburg Planner  
For Requirements Method # (circle): 1 2 3 4 5

Other:

For Requirements Method # (circle): 1 2 3 4 5

:

# Appendix I. Requirements Analysis Experience Survey

Name: \_\_\_\_\_ Group Number: \_\_\_\_\_

Group Role: \_\_\_\_\_ Meeting Number: \_\_\_\_\_

## RA Experience Questionnaire

NOTE: Information supplied **WILL NOT** be used to assign participation grades for CS 5704 or CS 5734

Please indicate your level of experience using Requirements Analysis methods:

**Do not count any experience gained from this project or the CS 5704 class.**

Interviews (circle one)

completely new to me before 5704  
used 1-3 times before

familiar with, but never used  
have used extensively

Questionnaires (circle one)

completely new to me before 5704  
used 1-3 times before

familiar with, but never used  
have used extensively

Brainstorming / Idea Reduction (circle one)

completely new to me before 5704  
used 1-3 times before

familiar with, but never used  
have used extensively

Feature Definition (circle one)

completely new to me before 5704  
used 1-3 times before

familiar with, but never used  
have used extensively

Storyboards (circle one)

completely new to me before 5704  
used 1-3 times before

familiar with, but never used  
have used extensively

Use Cases (circle one)

completely new to me before 5704  
used 1-3 times before

familiar with, but never used  
have used extensively

Prototyping (circle one)

completely new to me before 5704  
used 1-3 times before

familiar with, but never used  
have used extensively

Requirements Management (circle one)

completely new to me before 5704  
used 1-3 times before

familiar with, but never used  
have used extensively

Others methods:

**THANK YOU!**

# Appendix J. Software Engineering Experience Survey

## Software Engineering Student Demographics Survey

Name: \_\_\_\_\_ Email Address: \_\_\_\_\_

Are you taking CS 5734 this semester? \_\_\_\_\_

Years of college education: \_\_\_\_\_

List All Computer Languages and level of expertise that you could use to build and demonstrate system mockups / interface prototypes :

Please Rate your Experience and ability on the following scale:  
0-No Experience 1-beginner, 2-intermediate, 3-expert

Description of ratings:

0- No Experience: No familiarity or experience

1- Beginner: Minimal formal experience, minimal familiarity (0-3 months)

2- Intermediate: Some experience and familiarity (3-12 months, small projects)

3- Expert: Experienced and Fluent (1+ year, or one or more large projects)

Example: C level 2, Java level 3, C++ level 2, Visual Basic level 1.5, HTML/DHTML 1.5, Delphi 1.0

Concept	Experience Level 0-No Experience 1-beginner 2-intermediate, 3-expert	Describe Experience
Gathering Requirements from end users		
Gathering Requirements from customers/clients		
Experience developing system prototypes / mockups		
Conducting Requirements Negotiation with a client		
Writing Requirements Specification Documents		
Writing Design Specification Documents		
Writing Software Test Plans/Specifications		
UML		
Object Oriented Design		
Use of CASE Tools		
Use Case Analysis (for requirements engineering)		
Have seen/used requirements/functional specifications in software development		
Have seen/used design specifications in software development		

Please list experience significant to requirements analysis and system prototyping:

# Appendix K. The Belbin Self-Perception Inventory

## The Belbin Self-Perception Inventory

The following is from Belbin's original work on team roles (Belbin, 1981).

For each of the following sections, **distribute ten (10) points** among the 8 sentences that you think best describe your behavior. These points may be distributed among several sentences: in extreme cases they might be spread among all 8 sentences or ten points may be given to a single sentence. Enter the points in the spaces in front of each sentence. For example, for section 1, you might give five points to statement 2, two points to each statement 4 & 5, and one point to statement 7. (Suggestion: Read all of the sentences, crossing out the ones that are not true or hardly true, then distribute points among those sentences left.)

I. What I believe I can contribute to the team:

1. \_\_\_ I think I can quickly see and take advantage of new opportunities.
2. \_\_\_ I can work well with a very wide range of people.
3. \_\_\_ Producing ideas is one of my natural assets.
4. \_\_\_ My ability rests in being able to draw people out whenever I detect they have something of value to contribute to group objectives.
5. \_\_\_ My capacity to follow through has much to do with my personal effectiveness.
6. \_\_\_ I am ready to face temporary unpopularity if it leads to worthwhile results in the end.
7. \_\_\_ I can usually sense what is realistic and likely to work.
8. \_\_\_ I can offer a reasoned case for alternate courses of action without introducing bias or prejudice.

II. If I have a possible shortcoming in teamwork, it could be that:

1. \_\_\_ I am not at ease unless meetings are well structured and controlled and generally well conducted.
2. \_\_\_ I am inclined to be too generous towards others who have a valid viewpoint that has not been given proper airing.
3. \_\_\_ I have a tendency to talk too much once the group gets on to new ideas.
4. \_\_\_ My objective outlook makes it difficult for me to join in readily and enthusiastically with colleagues.
5. \_\_\_ I am sometimes seen as forceful and authoritarian if there is a need to get something done.
6. \_\_\_ I find it difficult to lead from the front, perhaps because I am over-responsive to group atmosphere.
7. \_\_\_ I am apt to get caught up in ideas that occur to me and so lose track of what is happening.
8. \_\_\_ My colleagues tend to see me as worrying unnecessarily over detail and the possibility that things may go wrong.

III. When involved in a project with other people:

1. \_\_\_ I have an aptitude for influencing people without pressurizing them.
2. \_\_\_ My general vigilance prevents careless mistakes and omissions being made.
3. \_\_\_ I am ready to press for action to make sure that the meeting does not waste time or lose site of the main objective.
4. \_\_\_ I can be counted on to contribute something original.
5. \_\_\_ I am always ready to back a good suggestion in the common interest.
6. \_\_\_ I am keen to look for the latest in new ideas and developments.
7. \_\_\_ I believe my capacity for judgment can help to bring about the right decisions.
8. \_\_\_ I can be relied upon to see that all essential work is organized.

IV. My characteristic approach to group work is that:

1. \_\_\_ I have a quite interest in getting to know colleagues better.



2. \_\_\_ I am not reluctant to challenge the views of others or to hold a minority view myself.
3. \_\_\_ I can usually find a line of argument to refute unsound propositions.
4. \_\_\_ I think I have a talent for making things work once a plan has to be put into operation.
5. \_\_\_ I have a tendency to avoid the obvious and to come out with the unexpected.
6. \_\_\_ I bring a touch of perfectionism to any job I undertake.
7. \_\_\_ I am ready to make use of contacts outside the group itself.
8. \_\_\_ While I am interested in all views I have not hesitation in making up my mind once a decision has to be made.

V. I gain satisfaction in a job because:

1. \_\_\_ I enjoy analyzing situations and weighing up all of the possible choices.
2. \_\_\_ I am interested in finding practical solutions to problems.
3. \_\_\_ I like to feel I am fostering good working relationships.
4. \_\_\_ I can have a strong influence on decisions.
5. \_\_\_ I can meet people who may have something new to offer.
6. \_\_\_ I can get people to agree on a necessary course of action.
7. \_\_\_ I feel in my element where I can give a task my full attention.
8. \_\_\_ I like to find a field that stretches my imagination.

VI. If I am suddenly given a difficult task with limited time and unfamiliar people:

1. \_\_\_ I would feel like retiring to a corner to devise a way out of the impasse before developing a line.
2. \_\_\_ I would be ready to work with the person who showed the most positive approach.
3. \_\_\_ I would find some way of reducing the size of the task by establishing what different individuals might best contribute.
4. \_\_\_ My natural sense of urgency would help to ensure that we did not fall behind schedule.
5. \_\_\_ I believe I would keep cool and maintain my capacity to think straight.
6. \_\_\_ I would retain a steadiness of purpose in spite of the pressures.
7. \_\_\_ I would be prepared to take a positive lead if I felt the group was making no progress.
8. \_\_\_ I would open up discussions with a view to stimulating new thoughts and getting something moving.

VII. With reference to the problems to which I am subject to working in groups:

1. \_\_\_ I am apt to show my impatience with those who are obstructing progress.
2. \_\_\_ Others may criticize me for being too analytical and insufficiently intuitive.
3. \_\_\_ My desire to ensure that work is properly done can hold up proceedings.
4. \_\_\_ I tend to get bored rather easily and rely on one or two stimulating members to spark me off.
5. \_\_\_ I find it difficult to get started unless the goals are clear.
6. \_\_\_ I am sometimes poor at explaining and clarifying complex points that occur to me.
7. \_\_\_ I am conscious of demanding from others the things I cannot do myself.
8. \_\_\_ I hesitate to get my points across when I run up against real opposition.

**VIRGINIA POLYTECHNIC INSTITUTE AND STATE UNIVERSITY****Informed Consent for Participants  
of Investigative Projects**

Title of Project: Groupware to facilitate Distributed Requirements Engineering: An Empirical Study

Investigator(s): Wes J. Lloyd, Dr. Mary Beth Rosson, Dr. Stephen Edwards, Dr. James D. Arthur

**I. The Purpose of this Research/Project**

This empirical study will study the effectiveness of groupware in enabling distributed requirements engineering. We will seek to obtain an understanding of how existing groupware supports distributed requirements engineering. We seek to gain an understanding of how to improve groupware to support distributed requirements engineering. From our observations we may approach the development of a requirements gathering process for distributed groups.

**II. Procedures**

Subjects will be assigned into groups and will role-play as Software Engineers or customers in remote requirements gathering meetings. Each group will participate in four sessions of up to seventy-five minutes each. Subjects will use computer based audio/video conferencing systems available in Virginia Tech research labs. Subjects role-playing as software engineers and customers will exchange email as necessary to clarify information outside of arranged meetings.

The Software Engineers will produce a requirements specification document from the meetings with customers. The requirements specification document will describe the functionality of the software system requested by the customers. This document will be produced outside of the arranged meetings.

The customers will have the responsibility to role-play as motivated customers who demand a quality software system from the software engineers. The customers will be given a system description. They will role-play as motivated and demanding users who want the software engineers to develop a good system.

**III. Risks**

The empirical study involves no unusual risks to the subjects participating.

**IV. Benefits of this Project**

Software engineers will be recruited from the CS 5704 Software Engineering course. Their participation in this experiment will augment their education in software engineering. Students in a software engineering course frequently study requirements gathering processes, and produce a “requirements specification” document.

Customers will be recruited from the graduate CS 5734 Computer Supported Cooperative Work (CSCW) course. The CSCW course focuses on the study of using computers to support and facilitate work processes both remote and local. This study will have students participate in a distributed requirements gathering activity. This experiment will augment the education of students in the CSCW course by allowing them to experience first hand real issues encountered when using computers to support remote work. (one topic within the CS 5734 course.)

Subjects participating in the study are free to contact the investigators following the conclusion of the experiment to obtain a summary of the research results.

## **V. Extent of Anonymity and Confidentiality**

The names of subjects will not be used in published experimental write-ups. References to subjects in experiment write-ups will be made via code-names. Code name identification will remain unpublished and only available to the research committee. Unpublished data will only be available to the research committee.

Audio and video recordings of distributed meetings if made will not be distributed. Meeting records will allow the research committee to observe both sides of interaction at distributed meetings. Meeting recordings could be implemented on standard video recording equipment. Also some recordings may be taken using capabilities integrated in the software used to support the distributed meetings. Once meeting information is transcribed and recordings are deemed as no longer necessary the recordings will be erased or discarded. The experimenter will conduct transcribing of recordings to a written event script.

## **VI. Compensation**

Due to the limitations of research funding subjects participating in this experiment cannot be compensated at this time.

## **VII. Freedom to Withdraw**

If necessary, subjects participating in this study may withdraw from participation at any time without penalty.

## **VIII. Approval of Research**

This research project has been approved, as required, by the Institutional Review Board for Research Involving Human Subjects at Virginia Polytechnic Institute and State University, by the Department of Computer Science.

## **IX. Subject's Responsibilities**

I voluntarily agree to participate in this study. I have the following responsibilities:

- Role-playing as either a software engineer or customer in the study.
- Software engineers must produce a functional specification of the product requested by the customers.
- Customers must role-play as users or stakeholders interested in foreseeing that a quality system is developed.
- Participation in the study will involve four distributed meetings of up to 75 minutes each, a training session on conducting distributed meetings, and an overview of the requirements analysis process that is to be used in the project.
- Customers and engineers may need to collaborate off line using email to clarify details.

## **X. Subject's Permission**

I have read and understand the Informed Consent and conditions of this project. I have had all my questions answered. I hereby acknowledge the above and give my voluntary consent for participation in this project.

If I participate, I may withdraw at any time without penalty. I agree to abide by the rules of this project.

---

Signature

Date

Should I have any questions about this research or its conduct, I may contact:

Wes J. Lloyd	(540) 200-1861
Dr. Stephen Edwards (SE)	(540) 231-5723
Dr. Mary Beth Rosson (CSCW)	(540) 231-6470
Dr. James D. Arthur (SE)	(540) 231-7538

---

E. R. Stout  
Chair, IRB,  
Research Division

Phone

# Vita

---

Wesley James Lloyd was born in Cleveland, Ohio in April of 1975. His interest in computers can be traced as far back as 1983 when his dad Marlin, purchased the first of many home computers. In 1988 he obtained an amateur radio license and began using computers as the secretary and newsletter editor for a local amateur radio club in Huron, Ohio. He worked on his senior yearbook, and was referred to as “byte-head” by his high school librarian. He found himself working in the computer field right after graduating high school in 1993. His first project consisted of converting an apple II computer system to create and send CNC programs to a machine which cut two dimensional shapes out of sheet metal.

Wes attended the University of Toledo to obtain a bachelor’s degree in computer science and engineering, obtained in the fall of 1997. After graduating from the University of Toledo he worked as a consultant for the world’s largest glass maker, Owens Illinois, where he was a team member on a software development team to convert a legacy mainframe control system to a client/server system to support process control for any glass making furnace. This multi-year, multi-million dollar development project sparked interest in software engineering processes. In the fall of 1999 Wes attended graduate school at Virginia Tech in pursuit of a Master of Science degree in Computer Science with a focus in Software Engineering. Upon completion of this degree in July of 2001, Wes will work for Hewlett Packard in Fort Collins, Colorado. He plans to continue to obtain experience in software engineering and

Wes has been a member of various professional organizations including: ACM, Golden Key National Honor Society, Eta Kappa Nu an Electrical/Computer Engineering Honor Society, Upsilon Phi Epsilon a Computer Science Honor Society, and Alpha Phi Omega a national service organization.