

REQUIREMENTS DETERMINATION OF INFORMATION SYSTEMS:
USER AND DEVELOPER PERCEPTIONS OF FACTORS
CONTRIBUTING TO MISUNDERSTANDINGS

by

Chad A. McAllister

A Dissertation Presented in Partial Fulfillment

Of the Requirements for the Degree

Doctor of Philosophy

Capella University

August 2006

UMI Number: 3226800

Copyright 2006 by
McAllister, Chad A.

All rights reserved.

UMI[®]

UMI Microform 3226800

Copyright 2006 by ProQuest Information and Learning Company.
All rights reserved. This microform edition is protected against
unauthorized copying under Title 17, United States Code.

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

© Chad A. McAllister, 2006

REQUIREMENTS DETERMINATION OF INFORMATION SYSTEMS:

USER AND DEVELOPER PERCEPTIONS OF FACTORS

CONTRIBUTING TO MISUNDERSTANDINGS

by

Chad A. McAllister

has been approved

August 2006

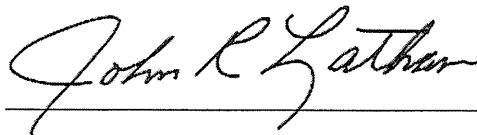
APPROVED:

JOHN R. LATHAM, Ph.D., Faculty Mentor and Chair

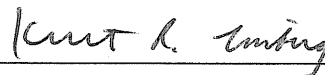
PERRY C. HAAN, DBA, Committee Member

ALAN M. DAVIS, Ph.D., Committee Member

ACCEPTED AND SIGNED:



JOHN R. LATHAM, Ph.D.



Kurt Linberg, Ph.D.
Dean, School of Business & Technology

Abstract

The development of information systems involves knowing what to create and how to create it. What to create is the area of requirements determination, which is often identified as the most difficult part of bringing an information system into existence. Two key stakeholder groups in information systems projects are users and developers, who must understand and agree upon the requirements. Misunderstandings between these two groups can lead to several negative consequences, including jeopardizing the success of the system, increasing the cost and time of the project, and risking the project being cancelled. Although several techniques have been created to help users and developers better understand requirements, little research has been conducted to learn why users and developers continue to misunderstand requirements. The purpose of this study is to identify factors that contribute to users and developers misunderstanding requirements for information systems. The findings of the study contribute to a theoretical foundation for future research, allowing for the creation of more effective and efficient techniques for understanding requirements.

Dedication

This dissertation is dedicated to my family. First, to my wife whose loving support and encouragement made my journey toward completing a PhD degree possible. Second, to my children who patiently tolerated daddy working on his dissertation and who I know will grow up to value education. Third, to my mother who still does not understand what I do but will always love me. Fourth, to my father who never knew I started this journey but who I know would be proud of me if he was still alive. And most importantly, to my Heavenly Father who taught me to seek wisdom and showed me what love is.

Acknowledgments

Acknowledging all of the people who have contributed to this work has much in common with Academy Award Winners' Oscar speeches that start "I wish to thank everyone that had anything to do with this." There are so many to whom I am grateful for helping me start, persevere, and complete this dissertation, but just as Oscar winners are encouraged to keep their speeches short, I will do the same here.

My wife is my best friend, a loving mother to my children, and a first-class editor who made this dissertation not only a possibility, but stood by me to see it become a reality—I will always love you. I am grateful to the three organizations that participated in this study and provided a forum for data collection—thank you for trusting me. I appreciate Tom Ward who I met as a fellow student and who helped to show me the way by allowing me to participate as a peer member of his dissertation committee—thank you Dr. Ward. Valerie Anderson, former Silicon Valley executive and venture capital fund manger turned counselor for teens gave me inspiration to begin this work through her own example of reentering higher education—you inspired me.

I could not have asked for a better dissertation committee. Their knowledge, personalities, and concern for me doing my best made creating this dissertation a truly wonderful experience. I am indebted to Dr. Al Davis who I "connected with" first through his writings and then in person. As an international requirements engineering expert, he not only provided credibility to my work, he also provided me with invaluable knowledge and lucid insights. Dr.

Perry Haan, business marketing educator and practitioner, offered much encouragement and his experienced perspective on the gaps that exist between marketing and development groups. Most importantly, I am grateful to my mentor, Dr. John Latham. When I began seeking a mentor, fellow students described John as “demanding,” which was what encouraged me to learn more about him. His engineering experience, passion for excellence, and huge amount of patience for me made my journey as a doctoral student so very rewarding.

Table of Contents

Acknowledgments.....	iv
CHAPTER 1. INTRODUCTION	1
Introduction to the Problem	1
Background of the Study	3
Statement of the Problem.....	4
Purpose of the Study	4
Rationale	5
Research Questions.....	5
Conceptual Framework.....	6
Significance of the Study	8
Definition of Terms.....	9
Assumptions and Limitations	11
Nature of the Study	11
Organization of the Remainder of the Study	12
CHAPTER 2. LITERATURE REVIEW	13
Overview of Chapter.....	13
Introduction to Developing Information Systems.....	13
Requirements and Requirements Determination	20
Why Requirements Matter.....	21

Contemporary Approaches to Determining Requirements	26
Factors Related to Misunderstanding Requirements	36
Assessment of Current Literature	55
Enhanced Conceptual Framework	57
CHAPTER 3. METHODOLOGY	59
Review of the Purpose of the Study.....	59
General Research Philosophy	59
Research Questions.....	60
Research Approach.....	61
Appropriateness of Approach	62
Sampling Plan	70
Profile of Participating Organizations	71
Data Collection and Analysis Process	73
Reliability, Validity, and Generalizability	76
Ethical Considerations	78
CHAPTER 4. DATA COLLECTION AND ANALYSIS.....	81
Purpose of the Study and Research Questions.....	81
Data Collection and Analysis of Users' Perceptions	81
Data Collection and Analysis of Developers' Perceptions.....	101
Analysis of User and Developer Data.....	120

CHAPTER 5. RESULTS, CONCLUSIONS, AND RECOMMENDATIONS.....	144
Summary and Discussion of Results.....	144
Conclusions.....	169
Recommendations.....	171
Closing Observations.....	175
REFERENCES	177
APPENDIX A: TEMPLATE FOR REQUESTING PARTICIPATION.....	188
APPENDIX B: SURVEY INVITATION.....	190
APPENDIX C: NOMINAL GROUP TECHNIQUE PROCEDURES.....	191
APPENDIX D: NGT PARTICIPANT ATTENDANCE SHEET	199
APPENDIX E: NOMINAL QUESTION WORKSHEET.....	200
APPENDIX F: FACTOR VOTING WORKSHEET.....	201
APPENDIX G: USER SURVEY.....	202
APPENDIX H: DEVELOPER SURVEY	220
APPENDIX I: ALL FACTORS CREATED BY USERS	243
APPENDIX J: USER THEMES BASED ON DISCUSSION IN FOCUS GROUPS	247
APPENDIX K: QUALRUS SCRIPT FOR LISTING CODES FOR EACH FACTOR	251
APPENDIX L: AHP INCONSISTENCY VALUES FOR USER PARTICIPANTS	252
APPENDIX M: ALL FACTORS CREATED BY DEVELOPERS.....	253
APPENDIX N: DEVELOPER THEMES BASED ON DISCUSSION IN FOCUS GROUPS .	258

APPENDIX O: AHP INCONSISTENCY VALUES FOR DEVELOPER PARTICIPANTS... 263

APPENDIX P: INTERDEPENDENCIES BETWEEN THEMES..... 264

APPENDIX Q: INSTITUTIONAL REVIEW BOARD EXCERPTS..... 272

List of Tables

Table 1. Activities Common in the Development of Information Systems.....	17
Table 2. Common Requirements Activities.....	18
Table 3. JAD Workshop Activities.....	30
Table 4. Factors Found in Literature.....	52
Table 5. Research Question Hierarchy	60
Table 6. Summary of Nominal Group Technique Activities.....	65
Table 7. AHP 9-Point Scale.....	67
Table 8. Example of Pairwise Weightings.....	68
Table 9. Descriptive Statistics for Each Participating Focus Group.....	73
Table 10. Users’ Top Voted-For Factors	82
Table 11. Combined User Factors and Supporting Factors	84
Table 12. Number of Themes Generated by Users.....	87
Table 13. Synthesized Themes Based on User Discussion.....	88
Table 14. Percent of Codes in Common	90
Table 15. Comparison of Top-9 Factors and Synthesized Themes With 20% or More Support .	92
Table 16. Participants Divided into Groups Based on the Inconsistency of Their Responses	97
Table 17. User Factor Rankings by Groups Based on Inconsistency.....	99
Table 18. Inconsistency Values by Organization for User Participants	99
Table 19. User Factor Rankings by Organization.....	101
Table 20. Developers’ Top Voted-For Factors.....	102
Table 21. Combined Developer Factors with the Supporting Factors Expressed by Each Developer Group	104
Table 22. Number of Themes Generated.....	107

Table 23. Synthesized Themes from Factors Identified by Developer Focus Groups	108
Table 24. Percent of Codes in Common	109
Table 25. Comparison of Developer Top-10 Factors and Synthesized Themes.....	111
Table 26. Developers Divided into Groups Based on the Inconsistency of Their Responses....	116
Table 27. Developer Factor Rankings Groups Based on Inconsistency.....	117
Table 28. Inconsistency Values by Organization for Developer Participants	118
Table 29. Developer Factor Rankings by Organization.....	120
Table 30. Summary of Factors Ordered by Users.....	122
Table 31. Summary of Factors Ordered by Developers	124
Table 32. Summary of Themes Ordered by User Support.....	127
Table 33. Descriptive Statistics of AHP Weights.....	132
Table 34. Factors Created in the Study Related to Factors Found in Literature.....	135
Table 35. User and Developer Factors Not Found In the Literature Review	138
Table 36. Expected Issues With Requirements Determination Techniques Based on User and Developer Top Factors	142
Table 37. UPFs and DPFs Ordered by User Weight	146

List of Figures

Figure 1. Conceptual framework for studying what influences users and developers to misunderstand requirements.	7
Figure 2. Waterfall SDLC depicting iteration between steps.	14
Figure 3. Depiction of staged delivery SDLC.	16
Figure 4. Simplified view of developing an information system, emphasizing requirements.	19
Figure 5. Factors responsible for software development problems.	24
Figure 6. Relative cost to identify and correct defects at different phases of software development.	26
Figure 7. Davis' conceptual framework of requirements determination involving users and developers.	45
Figure 8. User and developer four-stage communication model.	48
Figure 9. Assessment of literature and connections between literature and the research presented in this study.	56
Figure 10. Enhanced conceptual framework for studying what influences users and developers to misunderstand requirements.	58
Figure 11. Data analysis activities for Phase I and Phase II.	75
Figure 12. Importance of user factors normalized to 1.0.	94
Figure 13. Importance of user factors showing AHP weights summing to 1.0.	95
Figure 14. AHP weights for each user factor shown with standard deviation error bars.	96
Figure 15. Comparison of importance based on user groups with decreasing inconsistency values.	98
Figure 16. Importance of factors as judged by users for each organization.	100
Figure 17. Importance of developer factors normalized to 1.0.	113

Figure 18. Importance of developer factors showing AHP weights summing to 1.0.....	114
Figure 19. AHP weights for each DPF shown with standard deviation error bars.....	115
Figure 20. Comparison of importance based on groups with decreasing inconsistency values.	117
Figure 21. Importance of factors as judged by developers from each organization.....	119
Figure 22. Comparing user and developer factors.....	121
Figure 23. Box plot comparing AHP weights for developer and user factors.....	130
Figure 24. Most- and least-important factors identified by users and developers.....	131
Figure 25. Summary of AHP weights for each UPF and DPF.....	148
Figure 26. Graphical conceptual framework of the research problem with UPFs and DPFs listed.....	149
Figure 27. Three dimensions involved in misunderstanding requirements for information systems.....	176
Figure P-1. Themes related to Key Users are Not Available.....	265
Figure P-2. Themes related to Users Do Not Know What They Want.....	267
Figure P-3. Themes related to Users and Developers Operate from Different Frames of Reference.	269
Figure P-4. Themes related to Users are Intimidated by Developers	270

CHAPTER 1. INTRODUCTION

Introduction to the Problem

John Gray's popular book on relationships proposes that men and women are prone to misunderstand each other. A culture gap, due to each group figuratively being from different worlds, is responsible for misunderstandings (1992). He said, speaking metaphorically of the communication difficulties between men and women, "The Martian languages and Venusian languages had the same words, but the way they were used gave different meanings" (p. 59). A similar problem exists among users and developers of information systems. Coughlan, Lycett, and Macredie (2003) suggested that "eliciting requirements involves activities that are intensely communicative and increase in significance when one considers the 'culture gap' or basic semantic differences dividing two groups such as users and designers attempting to engage in meaningful dialogue" (2003, p. 525). The different worldviews, personalities, and mental frameworks users and developers bring to the creation of an information system can contribute to misunderstandings about requirements (Al-Rawas & Easterbrook, 1996; Coughlan et al., 2003; Kudikyala & Vaughn, 2005).

Understanding what an information system needs to do, which encompasses determining the requirements, is widely considered the most difficult step in the development of the system. In doing so, effective communication between users and developers is a critical success factor (Duggan & Thachenkary, 2003). During the initial phase of any information system development, software developers are challenged to uncover, understand, and specify the user requirements (Davis et al., 1997). Errors made during this phase can be responsible for up to 60% of the cost of a project, cause schedules to more than double, and may ultimately result in users rejecting the information system. To reduce these negative consequences, users and

developers must share a common understanding of the requirements for the system being developed.

Although the development of information systems has matured in the last 20 years, many systems continue to be of low quality because of misunderstood requirements (Duggan & Thachenkary, 2003). Information systems literature, as well as literature on new product development, contains several examinations of the role and importance of requirements. Further, techniques such as Quality Functional Development, Joint Application Design, and Observation Studies, as well as others, have been applied to improve the elicitation of requirements and minimize misunderstandings between developers and users. Although these techniques have been helpful, their existence is not grounded in knowledge of *why* misunderstandings occur. Rather, they have been created in response to symptoms of the underlying problem. What is lacking is evidence for why the misunderstandings occur.

The situation examined in the present study is reflective of a common practice in software development: to rush to a solution before the problem is well understood. It is also common in the scientific process, where a problem is identified, a solution is devised, its effectiveness is tested, and the solution continues to be used if the effectiveness is reasonable. The techniques used to improve the quality of the requirements for an information system and the users' and developers' understanding of the requirements are beneficial. However, without knowledge of the factors involved in misunderstanding requirements, they may be suboptimal and new techniques could be created that are more effective.

The present study seeks to identify the factors that contribute to users and developers misunderstanding requirements. Further, the factors will be prioritized to determine the importance users and developers assign to each factor.

Background of the Study

Mrenak made the observation that the development of information systems is concerned with two core problems (1) "figuring out what to do" and (2) "figuring out how to do it" and that the first problem has received the least attention (1990, p. 17). The "figuring out what to do" part of developing information systems is known as requirements determination, which is intended to produce a clear understanding of the problem that needs to be solved and the requirements for the information system. A correct, complete understanding of requirements is the foundation for quality software and reduces the cost of an information system development project. However, communication problems and misunderstandings between stakeholders, particularly between users and developers, create requirements determination difficulties (Coughlan et al., 2003). Brooks, who is frequently cited for his early perspective of the difficulty understanding requirements said:

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. Therefore, the most important function that the software builder performs for the client is the iterative extraction and refinement of the product requirements. (1987, p. 17)

The importance of getting the requirements for an information system right—clearly understanding the requirements—is directly related to the success, cost, and duration of the development project. A good information system solves the problems it was created for and meets the expectations of its users (Tiwana & Keil, 2004). Misunderstanding the requirements for the system leads to dissatisfied users, increased project cost, and longer project schedules (Hickey & Davis, 2004; Wiegers, 2003). Negligible research has been conducted to understand

why misunderstandings occur. Improved knowledge of why requirements are misunderstood, particularly between users and developers, will begin a theoretical foundation for creating means of improving requirements determination. The results of this research will benefit the future of requirements determination, aid the creation of successful information systems and may have applicability to other types of software development projects.

Statement of the Problem

Users and developers misunderstand requirements for information systems, which leads to requirement errors and the development of inappropriate software, contributing to increased project cost and schedule while decreasing project success. Although literature contains assumptions and hypotheses for factors influencing misunderstanding of requirements, little research exists. The problem being studied is to determine what users and developers of information systems say are the factors that influence why requirements are misunderstood. Further, the prioritization of the factors will be determined from the separate perspectives of users and developers.

Purpose of the Study

The purpose of the study is to begin building the theoretical foundation for why users and developers misunderstand requirements for information systems. This will enable the creation of more effective and efficient techniques for understanding requirements. To limit the scope of the study, the term *information systems* is confined to software systems created in-house by an organization to be used within the organization and does not include outsourcing situations. Insights will be derived from three areas: (a) factors influencing misunderstood requirements as identified by users and developers, (b) priority of the factors assigned by users and developers, and (c) similarities and differences between users and developers in the generation and prioritization of factors.

By studying what influences developers and users to misunderstand requirements, software project managers can begin seeking ways to reduce these influences, therefore reducing misunderstandings. The result is expected to ultimately enable the creation of software that is more successful with its users, decreases development costs, and provides better schedule control. For information systems, these benefits also improve a systems' return on investment.

Rationale

The original intent of this study was to investigate practical means of bridging the gap between users and developers of information systems by creating improved requirements determination techniques. Doing so has the potential to significantly improve user satisfaction of information systems while decreasing costs. Although the user-developer gap is well acknowledged in literature, a theoretical framework for understanding the gap does not exist (Wieggers, 2003). Consequently, the research focus was altered to contribute to a theoretical foundation of why users and developers misunderstand requirements, often resulting in an expectation gap between what users wanted and what they received.

Research Questions

The research question hierarchy of Cooper and Schindler (2003) is used to construct the questions answered in this research, and is fully presented later in the Chapter 3, including the business dilemma, management question, and research questions. The primary research questions are:

1. Which factors do users and developers believe cause misunderstandings about the requirements for information systems?
2. Which factors do users and developers believe have the most impact on misunderstandings?

3. What is the difference between users' and developers' perceptions of these factors?

Answers to these questions have not been found in previously published research. The most similar study found examined critical productivity factors for requirements determination identified by users and developers (Havelka, 1994). A lack of knowledge exists about why requirements for information systems are misunderstood.

Conceptual Framework

According to Miles and Huberman (1994), a "conceptual framework explains, either graphically or in narrative form, the main things to be studied—the key factors, constructions or variables—and the presumed relationships among them" (p. 18). Figure 1 shows the conceptual framework for the present study.

As highlighted on the figure, the key variables under investigation are the *user perceived factors* (UPFs) that influence misunderstanding requirements from the perspective of users and the *developer perceived factors* (DPFs) that influence misunderstanding requirements from the perspective of developers. These variables affect the *problem domain*, consisting of business objectives, users, developers, and requirements. The requirements stem from the *business objective* that needs to be addressed by an information system. Users formulate a *concept of the business objective*, as do developers. Users and developers work together via *knowledge sharing* to describe their concepts of the business objective and the requirements as they view them, producing a *user understanding of requirements* and a *developer understanding of requirements*. The result is *users' view of requirements* and *developers' view of requirements*. A subset of these requirements is shared between users and developers—the *common understanding of requirements*. Other requirements are not understood by users and developers correctly, causing *misunderstood requirements*.

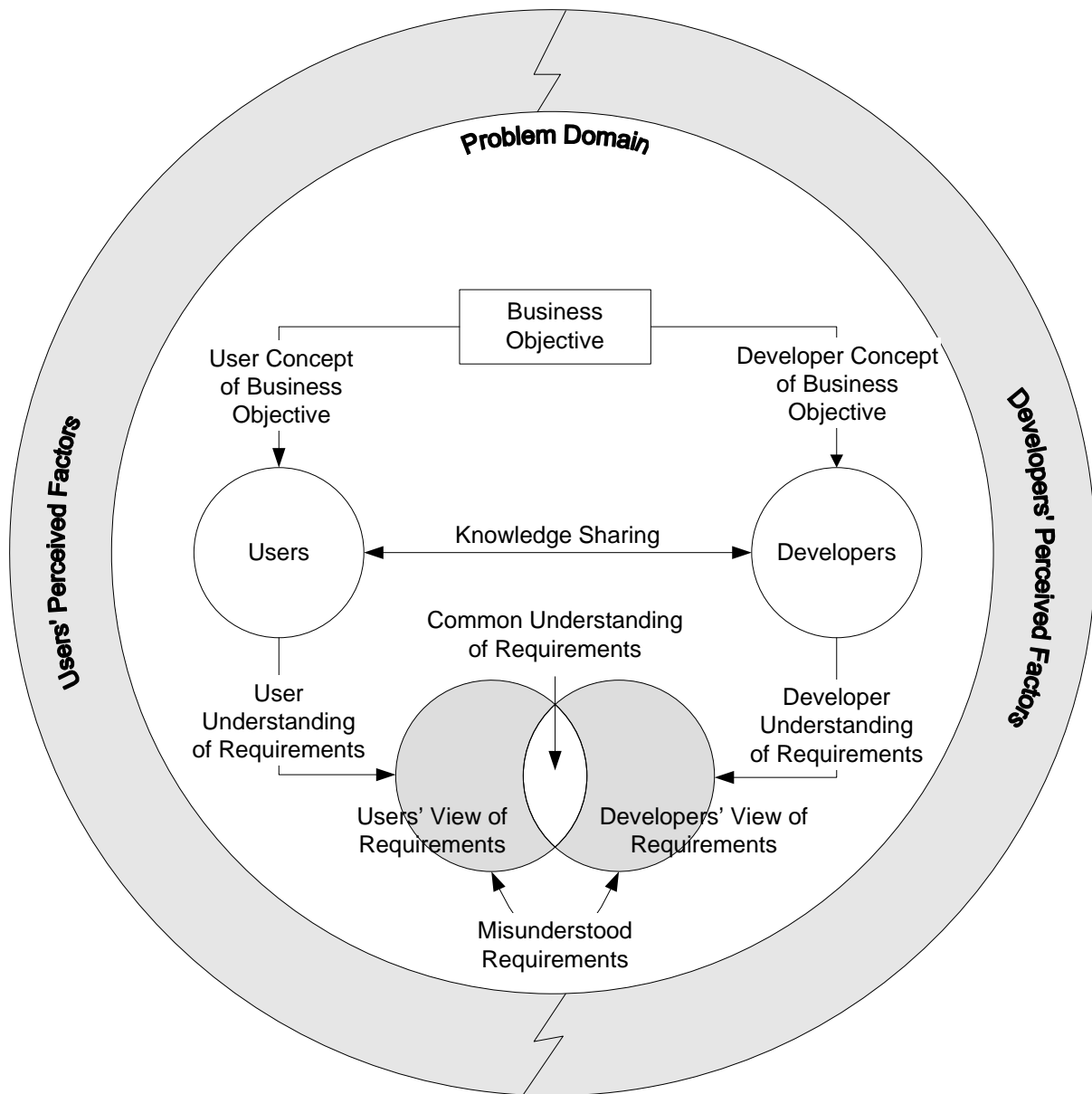


Figure 1. Conceptual framework for studying what influences users and developers to misunderstand requirements.

This conceptual framework was developed as a synthesis of existing frameworks from literature. First, Mrenak (1990) noted that both users and developers have a concept of the

problem and the corresponding information system that is needed. Over time, the users' and developers' concepts evolve in different contexts and can diverge. The context for users is associated with the real-world problem while the context of the developer is associated with the software contract, resulting in different concepts of the problem. Second, information systems are created in a process of knowledge sharing between users and developers, with users owning the knowledge about the problem and what the software must do and developers owning the technical knowledge about how software is constructed (Tiwana & Keil, 2004). Knowledge sharing occurs using one or more communication vehicles, such as interviews, surveys, requirements workshops, user interface prototypes, and the like (Keil & Carmel, 1995).

Both the perception of the problem and the knowledge sharing that occurs are influenced by conflicting goals between users and developers, with users responding to organizational imperatives and a need to complete one or more tasks while developers are concerned with technical imperatives and wish to construct a technically excellent solution (Cushing, 1990). These and other factors are expected to influence the creation of requirements that are understood differently between users and developers. Additional literature is explored in Chapter 2.

Significance of the Study

Misunderstood requirements is one of the three major reasons why 66% of information systems fail to meet users' needs, are delivered late, or exceed budgets (Standish, 2002, 2005; Walsh, 2003; Xia & Lee, 2004). The other two reasons are also related to requirements: lack of user involvement and changing requirements. Gaining knowledge about why users and developers misunderstand requirements will provide a foundation for improving information systems and possibly improving the success rate of information systems.

Based on the three primary research questions of the present study, conclusions were drawn principally from three areas:

1. The UPFs and DPFs that influence users and developers to misunderstand requirements for an information system.
2. The similarities and differences in UPFs and DPFs prioritizations.
3. The similarities and differences in the definitions of UPFs as defined by users and the DPFs as defined by users.

By knowing why requirements are misunderstood, one will be better prepared to devise ways to improve users' and developers' understanding of requirements. Although many methods have been proposed for this, encompassed under the philosophy of Voice of the Customer (VOC), a theoretical knowledge of the factors responsible for misunderstanding is lacking. With knowledge of the factors, enhancements to the VOC philosophy and specific techniques, as well as the creation of new methods, can lead to more effective and efficient requirements determination processes. An immediate outcome of the present study is the ability of software project managers to reduce misunderstandings by reducing the factors that influence misunderstandings. Project managers will be better informed to identify factors on their development projects that are jeopardizing the projects' success. Further, creators of requirement elicitation techniques may learn ways to improve their processes.

Definition of Terms

User. Users of information systems are "those individuals with a direct or indirect interest in the software product and the real-world problem solved by the software product. User concepts reflect the most relevant perceptions, experience, and preferences because they are most strongly influenced by the real-world problem and because it is the user who will eventually

apply the software product" (Mrenak, 1990, p. 19). Users are a source of the "what" of an information system.

Developer. A developer of information systems is "... usually not associated with the [user], [but] translates the requirements of the problem into a computer and software solution. Software developers are interested in fulfilling the requirements of the software development contract" (Mrenak, 1990, p. 19). Developers provide the "how" of an information system.

Requirements Determination. "The overall process of getting at, analyzing and documenting the requirements." This includes (a) "Functional features (description of the requirements, e.g., provide for automatic order generation based on pre-established reorder level)," (b) "Nonfunctional arrangements (performance and reliability stipulations)," and (c) "Constraints" (Duggan & Thachenkary, 2003, p. 392). Synonyms for requirements determination include requirements engineering, requirements definition, requirements management, and the like.

Requirements Elicitation. Requirements elicitation is at the heart of requirements determination and involves "learning, uncovering, extracting, surfacing, or discovering needs of customers, users, and other potential stakeholders" (Hickey & Davis, 2004, p. 67). Several approaches are used to elicit requirements, such as interviews, surveys, and observations.

User Perceived Factor. User perceived factors (UPFs), which will be identified in this study, are those variables users say are involved in requirements determination of information systems that influence requirements being misunderstood.

Developer Perceived Factor. Developer perceived factors (DPFs) are similar to UPFs, except they are the factors from the point of view of developers. DPFs will be identified in this study.

Assumptions and Limitations

Insights are expected to be gained about why users and developers misunderstand requirements. Limitations of the study are largely due to its exploratory nature and include:

1. Relative small sample size that may not be representative of information systems development across the company or in other companies.
2. Variations in software development processes may impact the factors users and developers would identify.
3. Factors and their judged importance may differ based on project success, experience, and other variables that cannot be sufficiently controlled or examined with a small number of participants. A broader study is needed to investigate such dependencies.
4. The study focused on misunderstandings of requirements between users and developers. Although these are typically the two key groups in the development of software, influences of other stakeholders were not considered.
5. The factors identified are from an information systems domain and may or may not be the same factors that would be generated from other domains, such as commercial software development or new product development in the manufacturing domain (Keil & Carmel, 1995). However, the study can be reproduced to test the validity of the results in these domains.
6. Butler and Fitzgerald found users to be diverse and not from a homogenous group (1997). They can belong to different organizational groups, vary in world views, and possess varying institutional perspectives. The results found in this study may not be representative of all users, and the same is expected to be true of developers.

Nature of the Study

Two phases were used to collect data. First, a qualitative phase using the Nominal Group Technique (NGT) with small groups of users and developers involved in requirements

determination generated the most influential factors influencing misunderstood requirements. A second phase quantitatively assessed the importance of each factor critical to users and developers by prioritizing them using the Analytical Hierarchy Process (AHP). Data analysis assessed how users and developers are similar and different in their selection and prioritization of factors. A thematic analysis of the NGT sessions created a better understanding of the different perspectives users and developers have.

Organization of the Remainder of the Study

Chapter 2 provides a review of related literature, beginning with an understanding of requirements and their impact on the success or failure of an information system. The review continues, covering user participation, contemporary approaches for determining requirements, and factors believed to influence misunderstandings. It concludes with an assessment of the current literature. Chapter 3 presents the research methodology chosen to answer the research questions, a justification for employing NGT and AHP, the sample plan, plans for data collection, analysis, and reporting, and concerns with reliability, validity, and generalizability. Chapter 4 presents the data collected from participants in the study and the analysis of the data in light of the research questions. Chapter 5 briefly summarizes the purpose of the study, the research methodology, the key findings, and offers conclusions from the analyzed data found in Chapter 4. Chapter 5 also contains recommendations for further research.

CHAPTER 2. LITERATURE REVIEW

"Miscommunication and misunderstanding between software developers and users are at the heart of the requirements identification dilemma" [italics added] (DeBellis & Haapala, 1995, p. 35).

Overview of Chapter

The chapter begins by setting the stage for the necessity of clear requirements and the negative impact of misunderstood requirements on an information system. This foundation is laid by (a) reviewing common information system development activities and the role of requirements in software development life cycles, (b) describing requirements and requirements determination, and (c) building a case for the negative impact of misunderstanding requirements. Given the importance of clearly understanding requirements, several contemporary approaches for determining requirements are discussed and an analysis of their creation provided. The analysis identifies a lack of fundamental knowledge of the factors that influence misunderstood requirements. Consequently, a review of related literature is presented to identify potential factors. The chapter concludes with an assessment of current literature relevant to the problem of misunderstood requirements for an information system and a description of an enhanced conceptual framework based on the literature.

Introduction to Developing Information Systems

The body of literature related to the development of information systems is immense. Before focusing specifically on the applicable sources of requirements literature, a general understanding of software development methodologies used in the creation of information systems is useful to put the importance of requirements in perspective with other activities.

Frequently such methodologies are known as "software development life cycles" (SDLCs), which are prescriptive models that describe a sequence of activities for creating software systems (Scacchi, 2001). SDLCs began in the early days of programming as a means to manage software projects by providing a step-by-step process that typically moved from requirements, to design, to coding, to test. Basic SDLCs are covered in this section to show the relationship of requirements determination to other activities involved in the development of information systems. Further, SDLCs are a moderating variable of the research, and additional studies are needed to understand the impact of various SDLCs on users' and developers' perceived factors for misunderstanding requirements.

Perhaps the best known and most misunderstood SDLC is the Waterfall, depicted on Figure 2, which was created based on Royce's (1970) experience managing large aerospace software projects in the 1960s. The name is reflective of the nature of a waterfall, with the development of software flowing from step to step as a waterfall does. The purpose of the Waterfall was to provide a rational means for managing large software development by breaking a project into several sequential activities (Scacchi, 2001).

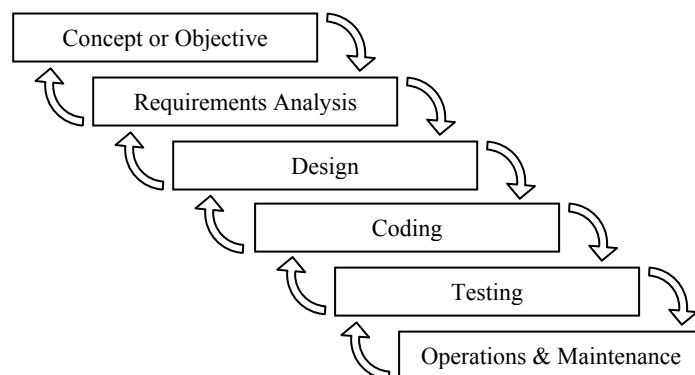


Figure 2. Waterfall SDLC depicting iteration between steps.

Royce proposed successive iterations between each discrete phase so as the design becomes more detailed it is iterated with the next phase and the previous phase. He observed that a purely sequential process did not accommodate changes in requirements after they had been defined, and considered the process depicted on Figure 2 to be inviting failure and likely doubling the cost of a project if extra care was not taken. To overcome this problem, he provided additional guidance for minimizing development risk. These included creating detailed documentation for each phase of development, iterating the product so the operational release is at least the second version of the software, and involving the customer at key periods to get official approval of requirements. Inaccurately, the Waterfall SDLC is frequently characterized as linear and not the iterative approach that Royce had intended.

Another popular class of SDLCs is known for its incremental and iterative development (IID) characteristics. The adage, build a little, test a little, build a little, aptly describes IID approaches, where software is created through cycles, with each cycle adding required functionality. With this as a working description, several SDLCs could be considered to be IID approaches, including but not limited to Incremental, Spiral, Staged Delivery, and Evolutionary Delivery (Futrell, Shafer, & Shafer, 2002; McConnell, 1996). Although IIDs are commonly thought to be the modern replacement to the Waterfall (Laplante & Neill, 2004), such approaches are circa that of the Waterfall. Larman and Basili (2003) attribute the roots of IID to the quality movement started by Walter Shewart and popularized by W. Edwards Deming (Russell & Taylor, 2003). Early examples of using IID cited by Larman and Basili were the X15 hypersonic aerospace program of the 1950s (Dana, 1993) and later NASA's Project Mercury of the 1960s. Others, notably Gerald M. Weinberg, Tom Gilb, and Robert Glass, also discussed the application of IID in the 1960s (see Larman & Basili, 2003 for additional references).

An example of an IID SDLC is staged delivery, which was popularized by Steve McConnell (1998). This approach begins with developing requirements, then the requirements

are incrementally implemented starting with the most important requirements first. Each stage progressively demonstrates more functionality. The software is complete when the final stage has incorporated all requirements. Since requirements are likely to evolve between stages as the problem is better understood, requirements can be reassessed at the beginning of each stage. The process is depicted on Figure 3.

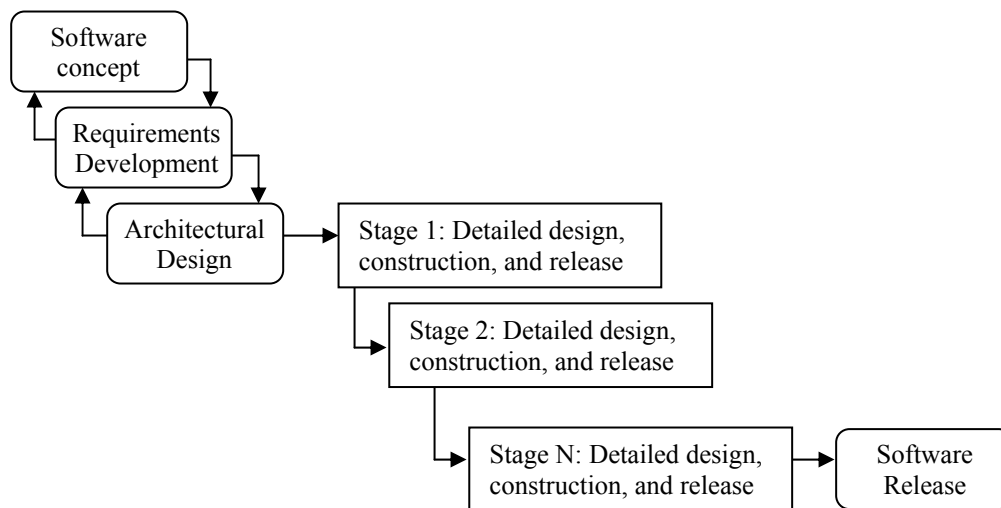


Figure 3. Depiction of staged delivery SDLC.

"Agile" software development is the most recently named SDLC category, with its formal roots originating in 2001, although arguments can be made that agile principles, albeit in different terminology, date back to the beginning of software (Larman & Basili, 2003).

Waterfall, and to a lesser degree, IID approaches, are document driven and can be procedurally heavy. In contrast, Agile approaches are meant to be exceptionally minimalistic, flexible, and responsive to users' changing needs. Throughout the 1990s several software engineers pursued these goals, using so-called "lightweight" methodologies (Fowler & Highsmith, 2001). In 2001,

experts met to discuss aspects of various approaches and officially created the "Manifesto for Agile Software Development," which expresses the four most important characteristics of Agile development (Beck et al., 2001): (a) individuals and interactions over processes and tools, (b) working software over comprehensive documentation, (c) customer collaboration over contract negotiation, and (d) responding to change over following a plan. Another consistent theme with Agile approaches is a recognition that software development is non-linear and non-repeatable because of numerous factors that can induce change during a project, such as changes in requirements, technology, and team composition (Williams & Cockburn, 2003).

Two popular Agile approaches include Extreme Programming (XP) (Beck, 2000) and SCRUM (Schwaber, 2004). Several other approaches, which some would argue are more philosophies than methodologies or SDLCs, also exist (see AgileAlliance, 2005 for additional examples). Agile approaches cannot be diagrammed as easily as other SDLCs and may appear to be more reactive than predictive, but they still contain the basic software development tasks with a focus on short development iterations, incremental planning and requirements creation, and evolutionary design (Abrahamsson & Koskela, 2004, Extreme Programming section).

A synthesis of several SDLCs produces the software development activities or phases that are common to developers of information systems, shown in Table 1. Regardless of the specific SDLC employed, the significant activities remain the same.

Table 1. *Activities Common in the Development of Information Systems*

Activity	Description
Requirements Determination	"Involves all life-cycle activities devoted to identification of user requirements, analysis of the requirements to derive additional requirements, documentation of the requirements as a specification, and validation of the documented requirements against user needs, as well as processes that support these activities" (DoD, 1991).
Design Phase	"The period of time in the software life cycle during which the designs for architecture, software components, interfaces, and data are created, documented, and verified to satisfy requirements" (IEEE, 1990a).

Table 1. *Activities Common in the Development of Information Systems, Continued*

Activity	Description
Code Construction	"The transforming of logic and data from design specifications (design descriptions) into a programming language" (IEEE, 1990a).
Test Phase	"The period of time in the software life cycle during which the components of a software product are evaluated and integrated, and the software product is evaluated to determine whether or not requirements have been satisfied" (IEEE, 1990a).
Operations and Maintenance Phase	"The period of time in the software life cycle during which a software product is employed in its operational environment, monitored for satisfactory performance, and modified as necessary to correct problems or to respond to changing requirements" (IEEE, 1990a).

Note. These definitions are used by the Software Engineering Institute (SEI, 2004).

Of these activities, there is general agreement in literature and by practitioners that requirements determination is the most critical, most troublesome, and least understood part of developing an information system (Hevner & Harlan, 1995). Requirements determination consists of several tasks, which are described in Table 2 using the descriptions provided by Hickey and Davis (2004, p. 67). Although literature contains various titles for these activities, the tasks generally exist under one name or another.

Table 2. *Common Requirements Activities*

Activity	Description
Elicitation	Learning, uncovering, extracting, surfacing, or discovering needs of customers, users, and other potential stakeholders.
Analysis	Analyzing the information elicited from stakeholders to generate a list of candidate requirements, often by creating and analyzing models of requirements, with the goals of increasing understanding and searching for incompleteness and inconsistency.
Triage	Determining which subset of the requirements ascertained by elicitation and analysis is appropriate to be addressed in specific releases of a system.
Specification	Documenting the desired external behavior of a system.
Verification	Determining the reasonableness, consistency, completeness, suitability, and lack of defects in a set of requirements.

Within the spectrum of information system development activities, this study is focused on requirements determination and specifically why users and developers of information systems misunderstand requirements. Although such a focus may warrant an emphasis on requirements elicitation because this is when requirements are first shared, it is not the activity responsible for identifying misunderstandings, although this may naturally occur. Misunderstandings may be identified by any of the activities performed in requirements determination, or at any other place in an SDLC. Further, misunderstood requirements may not be found until after the information system has reached Operations and Maintenance and users interact with the system in their operational environment. Consequently, with the focus on establishing well understood requirements, the development of an information system can be simplified to three main activities, summarized on Figure 4: (a) identifying the problem or business objective that needs to be solved, (b) developing the requirements for a solution in response to the problem, and (c) creating the information system in accordance with the requirements.

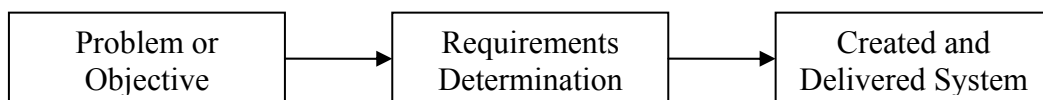


Figure 4. Simplified view of developing an information system, emphasizing requirements.

With an understanding of the role of requirements in creating an information system, the foundation for understanding the importance of clearly understood requirements continues to be built in the next sections by discussing requirements and requirements determination in more detail.

Requirements and Requirements Determination

What are Requirements

Requirements include business- and user-related perspectives of problems, needs, wants, desires, features, capabilities, objectives, goals, and so forth (Hickey & Davis, 2004). Darlington and Cully (2002) define requirements as “the expression of customer needs in relation to the real world” (p. 376) while Coughlan et al. (2003) relate requirements to what users value.

Other perspectives on requirements exist in literature, such as making a distinction between functional requirements—what users want the system to do—and non-functional requirements—constraints or capabilities that are not visible to users (Wiegers, 2003). In this study, the term *requirements* is consistently used in the Hickey and Davis (2004) connotation of user-related needs and objectives. Davis further clarifies what a requirement is and is not by providing two tests that a valid requirement must pass (Davis, 2005): (a) the satisfaction of the requirement must be externally observable and not require an internal understanding of the system to be viewable, and (b) the requirement must meet a desire or help to satisfy a need of users or other stakeholders.

What is Requirements Determination

Requirements determination is “a systematic approach to eliciting, organizing, and documenting the requirements of the system, and a process that establishes and maintains agreement between the customer and the project team on the changing requirements of the system” (Leffingwell & Widrig, 2000, p. 16). This process involves more than understanding the features and capabilities needed by users. It also involves the “identification of goals, assumptions, opinions, and desires of users” (Browne & Rogich, 2001, A Model ... section). The elements of requirements determination have been described in a variety of similar ways with requirements elicitation as the cornerstone activity (Browne & Rogich, 2001; Herlea, Jonker,

Treur, & Wijngaards, 2002; Jin, Bell, Wilkie, & Leahy, 2003; Lamsweerde, 2000; Saiedian & Dale, 2000).

Lamsweerde (2000) provided a historic review of requirements determination. He summarized several reasons why requirements determination is complicated, including:

1. Creating requirements involves more than just the proposed information system. It also encompasses the environment the software will operate in, becoming a problem with social-technical, social-economic, and other dimensions.
2. Both functional and non-functional concerns need to be addressed in the requirements. Non-functional issues involve safety, security, performance, usability, maintainability, reliability, cost, and the like, and they may be in conflict with each other. For example, a highly reliable system may also be very costly.
3. The stakeholders involved in the requirements process, such as users and developers, have different backgrounds, motivations, perceptions, concerns, personalities, and other factors that can create conflicting viewpoints and misunderstandings.

To complete the foundation of the importance of requirements, the next section presents the negative consequences of misunderstanding requirements.

Why Requirements Matter

The Importance of Requirements is Well Established

Practitioner experience and numerous research studies show that a major source of a system's success or failure is tied to understanding the requirements for the system—a clear understanding of the problem that needs to be solved and the requirements of the solution leads

to a system that satisfies users. Anything short of a clear understanding is a recipe for failure and dissatisfaction. This was recognized years ago as the adoption of information systems increased in businesses (Brooks, 1987) and it is still a leading factor in the creation of systems today (Wieggers, 2003). Accurately and completely understanding requirements continues to be one of the most important activities in the development of information systems (Herlea, 1999; Hickey & Davis, 2004; Moody & Sindre, 2003).

When ranked, understanding requirements consistently appears at or near the top of activities that contribute to a successful system. Osmundson, Michael, Machniak, and Grossman found it ranked among the top four success factors (2003). Browne and Rogich, based on their research and that of others, said "A principal reason that systems do not meet user expectations is the failure of the development process to yield a complete and accurate set of requirements" (2001, Requirements Determination section). Further, incorrect, lacking, misunderstood, and incomplete requirements are top reasons why systems fail (Havelka, 2003). Several other studies reached similar conclusions about the importance of fully understanding software requirements (Havelka, 2003; Ibanez & Rempp, 1996; Kenney & Leggiere, 2003; Schmidt, Lyytinen, Keil, & Cule, 2001; Standish, 2005; Valenti, Panti, & Cucchiarelli, 1998; Walsh, 2003).

Misunderstood Requirements Are Associated with Failure

The Standish Group, known for its research of information system projects, defines a successful project as one that is "completed on time and budget and delivers all originally specified functions and features" (Kenney & Leggiere, 2003). Since 1994 the Standish Group has published its Chaos report that gives an accounting of successes and failures of information systems projects (Standish, 2005). The 2002 Standish Chaos report found that 66% of information systems projects fail, a number that has varied little since their original report (Walsh, 2003; Xia & Lee, 2004). Lack of user involvement, misunderstood requirements, and changing requirements are cited as the key factors for project failures.

Schmidt, Lyytinen, Keil and Cule (2001) conducted an international Delphi study of risk factors for software project failures and determined that misunderstanding user requirements was ranked as the third most important risk out of a total of 29 risks. Another indication of the importance of well-understood requirements is reflected in Cocomo II, a software estimation model. It predicts the size of a software application before it is constructed based on several factors such as programmer capability, tool experience, or development process. The skill of the requirements analyst, the person responsible for requirements, carries the most weight in the Cocomo model, underscoring the significance of requirements (McConnell, 2000).

As part of a 20-month European study to improve the development of quality software, organizations were surveyed to assess their current software engineering practices. A reported 3805 respondents categorized their main problems with software development against 12 factors as *major problem*, *minor problem*, or *never a problem*. The two factors most responsible for problems were *requirements specifications* and *managing customer requirements*, shown on Figure 5 outlined by a box.

In an international investigation of risk factors of information systems development using the Delphi technique, "misunderstanding the requirements" was the second most important risk, closely following "lack of top management commitment to the project" (Keil, Cule, Lyytinen, & Schmidt, 1998). Reflecting the insights of the Delphi experts, the researchers said, "Without a proper systems analysis to develop a complete and accurate set of requirements there is a distinct possibility of building a system that no one wants to use" (p. 79).

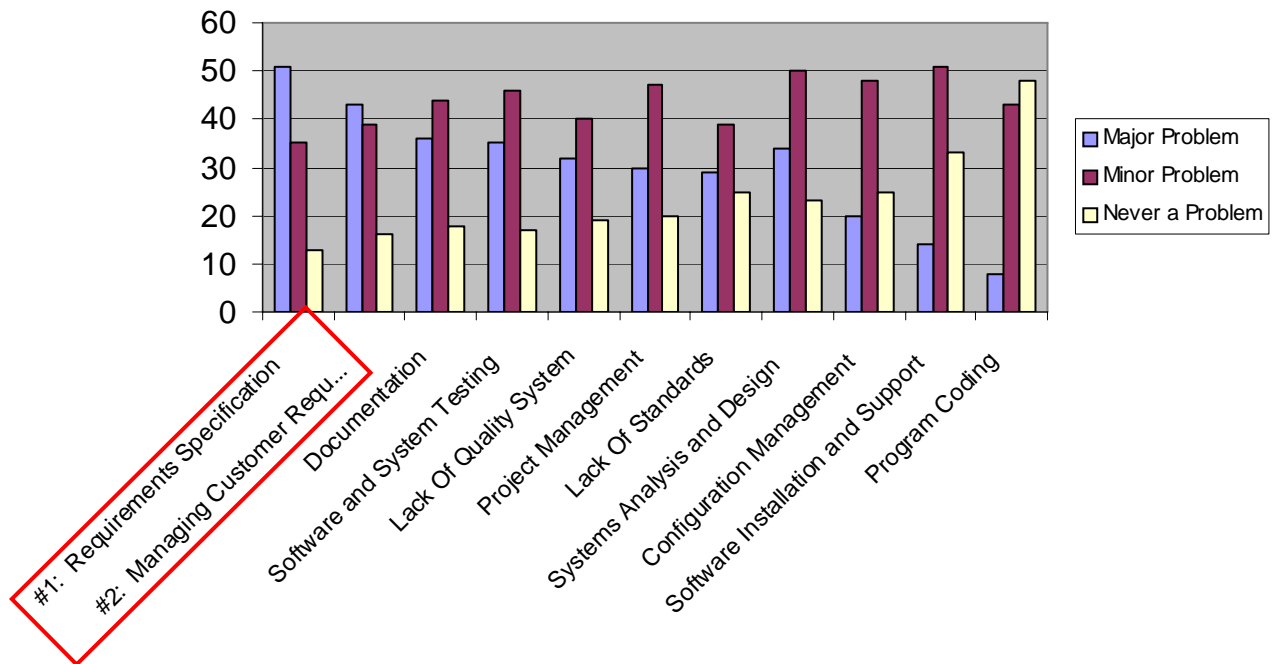


Figure 5. Factors responsible for software development problems.

From "European User Survey Analysis," by M. Ibanex and H. Rempp, 1996, retrieved 6/6/2004 from <http://www.esi.es/VASIE/Reports/All/11000/Download.html>. Copyright 1996 by ESI. Adapted with permission.

The Cost of Requirement Errors

In addition to the importance of understanding requirements to the success of a system, the price of misunderstanding them is high. Errors from inadequate requirements determination have been found to be responsible for 40 to 60% of all defects in the development of a system (Davis, 1995; Moody & Sindre, 2003; Wiegers, 2003). These errors account for 25 to 40% of the total budget of the system (Leffingwell & Widrig, 2000). Wiegers adds, "Nonetheless, many organizations still practice ineffective methods for these essential project activities. The typical outcome is an expectation gap, the difference between what developers think they're supposed to build and what customers really need" (p. 4).

To put the cost of requirement errors in perspective, recall the information system development activities presented previously in Table 2. The relative cost for detecting and

correcting a requirement error during each of these activities is depicted on Figure 6. The numbers are a compilation of studies at several companies that were first compiled by Boehm (1981) and discussed by Davis (1993). Given an approximate relative cost of one to detect and correct a defect in the requirements determination phase, the cost increases with each phase as designs are created, code is constructed, and so on, until the approximate relative cost is 200 times greater in the maintenance phase. This means that an inadequate requirements determination process, one that does not correctly identify the problem to be solved, does not discover all necessary requirements, or misunderstands requirements, can result in errors that are as much as 200 times more expensive to correct compared to an adequate requirements determination process that creates error-free requirements.

Others have also made the observation that it is considerably less expensive to correct requirements before they are implemented and tested (Darlington & Culley, 2002; Davis, 1993; Moody & Sindre, 2003), corroborating that the cost savings can be as much as 200 times. Even if lifecycle approaches that emphasize users' requirements are used, such as evolutionary prototyping or rapid application development (Elliott, 2000), the cost to correct requirements errors grows the longer they remain undetected. Leffingwell and Widrig (2000) give two reasons for this increasing cost: (a) time invested in all other activities (e.g., design, coding, etc.) is wasted if requirements are wrong, and these activities will have to be redone, and (b) the natural assumption that errors discovered during testing or inspection are related to design, which causes considerable time to be wasted until it is recognized that they are from errors in requirements. The cost of requirement errors is not limited to the price of creating the system. If the requirements are wrong or incomplete, the system may not be used, or its use results in lower productivity.

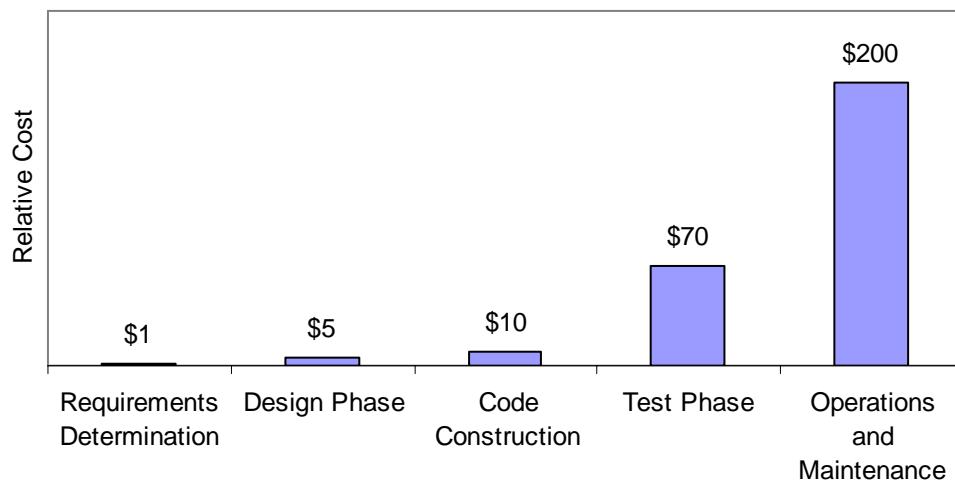


Figure 6. Relative cost to identify and correct defects at different phases of software development.

Contemporary Approaches to Determining Requirements

With such importance given to understood requirements, it is no surprise that several approaches have been used to improve requirements determination, and consequently, users' and developers' understanding of requirements. Some are more user-oriented techniques, such as a natural language approach (Lu, Jin, & Wan, 1995), while others are more developer-oriented techniques, such as employing intelligent computer aided design (Williamson & Healy, 2000). Lamsweerde (2000) provides brief descriptions of dozens of requirement approaches and Hickey and Davis (2004) note that several approaches exist: “Interviewing, questionnaires, observation, modeling, prototyping, and collaborative requirements workshops are just a few of the many hundreds of elicitation techniques available today” (p. 74).

This section reviews a few popular and contemporary approaches, including voice of the customer, quality function deployment, interviews, requirement workshops, prototyping, scenarios, observation, user advocate, and software development methodologies. The section

concludes with an assessment as to how the approaches impact users and developers misunderstanding requirements.

Voice of the Customer

VOC is intended to help developers understand the needs of users. VOC is a component of QFD, discussed next, but also used in other customer-focused methodologies (Griffin & Hauser, 1993). As such, it is a philosophy of development rather than a specific technique. Simply, VOC captures the needs of users (customers) in their own words, and prioritizes the importance of each. VOC is seen in quality programs, such as Total Quality Management (TQM), because quality is frequently equated with user satisfaction. This is evident in the Baldrige National Quality Award core values that stress the importance that "quality and performance are judged by an organization's customers" (NIST, 2005, p. 1). VOC can be accomplished using a number of techniques, some of which are described elsewhere in this section, but one-on-one interviews are most often used (Katz, 2004).

Quality Function Deployment and House of Quality

Quality Function Deployment (QFD) was created in Japan, originally at the Mitsubishi Shipyard in 1972 as an element of Total Quality Management, and was later refined by Toyota to improve manufacturing (Eldin, 2002). Other organizations have adopted QFD processes as they strive to become more customer-driven (Hales, Lyman, & Norman, 1994) and benefit from claims of up to 60% reduction in design costs (Griffin & Hauser, 1993). QFD has been applied to the development of software and combined with other methodologies to improve information systems (Duggan, 2003; Pai, 2002; Yang, Jang, Yeun, Lee, & Lee, 2003; Zrymiak, 2003).

QFD is composed of several tools, including "the use of focus groups, house of quality (HOQ), affinity diagrams, tree diagrams, benchmarking, value engineering, and market research" (Eldin, 2002, p. 28). Of these, HOQ is most helpful in recognizing conflicts and prioritizing requirements. An HOQ is constructed as a matrix that shows the relationships between user

requirements, known as the voice of the customer (VOC) or the *what*, and design requirements, also called the voice of the developer or the *how*. The combination of user and developer views is intended to improve the understanding of requirements. Using HOQ involves prioritizing or weighting user requirements. This process helps users to better understand what is most important to them and helps developers understand how users decide to make tradeoffs. The critical component of HOQ is correlating customers' requirements with developers' requirements. This is accomplished by taking each combination of customer and engineering requirements (the *what's* and *how's*) and specifying if a weak, moderate, or strong correlation exists. Next, the engineering requirements are analyzed in pairs to determine whether a correlation exists. Pairs of engineering requirements with negative correlation indicate a conflict to be resolved.

A completed HOQ matrix provides a visual means of identifying up to nine problems with the requirements (Mazur, 1997). For example, a blank row in the matrix indicates that a user requirement has not been accounted for in the engineering requirements. The HOQ provides a way of bridging the work done by users in specifying system requirements and the work developers do to turn requirements into a completed system. When conflicts are identified, users and developers together can work to resolve them or use them as a point requiring innovation.

Interviews

According to Browne and Rogich (2001), interviews are used more than any other requirements elicitation technique. They studied interviewing techniques using directed questions; that is, questions that help users focus on specific aspects of a system. There are two types of interviews: domain-specific and domain-generic.

Domain-specific interviews are conducted by individuals with a deep understanding of the problem domain. By knowing the domain well, interviewers bring an immediate appreciation and understanding of the users' problems to the task of eliciting requirements. They have prior knowledge about the environment that can help others understand the problem (Hickey & Davis,

2004). The disadvantage of prior domain knowledge is that it can cause implicit requirements to be ignored and never made explicit (Browne & Rogich, 2001), which could restrict innovative thinking (Coughlan et al., 2003; Curtis, 1990).

Browne and Rogich (2001) created a domain-generic approach to interviewing that produces results comparable to domain-specific approaches. They synthesized theories about human problem solving, cognitive psychology, and information systems development and developed a list of open-ended questions that can be used for any problem domain. This interview approach has three advantages: (1) analysts do not need prior domain knowledge about the system, (2) the same approach can be used in multiple types of environments, and (3) because analysts do not fully understand the problem, biases and past experiences do not taint the requirements (Berry, 2002).

Workshops and JAD

Leffingwell and Widrig (2000) recommended workshops as the most useful elicitation technique, and Coughlan et al. (2003) found them to be widely used. Workshops, also called facilitated workshops, requirement workshops, user centered designs, and the like, focus on user-developer interaction. They are designed to convene key stakeholders for the purpose of reaching a consensus about the requirements for a software system. Workshops focus on understanding the problem, brainstorming solutions, and formulating an action plan. By the end of a workshop, requirements are known, political positions have been discussed, and a strong foundation has been laid for future negotiations about software requirements (Gottesdiener, 2002).

One commercially developed workshop methodology is known as Joint Application Design (JAD). This formal workshop approach was created by IBM in the 1970s to improve on frequently used interview-based approaches. The purpose of JAD is to get "the right people in a room together with a skilled neutral facilitator and, in a week or less, find out exactly what the user wants" (Mrenak, 1990, p. 21). JAD accomplishes requirements determination by

emphasizing collaboration and communication between key stakeholders, including users and developers (Duggan, 2003). An independent facilitator is relied upon to enhance communications between each of the participants, reduce conflict, and increase rapport. Table 3 lists the primary activities that occur in a JAD workshop.

The advantages of workshops are that they bring stakeholders together, break down walls between people, create rapport helpful for future requirement negotiations, and provide everyone with a big picture of the problem. Workshops are less valuable when key people are not involved. Key stakeholders may believe they do not have the time to attend a multi-day workshop, and they may send a surrogate to the workshop. If the surrogate cannot make decisions or does not understand the problem, workshops become unproductive (Coughlan et al., 2003). Further, workshop success is dependent on the quality and experience of the workshop facilitator. They also place the emphasis on completing requirements determination, which may not be practical if requirements are likely to change or if an incremental SDLC is used.

Table 3. *JAD Workshop Activities*

Stage	Activities
1. Project definition	<ul style="list-style-type: none"> a. Determine system purpose, scope, and objectives b. Identify JAD team members c. Establish project schedules
2. Background research	<ul style="list-style-type: none"> a. Gather background details about the user requirements b. Explore the technical, social, political implications c. Consider general system issues, agree what needs to be decided in the session
3. Pre-workshop prep	<ul style="list-style-type: none"> a. Prepare for the session b. Finalize logistics for the meeting c. Procure visual aids, working documents, and other meeting apparatus d. Train the scribe(s)

Table 3. *JAD Workshop Activities, Continued*

Stage	Activities
4. The workshop	a. Pool the information and knowledge of JAD team members in the analysis of potential solution b. Generate solutions (system requirements) during the three- to five-day session c. Finalize and document meeting decisions
5. Final documentation	Prepare the final document and capture decisions and agreements arrived at during the workshop

Note. From "Higher Quality Requirements: Supporting Joint Application Development with the Nominal Group Technique," by E. W. Duggan and C. S. Thachenkary, 2003, *Information Technology and Management*, 4(4) p. 394. Copyright 2003 by Kluwer Academic Publishers. Adapted with with kind permission of Springer Science and Business Media.

Prototypes

Prototypes are defined as a “a preliminary type, form, or instance of a system that serves as a model for later stages or for the final, complete version of the system” (IEEE, 1990b, p. 60). They are used to help developers, users, and customers better understand the requirements of the system. Prototypes can be simple paper mock-ups, elaborate computer presentations, or other modes of showing the software’s features (Liu & Khooshabeh, 2003). Prototypes help initiate ideas, extend existing ideas, and validate or correct ideas (Williams, 2002). Prototypes enable developers to present more information than written descriptions of software, and they can make the software more understandable to clients. However, in order to serve their purpose, prototypes must be easily changed, and users must understand that a prototype is not a complete software system. If users believe the prototype is a finished product, the developer may be unable to compile an accurate description of software requirements.

Scenarios

Scenarios are described as real-world stories about how a software system should work (Lu et al., 1995). Natural language, pictures, formal modeling representations, and other means have been used to capture scenarios (Sutcliffe, 2003). They may be used in conjunction with other elicitation methods, such as prototypes, as a way to test the flow of events after an action is

taken. Scenarios have value when they are familiar to users, but the selection of scenarios and the proper number of them are critical to the success of scenario-based elicitation. Scenarios are helpful because they anchor requirements to the users' environment and provide a way for users' stories and experiences to be generalized and represented in the software requirements. There are problems with the scenario approach: the possibility that they represent atypical events because this is what users are more likely to remember, determining if sufficient scenarios have been gathered to reflect the problem completely, and knowing if the right scenarios have been described.

Observation

The use of observation studies provides a social-technical perspective on requirements determination (Jirotko & Goguen, 1994). The observation approach, also called field studies, task analysis, ethnographic research, and contextual interviewing, is a method of "observing human interactions in their social, physical and cognitive environments" (Spillers, 2004a, What is Ethnography section). Basically, the observation approach allows the developer to discover what a user does during the workday. Observation studies are a staple of usability and human computer interaction specialists, who frequently claim that it is more important to watch what users do than it is to listen to what they say (Nielsen, 2001). Observations are a rapid way to understand users' tasks, objectives, and expectations in the context of their work environment (Spillers, 2004b). For requirements elicitation, observations can provide more insight into the problem and reveal implicit requirements, but they do not always lead to innovative ideas about how to solve the problem.

User Advocate

Much has been written about the differences between users and developers of software systems and how these differences impact requirements (Borenstein, 1991; Browne & Rogich, 2001; Darlington & Culley, 2002; G. B. Davis & Monroe, 1987; Elliott, 2000; Eriksson &

Penker, 1998; Havelka, 2003; Havelka & Lee, 2002; Jin et al., 2003; Kazmierczak, Dart, Sterling, & Winikoff, 2000; Klockner, Pankoke-Babatz, & Prinz, 1999; Lamsweerde, 2000; Saiedian & Dale, 2000; Stary, 2002). According to Klockner et al. (1999), one of the more serious communication problems is caused by developers: “Designers [developers] had a tendency to defend the features they designed, which would be a hindrance to understanding the user perspectives to this feature, as opposed to pure user advocates who could concentrate on the users’ perspectives” (p. 378).

Klockner, et al. (1999) promote the use of a user advocate to bridge the gap between users and developers and improve their understanding of requirements (Gulliksen & Lantz, 2003). The concept of a user advocate recognizes that users are not system designers and developers are not users (Nielsen, 1993). It provides an interface or bridge between the two different perspectives of these groups. Instead of asking users and/or developers to learn a new language, which may be required in modeling, interviewing, or workshop approaches, the user advocate acts as a translator.

Including a user advocate during requirements elicitation has many advantages. Developers believe user advocates help communicate their sense of ownership and pride in the software. Although these are positive qualities, unabated, the developers’ sense of ownership and pride can hinder the development of quality software. Requirements and design work may need to be radically altered or completely abandoned in favor of a new direction, and users may be intimidated by the developers’ enthusiasm. User advocates can encourage the tendency of users to view ideas in a “yes, but” manner, which can draw out true requirements (Leffingwell & Widrig, 2000). User advocates can act as a moderating influence and stop developers from rushing into development before the software requirements have been fully analyzed. This can help cut down on costly errors that may not show up until later in the development process.

Users may not appreciate how difficult it is to create a software system. They may have unrealistic expectations about how easy it will be to make changes. They view their involvement in requirements elicitation as something that can happen later, after they see the developed system. User advocates can help users understand that even though the system is developed with software it is not easy, timely, or inexpensive to make changes to the system.

In addition, user advocates understand the perspectives of users and developers, but they are not domain experts in either area. This lack of knowledge is an advantage because a user advocate who is unfamiliar with the domain of the problem can help uncover unstated assumptions (Berry, 2002). Also, by not being a developer, requirements for the system will not be tainted by a prior understanding of technology. This allows requirements to naturally flow from users in a top-down, user-focused manner (Stary, 2002). Having developers responsible for requirements determination significantly limits this possibility and may produce a bottom-up, technology-focused set of requirements. The user advocate also benefits from having limited knowledge about the users' domain because this lack of knowledge can help users and developers identify assumptions or tacit knowledge involved in the problem (Browne & Rogich, 2001). However, as with workshops, the facilitation skills of a user advocate and ability to develop rapport with both users and developers constrains their usefulness.

Assessment of Contemporary Requirement Determination Approaches

Although not an exhaustive list of approaches for determining requirements, the above discussion presents several frequently used methods to help users and developers better understand requirements. Their existence, along with numerous other methods, is evidence of the difficulties user and developers have with requirements for information systems. Although the approaches strive to improve understanding of requirements, they fail to recognize why misunderstandings occur. It is more accurate to describe such misunderstandings as a symptom of a problem, rather than the problem itself. The fundamental problem is below the surface, and

may be directly related to the reasons users and developers misunderstand requirements, which is the focus of the research presented in this study.

As an example of an underlying problem, consider Bostrom's and Thomas' recognition of the importance of effective communication concerning the requirements for an information system, adding: "Although recently a number of techniques have emerged to facilitate requirements definition (e.g., structured design methodologies), the key remains effective communication between system developers and users (1983, p. 1). Bostrom's and Thomas' work is an early and often repeated example of the software industry's response to the widely recognized and studied problem of poor requirements. They recognized a problem, proposed a solution by introducing a new methodology, applied the solution to software development projects, and then measured the effectiveness of the solution for improving the quality of requirements and the success of the constructed information system. The question remains: did Bostrom and Thomas identify the correct problem?"

For further examples of possibly missing the fundamental problem, consider the motivation for the creation and use of each requirements approach previously discussed. VOC is in recognition of the need to better understand users' requirements in users' terms. QFD, specifically the HOQ component, is intended to provide users and developers with an improved understanding of requirements, missing requirements, and conflicting requirements. Interviews are a simple way of acting on the VOC philosophy by asking users what they want. JAD, and workshops in general, was created to overcome limitations of interviews and dissatisfaction with their use. Prototypes are a means to help users and developers better understand requirements, putting the adage "a picture is worth a thousand words" into action (Leffingwell & Widrig, 2000). The Klockner, et al. (1999) user advocate approach is in response to the recognition that users and developers have different perspectives and difficulties sharing information. None of

the approaches are grounded in an understanding of factors contributing to misunderstood requirements.

Although Guinan and Bostrom's observation is more than two decades old, it continues to hold true:

It is still too often the case that [information systems] are developed behind schedule, over cost, do not do as much as promised, and do not satisfy their users. In the last few years, we have been bombarded by techniques, methods, "optimal ways" in which to approach system development. For example, techniques such as prototyping and data flow diagrams have been developed to make the requirements definition process more effective. These methods provide direction and structure to the development process. These methods, however, require the interaction/communication between user(s) and developer(s) to generate necessary requirements information. Thus, they may be unsuccessful unless effective communication patterns are used by developers and users. (1984, p. 3)

Since Guinan and Bostrom made this observation, several additional requirements determination approaches have been created, but users and developers continue to misunderstand requirements. Each approach is striving to improve requirements determination, but they miss the fundamental problem by not first understanding the factors that contribute to misunderstanding requirements.

Factors Related to Misunderstanding Requirements

A large body of literature exists related to the topic of requirements for information systems. Several researchers have studied success and risk factors for the development of an information system, and some have studied success and risk factors specifically for requirements determination. Communication between stakeholders, particularly between users and developers,

is frequently regarded as a key contributor to the success or failure of a project and communication models have been presented. However, within this body of literature, little research has been published that directly studies factors that influence the misunderstanding of requirements for an information system. Although this void in research exists, several factors are discussed in literature. For example, DeBellis and Haapala (1995) offered four reasons why users and developers misunderstand requirements:

1. Paper-based documents are typically used to capture and convey requirements, which are prone to ambiguity, omissions, and misinterpretations.
2. Users and developers do not share a common frame of reference and do not invest time building a common language for dialog.
3. Users may not understand the real requirements themselves until they have interacted with early versions of the information system.
4. Even if requirements are understood, the time between requirements determination and deploying the information system may allow the requirements to change.

Additional factors are presented below, grouped in five general categories, (a) Developer Bias, (b) User Bias, (c) Different Worlds, (d) Process, and (e) Communication. Many of the factors can be classified in more than one category. For example, conflicts between developers and users are reflective of their different worlds as well as communication issues. For simplicity, the factors are only discussed once even when they apply to more than one category.

Developer Bias Related Factors

Requirements determination approaches are often from one of two perspectives: developer or user. Literature on the topic is filled with models, frameworks, techniques, and step-by-step procedures written by developers for developers, while fewer are written to capture the users' perspective. When developers approach requirements, they apply modeling and analysis

techniques native to their domain, such as use cases, entity-relationship diagrams, object-oriented analysis, and formal specification languages (Jin et al., 2003). This is the developer perspective, and some have suggested that users be trained on these approaches before they participate in creating the requirements for a system (Eriksson & Penker, 1998). Users view requirements in different terms because they are not concerned with how the system works but what the system will do for them. They are unfamiliar with the modeling and analysis methods used by developers and generally do not care to learn about objects, actors, use cases, dependencies, and concepts common to these methods.

A useful framework that lends insight into this bias consists of four critical processes for the creation of information systems, shared by Elliot (2000): lifecycles, communication mechanisms (modeling), team working, and contracting. He points out that much of the difficulty between the two perspectives comes from the SDLCs that have been created from the developer's perspective as a way to evolve an idea into an operational and maintainable system. Many SDLCs treat interaction with users as a black box, accepting inputs from them (e.g., requirements) and producing outputs to them (e.g., deliverables, final system). Elliott shared that "...these paradigms are probably the root cause of [user] 'dissatisfaction' experiences ..." because they "...emphasize developer's activities" (2000). His examination of broad lifecycle categories, which included waterfall, evolutionary prototyping, and rapid application development (RAD), concluded with the assessment that they assume users have already performed analysis to determine their requirements for an information system. It is because of this assumption that developers expect to have requirements provided to them and that the requirements are complete and unchanging.

As pointed out by Stry, the risk of a developer-oriented bottom-up approach is that requirement determination will be constrained to the constructs of the computer aided design system, just as requirements can be constrained by the choice of technology used. As he said, "In

the beginning of the development software-engineering principles should not dominate and thus bias [requirements determination] and design towards technology" (2002, p. 438).

Joshi (1992) introduced the term *friction* between users and developers as a factor that limits interaction. He encourages developers to be tactful and maintain a cooperative relationship with users, which may not be their first reaction to users. As an example of friction, Guinan and Bostrom (1984) observed that developers tend to believe that their work is superior to that of users and that technical concerns are more important than organizational concerns. Yeh and Tsai, who investigate hostility between users and developers, believe friction is inevitable and that conflict should be expected between users and developers (2001).

User Bias Related Factors

Unfortunately, focusing requirements determination on users does not rectify the difference in perspectives between users and developers. Stry researched the gap between requirements determination and design in system development (2002). Design focuses on the specification of the software and how it should be constructed. Stry suggested that if requirements and design are approached from the user's perspective, referred to as user-centered design, then the problem remains as to how users should be involved and how to translate their involvement into the construction of the information system. He identified three factors for the gap between requirements and design (p. 428):

1. Transparent and traceable development procedures;
2. Common 'languages' for mutual understanding, thus coupling the engineering with the human-centered perspective on systems and their development process;
3. Successful collaboration between users performing work tasks and developers (based on transparent processes and common understanding).

These factors point to a difference in communication and understanding between users and developers, focusing on the need for a common framework between the two.

The problem is that common frameworks do not exist and would require one or both parties to learn something that is outside of their general experience base (Eriksson & Penker, 1998; Saiedian & Dale, 2000). For example, developers could become experts in ethnographic research techniques to improve their understanding of the users' context and work environment. Or, users could learn one of several modeling languages, such as the Unified Modeling Language (OMG, 2005) that express the design of an information system before it is created the way blueprints convey the design of a building before it is constructed. Such notions are not practical because most users and developers do not have time for these activities, or more importantly, the disposition to learn them.

Havelka (2003) sought to fill a void in the literature concerning the factors that affect the quality of requirements. The aim of his research was to answer the question: what are the factors that affect the quality of the requirements determination process as perceived by users. He used NGT with users of information systems who were involved in the requirements determination process to create a list of prioritized factors that contribute to quality requirements determination. The 33 factors were categorized into five areas: (a) technical, (b) organizational, (c) process, (d) management, and (e) personnel. Surprisingly absent from the factors identified by users is any notion of requirement creep, an issue often discussed by developers as the inability of users to completely define requirements before design and coding start. He also found that users preferred requirement elicitation techniques that were as unobtrusive as possible and kept their participation to a minimum—they preferred methods that did not interfere with their work. From this finding, users appear to be reluctant to work with developers, just as developers are reluctant to work with users, but possibly for different reasons.

Different Worlds Related Factors

Several publications refer to the general differences between users and developers. Users and developers view the world through different conceptual frameworks, mental models, and

perspectives (Bostrom & Thomas, 1983; Guinan & Bostrom, 1984; Kudikyala & Vaughn, 2005). Differences have also been found in the general personality types of the two groups (Bostrom & Kaiser, 1982).

Jin, Bell, Wilkie, and Leahy recognized the disparity of user and developer perspectives as a central problem to requirements determination:

We argue that the misunderstanding between the two communities of people: [developers] and [users], the ignorance of [developers] to the application domain and the vagueness of [users] to the goals of the target system are the biggest obstacles in determining target system requirements. (Jin et al., 2003, p. 55)

An interpretation of this difference is that developers must better understand the context and environment of users' work and users must have effective ways to convey their needs.

A study by Stary (2002) found different approaches to requirements and design based on the perspective of business-process specialists. This type of specialist, also called a business analyst, systems analyst, or simply analyst (Davis, 2005) is frequently employed to be responsible for the requirements of an information system. More developer-oriented specialists involved users only in requirements while more user-oriented specialists involved users in both requirements and design. Further, a difference in approaches was observed. A developer-oriented approach to requirements, referred to as a bottom-up approach, views requirements in light of what is technically feasible, while a user-oriented approach, or top-down, views requirements in the context of use and expects technology to adapt as much as possible.

Stary (2002) concludes that a gap exists between requirements and design because of the differences between business-process specialists typically responsible for requirements and the developers typically responsible for creating a system design from the requirements. The two groups belong to different worlds and can encounter difficulty transferring knowledge from requirements to design.

Conflicts between users and developers were studied by Yeh and Tsai (2001), citing previously researched reasons for conflicts to include: intergroup hostility, poor communications, negative perceptions of the other group, and frequency of interpersonal interaction. They examined conflicts that were substantive in nature (related to practices, policies, procedures, roles, and responsibilities) and those that were emotional in nature (related to personal perceptions and feelings). Due to differences noted between users and developers, conflicts are expected to be common during the development of an information system.

Other indications of the different worlds users and developers operate in stems from their goals and motivations (Cushing, 1990). Users' goals are associated with conducting a function and can be viewed as organizational imperatives. Developers have goals that are related to implementing a quality (in technical terms) system and can be viewed as technological imperatives. Further, personal differences between those involved in the development of an information system, such as differences in language, experience, ambition, knowledge, and interests contribute to discrepancies in conceptualizations of the system (Makrygiannis & Enquist, 1998).

Havelka and Lee (2002) identified differences in users' and developers' perspectives of the top-ten critical success factors for the development of an information system. The factors that users found important but were not ranked in the top-ten by developers include (a) financial resources, (b) IS communication, (c) IS technical skills, and (d) project team's authority. In contrast, developers emphasized four factors that were not in users' top-ten ranking: (a) planning, (b) stability of requirements, (c) user commitment to the project, and (d) users' understanding of needs. From this study, developers place more emphasis on users' requirements and participation in the project, while users are more concerned with developers' ability to communicate and their technical skills to accomplish the project. A simpler way to look at the differences is that users want developers to relate to them in their terms while developers want users to provide stable

and complete requirements that they need to do their job. Havelka and Lee's findings show a clear distinction between the perspectives of users and developers, increasing the credibility of the notion that they operate from different worlds.

Keil, Tiwana, and Bush (2002) used the Delphi technique to examine differences in how users and managers of development perceive risk in the development of information systems. Not only did users and managers of development prioritize risks differently, they selected different risks as being important. In this study, managers ranked "misunderstanding the requirements" as the second most important risk while users ranked it sixth out of 23 identified risk factors. Users identified "lack of effective development process/methodology" as the most important risk factor while managers did not include this risk factor in their analysis. This stark difference in perceptions between the users and managers raises the question of whether each group is thinking about the factors in the same way and possibly confusing content with process. One user said "Improperly structured development causes an improperly developed product which does not meet the requirement of either the end users or the developers" (Keil et al., 2002, p. 112). The user, and other users who made similar statements, may be focusing on the requirements determination process without putting it in terms of "misunderstanding user requirements." This apparent dichotomy highlights the different perspectives of users and developer managers and reinforces the need for further research. It is reasonable to believe that users understand user requirements and would not place an emphasis on this problem while developer managers understand the development process and would not place an emphasis on this problem.

These and several similar issues have been identified by researchers as problems with requirements determination that stem from differences between users and developers (Browne & Rogich, 2001; Darlington & Culley, 2002; Davis & Monroe, 1987; Saideian & Dale, 2000). A summary of some of the problems believed to result from these differences include:

1. Requirements are often incomplete or inconsistent.
2. They do not reflect users' real needs.
3. They are expensive to change after they have been agreed upon.
4. Communicating requirements clearly to all those involved in the development of the system is difficult.
5. Communication between users and developers is generally poor.
6. Differences exist in the knowledge and experiences between users and developers.
7. Technology-based solutions are used instead of user-driven solutions.

Process Related Factors

Davis (1982) provided three reasons why obtaining correct and complete requirements for information systems is difficult (p. 5):

1. The constraints on humans as information processors and problem solvers.
2. The variety and complexity of information requirements.
3. The complex patterns of interaction among users and [developers] in defining requirements.

These difficulties can contribute to low quality requirements when users are simply asked to specify their needs. Davis proposed a process for selecting one or more requirements determination methodologies to improve the ability of users and developers in creating requirements. His framework of the dependent variables and human characteristics that make requirements determination difficult and the mitigating effects of properly chosen methodologies is shown on Figure 7. The inputs of this framework are characteristics related to the system and its development and characteristics related to human limitations concerning requirements. These characteristics impact the ability of users and analysts (or developers if they are acting as analysts) to specify, elicit, and evaluate requirements. Davis then suggests that users and analysts

can better specify, elicit, and evaluate requirements if an appropriate requirements determination methodology is used.

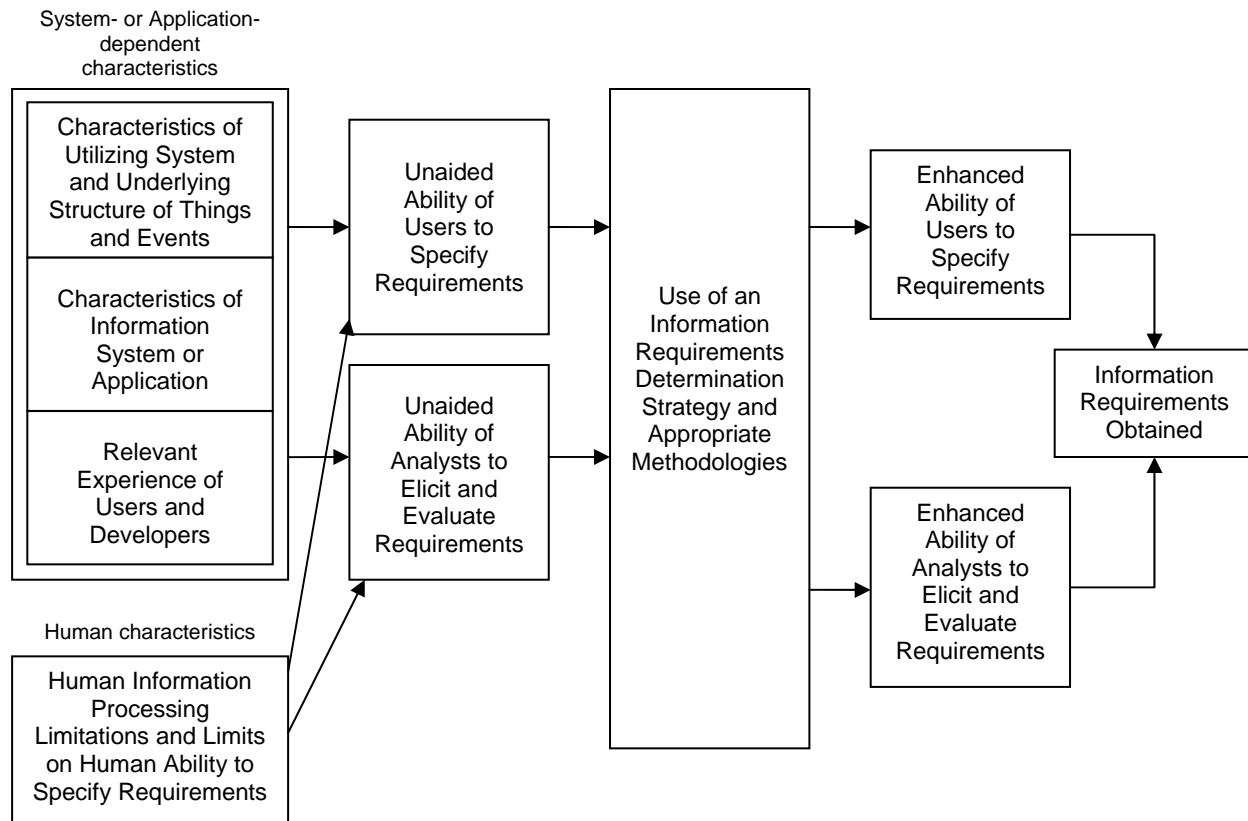


Figure 7. Davis' conceptual framework of requirements determination involving users and developers.

From "Strategies for Information Requirements Determination," by G. B. Davis, 1982, *IBM Systems Journal*, 21(1), p. 11. Copyright 1982 by IBM. Adapted with permission of the publisher.

Fisher (1999) reported on the results of a survey and a case study researching the value of technical communicators in the development of information systems. Technical communicators are typically involved in the creation of documentation for users, such as a "user guide" that describes how to operate the information system. On system development projects where technical communicators were more involved, working with both users and developers as a user advocate, users rated the success of the systems higher.

A contributing factor is the perspective viewed as being in control of the process. If developers drive the creation of a system, users may feel threatened and resist the change. If users drive, developers may not give the project the importance it warrants (Havelka, 2003).

Another factor is focusing on solutions before really understanding the problem and missing opportunities for more valuable solutions (NIST, 2005). Asking users to communicate a complex problem to developers is a difficult task by itself; asking them to communicate a poorly understood problem is even more challenging (Mrenak, 1990). DeBellis and Haapala described the situation by saying: "Introducing new technology affects user needs in unanticipated ways, and because those needs are often misunderstood to begin with, opportunities are missed" (1995, p. 35). As the understanding of the problem evolves over time, users' concept of it and possible solutions also evolve. Consequently, asking users to define requirements, sign a contract that they are complete and accurate, and then expect the requirements to not change as users' understanding improves, is ensuring development problems.

Developers may interpret this natural evolution of problem understanding as a weakness of users. A common belief held by developers is that users do not know what they want until they see it. The issue is not that users do not know, but that a characteristic of human behavior is the difficulty people have articulating what they want before they see it or are exposed to scenarios in the context of the problem (Lamsweerde, 2000; Yeh & Tsai, 2001). Requirements are difficult to determine because users are unsure what is possible, have trouble describing the problem, or do not sufficiently understand the problem (Kazmierczak et al., 2000). Users are much better at reacting to a design once they see it in their work environment (Nielsen, 1993). Bostrom and Thomas (1983) observed that the issue is not really that users do not know what they want, but that they have trouble articulating what they want. When developers are confronted with missing information from users, they may fill the void based on their own

understanding of the problem, mental model, and experience, instead of asking users for more information.

Another way to look at the situation is to view the information system as a catalyst for change, which causes users to have difficulty specifying requirements for something to which they are unaccustomed. Consequently, users are often uncomfortable responding to requests to describe requirements for a system and need a means to be prompted for input, such as using prototypes to create discussion (Saiedian & Dale, 2000). This difficulty with communicating the needs for a system is why requirements determination is important and must begin with an understanding of goals, objectives, environment, and the like before specific requirements are discussed.

Communication Related Factors

Lindqvist (2003) suggested that those involved in information systems development must learn to communicate effectively with each other, and that "What is important is to reduce miscommunication and increase efficiency in the communication process, between the IS developer and the IS user/client" (2003, Introduction section). He proposed a model consisting of three layers that impact effective communications: (a) culture, which includes both personal (nationality, religion, politics) and business culture and is dependent on the environment; (b) context, which encompasses the ideas present during the information system development and is dependent on the situation; and (c) concept, which involves the ideas surfaced during development and is dependent on the application. Although Lindqvist recognized the need for effective communications, Gallivan and Keil provide insight into the lack of knowledge about user-developer communications: "Much of the prior research on user participation assumes that user-developer communication will ensure that the resulting system will be designed to meet users' needs and will be accepted by them. The nature and quality of the communication between

users and developers, however, remains an understudied aspect of user participation" (2003, p. 37).

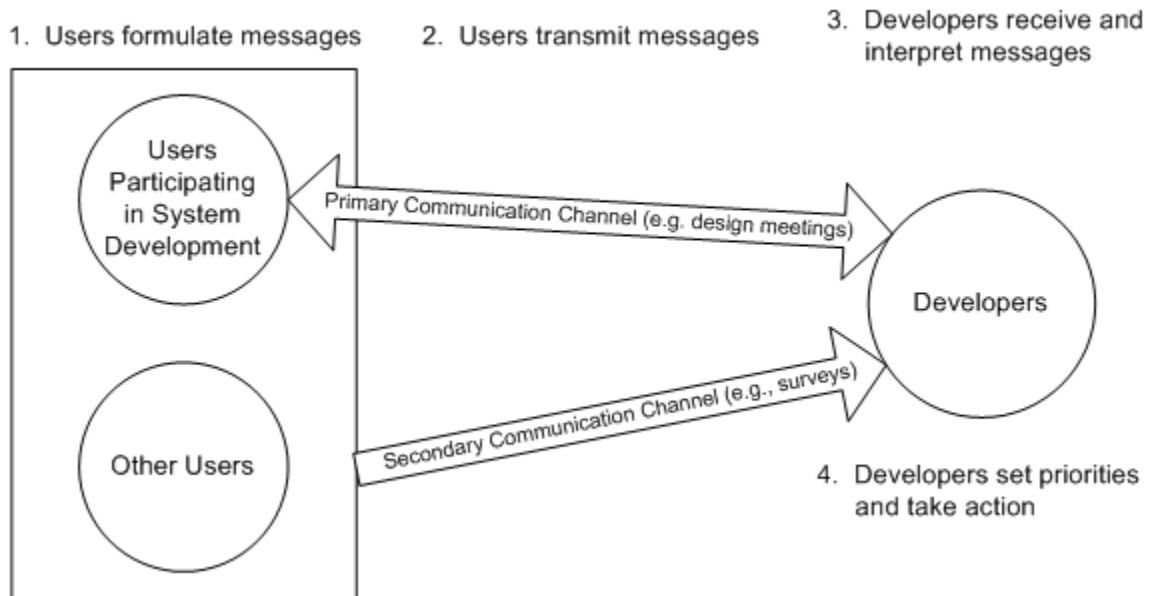


Figure 8. User and developer four-stage communication model.

From "The User-Developer Communication Process," by M. J. Gallivan and M. Keil, 2003, *Information Systems Journal*, 13(1), p. 43. Copyright 2003 by Blackwell Publishing Limited. Adapted with permission of the publisher.

The goal of Gallivan's and Keil's (2003) research was to introduce a process model to explain why communications are effective or ineffective. They observed a common theme running through information systems literature, which was that user-developer interpersonal communications are most important in creating a successful system. Based on their fieldwork, they developed a communication model, shown on Figure 8. The four activities in the model include: (1) users begin thinking about the problem to be solved and what they will tell developers, (2) users provide the messages to developers over one or more communication channels, (3) developers receive the messages and interpret them, and (4) developers set priorities and take actions based not on the users' messages but on the developers interpretation of the messages. Consequently, misunderstanding or misinterpretation could start at any stage of

their model, from message formulation, to message transmission, to message interpretation, and finally to taking action on the message.

Pan and Yanp (1999) studied the communication patterns of users and developers involved in the creation of information systems. They highlight the problem of understanding requirements during the beginning of a project (the R&D stage in their terms) and the need for effective information sharing:

Developers face the highest requirement uncertainty at R&D stage [3, 26]. Such uncertainty can only be reduced through frequent information processing between users and developers. In addition to requirement uncertainty, developers face the greatest equivocality of information at R&D stage too because of the different terminology and different focuses between users and developers. Thus, the richest information processing is required at R&D stage in order to avoid problems caused by equivocality. (p. 2-3)

Enquist and Makrygiannis studied the frequency and impact of misunderstandings between stakeholders, focusing on those involved in creating requirements as well as developers (1998). They found that misunderstandings occur frequently and have negative consequences on the development of an information system. Although concerned with the entire software development process and not only requirements, the two most common causes of misunderstandings were found to be "Unclear/ incompletely expressed information" and "Differences in concepts and frame of reference" between the stakeholders (Makrygiannis & Enquist, 1998).

Another factor is that although most requirements determination approaches are steeped in communications, developers are no more likely to be communication experts than anyone else (Mrenak, 1990). Developers may have the responsibility for gathering and understanding the requirements but lack specialized communication skills for the task. Moore (2003) found that the

communication difficulties between users and developers is a root cause of requirement errors, saying "End-users and requirements analysts essentially speak two different languages" (p. 1). Carlsson (2000) echoes the communication difference, pointing out that users and developers talk about the same concepts in different terms and their perception of each other may lead to misunderstandings. This contributes to the well recognized gap between users and developers.

The communication medium and its associated amount of *bandwidth*, or ability to effectively enable the exchange of information, is another contributing factor. A low-bandwidth medium such as the telephone may hinder understanding while a higher bandwidth medium, such as a face-to-face meeting using a white board to capture ideas, may improve understanding (Carlsson, 2000). Keil and Carmel (1995) examined using a variety of communication mediums, called user-developer links, which are "ways in which [users] and developers exchange information during the development process" (p. 36). A few examples of links include facilitated workshops, surveys, user interface prototypes, and interviews. They found that successful projects generally employ a greater number of links than less successful projects do, thus increasing communication bandwidth between customers and developers and improving understanding. Also, the use of customer surrogates, intermediaries that help to define requirements on behalf of customers, was found to be associated with less successful projects.

Rapport also contributes to communication difficulties and the pervasiveness of requirements misunderstandings. The more "in sync" users and developers are with each other, the better their communication may be (Guinan & Bostrom, 1984). Negative feedback may not be shared by users because some topics are considered undiscussable or past experience tells users that the feedback will be ignored (Gallivan & Keil, 2003). Further, developers tend to be defensive about their work, which can hinder open communication with users and their ability to discuss requirements (Klockner et al., 1999). Worse, developers may talk-down to users,

believing that users are "too unsophisticated to understand the glories of a real computer system" (Borenstein, 1991, p. 70).

Finally, two communication frameworks, developed specifically for eliciting and understanding software requirements, add insights into users and developers misunderstanding requirements. Al-Rawas and Easterbrook (1996) found three major barriers to communication, while Coughlan et al.'s (2003) study identified four barriers to communication. The synthesis of the two frameworks creates four problems that can hinder communication during requirements elicitation:

Lack of probing, feedback, and clarification. Needs are provided by users, with insufficient feedback from developers to clarify their understanding. This results in a failure to create a shared understanding of the requirements and the big picture. It can also mean that the wrong problem is being addressed or that it is approached from a suboptimal position. For example, users may be asking developers to make an existing process faster/better/cheaper when a new innovative process is needed.

A gap between user and developer perspectives and problem-solving approaches. Users prefer to discuss the problem in business terms, while developers prefer to discuss it in modeling terms. Users and developers operate in two different worlds, use different terminology, have different motivations, and do not appreciate each other.

Incongruence between the problem and the stakeholders. Al-Rawas and Easterbrook (1996) as well as Coughlan et al. (2003) emphasize the importance of having the correct stakeholders involved in requirements elicitation and suggest that they must be available for negotiations or when problems need to be solved. Also, organizational culture and politics can hamper involvement or create biased perspectives.

Trained, experienced, and independent facilitator. A project manager or business analyst may have the role of facilitator during requirements elicitation. Without proper training and

experience with elicitation facilitation, the previously discussed communication problems may continue unnoticed or unresolved. In addition, if the facilitator has ties to a stakeholder group, such as users or developers, he or she may be considered biased, which causes distrust.

Summary

Table 4 contains a summary of the above factors found in literature that may contribute to misunderstanding requirements. The five categories of factors are coded as F1, F2, ..., F5 and the factors that appear in each category are coded as a .1, .2, ..., .n.

Table 4. *Factors Found in Literature*

Category	Factor	Source
F1. Developer Bias	F1.1. Developers view requirements in terms of modeling and analysis techniques.	Jin et al. (2003)
	F1.2. Users are unfamiliar with modeling and analysis techniques used by developers.	Eriksson and Penker (1998)
	F1.3. Software development methodologies used to create information systems assume users have already analyzed the requirements for the system.	Elliott (2000)
	F1.4. Software engineering principles dominate requirements determination and result in technology-centric designs instead of user-centric designs.	Stary (2002)
	F1.5. Developers believe their work is more important than that of users.	Guinan and Bostrom (1984)
F2. User Bias	F2.1. Users desire transparent development procedures, a common language to create mutual understanding, and successful collaboration.	Stary (2002)
	F2.2. A common framework is missing for users to effectively communicate with developers and neither party desires to learn the business of the other to aid communication.	Eriksson and Penker (1998), Saideian and Dale (2000)
	F2.3. Users prefer requirements determination methods that do not interfere with their work.	Havelka (2003)

Table 4. *Factors Found in Literature, Continued*

Category	Factor	Source
F3. Different Worlds	F3.1. Users and developers view the world through different conceptual frameworks, mental models, and perspectives.	Bostrom and Thomas (1983), Guinan and Bostrom (1984), Kudikyala and Vaughn (2005)
	F3.2. Developers lack understanding of the problem domain while users are vague about their needs.	Jin et al., 2003
	F3.3. Users and developers tend to be associated with different personality types.	Bostrom and Kaiser (1982)
	F3.4. A gap exists between users, who have a business-process perspective, and developers, who have a technical perspective, leading to different requirement determination processes.	Sary (2002)
	F3.5. Conflict naturally exists between users and developers, in part because of negative perceptions one group has of the other.	Yeh and Tsai (2001)
	F3.6. Users and developers have different goals and motivations.	Cushing (1990)
	F3.7. Users and developers exhibit differences in language, experience, ambition, knowledge, and interest.	Makrygiannis and Enquist (1998)
	F3.8. Users and developers select different factors as important to the success of a project.	Havelka and Lee (2002)
	F3.9. Users and development managers select different risks as important to the development of an information system.	Keil, Tiwana, and Bush (2002)
F4. Process	F4.1. Prematurely adopting a solution before the problem is well understood.	NIST (2005)
	F4.2. Attempting to explain a poorly understood problem.	Mrenak (1990)
	F4.3. The introduction of an information system may change the way the problem concept is understood.	DeBellis and Haapala (1995)
	F4.4. Difficulties articulating what is needed before seeing what is possible in the proper context.	Lamswerde (2000), Kazmierczak et al. (2000), Saiedian and Dale (2000)
	F4.5. When developers are faced with missing requirements, they tend to create them based on their understanding of the problem.	Bostrom and Thomas (1983)

Table 4. *Factors Found in Literature, Continued*

Category	Factor	Source
	F4.6. Users may resist if developers are driving the project while developers may regard the project as unimportant if users are driving the project.	Havelka (2003)
	F4.7. Complex patterns of interaction exists between users and developers.	Davis (1982)
	F4.8. Technical communicators working as user advocates improve system success.	Fisher (1999)
F5. Communication	F5.1. Although requirements determination techniques require communication proficiencies, developers who are responsible for these techniques are not likely to be communication experts.	Mrenak (1990)
	F5.2. Developers and users must learn to communicate more efficiently, incorporating culture, context, and concept in their communications.	Lindqvist (2003)
	F5.3. User-developer interpersonal communications are the most important factor in the success of an information system.	Gallivan and Keil (2003)
	F5.4. No single requirements determination technique solves all of the problems.	Byrd, Cossick, and Zmud (1992)
	F5.5. Users and developers need to frequently share and process information during requirements determination.	Pan and Yanp (1999)
	F5.6. Effective communication is more important than specific requirements determination techniques.	Bostrom and Thomas (1983)
	F5.7. Misunderstandings are most commonly the result of incompletely expressed information and differences in frame of reference between users and developers.	Makrygiannis and Enquist (1998)
	F5.8. Users and developers speak two different languages, using the same terms for different concepts or different terms for the same concepts.	Moore (2003), Carlsson (2000)
	F5.9. The chosen communication medium can hinder effective communication.	Carlsson (2000)
	F5.10. User-developer rapport impacts communication effectiveness.	Guinan and Bostrom (1984)
	F5.11. Negative feedback may not be shared.	Gallivan and Keil (2003)

Table 4. *Factors Found in Literature, Continued*

Category	Factor	Source
	F5.12. Developers insufficiently probe for clarification and ask for feedback from users.	Al-Rawas and Easterbrook (1996), Coughlan et al. (2003)
	F5.13. A properly trained facilitator can improve communication effectiveness.	Al-Rawas and Easterbrook (1996), Coughlan et al. (2003)
	F5.14. The use of customer surrogates limits project success while increasing the number of elicitation techniques improves project success.	Keil and Carmel (1995)
	F5.15. Developers tend to hinder open communications because they are defensive about their work.	Klockner, Pankoke-Babatz and Prinz (1999)

Note. Source indicates the citation used in this study for the factor, but does not imply that other cited authors have not also discussed the same factor.

Assessment of Current Literature

There is wide agreement in literature for the difficulty of creating requirements that are understood by both users and developers. Although the development of information systems has been examined from multiple dimensions, little is known about why users and developers misunderstand requirements, and studies have not prioritized the factors involved. Much of the research in requirements determination has focused on methodologies and techniques for improving the requirements generation and gathering process. Havelka provided a review of the literature, finding support for techniques including: user prompting, a prioritization process for collecting requirements from distributed stakeholders, socio-technical approaches, an approach based on grounded theory from qualitative research, cognitive interviews, object-oriented approaches, structured interviews, use of focus groups, the application of imagery for requirements creation, facilitated teams, critical success factors, organizational classification, and many others (see Havelka, 2003 for details).

Therefore, as depicted on Figure 9, from previous research one knows (a) the risks and success factors involved in the creation of information systems along with a detailed understanding of methodologies for their creation, (b) the risk and success factors related to requirements determination and several techniques for determining requirements, (c) the need for

more effective requirements determination approaches that improve the creation of clearly understood requirements, and (d) the lack of basic knowledge for why users and developers misunderstand requirements.

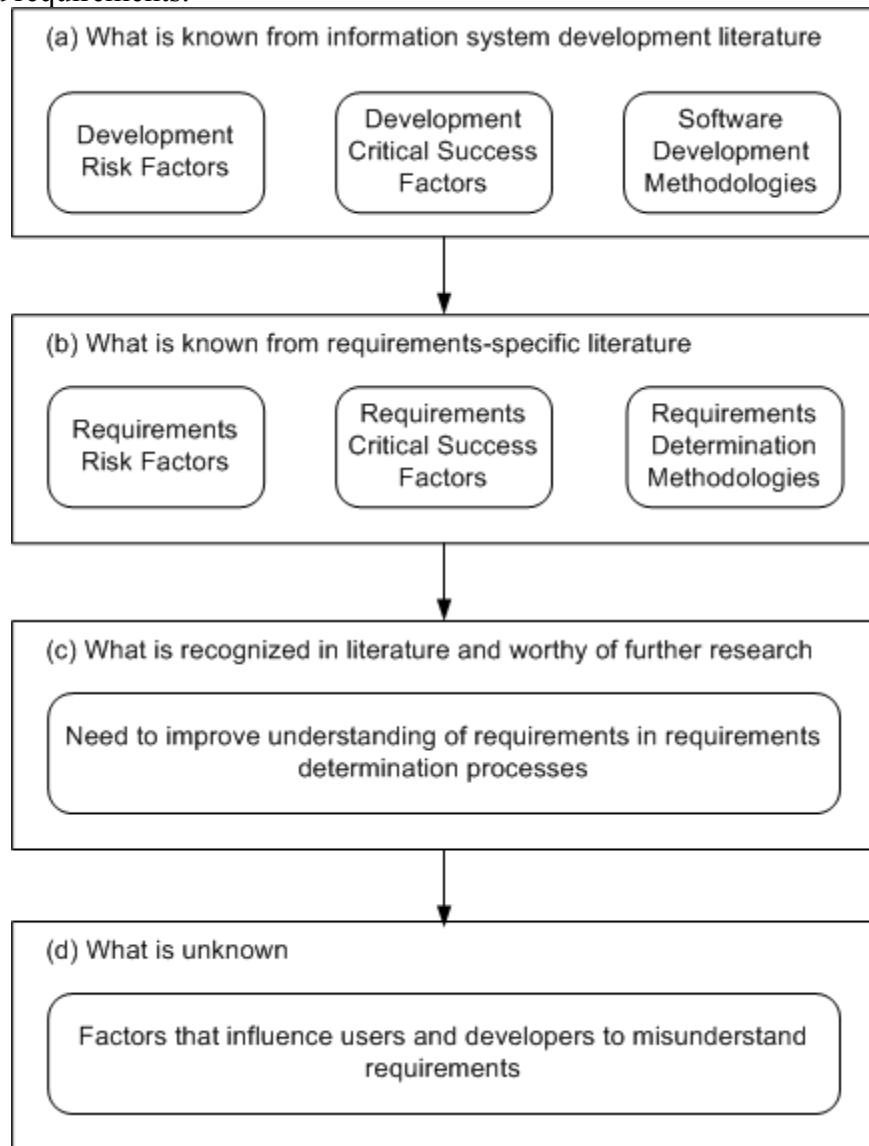


Figure 9. Assessment of literature and connections between literature and the research presented in this study.

With knowledge of the underlying variables—factors that contribute to users and developers misunderstanding requirements—it may be possible to develop more effective requirement determination approaches and improve users' and developers' understanding of requirements. The present study provides the first steps in creating this knowledge by identifying and prioritizing the factors users' and developers' perceive as influencing misunderstanding requirements for information systems.

Enhanced Conceptual Framework

The conceptual framework previously presented on Figure 1 has been enhanced based on the findings of the literature review. The enhanced conceptual framework is shown on Figure 10.

From existing literature, factors have been discussed that may influence users and developers misunderstanding requirements for an information system. These factors are referenced in the figure by their corresponding categories: F1 Developer Bias, F2 User Bias, F3 Different Worlds, F4 Process, and F5 Communication. The primary areas researched in the present study are those that are highlighted in gray. UPFs and DPFs were identified through investigation since the literature shows that such factors have not been previously studied, only discussed as possible contributors to the problem of misunderstood requirements. UPFs and DPFs were prioritized to understand the importance or weight users and developers give to the identified factors. Finally, the prioritized UPFs and DPFs, along with their similarities and differences, provide insights into why users have one view of requirements and developers have another view of requirements for an information system.

Factors Found in Literature

F1: Developer Bias, F2: User Bias, F3: Different Worlds, F4: Process, F5: Communications

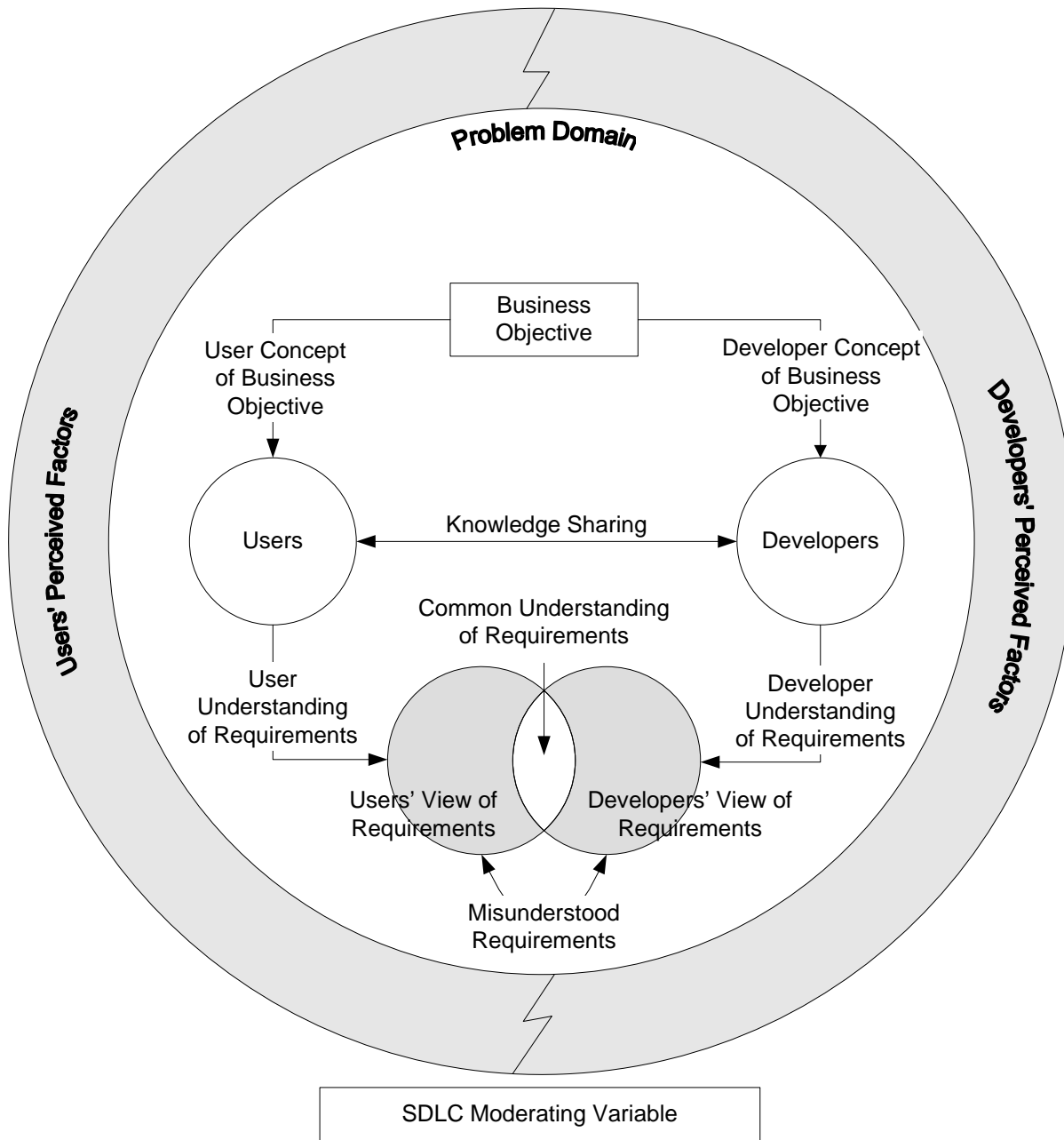


Figure 10. Enhanced conceptual framework for studying what influences users and developers to misunderstand requirements.

CHAPTER 3. METHODOLOGY

Review of the Purpose of the Study

Information systems often fail to satisfy users' expectations because developers misunderstand users' requirements. In fact, Guinan and Bostrom (1984) state that "the basic assumption is that a lack of understanding exists between the user and the developer which blocks effective communication" (1984, p. 5). Ineffective communication is only one of many reasons why developers misunderstand users' requirements. Instead of continuing to rely on assumptions about why requirements are misunderstood, this study sought to qualitatively identify the factors that influence misunderstandings about requirements and prioritize the importance of each factor from the perspective of users and developers, noting differences between the two perspectives. Consequently, this theory-building study may improve requirement determination processes in the future.

General Research Philosophy

The present study was guided by a philosophy of pragmatism. According to Miles and Huberman (1994), "any method that works—that will produce clear, verifiable, credible meanings from a set of qualitative data—is grist for our mill, regardless of its antecedents" (p. 3). This pragmatic approach is also discussed by Robson (2002), who acknowledges a basic incompatibility between the historical extremes of positivists and constructivists. He characterizes the pragmatic perspective as "whatever philosophical or methodological approach [that] works best for a particular research problem at issue" (p. 43). Regardless of the specific method, the pragmatist is concerned with defining the scope and intent of a study, ensuring

reliability and validity, and using an adequate research structure. The pragmatist uses a scientific method that includes systematic procedures, a skeptical interpretation of results, and an ethical code of conduct. Consequently, the research approach described in this chapter was chosen to answer the research questions without regard to epistemological preferences.

Research Questions

The research question hierarchy described by Cooper and Schindler (2003) was used to construct the questions for this study. Their hierarchy starts with an observed business dilemma that, in turn, stimulates a management question. The management question spawns research questions. The research method is chosen to answer the research questions, which answers the management question. Table 5 shows the research question hierarchy.

Table 5. Research Question Hierarchy

	Question or Statement
Business Dilemma	Information systems frequently fail to meet users' expectations, resulting in several possible negative outcomes, including jeopardizing the success of a system, increasing the cost and time of a project, and risking the cancellation of a project.
Management Question	How can management better understand why information systems fail to meet users' expectations and eliminate the misunderstandings that result in informational systems that fail to meet users' expectations?
Research Questions	<ol style="list-style-type: none"> 1. Which factors do users and developers believe cause misunderstandings about the requirements for information systems? <ol style="list-style-type: none"> 1.1. What factors do users think cause misunderstandings? 1.2. What factors do developers think cause misunderstandings? 2. Which factors do users and developers believe have the most impact on misunderstandings? <ol style="list-style-type: none"> 2.1. How do users prioritize the factors that cause misunderstandings? 2.2. How do developers prioritize the factors that cause misunderstandings? 3. What is the difference between users' and developers' perceptions of these factors? <ol style="list-style-type: none"> 3.1. How do the factors identified by users compare to those identified by developers? 3.2. Which factors identified by users were not identified by developers? 3.3. Which factors identified by developers were not identified by users? 3.4. Why is there a difference between users and developers' perceptions of these factors?

Research Approach

The study discussed in this dissertation was designed to uncover the factors (i.e., independent variables) that influence users' and developers' misunderstandings about the requirements (i.e., dependent variable) for an information system. Therefore, it is an exploratory theory-building study as opposed to confirmatory theory-testing study (Creswell, 2003). To answer the questions presented in Table 5, a two-phase field study was conducted to discover, describe, and prioritize factors that influence users' and developers' misunderstandings about the requirements for an information system. For each phase, participating organizations were contacted in person or by phone, email, or postal mail. The letter shown in Appendix A, which asks for their participation and describes the research and its importance, was used when contact was made by email or postal mail. The content of this letter was shared if contact was made in person or by phone. Follow-up contacts by phone and email were used to obtain participants' commitment to the study.

Phase I of Research

A purposeful sample comprised of small groups from three organizations engaged in the development of information systems was used in the present study. Pairs of small groups were formed from users involved in determining the requirements of an information system and developers from the same organization, resulting in three pairs of groups. These six small groups participated in focus groups.

The first phase of the present study identified factors using the Nominal Group Technique (NGT) with the focus groups. NGT has been applied in previous studies to identify factors related to information systems (Havelka & Lee, 2002), and Havelka, Sutton, and Arnold

(1998b) found that this technique is appropriate for this type of study. NGT revealed the critical factors that produce misunderstandings between users and developers of information systems. A thematic analysis was performed to remove duplicates and consolidate similar factors, which produced two distinct lists of factors: a list of factors identified by users and a list of factors identified by developers.

Phase II of Research

The second phase of the present study was designed to uncover the importance of each factor. Two survey instruments were created to prioritize the factors: (a) a survey that contained user-generated factors, which was only completed by the users who participated in the NGT groups; and (b) a survey that contained developer-generated factors, which was only completed by the developers who participated in the NGT groups. Participants were invited to complete the appropriate survey by email, using the invitation template shown in Appendix B. Each survey focused on the most critical factors identified by users and developers. Analytical Hierarchy Process (AHP), a robust prioritization technique, was used to determine the subjective importance of each factor. The results from each participant were aggregated to create the absolute weightings of factors for users and developers.

Appropriateness of Approach

Exploratory Mixed Method

A quantitative study that focused on factors that cause misunderstandings between users and developers, such as communication problems, personality differences, and conflicting frameworks, was initially considered for the present study, but there are no existing theories

about these factors. Therefore, a theory-testing approach was not as appropriate as an exploratory theory-building approach.

The aim of the present study was to determine the most important factors that cause misunderstandings between users and developers of information systems. Two steps were required to achieve this objective: first, the factors had to be identified, which was accomplished using a qualitative approach; and second, the importance of each factor had to be determined, which was accomplished using a quantitative approach. Therefore, the exploratory approach, which starts with a qualitative investigation that is followed by a quantitative investigation, was chosen for the present study (Creswell, 2003).

Nominal Group Technique

In 1968, Andre Delbecq and Andrew Van de Yen used results from social-psychological research that examined decision making and problem solving to develop the Nominal Group Technique. It is an effective technique for generating ideas, or in the case of the present study, factors, from small groups of people, and it is more effective than other group processes (Van de Yen & Delbecq, 1974). For example, Havelka, Sutton, and Arnold (1998) used NGT to examine the factors users and developers think influence the quality of information systems. Later, Havelka (2003) used NGT with users of information systems who were involved in a requirements determination process to create a list of prioritized factors that contribute to quality requirements determination.

NGT is considered the gold standard for techniques used to uncover factors from a group of participants (Valacich, Dennis, & Connolly, 1994). Duggan and Thachenkary (2003) identify several advantages of NGT:

1. It involves enforced participation, which prevents less involved or motivated group members from opting out of the process and increases the number of identified factors.
2. It is easy to use, which improves the repeatability of a study.
3. It has well-structured procedures that help reduce domination by more powerful participants.
4. It has a brainstorming nature that improves the generation and communication of factors.

The face-to-face nature of NGT provides opportunities for analyzing qualitative data shared by participants, which makes it a better choice than other tools, such as the Delphi technique. There is also empirical evidence that NGT generates more factors than Delphi, and participants are more satisfied with the NGT approach than with the Delphi approach, which may lead to improved participation (Van de Yen & Delbecq, 1974). In addition, Keil, Tiwana, and Bush (2002) encountered difficulties with the Delphi technique when they conducted a study comparing the risk factors associated with the development of information systems from the perspective of software managers and users. In addition, they also found that participants were not satisfied with the Delphi technique.

NGT consists of four steps: (a) introduction, (b) generation of factors, (c) listing of factors, and (d) evaluation of factors. These steps are summarized in Table 6.

Table 6. Summary of Nominal Group Technique Activities

Step	Description
1. Introduction	Participants are asked to introduce themselves and give a short description of their background. The activities to be performed by the focus group during the session are presented. This is followed by a short presentation about the scope of the problem being considered (i.e., misunderstanding requirements) and a description of NGT.
2. Generation of factors	Each participant is asked to silently and individually generate a list of factors they believe influence misunderstandings about requirements. The facilitator also participates, listing factors discussed in the literature and previous NGT sessions.
3. Listing of factors	The factors generated in Step 2 are listed in round-robin fashion on a flip chart and discussed and clarified by participants.
4. Evaluation of factors	After all factors are listed, the participants are asked to select 10 factors most important to them.

Note. From “Antecedents to Systems Development: Beliefs of Information Systems Specialists and Users,” by D. Havelka, 1994, Doctoral Dissertation, Texas Tech University. Copyright 1994 by D. Havelka. Adapted with permission of the author.

Analytical Hierarchy Process

Several types of problems are associated with ranking the influence of multiple factors on one or more dependent variables. For example, someone considering which car to buy may want to rank their alternatives based on price, performance, suitability, and general feel. When the factors consist of value judgments (i.e., subjective, qualitative information), a method is required that produces quantitative ranking and weighting. The process of determining this quantitative information has been described as selection, ranking, prioritizing, weighting, and so forth (Vaidya & Kumar, 2004). Although several methods can be used to rank factors from most important to least important, few methods provide a process for weighting the importance of the factors against a standardized scale. One that does use a standardized scale is the Analytical Hierarchy Process (AHP), an approach frequently used for ranking and weighting factors.

AHP was created in the 1970s by Wharton Business School Professor Dr. Thomas Saaty as a method to reduce complex decision making to a series of pairwise comparisons

(ExpertChoice, 2005). Saaty's intent was to create an analytical tool that reflects the normal cognitive approach to making decisions. AHP has been used for a wide variety of problems, which are summarized by Vaidya and Kumar (2004). Vaidya and Kumar examined 150 articles that describe practical applications of AHP and found that it is used for problems such as selection, evaluation, benefit-cost analysis, resource allocations, planning and development, priority and ranking, decision making, and forecasting. They also found strong support for using AHP in social science research. Steuer and Na (2003) conducted a similar review of AHP and examined its use in financial contexts. In addition, AHP has been used to prioritize the importance of factors that influence the quality of information systems. For example, Johansson, Wesslén, Bratthall, and Höst (2001) used an AHP variant called Incomplete Pairwise Comparison to analyze quality requirements in the development of software, and Badri (2001) used AHP to determine the priority of several factors related to a dependent variable.

Saaty (1999) describes AHP as a method that uses pairwise comparisons to determine priority from value judgments. There are many advantages to the pairwise technique: for example, participants can focus on making a single choice between two items at a time, other factors or influences are almost completely eliminated, and consistency is improved and can be verified (Schniederjans & Wilson, 1991). AHP has four basic steps for ranking a set of criteria:

1. Identify the criteria that impact the dependent variable.
2. For n criteria, conduct $n(n-1)/2$ pairwise comparisons and create a matrix of weighting coefficients.
3. Solve the linear equations to find the maximum eigenvalue, the corresponding consistency ratio, and the normalized weights for each criterion.

4. Validate weightings using the consistency ratio, and repeat all steps if inconsistency is suspected. (Vaidya & Kumar, 2004)

AHP uses a 9-point interval scale that is based on the human brain’s cognitive capacity to process information (Miller, 1956; Saaty, 1977). Table 7 describes the AHP 9-point scale.

Table 7. *AHP 9-Point Scale*

Intensity of Importance	Definition	Explanation
1	Equal importance	Two activities contribute equally to the objective
3	Moderate importance	Experience and judgment slightly favor one activity
5	Strong importance	Experience and judgment strongly favor one activity
7	Very strong or demonstrated importance	An activity is favored very strongly over another; its dominance is demonstrated in practice
9	Extreme importance	One activity is favored over all other activities
2, 4, 6, 8	For compromises	Sometimes it is necessary to interpolate a compromise judgment

Note. From “Decision Making for Leaders: The Analytic Hierarchy Process for Decisions in a Complex World,” by T. L. Saaty, 1999, p. 73. Copyright 1999 by RWS Publications. Adapted with permission publisher.

In AHP, responses from each participant are tabulated in a pairwise matrix, with the reciprocal of each evaluated pair assigned the reciprocal of the preference. A hypothetical example comparing three factors that influence the purchase of a car is shown in Table 8, and the results show a moderate preference for performance and price compared to safety. The geometric mean, which is the n^{th} root of the product of n criteria, is used to create a consensus value for each preference from all participants (Koksal & Egitman, 1998). Instead of the arithmetic mean, the geometric mean is used because it is less affected by extreme outsiders (Saaty, 1999) and better represents the aggregation of value judgments (E. Forman & Peniwati, 1998).

Table 8. *Example of Pairwise Weightings*

	Safety	Performance	Price
Safety	1	1/3	1/3
Performance	3	1	1
Price	3	1	1

After a consensus value is established, the geometric mean is computed for each row in order to produce an average weight for the importance of each factor. This weight is divided by the sum of all geometric means to produce a weight for each factor normalized between 1 and 0 (Baker et al., 2001). The normalized weights are called a priority vector, which can be used to calculate the consistency of judgments. This is made possible because a defined interval scale is used for ranking preferences. If the judgments are perfectly consistent, then any path through the matrix will equal the reciprocal of the opposite path. For any matrix containing judgments, the corresponding perfectly consistent matrix can be calculated by finding its eigenvalue (Saaty, 2003). Saaty and Foreman (ExpertChoice, 2005) created a software application called ExpertChoice that calculates the matrix that maximizes consistency. The result of these calculations is a consistency ratio (CR), which should generally be less than 5% for a three-by-three matrix, 9% for a four-by-four matrix, and 10% for larger matrices (Saaty, 1999). Larger CR values suggest an inconsistency in the preferences captured by pair-wise comparisons.

There are several benefits associated with using AHP to prioritize factors:

1. The use of pairwise comparisons makes AHP one of the simplest techniques for participants to use, and it is the most easily mastered weighting method (Dodgson, Spackman, Pearman, & Phillips, 2000).
2. It virtually eliminates participant bias in the weighting process by focusing on two criteria at a time (Badri, 2001).

3. Pairwise comparisons provide redundant information that can be used to check the consistency of the weighting and indicate if re-weighting should be considered (Saaty, 1999).

There are also some problems associated with AHP:

1. Ease of use decreases as the number of factors increases (Brugha, 2004). For example, 10 comparisons are required to weight five factors, 45 comparisons are required for 10 factors, 105 comparisons are required for 15 factors, and 190 comparisons are required for 20 factors. In the present study, the number of factors was limited to 10, which resulted in a manageable 45 comparisons.
2. AHP is less suitable when criteria are added after an initial weighting. When this occurs, the most defensible course of action is to re-weight all factors. In the present study, the factors were fully known before participants were provided the AHP survey.
3. Rank reversal is often cited as a problem with AHP (Leung & Cao, 2001). Rank reversal occurs when a new factor is introduced and reverses existing factors, even though the new factor is independent of existing factors (Forman, 1993). Although rank reversal is considered a problem or a consequence of any valid preference rating system (Foreman & Gass, 2001), it does not occur when a known, unchanging number of factors are rated, which was the case in the present study.

Sampling Plan

Research (Havelka, Sutton, & Arnold, 2001; Keil et al., 2002) has shown that users and developers have different views about which factors are important in the development of information systems. Therefore, it is reasonable to believe that users and developers would have different views about the factors that cause misunderstandings about the requirements for information systems. In order to capture these differences, the present study was designed to include a sufficient number of participants who are users and developers.

The number of participants in the present study was based on the Voice of the Customer research conducted by Griffin and Hauser (1993), which determined that 20 to 30 participants can express 90% or more of the requirements for a new product. The creation of requirements for a new product was considered analogous to creating factors that influence a concept (e.g., factors involved in misunderstanding requirements). In addition, Bock and Sergeant (2002) state that small sample sizes of 30 or less participants are sufficient when the purpose is to qualitatively generate a list of factors. Van de Yen and Delbecq (1974) suggest the use of groups with approximately seven participants.

As a result, the present study used three NGT groups of users and developers, with approximately seven participants each, which was expected to capture at least 90% of the factors participants believe contribute to misunderstandings about the requirements for information systems. This resulted in three user-developer pairs of NGT groups.

A purposeful dimensional sample was used in the present study. Merriam (1998) states that “purposeful sampling is based on the assumption that the investigator wants to discover, understand, and gain insight and therefore must select a sample from which the most can be

learned" (p. 61). Miles and Huberman (1994), based on the work of Johnson (1990), claim that a dimensional sample includes well-informed participants who represent the dimensions of variability sought in a study. The present study was designed to investigate two dimensions: (a) users who participate in requirements determination for an information system and (b) developers who create an information system. Users and developers were chosen from three organizations that met the following criteria:

1. The organizations were large enough to create pairs of NGT groups with approximately seven users and seven developers.
2. They were in different industries in order to obtain diversity in the sample and a broader perspective.
3. They were located in a city that makes conducting the NGT sessions feasible for the researcher.

Profile of Participating Organizations

Several organizations were contacted in order to identify three organizations that could each provide one group with approximately 7 to 10 information system developers and another group with approximately 7 to 10 information system users. These organizations were purposefully sought from different sectors and industries: technology, government, and non-profit. Consistent with the IRB consent form for the present study, the identity of each organization and the participants is anonymous.

Technology

The technology organization in the present study is referred to as Org1. This technology organization is a global leader in the design and creation of semiconductors, and it is a member

of the S&P 500. Although consultants have been used on information system projects, the organization has a large number of people (e.g., users, business analysts, and information system developers) who are frequently responsible for developing and managing information systems. This was the only organization in the present study that had a business analyst who created and translated information system requirements from users to developers.

Government

The government agency in the present study is referred to as Org2. This government agency has extensive information systems used to manage its interaction with the general public. This agency employs project managers and developers, and like the other organizations in the present study, it may use external consultants and vendors to complete projects. This agency provides tax assistance to businesses and individuals, which creates periodic inflexible deadlines for projects. Unlike the other two organizations in the present study, it is arduous for this agency to fire employees.

Non-Profit

The non-profit organization in the present study is referred to Org3. This non-profit organization has a large presence in the United States and operates in several international locations. Their information system staff consists primarily of managers and developers. They tend to use existing, off-the-shelf systems that can be customized for their needs. It is a Christian-based organization, founded on principles from the Bible and Judeo-Christian ethics. Employees consider themselves part of a ministry and call themselves ministry workers.

Table 9 provides descriptive information about the study participants from each organization. The research method required 20 to 30 users and 20 to 30 developers. The actual number of participating users was 22, and the number of participating developers was 24.

Table 9. Descriptive Statistics for Each Participating Focus Group

	Org1		Org2		Org3	
	Users	Developers	Users	Developers	Users	Developers
Development Process Used	Incremental, Unknown	Iterative, Incremental, Others	Iterative, Unknown	Iterative, Agile, Waterfall	Incremental, Waterfall, Unknown	Iterative, Incremental, Waterfall
Mean Number of IS Projects	3 or more	6 or more	6 or more	7 or more	4 or more	7 or more
Mean Years of Experience	6	12.3	12	17.8	6.1	13.2
Number of Factors Created	32	44	28	30	30	48
Number of Participants	6	7	7	8	9	9

Data Collection and Analysis Process

Data were collected in two phases. Phase I data were collected from focus groups using the Nominal Group Technique (Appendix C for detailed procedures). Each group created a list of factors that cause misunderstandings about requirements for an information system. At the conclusion of a focus group, each participant selected the five most important factors. This selection process produced a list of the top- x factors, where x was approximately 10 and determined by a natural break in the number of votes received by each factor. The top- x factors from all three user groups were aggregated to form the combined top- x factors for users, referred to as the User Perceived Factors (UPFs). The same process was used to form the combined top- x factors for developers, called the Developer Perceived Factors (DPFs). Therefore, Phase I produced two lists: (a) UPFs and (b) DPFs.

Phase II data were collected using a survey. The participants in Phase II had already participated in the Phase I focus groups. Participating users received a web-based survey asking

them to weight the importance of UPFs, considering two factors at a time. The survey was constructed to support data analysis using AHP. The same process was used with the participants who were developers. Appendix G contains the user survey, and Appendix H contains the developer survey. Participants accessed both surveys through the World Wide Web (SurveyMonkey, 2006).

Data analysis for Phase I and II is depicted on Figure 11. The analysis shown on this Figure was performed for the user groups and the developer groups. Phase I analysis examined data from the *1-NGT Focus Groups*: (a) a list of the most important factors and (b) an audio recording of a discussion about each factor. Each artifact was analyzed separately. The factor list was used to start the primary analysis path. The audio recording was analyzed separately and used to confirm the primary data analysis and to enhance the richness of the data.

The primary analysis path started with *2-Factors and Votes* produced by the focus groups. *3-Top-X Factors by Group* represent the factors that received the most votes in each group. These factors from each group were combined based on similar meanings and used to form a single user factor list: *4-Combined Top-X Factors*. The combined factors represent the factors created by each group. They are also the factors used in *11-Combined Factors for Surveys/AHP Hierarchy*, which aided participants in recognizing them as the factors identified in their focus group.

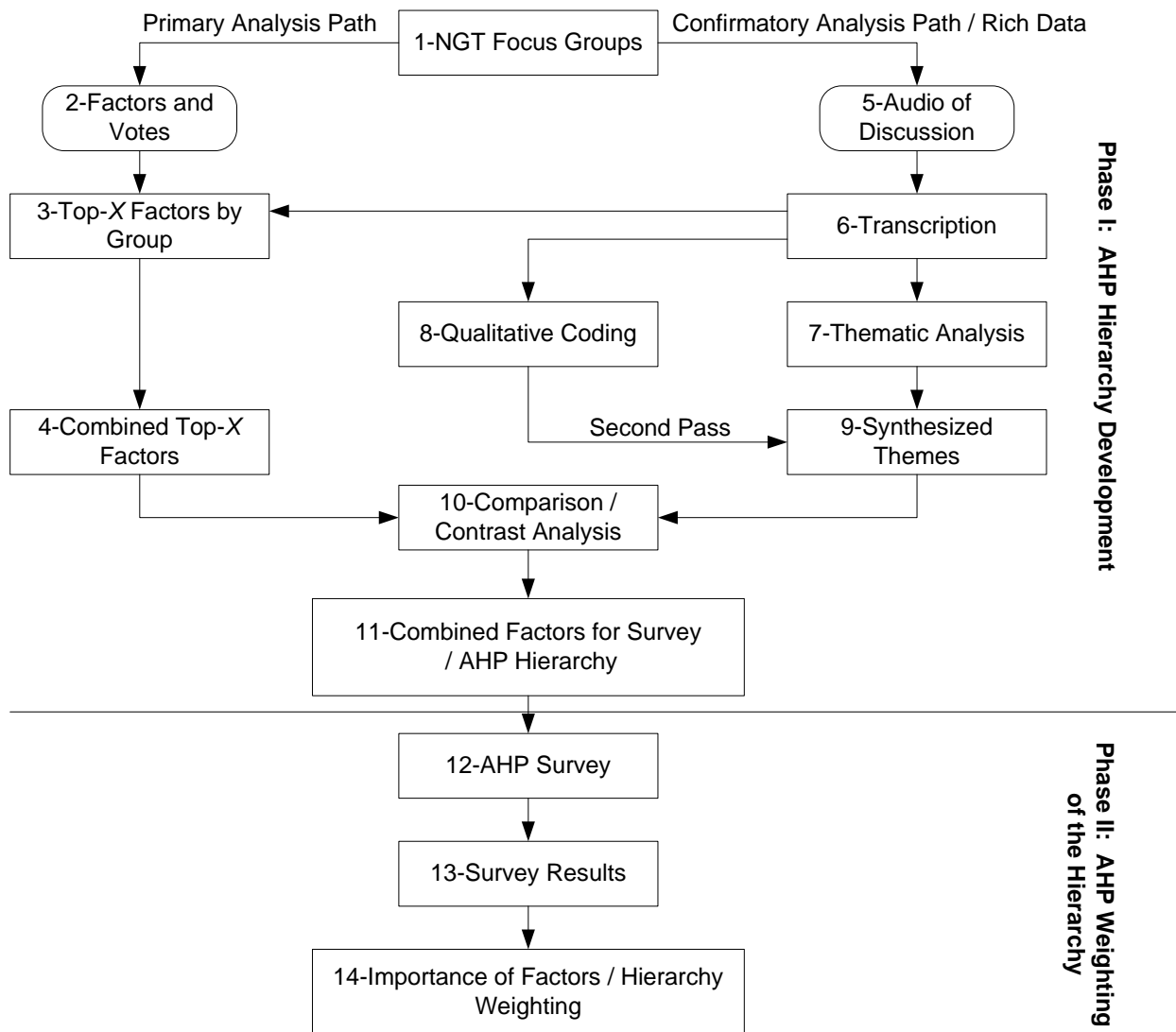


Figure 11. Data analysis activities for Phase I and Phase II.

The secondary analysis path started with *5-Audio of Discussion*, which is the recorded discussion of a focus group. The audio recording was transcribed to produce *6-Transcription*. The transcript was analyzed in two independent ways: first, in *7-Thematic Analysis*, common themes in the transcription were identified, and relationships between factors were noted, which produced a synthesized set of factors that reflected thematic clustering; and second, an

alternative analysis, *8-Qualitative Coding*, was performed on the transcript using Qualrus qualitative analysis software (Idea Works, 2006) to assign qualitative codes to text segments in the transcript. The results were used to verify the completeness and accuracy of the thematic clustering and resulted in *9-Synthesized Themes*. As a result, the analysis was checked using an independent analysis mechanism. Results of the primary and confirmatory analysis paths were analyzed in *10-Comparison/Contrast Analysis*, which resulted in the output for Phase I: *11-Combined Factors for Survey*. In AHP terms, this represents the hierarchy for the AHP model. It is a single level, flat hierarchy that used pairwise assessment to understand the importance assigned to each factor.

In Phase II, an analysis was performed on the combined factors from the user and developer surveys. In *12-AHP Survey*, a survey was created to determine the importance of each factor. The results were collected and formatted (i.e., *13-Survey Results*) and imported into ExpertChoice (2006) for analysis. ExpertChoice is a software application designed to perform AHP calculations and help interpret the data. The results of the ExpertChoice analysis produced *14-Importance of Factors/Hierarchy Weighting*. In AHP terms, the weight assigned to a factor represents the amount of responsibility or influence that factor has on misunderstanding requirements.

Reliability, Validity, and Generalizability

Any useful research must convince readers that the findings are trustworthy and believable: that is, they are reliable and valid. It is also important to show whether the findings can be generalized to a larger population or whether the findings only apply to the group under study.

Reliability is concerned with the repeatability of a study: that is, whether the same results can be produced if the same data collection and analysis methods are employed in a new study (Robson, 2002). Reliability is generally difficult to achieve in qualitative studies because it is difficult to replicate the identical circumstances of the original study. Reliability can be improved by using well-structured data collection techniques. For example, NGT does not rely on a skilled interviewer or facilitator to be successful, and this reduces researcher bias. Although the factors generated by the NGT groups in the present study were expected to vary, using the sampling approach suggested by Griffin and Hauser (1993), it was assumed that similar studies will produce at least 90% of the aggregated factors identified in the present study for similar organizations. AHP is also a highly structured and reliable technique. Given its internal consistency checking ability, it was assumed that the AHP results for similar studies will produce similar consistency values.

Validity is concerned with whether the measurements provide the information needed to answer research questions (Cooper & Schindler, 2003). Robson (2002) states it more simply: “Whether the findings are ‘really’ about what they appear to be about” (p. 93). Robson, using a typology provided by Maxwell (1992), describes three threats to validity: (a) description, (b) interpretation, and (c) theory. Description is concerned with providing an accurate account of what was heard and seen, and in the present study, the accuracy of what was heard and seen was ensured by using audio recordings of the NGT sessions and collecting data using the AHP instrument. Interpretation is concerned with reaching defensible conclusions. NGT and AHP guard against problems with interpretation, but in the present study, it was assumed that the interpretation of users’ and developers’ comments during NGT sessions could be questioned.

Therefore, in the present study, the procedures used for the thematic analysis are described to lend credibility to the interpretation of participants' comments (Miles & Huberman, 1994). The theory issue involves the possibility that there are alternative explanations for the findings of a study. The present study was designed to build theory, and NGT and AHP were used to identify and prioritize the factors that cause misunderstandings about the requirements for information systems. Future research is needed to discover whether the application of these factors and the results of the thematic analysis improve requirements determination.

Given the findings of Griffin and Hauser (1993), it was assumed in the present study that at least 90% of the factors that cause misunderstandings between users and developers for the three organizations involved would be identified using 20 to 30 participants. The NGT groups in this study were formed using 24 developers and 22 users, and these groups should have identified at least 90% of the factors that cause misunderstandings between developers and users. Therefore, the majority of UPFs and DPFs identified in the present study should be recognized by other users and developers. However, other uncontrolled variables may have had a meaningful impact on the factors. For example, the development method used or the success of an information system could cause the factors to differ. Also, the absolute weightings assigned using AHP vary within any sample of the population. In order to improve generalizability, the AHP survey should be conducted with a larger sample, account for more controlled variables, and allow for the impact of subgroups.

Ethical Considerations

Several ethical issues (for more detail, see Cooper & Schindler, 2003) were considered in the present study: (a) the researcher's code of ethics, (b) protecting participants, (c) identifying

conflicts of interests, (d) guarding against researcher bias, and (e) disclosing the researcher's qualifications. The researcher's code of ethics is based on integrity when conducting research, collecting data, drawing conclusions, and presenting findings. Conclusions are driven by the data only and not by expectations or hypotheses. The participants were protected by ensuring their anonymity (Miles & Huberman, 1994) and by not identifying the names of their organizations. In addition, participants were protected by adhering to the Institutional Review Board (IRB) process and Capella University guidelines (Capella, 2005):

1. Respect for persons, which involves recognition of the personal dignity and autonomy of individuals along with special protection of those people with diminished autonomy.
2. Beneficence, which entails an obligation to protect people from harm by maximizing anticipated research results while minimizing possible risks of harm.
3. Justice, which requires that the benefits and burdens of research be distributed fairly.

Conflicts of interest occur when a researcher is in a position to compromise a study's conclusions. The researcher in the present study did not receive payment for the research from any party and felt no pressure to produce results that represent a predetermined perspective. Researcher bias is always present in any investigation—it is the nature of the social sciences (Robson, 2002)—and it was minimized by observing good research practices, including retaining a skeptical attitude about the conclusions. The research method was chosen for its ability to address the research questions and protect against researcher bias.

The researcher is qualified to conduct the present study as a result of being a PhD candidate and being guided and supervised by a doctoral committee. In addition, the researcher has more than 18 years experience directly related to the development of custom software solutions and information systems. The majority of this experience involved working directly with both users and developers in order to develop systems that satisfy business needs. This experience enabled the researcher to witness the misunderstandings that occur between users and developers, and it motivated the researcher to conduct the present study.

CHAPTER 4. DATA COLLECTION AND ANALYSIS

Purpose of the Study and Research Questions

The primary purpose of the study discussed in this dissertation is to identify factors that influence misunderstandings about the requirements for information systems from the perspectives of developers and users. The following questions guided this study:

1. Which factors do users and developers believe cause misunderstandings about the requirements for information systems?
2. Which factors do users and developers believe have the most impact on misunderstandings?
3. What is the difference between users' and developers' perceptions of these factors?

This chapter addresses these questions and more detailed subquestions, and it includes a discussion of the analysis of users' and developers' perceptions of the factors that cause misunderstandings.

Data Collection and Analysis of Users' Perceptions

The present study used two questions to investigate UPFs that cause misunderstandings about the requirements for information systems:

1. Which factors do users believe cause misunderstandings about requirements?
2. How do users prioritize the factors that cause misunderstandings about requirements?

Step 1-NGT Focus Groups for Users

As was discussed in Chapter 3 and shown on Figure 11, data collection and analysis began by conducting focus groups with users of information systems who had been involved in

requirements determination activities. The focus groups were highly structured and used the NGT procedures presented in Appendix C.

Step 2-Factors and Votes for Users

The three user focus groups produced a list of 90 factors that cause misunderstandings about the requirements for information systems. Many of these factors were common between the groups. Each group participant selected the five most important factors. Appendix I lists all of the factors and the votes each received, expressed as the percentage of participants in the focus group.

Step 3-Top-X Factors by Group for Users

As a result of this selection process, the Top-X factors were identified for each group of users. The objective was to identify approximately 10 factors for each group. After the participants in a group voted for the most important factors, if a natural break in voting did not exist near the top-10 most-voted-for factors, participants would be asked to vote again in order to clarify the list. A natural break near the top-10 factors did exist for each of the user groups, and a second vote was not necessary. The users from Org1 selected nine factors, Org2 users selected 10 factors, and Org3 users selected 11 factors. These factors are shown in Table 10.

Table 10. *Users' Top Voted-For Factors*

Vote	Factor Identified by Participants
100%	Lack of appropriate analyst resources
71%	Users poorly articulate requirements
50%	Correct people are not involved in requirements determination
50%	Time/schedule drives projects

Table 10. Users' Top Voted-For Factors, Continued

Vote	Factor Identified by Participants
50%	Users do not understand available technology options
44%	Key people are not invited
44%	Key people start late on the project
43%	Terminology is difficult: user and developer jargon
43%	Users tend to be unable to separate what is currently in the system from what they need
33%	Business analysts/developers do not know business
33%	Business needs can change over the course of development
33%	Developers and users communicate differently
33%	Managers make decisions without understanding users' needs
33%	Needs change during the course of a project (e.g., 2 years)
33%	Not having enough requirements defined before development starts
33%	Not understanding interaction between systems/business processes
33%	Unclear roles and responsibilities
33%	Universe of affected users is not adequately identified or understood
33%	Users do not begin with the end in mind
33%	Users get stuck on an interim solution, thinking it is a long-term solution
33%	Users have conflicting needs; users use the system differently
29%	No communication between correct user and correct developer
29%	People involved in requirements determination may not be the right people
29%	Some users believe requirements determination is a waste of time
29%	Unclear system scope
29%	User/developer frame of reference
29%	Users do not realize what can and cannot be done

Table 10. *Users' Top Voted-For Factors, Continued*

Vote	Factor Identified by Participants
22%	Lack of broad knowledge of organization needed to make decisions
22%	Necessity versus nicety
22%	User confusion between business process and requirements: how versus need

Step 4-Combined Top-X Factors for Users

The most-voted-for factors from each user group were thematically combined into nine factors, referred to as the UPFs. The thematic analysis identified similar meanings between the factors and produced a general meaning for each group of factors. Table 11 lists the factors, by organization, that were combined to form more general factors.

Table 11. *Combined User Factors and Supporting Factors*

ID	Name	Org	Combined User Factor
U1	Key Users		Key users who have the information needed to determine requirements are not appropriately involved in requirements determination. This was expressed as <ul style="list-style-type: none"> 1 • Correct people are not involved in requirements determination 2 • People involved in requirements determination may not be the right people 3 • Key people start late on the project 3 • Key people are not invited 3 • Universe of affected users is not adequately identified or understood
U2	User Technology Knowledge		Users do not know what is possible. This was expressed as <ul style="list-style-type: none"> 1 • Users do not understand available technology options 2 • Users do not realize what can and cannot be done
U3	Deadlines		Deadlines drive projects, leaving inadequate time for requirements determination with the resources available. This was expressed as <ul style="list-style-type: none"> 3 • Not having enough requirements defined before development starts 1 • Time/schedule drives projects 2 • Lack of appropriate analyst resources

<i>Table 11. Combined User Factors and Supporting Factors, Continued</i>			
ID	Name	Org	Combined User Factor
U4	Roles		<p>People do not understand their role and the roles of others in requirements determination. This was expressed as</p> <ul style="list-style-type: none"> 1 • Unclear roles and responsibilities 3 • Managers make decisions without understanding users' needs 2 • Some users believe requirements determination is a waste of time
U5	Requirements Changes		<p>Requirements change during the creation of an information system, and the changes are not adequately addressed. This was expressed as</p> <ul style="list-style-type: none"> 1 • Needs change during the course of a project (e.g., 2 years) 3 • Business needs can change over the course of development
U6	Developer Business Knowledge		<p>Developers/IT lack knowledge about the business. This was expressed as</p> <ul style="list-style-type: none"> 1 • Business analysts/developers do not know business 3 • Lack of broad knowledge about organization needed to make decisions 1 • Not understanding interaction between systems/business processes 1 • Users do not start with the end in mind
U7	User Uncertainty		<p>Users are unclear about their needs and the priority of those needs. This was expressed as</p> <ul style="list-style-type: none"> 2 • Users poorly articulate requirements 3 • Necessity versus nicety 2 • Unclear system scope 1 • Users have conflicting needs; users use the system differently
U8	User Past Experience		<p>Users' experience with current systems limits their ability to create requirements for a new system. This was expressed as</p> <ul style="list-style-type: none"> 3 • User confusion between business process and requirements: how versus need 2 • Users are unable to separate what is currently in the system from what they need 3 • Users get stuck on an interim solution, thinking it is a long-term solution
U9	Different Perspectives		<p>Users and developers relate to each other differently. This was expressed as</p> <ul style="list-style-type: none"> 2 • Terminology is difficult: user and developer jargon 3 • IT and users communicate differently 2 • User/IT frame of reference 2 • No communication between correct user and correct developer

Step 5-Audio of Discussion for Users

The user focus groups were recorded using two digital audio recorders: one was a primary recorder, and the other was a backup. The audio recording was loaded directly from the

recorder into a WMA format on a personal computer. The audio quality for each group was good, and most participants' comments were understandable.

Step 6-Transcription for Users

ExpressScribe (2006) software was used to create a transcript of each focus group discussion. The text captured the discussion that followed the identification of each factor, which is part of the Nominal Group Technique (NGT).

Step 7-Thematic Analysis for Users

The factors identified in each user group's transcripts were organized into common themes. A theme would emerge from a discussion about many factors: For example, the theme of *key users not being available when necessary* emerged during discussions by all three user groups as they discussed several factors. The following list shows the factors identified by one user group that were combined to form the theme of key users not being available when necessary:

1. Users most valuable during the requirements determination process are the least available because they are too busy doing their existing work.
2. Most input comes from non-expert users.
3. The right users are not identified.
4. Managers do not prioritize requirements work.
5. Too few users are involved in requirements determination.
6. Individual users may not represent the needs of the business or the larger user group.

The same theme emerged from the following factors identified by another user group:

1. Key users are not available to participate in requirements determination.

2. Managers participate, but they are not the “worker bees” who will be using the system or they are not the best people to determine requirements.
3. Developers who actually do the coding should be involved in requirements determination.
4. Key users are too busy with other work, and managers do not make them available.
5. Key people do not stay involved during the project.
6. Key people cannot get together; conflicting schedules and other demands make group meetings nearly impossible.

Table 12 shows the total number of themes generated from each user group’s transcripts.

Table 12. Number of Themes Generated by Users

	Total
Org1	20
Org2	19
Org3	21

The synthesized themes shared by all user groups are shown in Table 13. This Table provides a short synopsis of each theme, but the complete dimensions of these themes are contained in Appendix J. The number of organizations that provided support for each theme is also provided, and this support is implied based on the number of votes for the factors that constitute a theme. This support was calculated by determining the factors that are associated with a theme, adding the votes for each of these factors to reach an implied total number of votes for the theme, normalizing this total to 100%, adding the normalized total for each organization, and dividing the total by three to find the mean. Therefore, the support for a theme is an indication of how important that theme was to all organizations, not to a single organization.

Table 13. *Synthesized Themes Based on User Discussion*

ID	Description of Theme	Number of Orgs	Support
UT1	User-developer translation: Users and developers speak different languages and need a translator	3	60%
UT2	Developers lack understanding about the business: Developers and business analysts do not adequately understand the organization's business	3	59%
UT3	Effect of time: Business needs change over time and can change requirements	3	58%
UT4	Key users are not involved in requirements: The best people are not part of the requirements determination process	3	50%
UT5	Articulation difficulties: Users have difficulty adequately describing their needs and may say one thing while visualizing something different	2	50%
UT6	User versus developer frame of reference: Users and developers have very different ways of looking at a problem	2	41%
UT7	Users do not understand technology: Users do not fully understand the options available to meet their needs	3	35%
UT8	Big picture understanding of the problem: Users lack a full understanding about the problem or objective and how it impacts the organization	2	34%
UT9	Terminology difficulties: Developers lack knowledge about business terminology, and users lack knowledge about developers' terminology	3	33%
UT10	Assumptions: Developers make assumptions about requirements instead of asking users questions, and users make assumptions about what the system will do	3	31%
UT11	Box thinking by users: Users' past experience limits their thinking about new systems and makes exercises that start with a blank sheet of paper difficult	3	24%
UT12	Developers know better: Developers create the information system they believe users need, not what is asked for by users	2	22%
UT13	Unclear who is responsible for requirements: Users do not want to be involved in requirements development, and developers should guide the creation of requirements	2	22%
UT14	Poor requirements reviews: Users are seldom asked to review requirements, and they are not notified about changes made to them	2	19%
UT15	Requirements documentation is poor: Requirements documentation is not understandable or complete	1	17%
UT16	Schedule drives projects: Deadlines drive projects and the functionality users receive	1	17%
UT17	Users lose hope: Some users believe requirements determination is a waste of time because they seldom see results for their work	2	16%

Table 13. *Synthesized Themes Based on User Discussion, Continued*

ID	Description of Theme	Number of Orgs	Support
UT18	Users are not equipped for requirements work: Users do not have skills for or experience with requirements determination	2	16%
UT19	Conflicting user needs: Users in the same group and different groups can have conflicting requirements for an information system	3	13%
UT20	Box thinking by developers: Developers create new systems based on previous systems	2	13%
UT21	Telephone game (Chinese Whispers): Communication breaks down as messages travel from users to the correct developer, sometimes through a business analyst	2	12%
UT22	Requirements are unexpectedly changed: Management changes requirements without involving users	1	12%
UT23	Design causes constraints: Developers' choice of technology and design limits what is possible from an information system	1	7%
UT24	Development starts before requirements are complete: Coding starts before requirements have been reviewed and completed	1	7%
UT25	Information gets lost: Requirements get lost and not acted on as they are relayed between developers	2	6%
UT26	Users do not prioritize requirement work: The best people are already too busy doing important work and are not made available for requirements determination	2	5%

Step 8-Qualitative Coding for Users

In order to check the thematic analysis that resulted in the 26 user themes, each user group's transcripts were imported into qualitative analysis software called Qualrus (Idea Works, 2006). Based on the discussion that surrounded each factor, codes were manually created to summarize the concepts shared by participants. A total of 41 codes were created for the users' transcripts. Many of the codes reflect concepts that were common between the user groups (see Table 14), and 76% of the codes were shared by two or more groups.

Table 14. *Percent of Codes in Common*

	Percent
Codes shared by all three groups	39%
Codes shared by two groups	37%
Codes present in only one group	24%

Researchers can create custom scripts in Qualrus for identifying information in coded data. In the present study, a script was developed for listing the codes associated with each factor, which is shown in Appendix K. The list produced in Qualrus was used to verify or enhance the prior thematic analysis for each user group. This was accomplished by searching for the factor numbers in the thematic analysis that were provided by Qualrus and verifying that the concept expressed in the Qualrus code was appropriately accounted for in the themes. For example, in one focus group, the communication problems code was associated with factors 5, 6, 7, 8, 13, and 25. The list of themes was searched for each of these factors and checked to ensure that they were appropriately attributed to a communication-related theme: for example, users lose hope, information gets lost, terminology difficulties, articulation difficulties, user-developer translation, key users are not involved in requirements, effect of time, and telephone game. Based on the discussion of a factor, one factor could be attributed to multiple themes. During this process, unaccounted factors were added to the thematic analysis, misplaced factors were corrected, and additional themes were created. This process ensured that the thematic analysis was an accurate synthesis of the factors identified by participants.

Step 9-Synthesized Themes

Table 13 already reflects the corrective actions taken as a result of the qualitative coding analysis.

Step 10-Comparison/Contrast Analysis for Users

The synthesized themes were compared to and contrasted with the UPFs. Only themes with a support of 20% or more were considered in the analysis; 20% was arrived at iteratively by examining how the synthesized themes with the most support compared to the UPFs, with the objective of comparing the most-supported themes to the most-voted-for user factors. Thirteen synthesized themes had support of 20% or more. All but one theme with 20% or more support was related to the PFs: *big picture understanding of the problem*, which had 34% support. This theme emerged from two user groups as they discussed factors such as *users do not start with the end in mind*, *users do not understand ultimate goal*, and *lack of broad knowledge about the organization needed to make decisions*. However, no factor addressing a big picture understanding received sufficient votes to make it a UPF. In contrast, all of the UPFs were related to the most-supported synthesized themes. The comparison between factors and themes is summarized in Table 15, which shows that many of the themes are applicable to more than one factor.

The purpose of the comparison shown in Table 15 was to investigate if the participants' discussions reflected the factors they selected as most important. In general, the synthesized themes confirmed the importance of the most-voted-for factors designated as the UPFs, with 12 out of 13 themes receiving 20% or more support for the UPFs. Only one synthesized theme could not be attributed to UPF.

Table 15. Comparison of Top-9 Factors and Synthesized Themes With 20% or More Support

ID	UPF	Synthesized Theme	ID
U1	Key users who have the information needed to determine requirements are not appropriately involved in requirements determination	The right people are not involved in requirements determination	UT4
		Developers know better	UT12
		Unclear who is responsible for requirements determination	UT13
U2	Users do not know what is possible	Users do not understand technology	UT7
		Box thinking by users	UT11
U3	Deadlines drive projects, leaving inadequate time for requirements determination with available resources	Effect of time	UT3
U4	People do not understand their role and the roles of others in requirements determination	The right people are not involved in requirements determination	UT4
		Assumptions are made by both developers and users	UT10
		Unclear who is responsible for requirements determination	UT13
U5	Requirements change during the creation of an information system, and the changes are not adequately addressed	Effect of time	UT3
U6	Developers/IT lack knowledge about the business	Developers lack understanding about the business	UT2
		Terminology difficulties	UT9
		Developers know better	UT12
U7	Users are unclear about their needs and the priority of those needs	Articulation difficulties	UT5
		Assumptions are made by both developers and users	UT10
U8	Users' experience with current systems limits their ability to create requirements for a new system	Users do not understand technology	UT7
		Box thinking by users	UT11
U9	Users and developers relate to each other differently	User-developer translation	UT1
		User versus developer frame of reference	UT6
		Terminology difficulties	UT9

Step 11-Combined Factors for Survey/AHP Hierarchy for Users

The original UPFs identified by the participants were used to form the users survey, which asked users to rate the importance of each factor.

Step 12-AHP Survey for Users

The purpose of the survey administered to users was to determine how much importance users place on each of the most-voted-for factors. This was investigated using a pairwise approach to rank and weight the factors. Participants were asked to choose between two factors at a time, selecting the one that was more important and then indicating the level of its importance. The survey supported AHP analysis and was consistent with other web-based AHP surveys (Finnie & Wittig, 1999). Nine factors had to be compared two at a time, and 36 comparisons were evaluated by each participant. Appendix G shows the content of the survey provided to the user participants in a web-based form.

Step 13-Survey Results for Users

Participants were told at the start and end of the focus groups that they would be asked to participate in a web-based survey. An email invitation to complete the survey was personally sent to each participant. Reminders were periodically emailed to participants who had not yet completed the survey. The survey was available for 6 weeks, and 21 of 22 participants completed the survey. The 95% completion rate is not surprising because participants knew they would be asked to complete the follow-up survey after the focus groups.

The survey responses were downloaded from the web-based survey provider and saved as a spreadsheet. Two types of data were collected for each factor: the factor that was more

important when compared two at a time and the factor’s level of importance based on a nine-point scale (previously described in Table 7). These data were translated into an array for importing into ExpertChoice (2006).

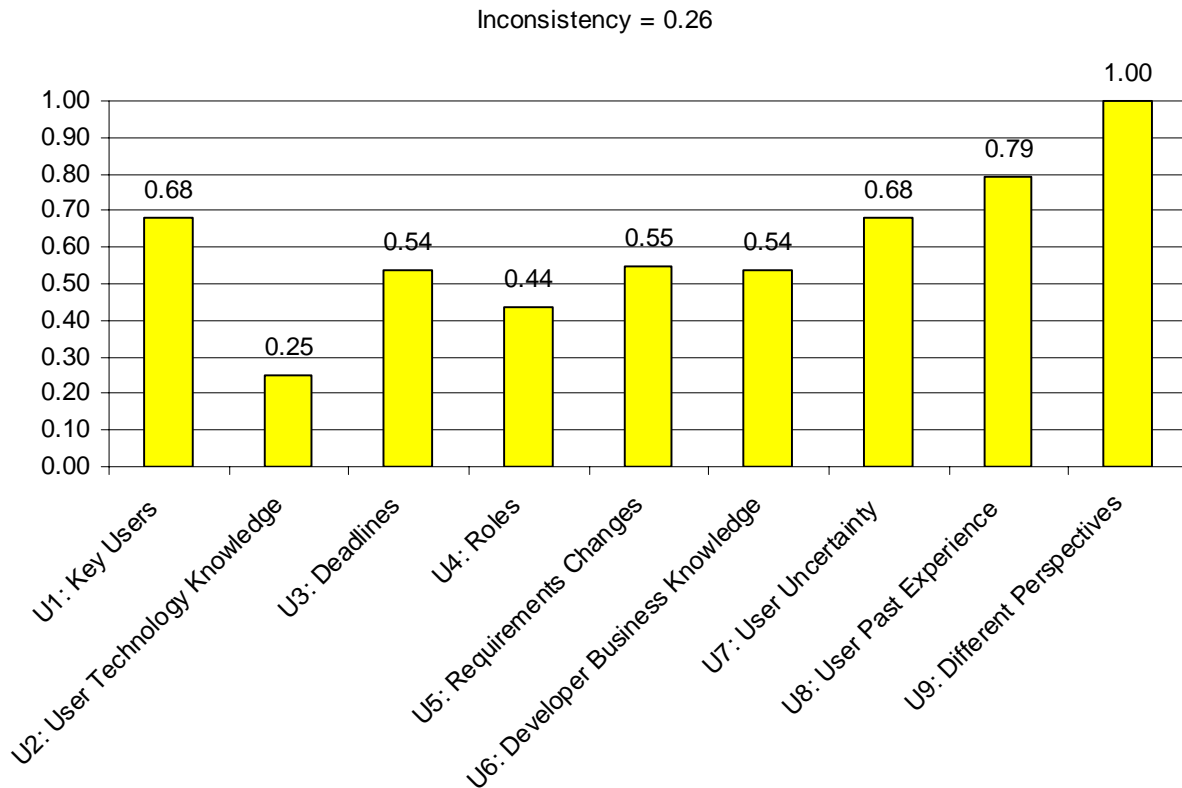


Figure 12. Importance of user factors normalized to 1.0

Step 14-Importance of Factors/Hierarchy Weighting for Users

The AHP algorithm was used to analyze the data from all user participants. Figure 12 illustrates the importance participants placed on each factor. These data have been normalized so the most-important factor (i.e., U9, *users and developers relate to each other differently*) has an importance equal to 1.0. The importance of the other factors is relative to the most important

factor: For example, U2 (*users do not know what is possible*) has an importance of 0.25, which is one fourth as important as U9.

The calculated AHP weights for each factor based on the pairwise comparisons, which sum to 1.0, are shown on Figure 13. These weights show the amount of influence a factor has on misunderstandings about requirements: for example, U9 has a weight of 0.18, which means that U9 exerts 18% of the influence on misunderstandings about requirements.

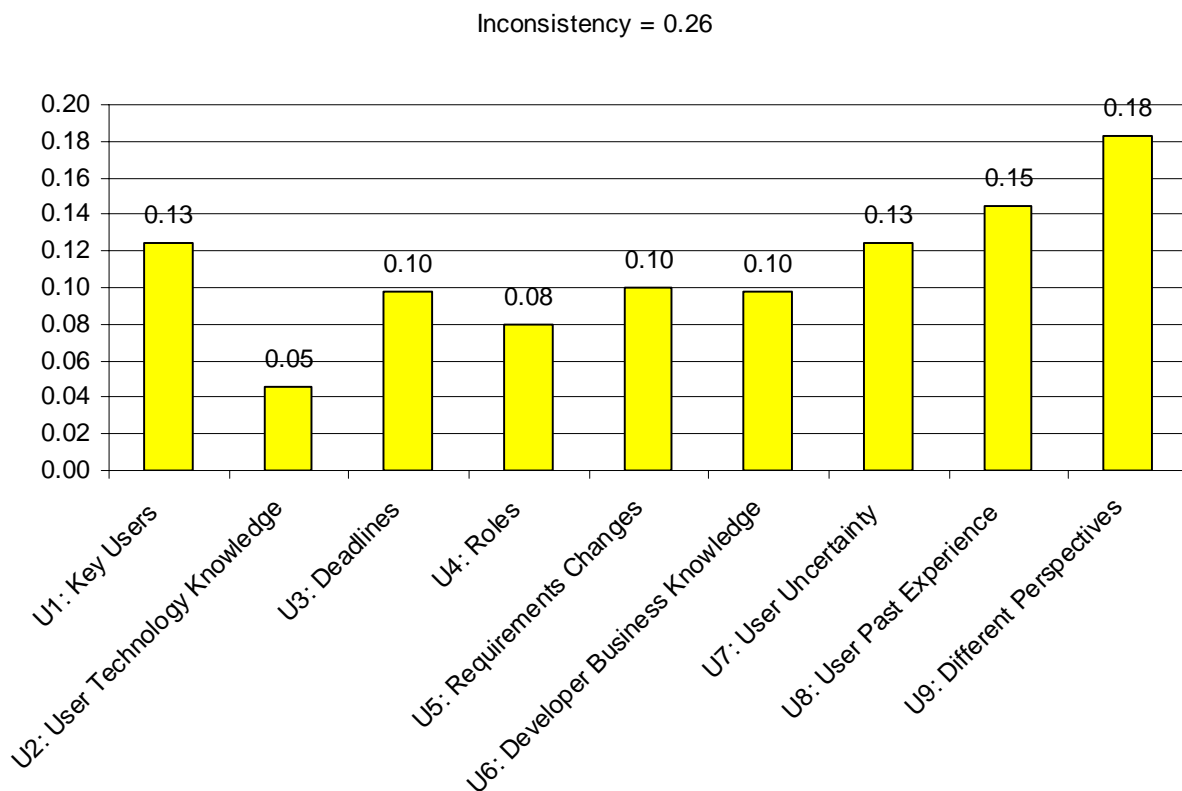


Figure 13. Importance of user factors showing AHP weights summing to 1.0

In order to interpret the AHP weights and identify significant differences between them, the standard deviation around the mean of the weights was overlaid on the data, as shown on Figure 14. The weights, which are equivalent to the importance participants placed on each

factor, are within one standard deviation for seven of the nine top user factors. Therefore, these seven factors share a similar importance and are responsible for a similar amount of influence on misunderstandings about requirements. Only U9 (*users and developers relate to each other differently*) is much more important than the other factors, and U2 (*users do not know what is possible*) is much less important than the other factors.

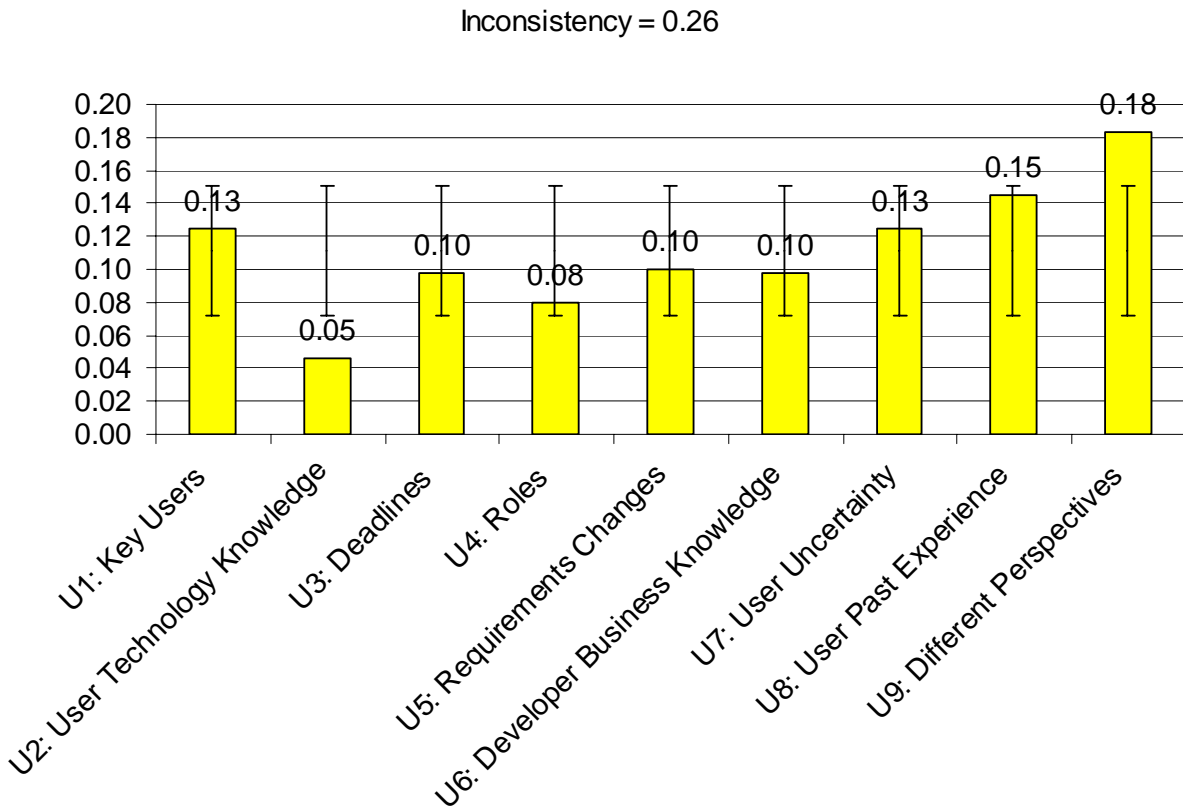


Figure 14. AHP weights for each user factor shown with standard deviation error bars.

AHP calculates an inconsistency index for each participant as well as the aggregated results for any n participants. This is possible because of the pairwise assessments participants made on the survey. For example, consider the statement “A is more important than B, and B is more important than C.” To be consistent, “A must be more important than C.” Using notation,

this is $A > B$, $B > C$, and $A > C$. If a participant responded that $A > B$, $B > C$, but $C > A$, then the response is inconsistent.

Saaty (1999) recommends keeping inconsistency for five or greater choices to 0.10 or less. Therefore, judging the impact of consistency is important for understanding the validity of the AHP results. The results shown on Figure 13 have an inconsistency of 0.26, which is well above Saaty’s recommendation. This higher value is a result of the large amount of inconsistency in each participant’s responses to the survey. Appendix L shows the inconsistencies for each participant, which ranged from 0.22 to 1.45, with an arithmetic mean of 0.64.

In order to examine the impact of inconsistency on the importance of the factors, several subgroups of participants were analyzed. This analysis compared the aggregated results from participants with individual inconsistencies of less than 0.67, 0.41, and 0.37 to the results from all participants. The number of participants in each category is shown in Table 16. The final group, with participants chosen based on individual inconsistencies of 0.36 or lower, had the lowest aggregated inconsistency; however, increasing the number of participants from three to eight, which are those with inconsistency of 0.40 or less, had little impact on the aggregated inconsistency.

Table 16. Participants Divided into Groups Based on the Inconsistency of Their Responses

Inconsistency Used to Create Group	Number of Participants	Inconsistency Value for Aggregate
All values	21	0.26
0.66 or less	12	0.19
0.40 or less	8	0.17
0.36 or less	3	0.16

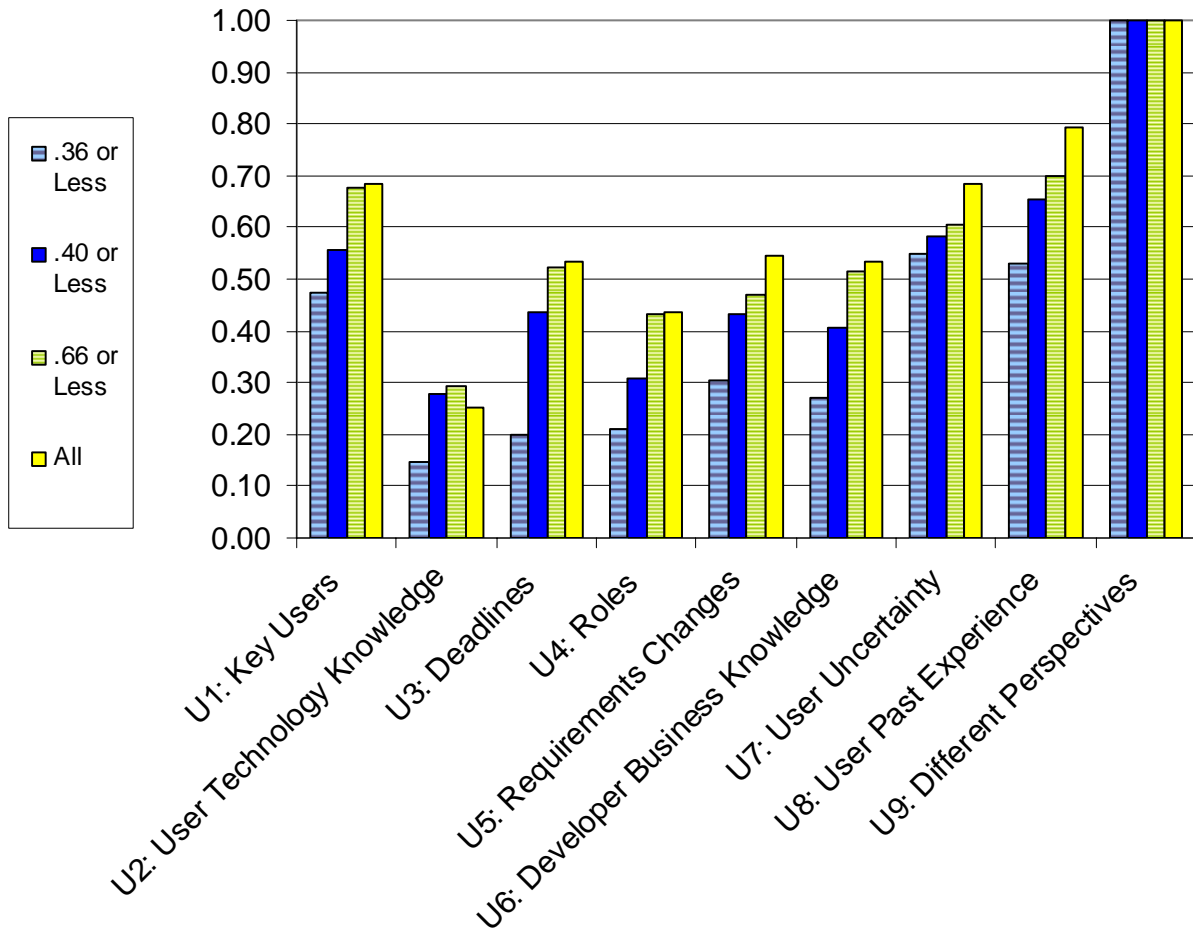


Figure 15. Comparison of importance based on user groups with decreasing inconsistency values.

The comparison of the aggregated results for each of these groups with the decreasing inconsistency is shown on Figure 15. The pattern of the chart remains similar for each group, even for the group that only contained three participants. The ranked order of the factors does vary, but this order is generally similar within one place of each other. Details are shown in Table 17.

Table 17. User Factor Rankings by Groups Based on Inconsistency

All		0.66		0.40		0.36	
Rank	Wgt	Rank	Wgt	Rank	Wgt	Rank	Wgt
U9: Different Perspectives	1.00	U9: Different Perspectives	1.00	U9: Different Perspectives	1.00	U9: Different Perspectives	1.00
U8: User Past Experience	0.79	U8: User Past Experience	0.70	U8: User Past Experience	0.66	U7: User Uncertainty	0.55
U7: User Uncertainty	0.68	U1: Key users	0.68	U7: User Uncertainty	0.58	U8: User Past Experience	0.53
U1: Key users	0.68	U7: User Uncertainty	0.60	U1: Key users	0.56	U1: Key users	0.47
U5: Requirements Changes	0.55	U3: Deadlines	0.52	U3: Deadlines	0.44	U5: Requirements Changes	0.31
U6: Developer Business Knowledge	0.54	U6: Developer Business Knowledge	0.52	U5: Requirements Changes	0.43	U6: Developer Business Knowledge	0.27
U3: Deadlines	0.54	U5: Requirements Changes	0.47	U6: Developer Business Knowledge	0.40	U4: Roles	0.21
U4: Roles	0.44	U4: Roles	0.43	U4: Roles	0.31	U3: Deadlines	0.20
U2: User Technology Knowledge	0.25	U2: User Technology Knowledge	0.29	U2: User Technology Knowledge	0.28	U2: User Technology Knowledge	0.15

Note. Wgt is the AHP weight calculated from participants' pairwise comparisons.

Finally, the results were examined by organization. The inconsistency values for each organization are shown in Table 18.

Table 18. Inconsistency Values by Organization for User Participants

Organization	Number of Participants	Inconsistency Value for Aggregate
Org1	6	0.33
Org2	6	0.26
Org3	9	0.25

The importance of factors by organization are provided on Figure 16, and the ranking details are shown in Table 19.

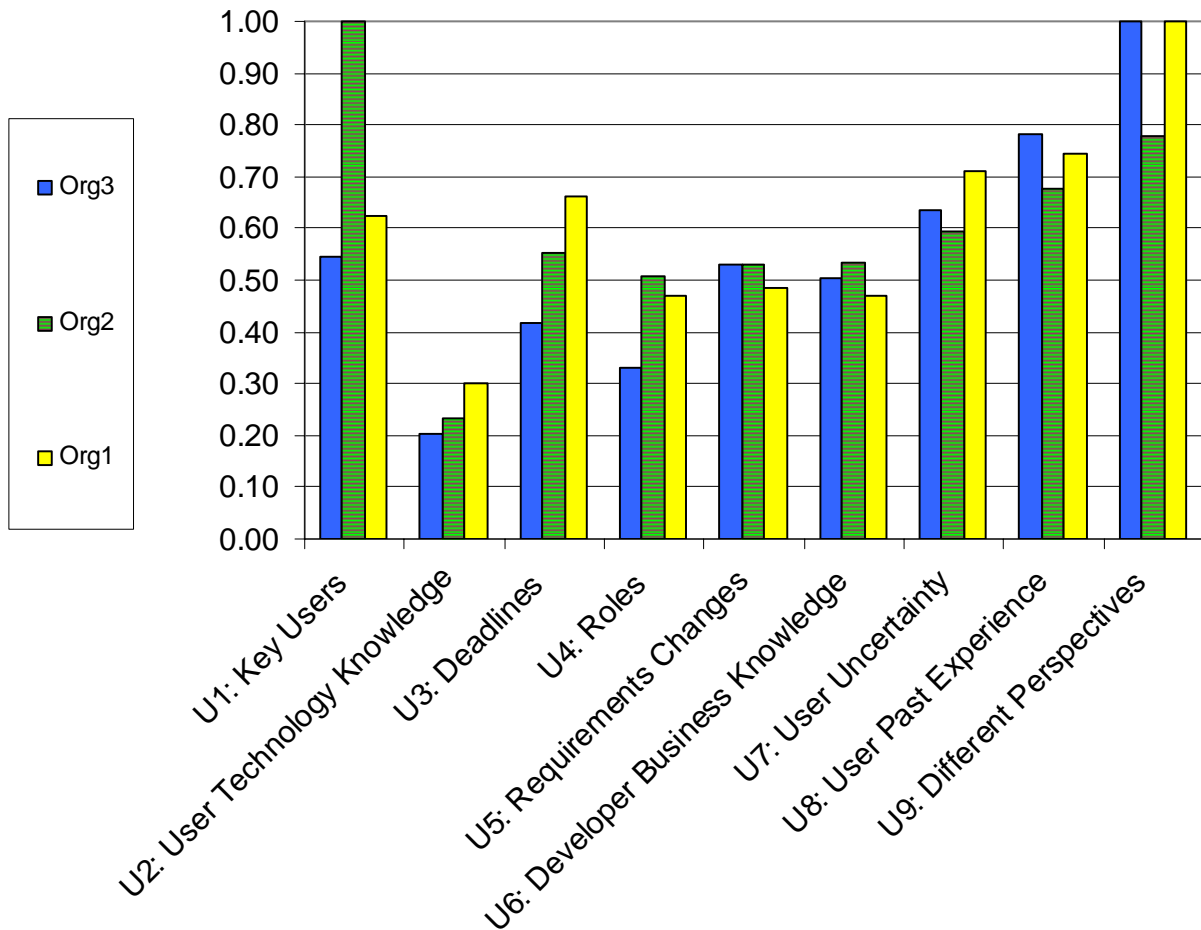


Figure 16. Importance of factors as judged by users for each organization.

With the exception of Org2, which placed more emphasis on U1 (*key users who have the information needed to determine requirements are not appropriately involved in requirements determination*), all three organizations tended to rank the factors in a similar way, with reversals generally existing only between factors of similar importance. In addition, the organizational rankings were similar to the rankings previously examined by groups of inconsistency. In these

six cases, U9, U8, U7, and U1 ranked as most important in five cases, with Org1 ranking U3 instead of U1 in the top four. Therefore, although the inconsistency values are larger than recommended by Saaty (1999), the results trend towards similar rankings even as inconsistency increases.

Table 19. *User Factor Rankings by Organization*

Org1		Org2		Org3	
Rank	Wgt	Rank	Wgt	Rank	Wgt
U9: Different Perspectives	1.00	U1: Key Users	1.00	U9: Different Perspectives	1.00
U8: User Past Experience	0.74	U9: Different Perspectives	0.78	U8: User Past Experience	0.78
U7: User Uncertainty	0.71	U8: User Past Experience	0.68	U7: User Uncertainty	0.63
U3: Deadlines	0.66	U7: User Uncertainty	0.59	U1: Key users	0.54
U1: Key users	0.62	U3: Deadlines	0.55	U5: Requirements Changes	0.53
U5: Requirements Changes	0.49	U6: Developer Business Knowledge	0.54	U6: Developer Business Knowledge	0.50
U4: Roles	0.47	U5: Requirements Changes	0.53	U3: Deadlines	0.42
U6: Developer Business Knowledge	0.47	U4: Roles	0.51	U4: Roles	0.33
U2: User Technology Knowledge	0.30	U2: User Technology Knowledge	0.23	U2: User Technology Knowledge	0.20

Note. Wgt is the AHP weight calculated from participants' pairwise comparisons.

Data Collection and Analysis of Developers' Perceptions

The present study used two questions to uncover developers' perceptions about the factors that cause misunderstandings about the requirements for information systems:

1. Which factors do developers think cause misunderstandings about requirements for information systems?
2. How do developers prioritize the factors that influence misunderstandings about requirements?

Step 1-NGT Focus Groups for Developers

The data collection and analysis of developers’ perceptions followed the same procedures as those for users. Therefore, not all the procedural details are repeated in this section. As with users, data collection and analysis began by conducting focus groups with developers of information systems.

Step 2-Factors and Votes for Developers

The participants in the developer focus groups identified 122 factors, with many of the factors in common between the groups. Appendix M lists all of the factors and the votes, as a percentage of the group, each received.

Step 3-Top-X Factors by Group for Developers

The top-X factors were identified for each group by asking participants to select the most important factors. The objective was to identify approximately 10 factors for each group. Two groups required a second round of voting to reduce the number to approximately 10 factors. Org1, after voting twice, produced a natural preference break between the eleventh and twelfth most-voted-for factors. Org2 also required a second vote, and the participants identified 13 top factors. Org3 selected 13 top factors in the first vote. The top factors identified by developers are shown in Table 20.

Table 20. Developers’ Top Voted-For Factors

Vote	Factor As Expressed by Participants
100%	Key players not always involved in requirements determination
57%	Developers-business analysts-users relationships can help or hinder requirements determination
57%	Lack of requirements review
57%	The telephone game: users to business analysts to developers communications

Table 20. Developers' Top Voted-For Factors, Continued

Vote	Factor As Expressed by Participants
57%	Users are not sure what they want, are vague about needs, and do not understand the problem
50%	Assumptions made by developers and users
50%	Lack of ongoing user involvement during design
44%	Appropriate people are not included; insufficient proxy for users
44%	Users do not understand requirements and cannot articulate requirements
43%	English is not the user's first language
38%	Formal versus informal requirements
33%	Technology is complicated
31%	Lack of enterprise planning; unable to understand the whole
29%	Business units do not create business plans
29%	Lack of strategic plans for the organization
29%	Managers do not provide users the time needed to determine requirements
29%	Not getting input from key users
29%	Team dynamics can help or hinder requirements determination
29%	Translation between IT and business is needed
29%	Users start with a preconceived notion about the solution
27%	Developers' lack of domain knowledge
27%	No formal organization-wide software development process
27%	Poor communication
27%	Users oversimplify requirements
25%	Fixed deadlines; deadlines impose change in requirements
25%	Lack of prototypes; users need a visual representation
25%	Users do not know what they want
25%	Users rush requirements definitions and create a quality problem

Table 20. Developers’ Top Voted-For Factors, Continued

Vote	Factor As Expressed by Participants
25%	Users/managers’ requirements leave out detail
24%	Adequate requirements documentation
24%	Organizational laziness with requirements determination
21%	Creating requirements is not a priority for customers; time is not allocated
21%	IT and users’ lack of listening skills
13%	Differences exist between users’ and developers’ viewpoint
13%	IT lacks business knowledge
13%	Lack of requirement documents; verbal versus written requirements
13%	Scope creep
13%	Users lack an understanding about the big picture

Step 4-Combined Top-X Factors for Developers

The most-voted-for factors from each developer group were thematically aggregated, which resulted in 10 DPFs. A thematic analysis identified similar meanings between the factors and derived a general meaning for these factors. Table 21 lists the factors, by organization, that were combined to form a more general description for similar factors.

Table 21. Combined Developer Factors with the Supporting Factors Expressed by Each Developer Group

ID	Name	Org	Combined Factor
D1	User Uncertainty		Users are uncertain about what they want and have difficulty articulating the problem and their needs
		1	<ul style="list-style-type: none"> • Users are not sure of what they want: vague about needs; do not understand the problem
		2	<ul style="list-style-type: none"> • Users do not know what they want
		3	<ul style="list-style-type: none"> • Users do not understand requirements and cannot articulate requirements
		3	<ul style="list-style-type: none"> • Oversimplifying requirements
		3	<ul style="list-style-type: none"> • Preconceived notion about solution

Table 21. *Combined Developer Factors with the Supporting Factors Expressed by Each Developer Group, Continued*

ID	Name	Org	Combined Factor
D2	Poor Communication		Poor communications result from making assumptions, leaving out details, not listening well, passing a message between too many people, and terminology differences between users and developers
		1	<ul style="list-style-type: none"> • Telephone game: users to business analysts to developers communications IT and users' lack of listening skills English is not first language
		2	<ul style="list-style-type: none"> • Assumptions made by developers and users
		2	<ul style="list-style-type: none"> • Users/managers' requirements leave out detail
		3	<ul style="list-style-type: none"> • Poor communication
D3	Requirements Reviews		Requirements prepared by developers are inadequately reviewed by users
		1	<ul style="list-style-type: none"> • Lack of requirements review
		2	<ul style="list-style-type: none"> • Lack of prototypes; users need a visual
D4	Different Perspectives		Users and developers have different perspectives, and a translator is needed
		1	<ul style="list-style-type: none"> • Developers-business analysts-user relationships can help or hinder requirements determination
		1	<ul style="list-style-type: none"> • Team dynamics can help or hinder requirements determination
		2	<ul style="list-style-type: none"> • Difference in users' and developers' viewpoint
		3	<ul style="list-style-type: none"> • Translation between IT and business
		3	<ul style="list-style-type: none"> • Technology is complicated
D5	Key users		Key users who have the information needed to determine requirements are not available or do not stay involved during the project
		1	<ul style="list-style-type: none"> • Not getting input from key users
		1	<ul style="list-style-type: none"> • Managers do not provide users with time for project work (i.e., defining requirements)
		1	<ul style="list-style-type: none"> • Creating requirements is not a priority for customers: time not allocated
		2	<ul style="list-style-type: none"> • Lack of ongoing user involvement during design
		2	<ul style="list-style-type: none"> • Key players not always involved in requirements determination
		3	<ul style="list-style-type: none"> • Business units do not create business plans
3	<ul style="list-style-type: none"> • Organizational laziness with requirements gathering 		
3	<ul style="list-style-type: none"> • Appropriate people not included (i.e., insufficient proxy) 		
D6	Deadlines		Users impose unreasonable schedules and rush requirements determination
		1	<ul style="list-style-type: none"> • Users rush requirements definitions and create a quality problem
		2	<ul style="list-style-type: none"> • Fixed deadlines; deadline imposes change in requirements

Table 21. *Combined Developer Factors with the Supporting Factors Expressed by Each Developer Group, Continued*

ID	Name	Org	Combined Factor
D7	Organizational Objectives		Users lack an understanding about how their immediate needs for an information system fit into their organizations' objectives and strategies
		2	<ul style="list-style-type: none"> • Users' lack of big picture understanding
		3	<ul style="list-style-type: none"> • Lack of strategic plans
		3	<ul style="list-style-type: none"> • Lack of enterprise planning; understanding the whole
D8	Ineffective Documentation		Users and developers do not maintain consistent and useful requirements documentation
		2	<ul style="list-style-type: none"> • Lack of requirement documents: verbal versus written
		2	<ul style="list-style-type: none"> • Formal versus informal requirements
		3	<ul style="list-style-type: none"> • Adequate documentation
D9	Developer Business Knowledge		Developers do not have sufficient knowledge about the organization's business
		2	<ul style="list-style-type: none"> • IT lacks business knowledge
		3	<ul style="list-style-type: none"> • Lack of domain knowledge
D10	Explicit Process		A predictable requirements determination process is not used or is not understood by users and developers
		2	<ul style="list-style-type: none"> • Scope creep
		3	<ul style="list-style-type: none"> • No formal organization-wide software development process

Step 5-Audio of Discussion for Developers

As was the case with the user groups, the developer focus groups were recorded using two digital audio recorders, and the audio recording was imported into a personal computer. The audio quality for each group was good, and almost all comments from participants were understandable.

Step 6-Transcription for Developers

The developers' discussions during the focus groups were transcribed and analyzed.

Step 7-Thematic Analysis for Developers

The transcribed discussions about factors were organized into common themes for each developer group. Frequently, the discussion about a factor produced multiple themes. Table 22 shows the total number of themes generated from the transcribed discussions conducted by each developer focus group.

Table 22. *Number of Themes Generated*

	Total
Org1	19
Org2	19
Org3	21

The 28 synthesized themes that emerged from the factors identified by all developer groups are shown in Table 23. More detail about the multiple dimensions represented in these themes is provided in Appendix N. Supporting concepts are listed after the theme, and the number of organizations that provided support for the theme is also provided. Support for the theme was implied based on the votes received by factors, calculated in the same way as the themes for the user groups. Only those themes that had non-zero support are presented.

Table 23. Synthesized Themes from Factors Identified by Developer Focus Groups

ID	Description of Theme	Number of Orgs	Support
DT1	Key users are not available or are not identified: The most useful users for requirements determination are the least available because they are too valuable doing their regular work	3	95%
DT2	Users do not know what they want: Users are uncertain and vague about their needs	3	64%
DT3	Big picture understanding: Users do not have a full grasp of the problem and how a solution fits into the organization's strategies and objectives	3	40%
DT4	Developers lack business knowledge: Developers do not understand the business domain and have little desire to learn about it	3	36%
DT5	Poor requirements documentation: Documentation is not maintained, contains conflicting requirements, and is difficult to use	3	32%
DT6	Requirements are not adequately reviewed: Users do not actively review requirements, and developers rely on email collaboration instead of in-person meetings	2	32%
DT7	Terminology: Developers are unfamiliar with users' terminology, and users are unfamiliar with developers' terminology	3	31%
DT8	Users are intimidated by developers: Some users are intimidated by some developers and do not freely share their ideas	2	29%
DT9	Translation is needed: Developers and users tend to speak different figurative languages, and a translator in the form of a business analyst is needed	2	28%
DT10	Insufficient detail: Users do not think about their requirements sufficiently to provide details needed by developers	2	26%
DT11	Telephone game (Chinese Whispers): Information is lost or distorted as it is passed from users to analysts to developers	1	24%
DT12	No standard development process: No consistent development process is in use or is not communicated and followed	1	21%
DT13	Past relationships: Positive relationships between users and developers aid requirements determination	2	18%
DT14	Users do not prioritize requirements work: Users have their normal work to complete, and requirements determination is not a priority	1	18%
DT15	Team members withhold opinions: Project team members may be reluctant to share information in meetings for fear of feeling stupid	1	17%
DT16	Users do not understand technology: Users do not care to know the details of technology and may find it intimidating	3	16%
DT17	Assumptions: Developers assume they understand the users' business, and users assume they know what developers are talking about	2	13%

Table 23. Synthesized Themes from Factors Identified by Developer Focus Groups, Continued

ID	Description of Theme	Number of Orgs	Support
DT18	Different perspectives: Developers, who are concerned with creating detailed software, have a different perspective or frame of reference than users	2	13%
DT19	The effect of time: Requirements evolve and are better understood over time	2	12%
DT20	Box thinking by users: Users think about information systems in terms of their past experience and do not realize there are other ways things can be done	2	12%
DT21	Personality differences: Users and developers process information differently, have different motivations, and work at different levels of detail	1	12%
DT22	Organizational culture: Culture influences how users and developers interact	1	11%
DT23	Language barrier: Requirements determination is complicated by non-native English-speaking team members	1	9%
DT24	Project deadlines: Deadlines cause requirements changes, scope changes, and resource availability changes	2	8%
DT25	Developers are arrogant: Developers may ignore users, falsify requirements, and would rather be coding than talking to users	2	6%
DT26	Users are resistant to change: A new information system disrupts users, and users dislike change	2	6%
DT27	Team dynamics: Team members want to spend time discussing the project with other people they like and are less likely to talk to someone about a question when the team dynamics are dysfunctional	1	6%
DT28	Developers lack listening skills: Developers are poor communicators and do not listen well	1	3%

Step 8-Qualitative Coding for Developers

In order to check the thematic analysis, the transcripts were manually coded to summarize the concepts shared by participants. A total of 46 codes were created, and 69% of the codes were shared between two or more groups (see Table 24).

Table 24. Percent of Codes in Common

	Percent
Codes shared by all three groups	39%
Codes shared by two groups	30%
Codes present in only one group	30%

A list of codes by factors was used to improve the previously created themes and account for missed factors, misplaced factors, and missing themes. The process ensured that the thematic analysis was an accurate synthesis of the participants' discussion.

Step 9-Synthesized Themes for Developers

Table 23 already reflects the corrective actions taken as a result of the qualitative analysis.

Step 10-Comparison/Contrast Analysis for Developers

The synthesized themes were compared to and contrasted with the DPFs. As with the user analysis, only themes with 20% or more support were considered in the developer analysis. Of the 12 themes with this support, only one theme was not included in the DPFs: *users are intimidated by developers* (29%). This theme emerged from the discussion about factors that received many votes from participants: for example, *the telephone game*, *appropriate people are not included*, and *team members withhold opinions or information*. However, participants did not specifically note that users are intimidated by developers. It was noted that intimidation can exist and that developers can be arrogant, but this was not captured in a developer-created factor.

One of the DPFs was not captured in the most-supported synthesized themes: *users impose unreasonable schedules and rush requirements determination*. The concept of schedule pressure surfaced in the discussion and is reflected in theme DT24 (*project deadlines*). However, this theme only received 8% support, and consequently, it was not considered because its support was less than 20%. The comparison of factors and themes is summarized in Table 25.

Table 25. Comparison of Developer Top-10 Factors and Synthesized Themes

ID	Top-10 Factor	Synthesized Theme	ID
D1	Users are uncertain about what they want and have difficulty articulating the problem and their needs	Users do not know what they want	DT2
D2	Developers and users communicate poorly	Terminology	DT7
		Telephone game (Chinese Whispers)	DT11
D3	Requirements prepared by developers are inadequately reviewed by users	Requirements are not adequately reviewed	DT6
D4	Users and developers have different perspectives, and a translator is needed	Translation is needed	DT9
D5	Key users who have the information needed to determine requirements are not available or do not stay involved during the project	Key users are not available or are not identified	DT1
D6	Users impose unreasonable schedules and rush requirements determination		N/A
D7	Users lack an understanding about how their need for an information system fits into their organization's objectives and strategies	Big picture understanding	DT3
D8	Users and developers do not maintain consistent and useful requirements documentation	Poor requirements documentation	DT5
		Insufficient detail	
D9	Developers do not have sufficient knowledge about the organization's business	Developers lack business knowledge	DT4
D10	A predictable requirements determination process is not used or is not understood by users and developers	No standard development process	DT12

The purpose of this comparison was to investigate whether the participants' discussion reflected their voting patterns. In general, the synthesized themes confirmed the importance of the most-voted-for factors (i.e., the DPFs), with 11 out of 12 themes receiving 20% or more support and accounting for 9 of the 10 top factors.

Step 11-Combined Factors for Survey/AHP Hierarchy for Developers

The original DPFs identified by the participants were used to form the developer's survey, which asked developers to rate the importance of each factor.

Step 12-AHP Survey for Developers

The purpose of the developer's survey was to determine the importance of each DPF. This ranking was determined using AHP and pairwise comparisons. The 10 factors had to be compared two at a time, and therefore, each participant compared 45 pairs of factors. Appendix H provides the content of the developer's survey.

Step 13-Survey Results for Developers

Participants were told at the start and end of the focus groups that they would be asked to participate in a web-based survey. An email invitation to complete the survey was personally sent to each participant. The survey was available for 6 weeks, and 23 of the 24 participants completed the survey. The 96% completion rate is not surprising because participants knew they would be asked to complete the follow-up survey after the focus groups. As with the user surveys, the developer survey responses were downloaded from the web-based survey provider and imported into ExpertChoice (2006), which was used to determine the importance of each factor.

Step 14-Importance of Factors/Hierarchy Weighting for Developers

Figure 17 shows the results of the AHP analysis. The data have been normalized so the factor (i.e., D5, *key users who have the information needed to determine requirements are not available or do not stay involved during the project*) judged most important is equal to 1.0. The importance of the other factors is relative to the most important factor: For example, D7, *users*

lack an understanding of how their need for an information system fits into their organizations' objectives and strategies has an importance of 0.48, which is nearly one half as important as D5.

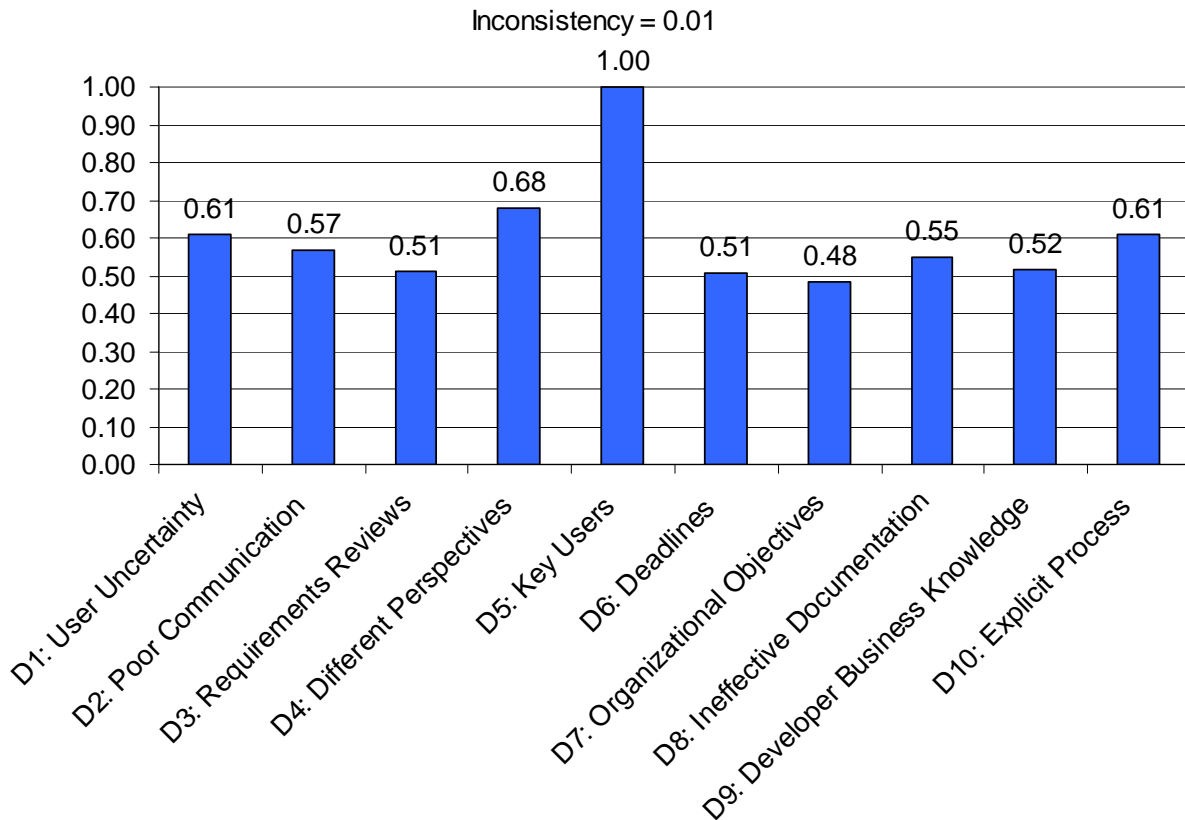


Figure 17. Importance of developer factors normalized to 1.0.

The calculated AHP weights for each DPF are shown on Figure 18. These weights show the amount of influence a factor has on misunderstandings about requirements.

In order to interpret the AHP weights and identify significant differences between them, the standard deviation around the mean of the weights was overlaid on the data (see Figure 19). The weights, which are equivalent to the importance of each factor, are within one standard deviation for 9 of the 10 top factors identified by developers; therefore, developers believe these nine factors are similar in importance. Only D5, *key users who have the information needed to*

determine requirements are not available or do not stay involved during the project appears more important than the other factors. No developer factor was significantly less important than the mean of the importance of the other factors.

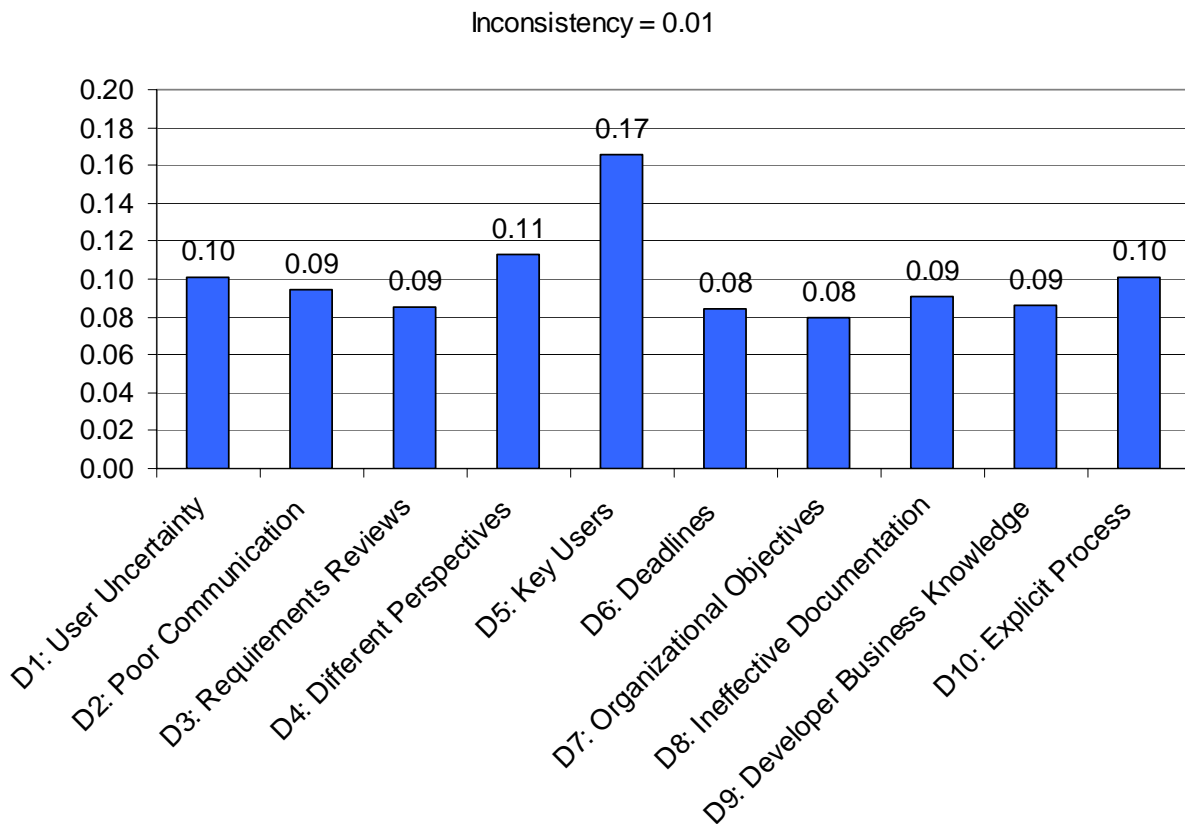


Figure 18. Importance of developer factors showing AHP weights summing to 1.0.

The results shown on Figure 18 have an inconsistency of 0.01, which is within Saaty’s (1999) recommendation of 0.10 or less. Appendix O shows the inconsistencies for each participant’s answers, which range from 0.02 to 0.96, with an arithmetic mean of 0.29. Consequently, although the aggregated inconsistency of all developer participants is very low, the individual inconsistency values are much higher. A closer examination of the data shows

that there was little consensus among participants. Some participants provided nearly opposite responses to other participants, which produces approximately the same mean values for each pairwise assessment. Therefore, the majority of factors are similar in importance and have a low inconsistency level, even though there was little consensus among participants. The lack of consensus among developers means that the AHP weights should be approached with a very skeptical attitude.

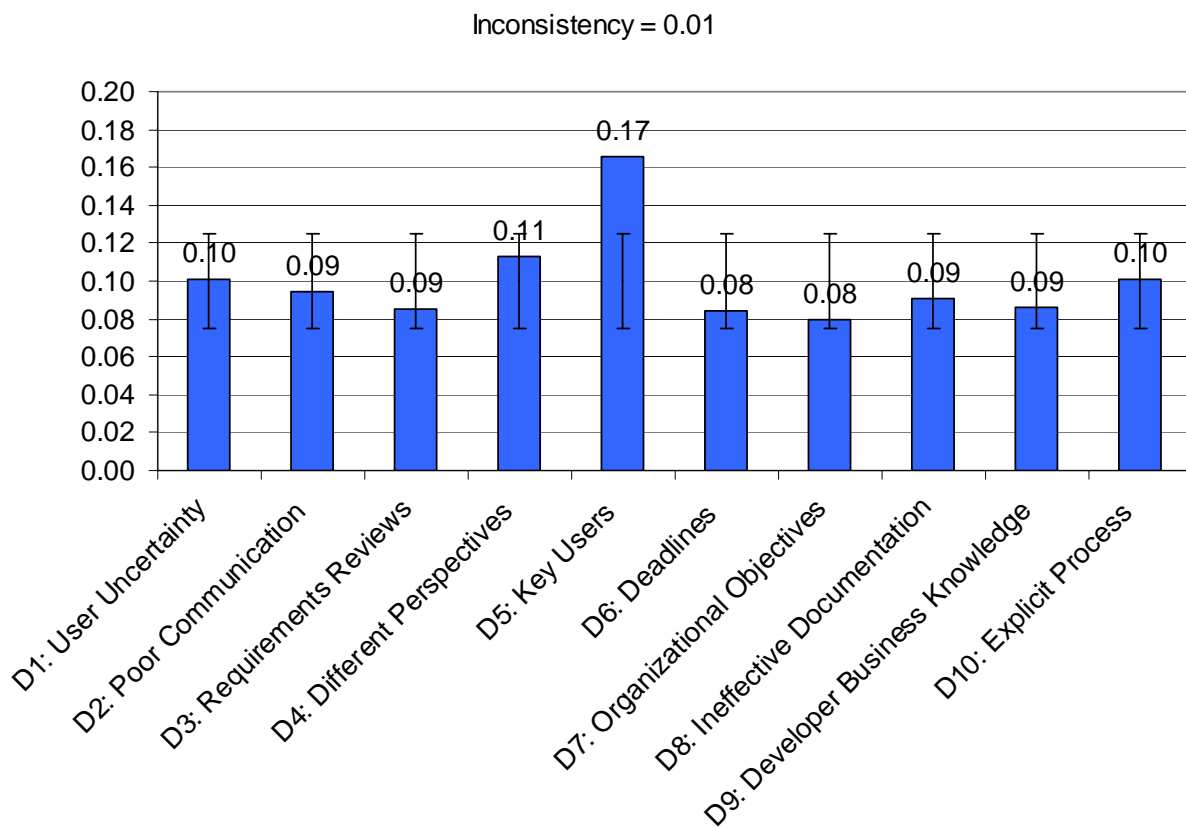


Figure 19. AHP weights for each DPF shown with standard deviation error bars.

Several participant subgroups were created in order to examine the impact of inconsistency on the importance of the factors. In this analysis, the aggregated results from

participants with individual inconsistencies of less than 0.67, 0.41, and 0.37 were compared to the results from all participants. The number of participants in each category is shown in Table 26.

Table 26. *Developers Divided into Groups Based on the Inconsistency of Their Responses*

Inconsistency Used to Create Group	Number of Participants	Inconsistency Value for Aggregate
All values	23	0.01
0.33 or less	17	0.01
0.20 or less	12	0.01
0.10 or less	5	0.01

The comparison of the aggregated results for each of these groups with decreasing inconsistency is shown on Figure 20. The pattern remains similar for each group, except for the 0.10 or less group, who weighted D1 and D2 more important than the other groups. The group rankings do vary, but each factor generally has a similar importance across the groups, with D5 clearly the most important. An exception is the group of participants who provided the most consistent responses and identified D5, D2, and D1 as most important, with the remaining factors of similar importance.

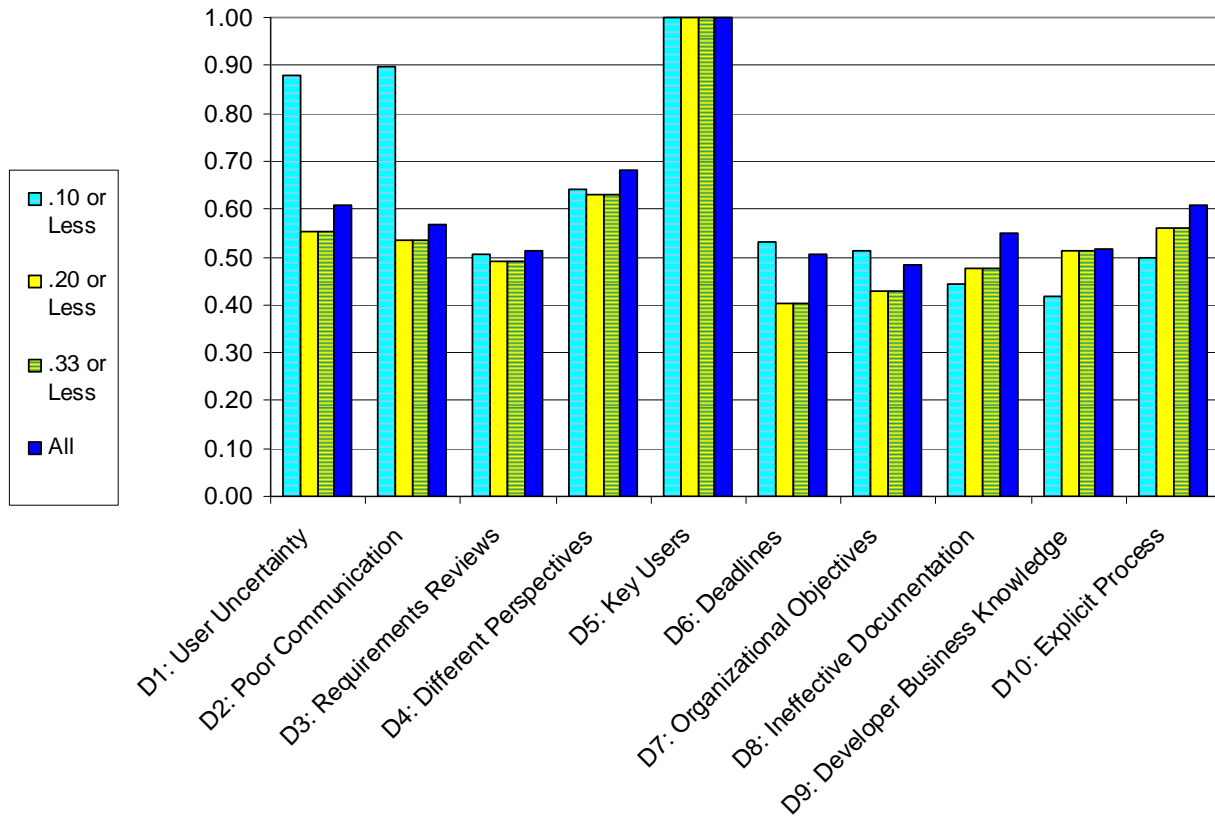


Figure 20. Comparison of importance based on groups with decreasing inconsistency values.

Details of the rankings shown on Figure 20 are provided in Table 27.

Table 27. Developer Factor Rankings Groups Based on Inconsistency

All		0.33		0.20		0.10	
Factor	Wgt	Factor	Wgt	Factor	Wgt	Factor	Wgt
D5: Key users	1.00	D5: Key users	1.00	D5: Key users	1.00	D5: Key users	1.00
D4: Different Perspectives	0.68	D4: Different Perspectives	0.63	D4: Different Perspectives	0.63	D2: Poor Communication	0.90
D1: User Uncertainty	0.61	D10: Explicit Process	0.56	D10: Explicit Process	0.56	D1: User Uncertainty	0.88
D10: Explicit Process	0.61	D1: User Uncertainty	0.55	D1: User Uncertainty	0.55	D4: Different Perspectives	0.64

Table 27. Developer Factor Rankings Groups Based on Inconsistency

All		0.33		0.20		0.10	
Factor	Wgt	Factor	Wgt	Factor	Wgt	Factor	Wgt
D2: Poor Communication	0.57	D2: Poor Communication	0.54	D2: Poor Communication	0.54	D6: Deadlines	0.53
D8: Ineffective Documentation	0.55	D9: Developer Business Knowledge	0.51	D9: Developer Business Knowledge	0.51	D7: Organizational Objectives	0.51
D9: Developer Business Knowledge	0.52	D3: Requirements Reviews	0.49	D3: Requirements Reviews	0.49	D3: Requirements Reviews	0.51
D3: Requirements Reviews	0.51	D8: Ineffective Documentation	0.47	D8: Ineffective Documentation	0.47	D10: Explicit Process	0.50
D6: Deadlines	0.51	D7: Organizational Objectives	0.43	D7: Organizational Objectives	0.43	D8: Ineffective Documentation	0.44
D7: Organizational Objectives	0.48	D6: Deadlines	0.40	D6: Deadlines	0.40	D9: Developer Business Knowledge	0.42

Note. Wgt is the AHP weight calculated from participants' pairwise comparisons.

Finally, the results were examined by organization. The inconsistency values for each organization is shown in Table 28, and the importance of factors are illustrated on Figure 21.

Table 28. Inconsistency Values by Organization for Developer Participants

Organization	Number of Participants	Inconsistency Value for Aggregate
Org1	7	0.02
Org2	8	0.03
Org3	9	0.02

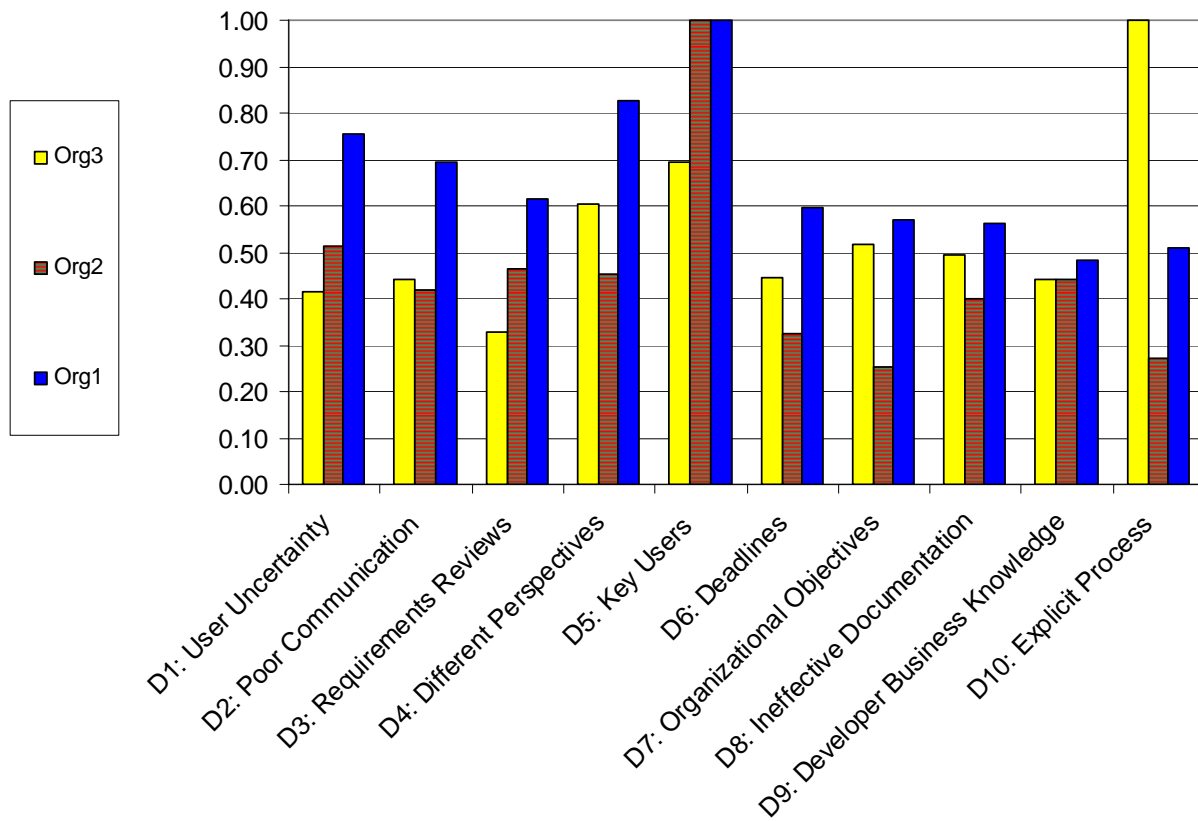


Figure 21. Importance of factors as judged by developers from each organization.

The details of the rankings by organization are shown in Table 29. Org1 and Org2 both identified D5 as the most important factor, while Org3 ranked this factor as the second-most important factor. Org3 identified D10 as the most important and D5 as the second-most important factor. At the time data were collected, Org3 participants had recently completed an organizational analysis, which may help explain their emphasis on D10 (*a predictable requirements determination process is not used or is not understood by users and developers*).

Table 29. Developer Factor Rankings by Organization

Org1		Org2		Org3	
Factor	Wgt	Factor	Wgt	Factor	Wgt
D5: Key Users	1.00	D5: Key Users	1.00	D10: Explicit Process	1.00
D4: Different Perspectives	0.83	D1: User Uncertainty	0.51	D5: Key Users	0.69
D1: User Uncertainty	0.75	D3: Requirements Reviews	0.46	D4: Different Perspectives	0.60
D2: Poor Communication	0.70	D4: Different Perspectives	0.45	D7: Organizational Objectives	0.52
D3: Requirements Reviews	0.62	D9: Developer Business Knowledge	0.44	D8: Ineffective Documentation	0.49
D6: Deadlines	0.60	D2: Poor Communication	0.42	D6: Deadlines	0.45
D7: Organizational Objectives	0.57	D8: Ineffective Documentation	0.40	D2: Poor Communication	0.44
D8: Ineffective Documentation	0.56	D6: Deadlines	0.32	D9: Developer Business Knowledge	0.44
D10: Explicit Process	0.51	D1: Explicit Process	0.27	D1: User Uncertainty	0.41
D9: Developer Business Knowledge	0.48	D7: Organizational Objectives	0.25	D3: Requirements Reviews	0.33

Note. Wgt is the AHP weight calculated from participants' pairwise comparisons.

Analysis of User and Developer Data

In this section, the following research questions are discussed:

1. How do the factors identified by users compare to those identified by developers?
2. Which factors identified by users were not identified by developers?
3. Which factors identified by developers were not identified by users?
4. What are possible explanations for the differences in the factors between users and developers?

Figure 22 illustrates an analysis that compared and contrasted the factors identified by users and developers. The analysis involved the three categories of data previously presented: (a) combined top-X factors (i.e., UPFs and DPFs), (b) themes, and (c) importance of factors. The remainder of this section deals with each of these categories.

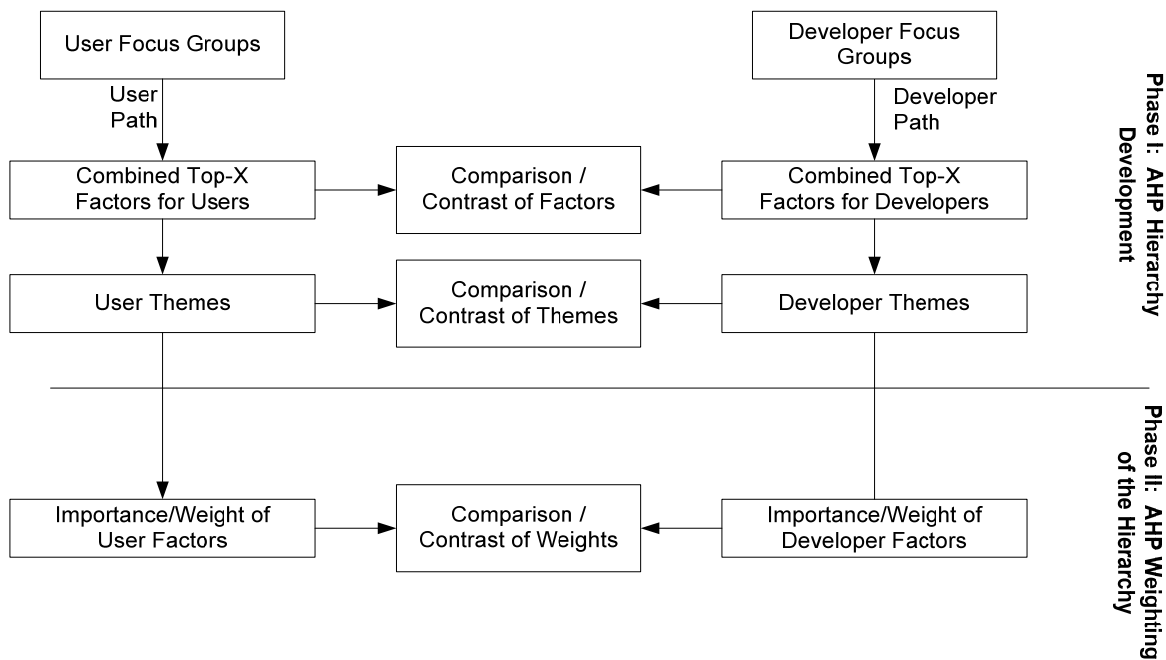


Figure 22. Comparing user and developer factors.

Analysis of UPFs and DPFs

Table 30 details the UPFs and DPFs and shows those factors identified by both users and developers, factors only identified by users, and factors only identified by developers. The factors are ordered from most-important factor to least-important factor by users. Users and developers identified five common factors, or 36% of the 14 top factors: (a) U9/D4, *users and developers have different frames of reference*; (b) U1/D5, *key users not involved in requirements determination*; (c) U7/D1, *users are unclear or uncertain about their needs*; (d) U3/D6, *project*

deadlines; and (e) U6/D9, *developers lack knowledge about the business*. All of these factors were described in similar ways by users and developers, except for the factor *project deadlines*. When users discussed *project deadlines*, they stated that adequate analyst or developer resources are not made available for requirements determination, and sometimes, an information system is not as functional as they want as a result of a project deadline. Developers stated that users rush requirements determination, which results in quality problems, and they are frustrated with user-imposed deadlines that do not account for the necessary engineering required to develop an information system.

Table 30. *Summary of Factors Ordered by Users*

User ID	User Description of Factor	Weight	Dev ID	Developer Description of Factor	Weight
U9: Different Perspectives	Users and developers relate to each other differently	0.18/1.0	D4: Different Perspectives	Users and developers have different perspectives, and a translator is needed	0.11/0.68
U8: User Past Experience	Users' experience with current systems limits their ability to create requirements for a new system	0.15/0.79	–	–	–
U1: Key users	Key users who have the information needed to determine requirements are not appropriately involved in requirements determination	0.13/0.68	D5: Key users	Key users who have the information needed to determine requirements are not available or do not stay involved during the project	0.17/1.0
U7: User Uncertainty	Users are unclear about their needs and the priority of those needs	0.13/0.68	D1: User Uncertainty	Users are uncertain about what they want and have difficulty articulating the problem and their needs	0.10/0.61
U5: Requirements Changes	Requirements change during the creation of an information system, and the changes are not adequately addressed	0.10/0.55	–	–	–

Table 30. Summary of Factors Ordered by Users, Continued

User ID	User Description of Factor	Weight	Dev ID	Developer Description of Factor	Weight
U3: Deadlines	Deadlines drive projects, leaving inadequate time for requirements determination with available resources	0.10/0.54	D6: Deadlines	Users impose unreasonable schedules and rush requirements determination	0.08/0.51
U6: Developer Business Knowledge	Developers/IT lack knowledge about the business	0.10/0.54	D9: Developer Business Knowledge	Developers do not have sufficient knowledge about the organization's business	0.09/0.53
U4: Roles	People do not understand their role and the roles of others in requirements determination	0.08/0.44	–	–	–
U2: User Technology Knowledge	Users do not know what is possible	0.05/0.25	–	–	–
–	–	–	D10: Explicit Process	A predictable requirements determination process is not used or is not understood by users and developers	0.01/0.61
–	–	–	D2: Poor Communication	Developers and users communicate poorly	0.09/0.57
–	–	–	D8: Ineffective Documentation	Users and developers do not maintain consistent and useful requirements documentation	0.09/0.55
–	–	–	D3: Requirements Reviews	Requirements prepared by developers are inadequately reviewed by users	0.09/0.51
–	–	–	D7: Organizational Objectives	Users lack an understanding about how their need for an information system fits into their organization's objectives and strategies	0.08/0.48

Note. – indicates no similar factor. The Weight column shows the AHP weight and then the AHP weight normalized to 1.0.

Table 31 shows the same information ordered from most-important factor to least-important factor by developers. Four of top-X factors were identified by users but not by

developers. Three of these four factors were self-abasing: (a) U2, *users do not know what they want*; (b) U4, *roles are not understood*; and (c) U8, *users' experience limits their thinking about new systems*. In the other factor (i.e., U5), users stated that *developers do not adequately address changes in requirements*.

Table 31 also shows that five top-X factors were identified by developers but not by users. In contrast to users, developers tended to place more responsibility on users for misunderstandings about requirements for an information system: (a) D3, *users inadequately review requirements* and (b) D7, *users do not understand how their needs are related to the organization's objectives*. Developers shared responsibility with users in three factors: (a) D2, *developers and users communicate poorly*; (b) D8, *users and developers do not maintain consistent and useful requirements documentation*; and (c) D10, *a predictable requirements determination process is not used or is not understood by users and developers*.

Table 31. Summary of Factors Ordered by Developers

User ID	User Description of Factor	Weight	Dev ID	Developer Description of Factor	Weight
U1: Key users	Key users who have the information needed to determine requirements are not appropriately involved in requirements determination	0.13/0.68	D5: Key users	Key users who have the information needed to determine requirements are not available or do not stay involved during the project	0.17/1.0
U9: Different Perspectives	Users and developers relate to each other differently	0.18/1.0	D4: Different Perspectives	Users and developers have different perspectives, and a translator is needed	0.11/0.68
U7: User Uncertainty	Users are unclear about their needs and the priority of those needs	0.13/0.68	D1: User Uncertainty	Users are uncertain about what they want and have difficulty articulating the problem and their needs	0.10/0.61
–	–	–	D10: Explicit Process	A predictable requirements determination process is not used or is not understood by users and developers	0.01/0.61

Table 31. Summary of Factors Ordered by Developers, Continued

User ID	User Description of Factor	Weight	Dev ID	Developer Description of Factor	Weight
–	–	–	D2: Poor Communication	Developers and users communicate poorly	0.09/0.57
–	–	–	D8: Ineffective Documentation	Users and developers do not maintain consistent and useful requirements documentation	0.09/0.55
U6: Developer Business Knowledge	Developers/IT lack knowledge about the business	0.10/0.54	D9: Developer Business Knowledge	Developers do not have sufficient knowledge about the organization's business	0.09/0.53
–	–	–	D3: Requirements Reviews	Requirements prepared by developers are inadequately reviewed by users	0.09/0.51
U3: Deadlines	Deadlines drive projects, leaving inadequate time for requirements determination with available resources	0.10/0.54	D6: Deadlines	Users impose unreasonable schedules and rush requirements determination	0.08/0.51
–	–	–	D7: Organizational Objectives	Users lack an understanding about how their need for an information system fits into their organization's objectives and strategies	0.08/0.48
U8: User Past Experience	Users' experience with current systems limits their ability to create requirements for a new system	0.15/0.79	–	–	–
U5: Requirements Changes	Requirements change during the creation of an information system, and the changes are not adequately addressed	0.10/0.55	–	–	–
U6: Developer Business Knowledge	Developers/IT lack knowledge about the business	0.10/0.54	D9: Developer Business Knowledge	Developers do not have sufficient knowledge about the organization's business	0.09/0.53

Table 31. Summary of Factors Ordered by Developers, Continued

User ID	User Description of Factor	Weight	Dev ID	Developer Description of Factor	Weight
U4: Roles	People do not understand their role and the roles of others in requirements determination	0.08/0.44	–	–	–
U2: User Technology Knowledge	Users do not know what is possible	0.05/0.25	–	–	–

Note. – indicates no similar factor. The Weight column shows the AHP weight and then the AHP weight normalized to 1.0.

Analysis of Themes

The themes synthesized from each focus group’s discussions support the top-*X* factors and also provide additional influencers on misunderstanding requirements. The themes that emerged from the user and developer transcripts are shown in Table 32. Users and developers had 15 themes in common, or 39% of the total of 39 themes. Another 11 themes were unique to users, and 13 themes were unique to developers. Users and developers described two themes (i.e., UT14/DT6, *poor requirements reviews*) in different ways. Users stated that they were not able to review requirements, developers did not discuss requirements documentation with them, and they did not know when developers and managers changed requirements. When developers discussed this theme, they stated that they were not involved in requirement reviews, users did not adequately review and understand requirements, and an approved set of requirements did not mean the requirements were correct. Developers also acknowledged an over-reliance on email to communicate with users and a desire to avoid interacting with users.

UT16/DT24 address *project deadlines* and were described in different ways by users and developers. This pair of themes was previously discussed in relation to a similar UPF/DPF pair listed in Table 30.

Table 32. Summary of Themes Ordered by User Support

User ID	User Theme	Support	Dev ID	Developer Theme	Support
User-only Themes					
UT5	Articulation difficulties	50%	–	–	–
UT12	Developers know better	22%	–	–	–
UT13	Unclear who is responsible for requirements	22%	–	–	–
UT17	Users lose hope	16%	–	–	–
UT18	Users are not equipped for requirements work	16%	–	–	–
UT19	Conflicting user needs	13%	–	–	–
UT20	Box thinking by developers	13%	–	–	–
UT22	Requirements are unexpectedly changed	12%	–	–	–
UT23	Design causes constraints	7%	–	–	–
UT24	Development starts before requirements are complete	7%	–	–	–
UT25	Information gets lost	6%	–	–	–
Both User and Developer Themes					
UT1	User-developer translation	60%	DT9	Translation is needed	28%
UT2	Developers lack understanding about the business	59%	DT4	Developers lack business knowledge	36%
UT3	Effect of time	58%	DT19	The effect of time	12%
UT4	Key users are not involved in requirements	50%	DT1	Key users are not available or are not identified	95%
UT6	User versus developer frame of reference	41%	DT18	Different perspectives	13%
UT7	Users do not understand technology	35%	DT16	Users do not understand technology	16%
UT8	Big picture understanding of the problem	34%	DT3	Big picture understanding	40%
UT9	Terminology difficulties	33%	DT7	Terminology	31%

Table 32. Summary of Themes Ordered by User Support, Continued

User ID	User Theme	Support	Dev ID	Developer Theme	Support
UT10	Assumptions	31%	DT17	Assumptions	13%
UT11	Box thinking by users	24%	DT20	Box thinking by users	12%
UT14	Poor requirements reviews	19%	DT6	Requirements are not adequately reviewed	32%
UT15	Requirements documentation is poor: Requirements documentation is not understandable or complete	17%	DT5	Poor requirements documentation	32%
UT16	Schedule drives projects: Deadlines drive projects and the functionality users receive	17%	DT24	Project deadlines	8%
UT21	Telephone game (Chinese Whispers)	12%	DT11	Telephone game (Chinese Whispers)	24%
UT26	Users do not prioritize requirement work	5%	DT14	Users do not prioritize requirements work	18%
Developer-only Themes					
–			DT2	Users do not know what they want	64%
–			DT8	Users are intimidated by developers	29%
–			DT10	Insufficient detail	26%
–			DT12	No standard development process	21%
–			DT13	Past relationships	18%
–			DT15	Team members withhold opinions	17%
–			DT21	Personality differences	12%
–			DT22	Organizational culture	11%
–			DT23	Language barrier	9%
–			DT25	Developers are arrogant	6%
–			DT26	Users are resistant to change	6%
–			DT27	Team dynamics	6%
–			DT28	Developers lack listening skills	3%

Note. – indicates no similar factor.

Of the 11 themes unique to users, four, or 36%, were considered users' responsibilities: (a) UT5, *users have difficulty articulating requirements*; (b) UT13, *users do not clearly know who is responsible for requirements*; (c) UT18, *users do not have requirements determination skills*; and (d) UT19, *users can create conflicting requirements*. The remaining 7 themes, or 64%, were considered developers' responsibilities.

Of the 14 themes unique to developers, two themes placed responsibility on developers: (a) DT25, *developers are arrogant* and (b) DT28, *developers lack listening skills*. Four of the themes place responsibility on users: (a) DT1, *key users are not available*; (b) DT2, *users do not know what they want*; (c) DT10, *users provide insufficient detail*; and (d) DT26, *users resist change*. The remaining seven themes place responsibility on developers and users.

Another analysis of themes was conducted to consider interdependences between themes. The themes are a surrogate for all of the factors generated by users and developers, not only the UPFs and DPFs. The purpose of this analysis was to examine if a theme, which may not be accounted for in a similar UPF or DPF, could have a significant impact on other themes and related UPFs and DPFs. These results were not central to the focus of the present study, but are deserving of further research. Appendix P presents this analysis.

Analysis of Weights

The importance of factors identified by users exhibited an inconsistency of 0.26, while the inconsistency of individual users ranged from 0.22 to 1.45. The high level of inconsistency indicates that the importance judgments were dubious, little consensus existed among users, and users had difficulty determining importance. However, the examination of various groups of inconsistencies, previously summarized in Table 17, clearly shows that U9, *users and developers*

relate to each other differently, is the most important factor to users. The least important factor is U2, *users do not know what is possible*.

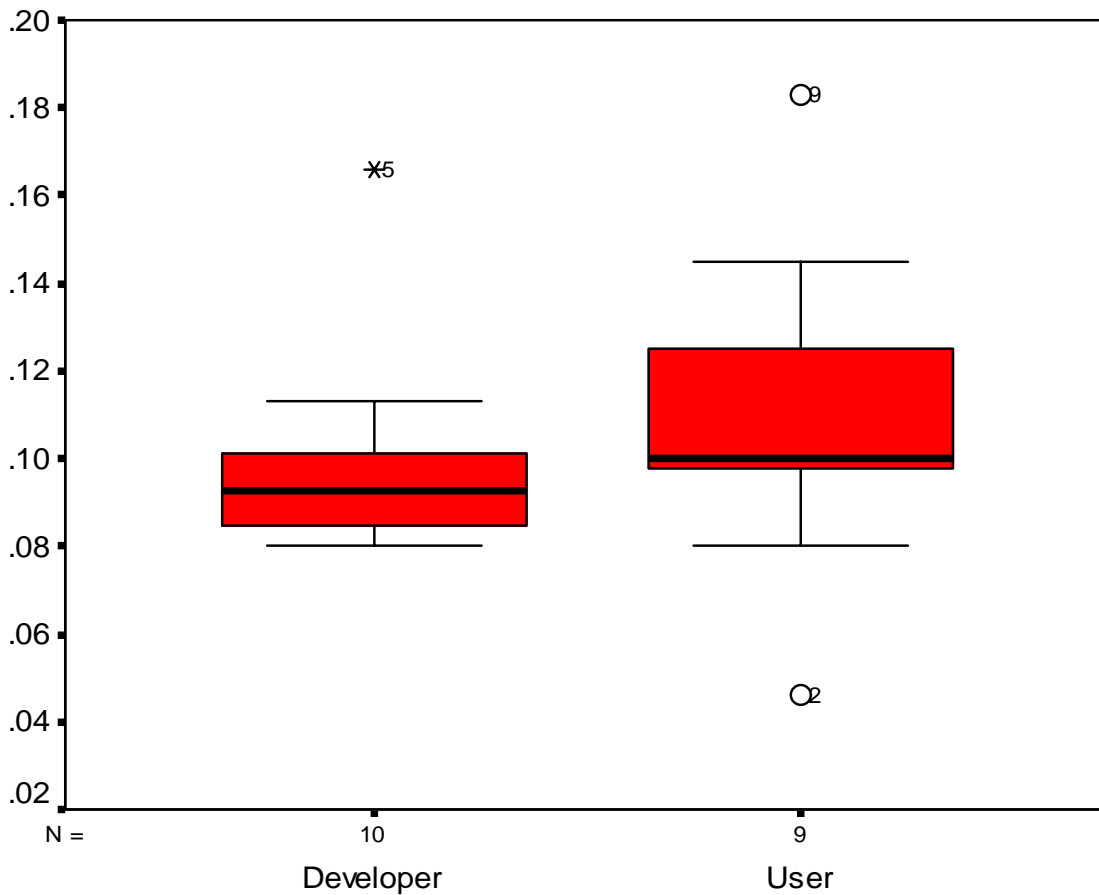


Figure 23. Box plot comparing AHP weights for developer and user factors.

In the case of developers, although the combined inconsistency of all developers was 0.01, there was little consensus among developers and their weighting of the factors is questionable. However, the most important factor was clear from the data: D5, *key users who have the information needed to determine requirements are not available or do not stay involved during the project*. The developers did not clearly identify a least important factor. Instead,

because of their lack of consensus and the use of the geometric mean in AHP, all the factors except for D5 were calculated to be of similar importance.

The box plot in Figure 23 concisely depicts the AHP weights for user and developer factors and is useful for quickly comparing the data. The developer factor weights are tightly grouped around the mean value, with the single outlier, D5, as the most important factor. The user factor weights are more spread out, and the mean is skewed towards the lower 50% of the weights. A most-important factor, U9, and a least-important factor, U2, are indicated by the outliers. Figure 24 summarizes the most- and least-important factors. Aside from the most and least important factors indicated by the data, the other weights should be skeptically considered because of the high inconsistency in participants' responses and their lack of consensus.

Most Important	U9: Users and developers relate to each other differently.	D5: Key users who have the information for determining requirements are not available or do not stay involved during the project.
Least Important	U2: Users don't know what is possible.	N/A: None judged as least important.
	Users	Developers

Figure 24. Most- and least-important factors identified by users and developers.

The descriptive statistics for the AHP weights are shown in Table 33. The percent difference was calculated as the change from the user value to the developer value. The mean for each is similar, as is the maximum weight, but the developer factors are more tightly located around their mean, with a smaller standard deviation and interquartile range.

Table 33. *Descriptive Statistics of AHP Weights*

Statistic	User Factors	Developer Factors	Percent Difference
Mean	0.1111	0.1001	-9.90
Minimum	0.046	0.08	73.91
Maximum	0.183	0.17	-7.10
Range	0.137	0.09	-34.31
Standard Deviation	0.039333	0.02521	-35.91
Interquartile Range	0.046	0.0193	-58.04

Possible Explanations for Differences Between User and Developer Factors

Users and developers identified five common factors that cause misunderstandings about the requirements for an information system but differed on the remaining nine factors. Although users identified factors that placed responsibility for misunderstandings about requirements on developers and developers identified factors that placed responsibility on users, they also frequently identified self-incriminating factors. As a result of these findings, it appears that both users and developers share responsibility for misunderstandings about requirements.

The results of the present study show that users and developers were reluctant to participate in requirements determination. Users are not trained for requirements work, already have full schedules, have found past information systems projects delivered disappointing results, and are hesitant to be associated with developing an information system over which they have little-to-no control. According to the users in the present study, developers are or should be responsible for requirements. Developers in the study recognized that they are not efficient communicators, do not understand the organization's business and do not care to learn more about it, and would rather be developing software code than working with users. According to

these developers, users are or should be responsible for providing requirements for developers. These findings show that users and developers are not motivated to participate in requirements determination and want someone else to be responsible for it.

The users and developers in the present study have different perspectives, and both users and developers recognize that they look at problems differently, enjoy different work, operate at different levels of detail, use different language, and require a translator. The users tend to be concerned with solving problems that relate to a business objective, while the developers tend to be concerned with solving problems that relate to software design and implementation. The users think about an information system project in terms of how it will change their work, which has a personal impact on them and may be positively or negatively perceived. The developers think in terms of what can and cannot be accomplished in a given timeframe with available resources. They may experience excitement or discomfort during the course of a project, but they are not affected the way users are after a system is developed. Instead, developers move on to the next project, with the exception of some maintenance work, while users must live with the information system for some time. These findings show that the fundamental difference between users' and developers' perspectives contributes to the differences in factors they consider important.

The results of the present study reveal conflicting views about the ability of business analysts to act as translators for users and developers. The two organizations that did not have business analysts consider them the answer to many requirements problems because analysts would be responsible for requirements determination and act as translators between the developers' and users' perspectives. However, the organization with a business analyst shared

many of the same problems as organizations that did not have business analysts. In addition, 57% of the developer group from this organization voted for the telephone game factor, selecting it as a most important influencer on misunderstanding requirements. The user group for the same organization stated that information was lost as it passed from users to analysts to developers. Therefore, the use of business analysts alone should not be expected to improve misunderstandings about requirements.

Developers approach requirements determination as a negotiation, and they expect to receive requirements from users and then negotiate in order to determine what requirements will actually be addressed. The negotiation may be based on several issues: development resources, schedule, technical feasibility, like or dislike for the people involved, like or dislike for the project, and the prospect of working with new technologies. Users never discussed any aspect of requirements determination in terms of a negotiation. They did state that developers will say what is or is not possible, but they also realize that it is not always true that a requirement is impossible. Therefore, users may lose trust in what developers tell them because users do not realize that they are in a negotiation.

Relationship to Factors Found in Literature

Several of the UPFs and DPFs relate to the factors found in literature. Further, the discussion from the focus groups support additional factors found in the relevant literature. Table 34 shows the relationship between the UPFs and DPFs and the factors from literature, previously shown in Table 4. The ID for the UPF and DPF is given when it is similar to the literature factor. Also, the Supported in Discussion column shows if discussions by users, developers, or both were similar to the literature factor. The absence of a relationship with a literature factor does not

necessarily suggest that the literature factor is invalid, only that it was not part of a focus group discussion.

Table 34. Factors Created in the Study Related to Factors Found in Literature

Category	Literature Factor	UPF	DPF	Supported in Discussion
F1. Developer Bias	F1.1. Developers view requirements in terms of modeling and analysis techniques.	N/A	N/A	No
	F1.2. Users are unfamiliar with modeling and analysis techniques used by developers.	N/A	N/A	Both
	F1.3. Software development methodologies used to create information systems assume users have already analyzed the requirements for the system.	N/A	N/A	User
	F1.4. Software engineering principles dominate requirements determination and result in technology-centric designs instead of user-centric designs.	N/A	N/A	User
	F1.5. Developers believe their work is more important than that of users.	N/A	N/A	User
F2. User Bias	F2.1. Users desire transparent development procedures, a common language to create mutual understanding, and successful collaboration.	N/A	N/A	Developer
	F2.2. A common framework is missing for users to effectively communicate with developers and neither party desires to learn the business of the other to aid communication.	N/A	N/A	Both
	F2.3. Users prefer requirements determination methods that do not interfere with their work.	N/A	N/A	Both
F3. Different Worlds	F3.1. Users and developers view the world through different conceptual frameworks, mental models, and perspectives.	U9	D4	Both
	F3.2. Developers lack understanding of the problem domain while users are vague about their needs.	U6	D9	Both
	F3.3. Users and developers tend to be associated with different personality types.	U9	D4	Both

Table 34. Factors Created in the Study Related to Factors Found in Literature, Continued

Category	Literature Factor	UPF	DPF	Supported in Discussion
	F3.4. A gap exists between users, who have a business-process perspective, and developers, who have a technical perspective, leading to different requirement determination processes.	U4	D10	Both
	F3.5. Conflict naturally exists between users and developers, in part because of negative perceptions one group has of the other.	N/A	N/A	No
	F3.6. Users and developers have different goals and motivations.	N/A	N/A	Both
	F3.7. Users and developers exhibit differences in language, experience, ambition, knowledge, and interest.	U9	D4	Both
	F3.8. Users and developers select different factors as important to the success of a project.	N/A	N/A	Both
	F3.9. Users and development managers select different risks as important to the development of an information system.	N/A	N/A	No
F4. Process	F4.1. Prematurely adopting a solution before the problem is well understood.	U3	D6	Both
	F4.2. Attempting to explain a poorly understood problem.	U7	D1	Both
	F4.3. The introduction of an information system may change the way the problem concept is understood.	U5	D1	Both
	F4.4. Difficulties articulating what is needed before seeing what is possible in the proper context.	U7	D1	Both
	F4.5. When developers are faced with missing requirements, they tend to create them based on their understanding of the problem.	N/A	N/A	Both
	F4.6. Users may resist if developers are driving the project while developers may regard the project as unimportant if users are driving the project.	N/A	N/A	Both
	F4.7. Complex patterns of interaction exists between users and developers.	U9	D4	Both
	F4.8. Technical communicators working as user advocates improve system success.	N/A	N/A	No

<i>Table 34. Factors Created in the Study Related to Factors Found in Literature, Continued</i>				
Category	Literature Factor	UPF	DPF	Supported in Discussion
F5. Communication	F5.1. Although requirements determination techniques require communication proficiencies, developers who are responsible for these techniques are not likely to be communication experts.	N/A	D2	Both
	F5.2. Developers and users must learn to communicate more efficiently, incorporating culture, context, and concept in their communications.	N/A	N/A	Both
	F5.3. User-developer interpersonal communications are the most important factor in the success of an information system.	N/A	N/A	Both
	F5.4. No single requirements determination technique solves all of the problems.	N/A	N/A	No
	F5.5. Users and developers need to frequently share and process information during requirements determination.	N/A	N/A	No
	F5.6. Effective communication is more important than specific requirements determination techniques.	N/A	N/A	Developer
	F5.7. Misunderstandings are most commonly the result of incompletely expressed information and differences in frame of reference between users and developers.	N/A	N/A	Both
	F5.8. Users and developers speak two different languages, using the same terms for different concepts or different terms for the same concepts.	N/A	N/A	Both
	F5.9. The chosen communication medium can hinder effective communication.	N/A	D8	Both
	F5.10. User-developer rapport impacts communication effectiveness.	N/A	N/A	Both
	F5.11. Negative feedback may not be shared.	N/A	N/A	Both
	F5.12. Developers insufficiently probe for clarification and ask for feedback from users.	N/A	N/A	Both
	F5.13. A properly trained facilitator can improve communication effectiveness.	N/A	N/A	No
	F5.14. The use of customer surrogates limits project success while increasing the number of elicitation techniques improves project success.	N/A	N/A	User
	F5.15. Developers tend to hinder open communications because they are defensive about their work.	N/A	N/A	User

Three of the UPFs were not discussed in the literature and include: (a) U1: *Key Users*, (b) U2: *User Technology Knowledge*, and (c) U8: *User Past Experiences*. Three of the DPFs were also not included: (a) D3: *Requirement Reviews*, (b) D5: *Key Users*, and (c) D7: *Organizational Objectives*. Table 35 summarizes these factors.

Table 35. User and Developer Factors Not Found In the Literature Review

User ID	User Description	Weight	Developer ID	Developer Description	Weight
U1: Key Users	Key users who have the information for determining requirements are not appropriately involved in requirements determination.	.13 / .68	D5: Key Users	Key users who have the information for determining requirements are not available or do not stay involved during the project.	.17 / 1.0
U2: User Technology Knowledge	Users do not know what is possible.	.05 / .25	—	—	—
U8: User Past Experience	Users' experience with current systems limits their ability to create requirements for a new system.	.15 / .79	—	—	—
—	—	—	D3: Requirement Reviews	Requirements prepared by developers are inadequately reviewed by users.	.09 / .51
—	—	—	D7: Organizational Objectives	Users lack an understanding of how their need for an information system fits into their organizations' objectives and strategy.	.08 / .48

Note. — indicates no similar factor. The Weight column shows the AHP weight and then the AHP weight normalized to 1.0.

Factors Related to Requirements Determination Techniques

One motivation for this research study was the acknowledgement that the plethora of requirements determination techniques has not significantly improved the problem of

misunderstanding requirements. The factors identified in this study provide a criteria for evaluating requirements determination techniques. The techniques previously described in Chapter 2 for requirements determination are discussed below in light of the UPFs and DPFs.

Quality Function Deployment and House of Quality. Developing an HOQ matrix strives to uncover user requirements provided by users and design requirements provided by developers. The combination of user and developer views is intended to improve the understanding of requirements. The HOQ provides a way of bridging the work done by users in specifying user requirements and the work developers do to turn requirements into a completed system. When conflicts are identified, users and developers work together to resolve them or use them as a point requiring innovation. The creation of an HOQ matrix would suffer from: key users not participating in the activity (i.e., U1 and D5); different perspectives of users and developers (i.e., U9 and D4); users being unclear or uncertain about their needs (i.e., U7 and D1); user's past experience limiting their ability to describe what best meets their needs (i.e., U8); an inadequate understanding of how the users' problem relates to the organizations objectives (i.e., D7); and others.

Interviews. Whether conducted by domain-experts with a deep understanding of the problem or domain-ignorant facilitators, interviews rely on gaining explicit knowledge from users. This most frequently used requirements determination technique is dependent on the skills, experience, and perspective of the interviewer as well as the skills, experience, and perspective of the interviewee. Consequently, this technique would be expected to be impacted by several factors, such as: key users not available to participate in interviews (i.e., U1 and D5); users constraining their answers because they do not know what is possible (i.e., U2); users are unclear about their needs when they are asked questions (i.e., U7 and D1); the past experience of users cloud the description of their needs (i.e., U8); users do not adequately review requirements

created from interviews (i.e., D3); and users do not relate their needs to the organization's objectives (i.e., D7).

Workshops and JAD. By convening key stakeholders for a few days to identify the problem, brainstorm solutions, and create a plan of action, workshops should be an effective requirements determination technique, and is regarded as very useful (Leffingwell & Widrig, 2000). They focus on users and developers interacting to better understand requirements. Although the purpose is to facilitate user and developer interaction, which is accomplished by getting them together in the same room, workshops would suffer from the similar issues as HOQ. Both workshops and HOQ are facilitated meetings with users and developers and would be subject to problems from not involving key users, encountering very different perspectives, users not clearly sharing their needs, and so forth.

Prototypes. The use of paper user interface mock-ups, computer presentations, and other representations of an information system's features provides users visual feedback of the requirements. Prototypes enable developers to present more information than written descriptions of requirements. The use of prototypes would encounter problems getting time with the best, most valuable users (i.e., U1 and D5); developers not fully understanding feedback from users because of their lack of business knowledge (i.e., U6 and D9); users constraining their feedback based on their past experiences with systems (i.e., U8); not recognizing misinterpretations due to users and developers approaching the problem from different perspectives (i.e., D4); and users not clearly knowing how their needs relate to other business needs and objectives (i.e., D7).

Scenarios. These real-world stories from a user's perspective describe how a system should work in the words of users. Scenarios are similar to interviews in the respect that each technique is directly asking users to provide a narrative of their needs. Consequently, scenarios would be expected to encounter problems with the same factors as interviews.

Observations. By watching what users do, observations are a rapid way to understand users' tasks, objectives, and expectations in the context of their work environment. Observations are unique compared to the other requirements determination techniques discussed here because they do not require users to change their work habits, make time for requirements determination, or articulate their needs. Consequently, observations would be expected to encounter few problems with factors for misunderstanding requirements. Observations, along with the other techniques, do not implicitly deal with requirements that change over the course of time (i.e., U5) and could be expected to encounter problems with reviewing requirements (i.e., D3).

User Advocate. Recognizing that users and developers have different perspectives, use different terminology, and have difficulty relating to each other, employing a user advocate can help bridge the gap between users and developers. A user advocate improves the success of other requirements determination techniques and is not used in isolation. For example, the person performing the user advocate role may choose to use interviews, prototypes, and other techniques for determining requirements. A user advocate would still encounter issues with gaining access to key users (i.e., U1 and D5); dealing with users who do not have a clear understanding of their needs (i.e., U7 and D1); finding requirements overly influenced by users' past experiences with systems (i.e., U8); and not sufficiently understanding how the users' needs relate to the organization's strategy (i.e., D7).

The factors and themes that are expected to negatively impact these requirements determination techniques are summarized in Table 36. One observation from this table is that all of the requirements determination techniques are negatively impacted by several factors. This suggests that multiple techniques used in conjunction would reduce these negative impacts. The selection of an improved set of techniques could be accomplished by choosing complimentary techniques that compensate for each other's weaknesses. This can be visually determined from the table.

As an example, HOQ and workshops, which are both facilitated group meetings, suffer from negative effects of a nearly identical set of factors and themes. Interviews and Scenarios are also similar in nature and impacted by the same factors and themes. User advocate has its own profile that does not include as many weaknesses as the other techniques, with the exception of observations, because it seeks to be a bridge between users and developers to improve requirements understanding. However, user advocate would be expected to be most susceptible from UT21 and DT11: *Telephone Game* –related problems because of the emphasis on third party facilitation. Observations stand out as being effected by the least number of factors and themes, and is the only technique that is not impacted by the reluctance of key users to be involved in requirements determination (i.e., U1 and D5). As with user advocate though, observations would suffer from *Telephone Game* issues as skilled observers are needed to gather ethnographic information and translate it into something useful to users and developers.

Consequently, from Table 36, combining observations with any of the other techniques would be expected to reduce requirements misunderstanding. An appealing combination of three techniques would be interviews or scenarios with observations and prototypes.

Table 36. *Expected Issues With Requirements Determination Techniques Based on User and Developer Top Factors*

Factor	HOQ	Interviews	Workshops	Prototypes	Scenarios	Observations	User Advocate
U1: Key Users	X	X	X	X	X		X
U2: User Technology Knowledge		X			X		
U3: Deadlines							
U4: Roles							
U5: Requirements Change	X	X	X	X	X	X	X
U6: Developer Business Knowledge	X		X	X			

Table 36. Expected Issues With Requirements Determination Techniques Based on User and Developer Top Factors, Continued

Factor	HOQ	Interviews	Workshops	Prototypes	Scenarios	Observations	User Advocate
U7: User Uncertainty	X	X	X		X		X
U8: User Past Experience	X	X	X	X	X		X
U9: Different Perspectives	X		X	X			
D1: User Uncertainty	X		X				X
D2: Poor Communication	X		X	X			
D3: Requirement Reviews	X	X			X	X	
D4: Different Perspectives	X		X	X			
D5: Key Users	X	X	X	X	X		X
D6: Deadlines							
D7: Organizational Objectives	X	X	X	X	X		X
D8: Ineffective Documentation							
D9: Developer Business Knowledge	X		X	X			
D10: Explicit Process							

CHAPTER 5. RESULTS, CONCLUSIONS, AND RECOMMENDATIONS

Summary and Discussion of Results

Summary of the Problem and Methodology

This study identified factors that influence the misunderstanding of requirements during the development of an information system. Other studies have shown that the misunderstanding of requirements is one of the leading contributors to requirement errors and the development of software that does not meet customers' needs, exceeds budget, and is delivered late. A review of the information systems and product development literature found several proposals for why misunderstandings occur, but little empirical evidence as to the causes. Consequently, this study sought to produce knowledge about misunderstanding requirements to provide the beginning of a theoretical foundation for greater understanding and future research.

Three primary research questions were:

1. Which factors do users and developers believe cause misunderstandings about the requirements for information systems?
2. Which factors do users and developers believe have the most impact on misunderstandings?
3. What is the difference between users' and developers' perceptions of these factors?

These questions were answered in two phases of research. In Phase I, focus groups of developers and users of information systems generated factors that influence misunderstandings. Three organizations participated in the study, with each providing a focus group of users and a focus group of developers, resulting in a total of six groups. The discussion in the focus groups

was structured using the nominal group technique to generate the factors and then select the most influential factors from those generated. Twenty-two users and 24 developers participated in the focus groups. Question 1 was addressed by each user group, generating between 28 and 32 factors and creating nine aggregated factors as having the most influence. The developer groups generated between 30 and 48 factors, creating 10 aggregated factors. Analytical Hierarchy Process (AHP) was used as an analysis tool in both Phase I and II. In terms of an AHP analysis, Phase I created the AHP hierarchy, which consisted of the factors organized in a single level and their impact on misunderstanding requirements as the single alternative.

In Phase II the participants from Phase I completed a survey asking them to use pairwise comparisons to rank and weigh the importance of the most influential factors aggregated from all three organizations. Users made judgments about the users' most influential factors and developers made judgments about the developers' most influential factors. Question 2 was addressed using AHP to analyze the survey results. In AHP terms, this resulted in weights for each factor that explain how much a factor contributes to misunderstanding requirements. Question 3, differences between users and developers factors, were examined using both the qualitative Phase I data and the quantitative Phase II data.

Summary of Findings

Table 37 presents the findings from Phase I and II for the 14 most influential factors that contribute to misunderstanding requirements from the perceptions of users (i.e., User Perceived Factors—UPFs) and developers (i.e., Developer Perceived Factors—DPFs).

This table also provides insight into the third research question by showing which of the top factors were described by both developers and users, and if they described the factor in a similar manner or an opposing manner. Further, it shows which factors were unique to users and which were unique to developers. Although all of these factors are important because they were

selected by users and developers from a much larger list of factors, the AHP weights provide a relative means of importance.

Table 37. *UPFs and DPFs Ordered by User Weight*

UPFs					DPFs				
ID	Factor	Description	Wgt	Rnk	Factor	Description	Wgt	Rnk	
1	Different Perspectives (U9)	Users and developers relate to each other differently.	0.18	1.0	Different Perspectives (D4)	Users and developers have different perspectives and translation is necessary for one group to understand the other.	0.11	0.68	
2	User Past Experience (U8)	Users' experience with current systems limits their ability to create requirements for a new system.	0.15	0.79	–	–	–	–	
3	Key Users (U1)	Key users who have the information for determining requirements are not appropriately involved in requirements determination.	0.13	0.68	Key Users (D5)	Key users who have the information for determining requirements are not available or do not stay involved during the project.	0.17	1.0	
4	User Uncertainty (U7)	Users are unclear about their needs and the priorities of those needs.	0.13	0.68	User Uncertainty (D1)	Users are uncertain about what they want and have difficulty articulating the problem and their needs	0.10	0.61	
5	Requirements Change (U5)	Requirements change during the creation of an information system and the changes are not adequately addressed.	0.10	0.55	–	–	–	–	
6	Deadlines (U3)	Deadlines drive projects, leaving inadequate time for requirements determination with the resources available.	0.10	0.54	Deadlines (D6)	Users impose unreasonable schedules and rush requirements determination.	0.08	0.51	
7	Developer Business Knowledge (U6)	Developers/IT lack knowledge of the business.	0.10	0.54	Developer Business Knowledge (D9)	Developers do not have sufficient knowledge of the organization's business.	0.09	0.53	

Table 37. *UPFs and DPFs Ordered by User Weight, Continued*

ID	UPFs				DPFs			
	Factor	Description	Wgt	Rnk	Factor	Description	Wgt	Rnk
8	Roles (U4)	People do not understand their role and the roles of others in requirements determination.	0.08	0.44	–	–	–	–
9	User Technology Knowledge (U2)	Users do not know what is possible.	0.05	0.25	–	–	–	–
10	–	–	–	–	Explicit Process (D10)	A predictable requirements determination process is not used or is not understood by users and developers.	0.01	0.61
11	–	–	–	–	Poor Communication (D2)	Developers and users communicate poorly.	0.09	0.57
12	–	–	–	–	Ineffective Documentation (D8)	Users and developers do not maintain consistent and useful requirements documentation.	0.09	0.55
13	–	–	–	–	Requirement Reviews (D3)	Requirements prepared by developers are inadequately reviewed by users.	0.09	0.51
14	–	–	–	–	Organizational Objectives (D7)	Users lack an understanding of how their need for an information system fits into their organizations' objectives and strategy.	0.08	0.48

Note. – indicates no similar factor. Wgt is the AHP weight. The weights for all UPFs sum to 1 as do all DPFs. The value reflects the amount a factor is responsible for the problem of misunderstanding requirements. Rnk is the rank of a UPF when compared to all other UPFs or the rank of a DPF when compared to all other DPFs. A rank of 1.0 signifies the most important factor and values less than 1.0 indicate the relative importance of a factor to the most important factor.

The weights convey the amount that a UPF or DPF is responsible for the problem of misunderstanding requirements. For example, a factor with a weight of 0.15 would be have been judged by the participants as being responsible for 15% of the problem.

Based on the AHP weights, one UPF and one DPF stood out as most important and one UPF as least important. The most important factors are highlighted with green in the table (darker gray when printed) and the least important factor is highlighted in yellow (light gray when printed). The most important UPF is Different Perspectives (U9), which was also judged to be important to developers. The most important DPF is Key Users (D5), which was also judged to be important to users. The least important UPF is User Technology Knowledge (U2), which did not have a similar DPF. Developers did not judge any of the DPFs to be significantly less important than the others. Figure 25 summarizes the amount of responsibility each UPF and DPF has for misunderstanding requirements.

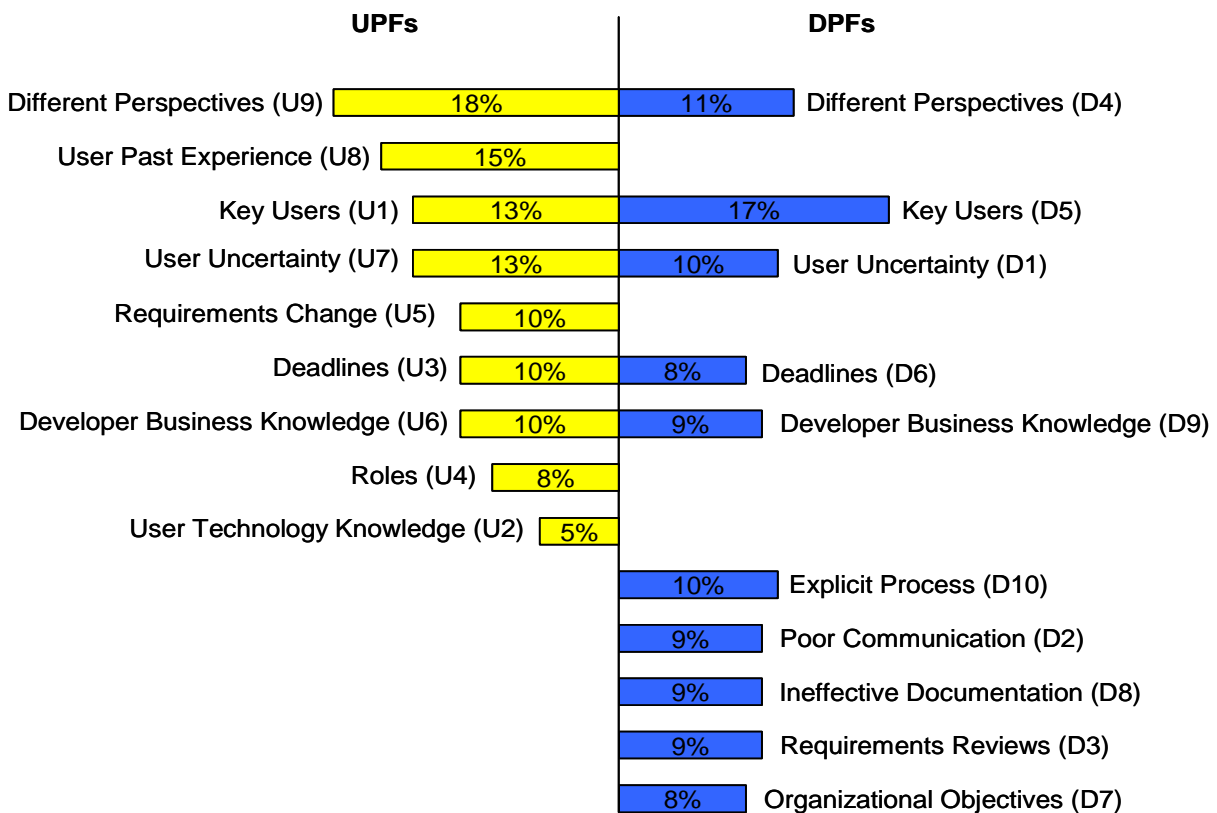


Figure 25. Summary of AHP weights for each UPF and DPF.

These UPFs and DPFs are the critical pieces of information needed to complete the graphical conceptual framework first presented on Figure 1. From the present study, the factors that contribute to misunderstanding requirements are shown added to the conceptual framework on Figure 26.

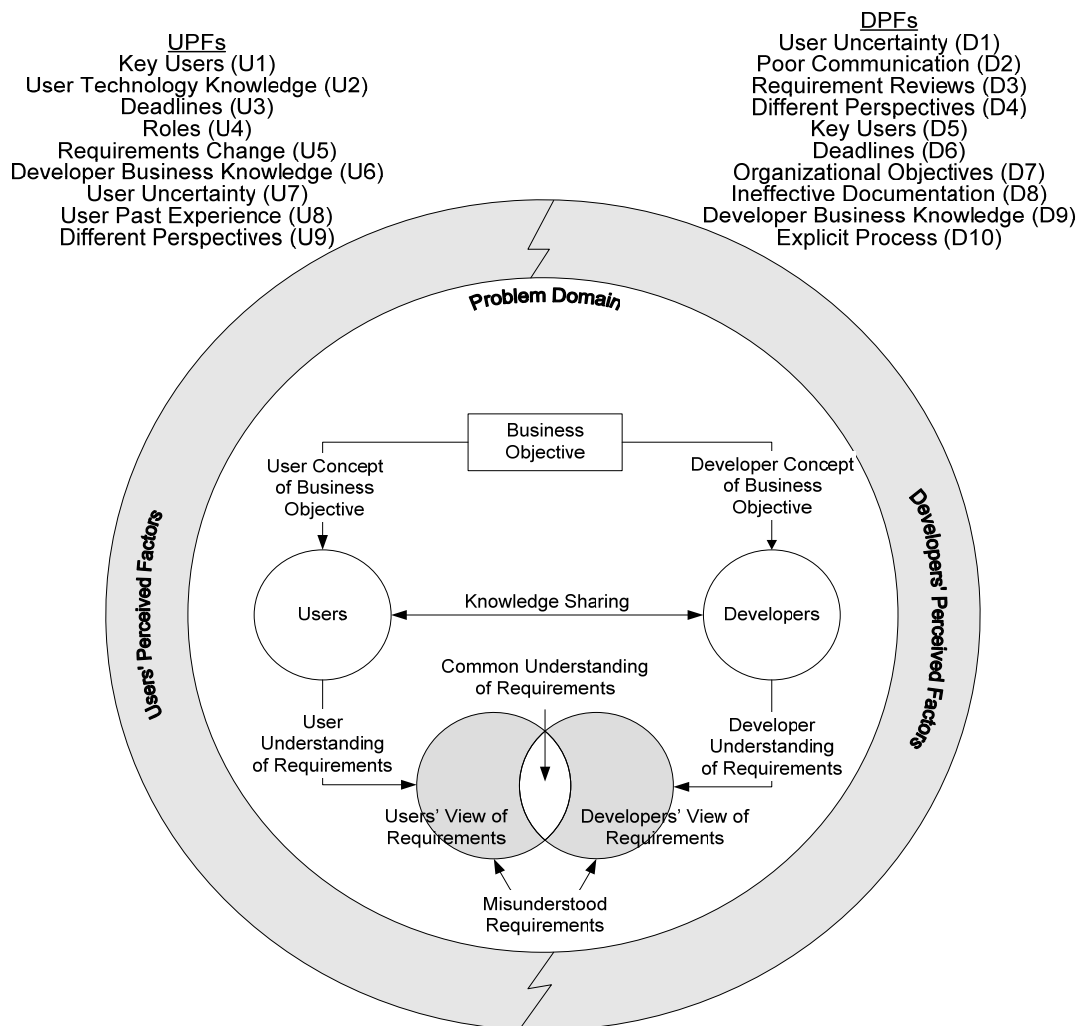


Figure 26. Graphical conceptual framework of the research problem with UPFs and DPFs listed.

Discussion of the Results: UPFs and DPFs

The key findings of this research study are the UPFs and DPFs, which are the factors judged as most important for contributing to the problem of misunderstanding requirements. The researcher expected some degree of finger pointing between users and developers, blaming each other for misunderstanding requirements. Instead, little finger pointing was observed. Both users and developers maintained a balance of identifying self-incriminating factors as well as identifying problems with the other group. They also tended to agree on the factors more than they disagreed.

Many similarities existed between the factors discussed by users and developers. Of the nine UPFs and the ten DPFs, five factors were in common and four of these were defined similarly. Both users and developers identified the following factors:

1. Key users are not adequately involved in requirements determination.
2. Developers lack sufficient knowledge of the business to better understand the needs of users.
3. Users are unclear and uncertain about their needs for information systems.
4. Users and developers have different perspectives, operating from different frames of reference.
5. Deadlines drive project schedules and negatively impact requirements. Users perceived developers' use of deadlines as an excuse to justify spending insufficient time on requirements determination. Developers viewed deadlines as being unreasonably imposed by users before they adequately considered the engineering effort.

Developers provided more consistent responses when judging the importance of factors than users, and did so even though they were assessing one more factor, which added nine

questions to their survey. The mean consistency ratio for developers was 0.29 while it was 0.64 for users. Also, each user group shared some form of hesitation about their developers not being very good focus group participants because they tend to not talk very much. On the contrary, the developer groups consistently could have used additional discussion time and generated more factors than the user groups. This may be because developers are immersed in the problem of creating useful information systems, which begins with clear requirements, while users only occasionally encounter this problem.

The key findings shown in Table 37 are discussed below, ordered by the ID shown in the table.

1 – Different Perspectives (U9) / Different Perspectives (D4). Both user and developer focus groups generated a factor that discussed how users and developers have different perspectives on the development of information systems that cause difficulties relating to each other. Because of the differences, translation was perceived as being necessary to help developers and users better understand each other. Users shared that developers have a completely different way of looking at a problem. Developers shared that the difference in perspective is necessary because they are accountable for a solution to the users' problems, while users only need to cope with the problem until it is solved.

Another aspect of a difference in perspectives was voiced as the arrogance of developers (supported by Guinan and Bostrom, 1984). Interestingly it was not voiced by users. Developers freely discussed in each developer focus group that they can be arrogant or at least can be perceived as being arrogant. Users did not initiate such a discussion. Instead, when the researcher asked one focus group of users if developers are arrogant, not only did they reject the notion, they defended developers and offered reasons why they may appear arrogant at times, such as being very busy. Further, in a developer focus group, their arrogance was emphasized as a

serious factor that contributes to misunderstandings, but they later chose other factors as being important, never elevating developer arrogance to a DPF.

Both users and developers suggested that a translator is needed to bridge the gap in perspective between users and developers and that a business analyst would be helpful. However, as is discussed later in item 11 – Poor Communication, a business analyst may not be able to minimize the impacts of Different Perspectives. Instead of a translator, which tends to isolate users from developers, a facilitator that engages users and developers may be more appropriate. A facilitator can help users better understand the perspective of developers and vice versa. This is analogous to a trained Meyers-Briggs Type Indicator (MBTI) facilitator that can help people of one temperament better work with people of another temperament. Likewise, a facilitator can help users and developers learn to relate to each other better by understanding each other's perspectives just as one can learn to better relate to someone with a different temperament.

Users weighted Different Perspectives as the most important of their nine factors, being responsible for 18% of the problem of misunderstanding requirements. Developers also weighted Different Perspectives highly, being 11% of the problem and ranked as 68% as important as their most important factor. The concept of users and developers having different perspectives has been previously suggested. Users and developers view the world through different conceptual frameworks, have differing mental models, and diverge in their perspectives (Bostrom and Thomas, 1983; Guinan and Bostrom, 1984; Stary, 2002; Kudikyala and Vaughn, 2005).

2 – User Past Experience (U8). Users recognized that their past experience impacts their thinking about requirements for a new information system. Users do not start with a blank sheet of paper when thinking about an information system. Instead, what they have seen and used in other information systems influences how they think about their current needs. They can more

easily consider how to automate and enhance what they currently do than to creatively consider different and potentially better ways of accomplishing their work.

Addressing this issue requires helping users look at their problem from a different perspective, which is what developers do. As in 1–Different Perspectives, a trained facilitator may help users consider alternative ways of approaching their problem as they discuss it with developers. Also, the use of brainstorming and creative thinking techniques could prove useful.

User Past Experience was judged to be responsible for 15% of misunderstanding requirements and ranks as 79% as important as the most important UPF. This factor was not directly discovered in existing requirements determination literature, but is indirectly supported in the practice of prototypes (Leffingwell & Widrig, 2000; Liu & Khooshabeh, 2003).

3 – Key Users (UI) / Key Users (D5). All of the focus groups discussed the importance of involving users with the proper experience and skills that can describe the needs for an information system. Although the involvement of key users was judged to be important, it frequently does not happen. Several reasons were identified for why key users are not available, such as their time is too valuable, they have their own procedures for being successful and do not need another information system, they have had poor experiences helping with past information systems, and they do not easily embrace change. One organization shared that users were asked to volunteer to aid in developing an information system but that they would lose their jobs at the end of the project because the new system would make them obsolete. All of these reasons point to motivation as a common dimension of key users' reluctance to participate in requirements determination; they are not personally or professionally motivated to do so.

As discussed in Chapter 4, using observations as part of a requirements determination process is a means of negating the impact of key users not participating. Key users can continue their normal work while they are observed performing tasks, which can be analyzed for

requirements. The use of observations helps to diminish the active role users must play in requirements determination.

Developers overwhelmingly selected Key Users as their most important factor, judging it to be responsible for 17% of the problem of misunderstanding requirements; 64% higher than the next most important factor with a weight of 0.11. It was also important to users with a weight of 0.13, making it their third most important factor and 68% as important as the most important factor users selected. Although the importance of having key or representative users involved in requirements determination is recognized by practitioners and the role of users in the development of information systems has been previously explored (Hartwick & Barki, 1994), the emphasis developers placed on the value of this factor does not appear to be reflected in literature.

4 – User Uncertainty (U7) / User Uncertainty (D1). An often-heard phrase from developers is that users do not know what they want, so it comes as no surprise that developers would select users being uncertain about their needs as an important factor. What was surprising is that users agreed that they are both unclear about their needs and they fail to set priorities. Users shared that they poorly articulate requirements, are not clear about the differences between a must-have and a like-to-have need, and encounter conflicts with other users when defining requirements. Developers shared that users do not articulate requirements well, are vague about their needs, lack a full understanding of the problem, and oversimplify their requirements.

As was the case with 2 – User Past Experience (U8), the use of brainstorming, problem solving, and creative thinking techniques may help users be more clear about their problem and requirements. Experience suggests that developers participating in these techniques with users would help developers better understand the problem from the users' perspective.

Users judged this UPF as being responsible for 13% of misunderstanding requirements and 68% as important as their most important factor. Developers judged it to be responsible for

10% of misunderstanding requirements and 61% as important as their most important factor. The difficulties involved with describing and explaining a poorly understood problem, which contributes to users' uncertainty, have been discussed by Mrenak (1990). Further, literature has addressed problems users have articulating their needs before seeing what is possible in their context (Lamswerde, 2000; Kazmierczak et al., 2000; Saiedian and Dale, 2000).

5 – Requirements Change (U5). Requirements can change for many reasons: (a) users learn more about the problem and evolve how they think about their needs, (b) business objectives change in response to market or competition changes, (c) the scope of the system is reduced because of resource contention or a change in priorities, (d) managers modify the requirements, (e) personnel on the project team change, and the like. A contributing dimension to changes in requirements is the effect of time. The longer it takes to develop an information system, the more likely changes will occur. Users in the focus groups expect requirements to change over the course of a project, but are not confident that developers will respond to requested changes. Instead, they expect to be told that changes to the requirements will be prioritized in the second release of the system and that the requirements for the initial release are frozen. Developers did not have a related factor. This is likely because developers have an effective way to deal with changes, which is to defer them until another release is planned.

Users expressed this as an important factor even though several software development processes were in use, including incremental processes that have provisions for handling changing requirements. Perhaps different management of an information systems project is needed to better incorporate changes to requirements while the system is under construction. For example, SCRUM is a software management approach that is frequently adapted to agile development processes (Schwaber, 2004). One of the benefits of SCRUM is developing working code in 30-day increments, with a review of requirements at the start of each 30-day cycle and the means to break a cycle if requirements significantly and suddenly change.

Users judged the Requirements Change UPF to contribute 10% to the problem of misunderstanding requirements and ranked it as 55% as important as their most important factor. Literature on software development lifecycles frequently addresses the reality of changing requirements (for examples, see McConnell, 1996). The existence of this UPF is evidence that existing processes are not adequately, from the perspective of users, dealing with changes in requirements.

6 – Deadlines (U3) / Deadlines (D6). Both users and developers discussed deadlines as being an important factor in misunderstanding requirements. Deadlines drive project schedules and negatively impact requirements. Users viewed deadlines as not leaving sufficient time for requirements determination given the developer resources made available. Users shared that developers are eager to start coding, and often do so before requirements have been adequately addressed. Developers viewed deadlines as being unreasonably imposed without users adequately considering the engineering effort required to construct the system. Developers also shared that users frequently miss their deadlines for providing requirements or completing requirement reviews.

Both users and developers held the other responsible for tension created by deadlines. However, developers expressed that users were unreasonable in their schedule expectations while users expressed that developers did not provide adequate time to understand their needs. Both parties expect deadlines to slip or functionality to be reduced to meet a deadline.

The negative impact of this factor may be reduced by openly discussing it with users and developers. A system delivered on time by developers but that fails to do what is needed by the users is a failed system, providing no value to the business. If both users and developers understood the triple constraint of software development—time, resources, and functionality always remain in balance so an impact to one will impact the other two—they may have more productive conversations for prioritizing requirements, planning incremental releases of the

information system, and understanding the relationship between slips in the schedule and cost or profit to the business. The tension around deadlines appears to limit communications, creating even more difficulties. Instead, it should be viewed as an opportunity to better understand the issues important to users and developers.

Users weighted Deadlines as being responsible for 10% of misunderstanding requirements while developers judged the responsibility as 8%. Users said it was 54% as important as their most important factor while developers said it was 51% as important as their most important factor. A natural tension over deadlines is to be expected, with developers often being uncomfortable with the amount of work they are asked to complete in a given period of time and users or other business representatives wanting to see information systems developed as quickly as possible to reap the anticipated business results (Davis, 2005).

7 – Developer Business Knowledge (U6) / Developer Business Knowledge (D9). This is the last factor to be discussed that is in common between users and developers. Users recognize that developers have an inadequate understanding of the organization's business. The users from the organization that employs business analysts extended the lack of business knowledge to include not only developers but analysts as well—the very people who are supposed to understand users' requirements and be able to translate them to requirements developers understand.

This factor also points out a different perspective between users and developers. Users expressed that developers are not starting an information system project with the end in mind. Users view a system as meeting a set of needs to improve business operations while developers view it as software that encapsulates specified functionality. The apparent disconnect in the end objective of users and developers could in part explain why developers are not motivated to become more knowledgeable about the business of the users.

Developers also recognized their weakness in understanding the organization's business. Some developers believe they could offer more value to users and their organization by better understanding what users do, but not all developers are concerned about gaining business knowledge. The issue is not merely one of motivation. Developers in one organization in the present study were told to not learn about the business but to respond to written requirements from users; that the job of developers was to code, not to understand what users do.

Developers need to have a straightforward means of learning about the organization's business and what users do. For example, they could shadow users for a few hours to a few days to gain an appreciation for their work. Further, they could attend industry seminars, read industry publications, participate in user training classes, and the like. Since it is rare that developers take part in any of these activities, then their motivation for learning more about the business needs to be questioned. If developers are not adequately motivated to gain business knowledge, they are unlikely to do so on their own.

Users judged that the Developer Business Knowledge UPF was responsible for 10% of misunderstanding requirements with a ranking of 54% as important as their most important factor. Developers judged the same factor to be responsible for 9% of misunderstanding requirements with a rank of 53%. Jin et al. (2003) also recognized developers lack an understanding of their organization's business.

8 – *Roles (U4)*. Users shared that they do not understand their role in requirements determination or the roles of others who are involved in requirements determination. Consequently, it is unclear to users who is responsible for requirements. Confusing the roles of users in requirements determination are managers who make decisions affecting requirements without involving users who have been or should have been part of the process. Further, users expressed that developers should ultimately be responsible for developing requirements, citing that users have no training in requirements determination and have demanding work that does

not provide time for outside projects. Users also expressed apathy for being involved in requirements determination because it has not historically resulted in information systems with the expected capabilities delivered when it was needed, the systems could not change quickly enough with changing business needs, and users have lost their jobs because of information systems.

Although there was not a similar DPF, developers are no more eager to be involved in requirements determination. They appreciate business analysts who are responsible for requirements and insulate developers from users so they can concentrate on the task of coding. One developer focus group shared that developers are only involved in requirements determination because users will not do the work, but that developers do not understand the business well enough to be doing the work. One could say this is a form of the blind leading the blind. Developers would rather be handed complete, detailed, accurate requirements that do not require assumptions to be made. However, they do not want so much detail as to limit their creative freedom.

Given the problem of role ambiguity, it could be improved by using a clearly understood requirements determination process that can be explained to both users and developers so they know what to expect and what their responsibilities are (Stary, 2002). However, a clear process alone does not address the motivational issues of neither users nor developers wishing to be involved in requirements determination. Whatever means is chosen to reduce the negative influence of this factor must consider how to motivate those who are required to participate.

Users judged Roles to be responsible for 8% of the problem of misunderstanding requirements, corresponding to being 44% as important as their most important factor. In addition to Stary's (2002) discussion of process as a way to address role ambiguity, Havelka (2003) found that users do not view their role as being responsible for requirements. Rather, users would prefer to have little involvement in requirements determination.

9 – *User Technology Knowledge (U2)*. Users believe they lack vision about what is possible because they do not understand technology options as well as developers. They shared that a better understanding of technology should help them more clearly describe what they want. The existence of this factor reinforces the confusion users have with their roles in requirements determination (UPF U4). Users need to be responsible for sharing their needs, not for translating those needs into a technology solution. This UPF can be attributed to process and motivation problems. Some processes clearly differentiate needs from technology, such as House of Quality (Zrymiak, 2003) that makes users responsible for defining their needs and developers responsible for describing the technology that will be used to meet the needs. The motivation aspect is related to the previous discussion on Roles, where developers do not care to be involved in requirements determination. Because of this, users may feel like they need to overcompensate and be able to specify their requirements in technology terms, or that they could create better requirements if they understood technology better. Users weighted this factor much lower than any of their others, with only a 5% contribution to the problem of misunderstanding requirements and 25% as important as their most important factor.

10 – *Explicit Process (D10)*. User and developer focus groups discussed process-related problems, but only developers identified the lack of an understood requirements determination process as an important factor in misunderstanding requirements. They expressed that a formal process is either not in use or has not been communicated to all parties involved in requirements determination. Such a process is viewed as beneficial in easing the problems with misunderstanding requirements, providing a means of dealing with scope creep, and clarifying users' and developers' roles in requirements determination.

All projects have a requirements determination process, even if it appears ad-hoc. In such cases, documenting the process and reviewing it with those involved in the project is a reasonable first step. For process adoption guidelines, the Carnegie Mellon Software Engineering

Institute provides several resources including the Capability Maturity Model Integration (SEI, 2006). Developers attributed 10% of misunderstanding requirements to the lack of an explicit process, which made it 61% as important as the most important factor.

11 – *Poor Communication (D2)*. Developers had no problem identifying the obvious—that users and developers communicate poorly. However, there are important aspects of poor communication that developers emphasized. One is that both users and developers make assumptions. Users may make assumptions when describing their needs because of unrecognized tacit knowledge they have of the problem or because they expect developers to have a basic understanding of their business. One group of users shared dismay when a new version of an existing system refreshed the screen after each data entry whereas the previous system allowed multiple data entries before refreshing the screen. The delay encountered with each refresh rendered the system unbearably slow to use. The users had assumed that the new system would work in a similar manner to the previous system regarding data entries and screen refreshes. When the users asked about this, the developers responded by saying that screen refreshing operations were not specified as requirements. This situation may be the result of developers insufficiently probing for clarification and details (Al-Rawas and Easterbrook, 1996; Coughlan et al., 2003).

When developers encounter requirements with insufficient detail, they likely make assumptions instead of consulting users to clarify the requirement and provide more detail. This issue was also addressed by Bostrom and Thomas (1983). Even though developers generally lack knowledge of the users' business, they believe they know what users need even if users do not tell them. Developers have found that this is a practical attitude and response to missing information because the project would unacceptably slow each time users had to be consulted.

Another key aspect of poor communications was what was called the *Telephone Game*, which refers to the childhood game of sharing a message with one person who passed it on to

another, who in turn passed it on to another, and so forth, until the last person hears a message that has little resemblance to the original message. In the context of the present study, the Telephone Game is the problem with messages becoming distorted as they are transferred from users to business analysts to developers. Only one of the three organizations used business analysts. This organization discussed many of the same factors that were shared by the other two organizations. That is, the presence of a business analyst did not result in vastly different problems—all three organizations shared similar obstacles to understanding requirements. However, both users and developers in the organization with business analysts discussed the Telephone Game. Specifically, they shared that communication breaks down as messages travel from a user to an analyst to one or more developers. Fifty-seven percent of the developer participants selected the Telephone Game factor as most important.

In stark contrast, the other two organizations that did not have business analysts viewed them as the means to solving many of their problems by translating between users and developers and being responsible for requirements. The need for a business analyst was associated with developers and users using different terminology, effectively speaking different languages, and having different perspectives on a problem. However, the organization with business analysts also identified all of these issues. Therefore, the employment of business analysts alone should not be expected to significantly improve requirements determination. Instead, the use, or possibly improper use, of analysts can increase poor communications between users and developers. Perhaps a reasonable approach, previously discussed in 1 – Different Perspectives (U9) / Different Perspectives (D4), is the use of a trained facilitator to bring users and developers together and aid more effective communications (Al-Rawas and Easterbrook, 1996; Coughlan et al., 2003).

Developers judged the Poor Communication DPF to be responsible for 9% of the misunderstanding requirements problem, which makes it 57% as important as their most

important factor. Much of the literature covering requirements determination discusses various aspects of the importance of communication and the existence of communication problems (e.g., Gallivan and Keil, 2003; Lindqvist, 2003; Mrenak, 1990). However, there is also an emphasis in the literature on the usefulness of business analysts (Joshi, 1992), which is brought into question by this DPF.

12 – *Ineffective Documentation (D8)*. Related to Poor Communication is the lack of useful requirements documentation. Although user and developer focus groups discussed documentation problems, only developers elevated the issue to become a DPF. They shared that requirements documentation is not updated when changes occur, that it is difficult to find what they need to know, and that inconsistencies and conflicts occur in documentation. Further, requirements that are shared verbally in meetings may never become formal requirements because they were not written down.

The ineffectiveness of requirements documentation raises a question about the purpose of the documentation. If the documentation primarily exists as a means for developers to defend what was implemented, then it is not focused on clarifying requirements. Users shared that when conflicts over system capabilities arise or when misinterpretations occur about the implementation, they are told the requirement was interpreted as written. Users are left feeling defensive about the letter of the requirements when they more naturally think in terms of the spirit of the requirements. Users view the objective of an information system as meeting a business need, while developers view it as reflecting the requirements. Therefore, requirements documentation could become a wall between users and developers—a crutch that hinders more effective communications.

This DPF is related to the desire of developers to be provided clear, concise, and complete requirements with little to no involvement of developers. If the requirements documentation provided by users did this, then developers would need little interaction with

users and could focus on coding the information system. However, the answer may not be in improving the quality of requirements documentation but in concentrating on the purpose of the documentation, which is to capture the needs for an information system. Perhaps time is better spent on other tools, such as prototyping and observation instead of perfecting documentation (Joshi, 1992). To this end, agile techniques that minimize documentation while maximizing user-developer interaction may reduce the negative impact of this DPF on misunderstanding requirements (Williams & Cockburn, 2003). The developer participants judged Ineffective Documentation to be responsible for 9% of misunderstanding requirements and 55% as important as their most important DPF.

13 – *Requirement Reviews (D3)*. Although developers selected Requirement Reviews as a DPF, users in the focus groups also discussed requirement reviews as an inadequate means of approving requirements. Developers shared that they do not consistently ask users to review requirements, that they prefer to rely on email for requirement review activities instead of meeting with users, and that they would rather not be involved in reviews. When they do provide requirements documentation to users to review, developers do not expect users to conduct an adequate review. Further, developers expressed that prototypes need to be used more frequently when reviewing requirements because users need a visual representation of the requirements, not merely a written document (Leffingwell & Widrig, 2000).

Although users did not create a related UPF for Requirements Reviews, they did discuss reviews. They shared that users are not often asked to review requirements. When they are asked they find the documentation to be poor and difficult to read. Further, the requirements are difficult to understand because they have been written from a developer's perspective (Elliott, 2000; Eriksson and Penker, 1998; Jin et al., 2003). Users do appreciate prototypes because they are useful for discussing what they do and do not want. Developers judged Requirement Reviews

to be responsible for 9% of misunderstanding requirements, which made it 51% as important as their most important factor.

14 – *Organizational Objectives (D7)*. The developer participants perceived that users lack an understanding of how user needs are related to the organization's objectives and strategy. Users are not discussing their problem in terms of the larger picture of the organization or in terms of the organization's strategy. Developers interpret this as users not having a full grasp of the problem and insufficient insight into the organization to reach an optimal solution. Without a complete picture of the needs in an organizational context, developers suspect they are not getting the correct requirements, or at least not all of the requirements.

This DPF contributes in part to the developer perception that users do not know what they want. When users express vague business objectives, developers question what they really need. Developers work with a greater level of detail than users concerning requirements. Although developers may not truly need or care about the higher level organizational objectives that a user's problem relates to, they want detailed information when discussing objectives. Developers also see users asking to automate or improve part of a process, such as filling in a form, without considering the larger problem. Although developers can respond to such requests, some developers first want to consider if the correct problem is being addressed or if they are merely making an existing process faster.

Addressing the Organizational Objectives DPF is challenging because many organizations struggle with their employees not knowing the organization's objectives and strategies. Further, organizations frequently have processes in place that discourage users working with others outside of their area of responsibility. Developers judged Organizational Objectives to be responsible for 8% of the misunderstanding requirements problem, making it 48% as important as their most important factor.

Discussion of the Results: User and Developer Profiles

The UPFs and DPFs provide valuable insights into the characteristics of users and developers. The research findings provide a logical basis for the creation of user and developer profiles. Although a subjective exercise, the formation of such profiles reflects the factors users and developers discussed. Knowledge of these profiles could be used to help users and developers understand each other, aid project management, improve team dynamics, and positively impact requirements determination.

User Profile. Users are knowledgeable about their business. They recognize that their past experience with information systems tends to influence and even constrain their thinking about new information systems. They would rather not be involved in requirements determination activities and expect developers to be responsible for creating requirements that are accurate and complete with as little input as possible from users. Further, the users who have the most experience and knowledge to best aid requirements determination are the ones who are least available and will not make adequate time to be involved. This same sentiment extends to users' managers who do not prioritize time for working on requirements. Their lack of desire to be involved in requirements determination feeds their lack of skill with determining requirements and provides the motivation for remaining ignorant about what is expected from them in a requirements determination exercise.

Users may be unclear discussing their needs because they may not have thought deeply enough about the problem they wish to address with an information system, or they may have difficulty finding words to articulate their vision for a system. Users also have difficulty prioritizing their needs. They become frustrated when requirements they share are changed or removed without their knowledge. They are accustomed to information systems being delivered late, being canceled during development, or not having the functionality they expected when

they are delivered. Consequently, they have lost hope in requirements determination exercises and have lower motivation to participate in developing information systems.

A significant frustration for users is the inability of developers to respond to changes that may occur in requirements over time. Changes may be introduced naturally as users better understand the problem and their needs, after viewing a system prototype, learning about the needs of a related group, or better understanding the larger organizational context that their specific problem fits into. Changes are also imposed when business objectives are modified in response to a turbulent market or a competitor's actions. Users expect that developers will tell them changes must be handled in the next release of the system, not the one they are currently working on, which likely will make the first release unusable.

Users believe if they had a better grasp of technology and what was possible that they could better determine requirements. They see deadlines as the means of constraining the functionality they receive, which may not be the functionality that makes a meaningful difference to them.

Users are most aware of the differences between themselves and developers. Relating to a developer requires more detailed communication, tolerating an arrogant attitude, and learning unfamiliar jargon. They would appreciate developers better understanding the business of the organization. Users are surprised when a developer does not realize that a capability should have been obvious without it being explicitly stated. Users believe if the developers knew the business of the organization, less mistakes would occur.

Developer Profile. Developers recognize themselves as being arrogant and may even go out of their way to intimidate users as a means to spend less time with them. Developers would rather be creating software than talking with users, who have very different motivations and perspectives. They prefer communicating via email and tend to avoid in-person meetings. Although they realize they lack knowledge of their business and what users do, and know they

could offer more value to the organization if they had this knowledge, they have little desire to learn about the business.

Just like users, developers also do not want to be responsible for requirements determination. They expect requirements to be provided to them, with sufficient detail so developers do not have to make assumptions but not with so much detail as to limit their creativity. Developers do not like users that overstep their bounds by providing a design for a system as opposed to requirements – “they’re not going to tell me what to do.”

They believe that a predictable and communicated requirements determination process that the entire organization used would resolve many of the problems encountered in creating requirements. It would lay out the roles and responsibilities of all involved and ensure that users sufficiently understood the problem, prepared business cases, and completed requirements as they should.

Developers are frustrated when users do not adequately review requirements documentation, and become even more frustrated when requirements change after the requirements document “has been signed off on.” They expect users to understand requirements documentation with little to no involvement from developers. When requirements do change, as is perceived as frequently the case, developers regard users as not knowing what they want. Due to this expectation that users do not know what they want, developers may assume that they know more about what is needed than the users.

Developers recognize that users are different from them and that they do not speak the same language. They want a business analyst or project manager involved that will help translate foreign terminology and even insulate the developers from the users. While developers are eager to try new technologies and new ways to solve problems, they view users as being stuck in a box created from their past experiences and less eager to deviate from what has worked before.

Developers may even make technology choices because of the personal learning experience or resume building opportunity, not because they are the most appropriate choices for the project.

Developers view determining requirements as a negotiation process. Users provide a list of what they want, often without priorities and without creating a common picture of the problem, and developers must discuss what can be done given the resources and time available. Developers do not expect to accomplish everything users need and may make assumptions about what is most important. Further, when details are missing, developers will act out of their own experience and creativity, which may result in something different than the user wanted.

Conclusions

This study is a first step in creating a foundational theoretical framework for misunderstanding requirements for information systems. The 14 factors created by users and developers are the key contribution to this framework. Many of the factors directly and indirectly highlight differences between users and developers. Improving understanding of requirements by these two groups is clearly a difficult problem; otherwise it would have been solved at some point in the history of information systems development that has resulted in a plethora of requirements determination techniques.

The problem is not merely one of translation between users and developers. As was pointed out in the present study, a business analyst was viewed by organizations not having one as the means to translate between users and developers while the organization that used business analysts expressed greater problems with communications. Perhaps a facilitator, not a translator, is needed to improve communications between users and developers. Business analysts play an important role, but they are not facilitators who bring users and developers together. Instead, as suggested in the present study, they tend to insulate users from developers.

However, software development processes that rely on close user-developer interaction may find it difficult to involve users as required or to get quality input from them. For example, the XP software development process requires users to write their own user stories, which is a description of what the system must do. Also, XP requires nearly daily user-developer interaction, recommending that an expert user be dedicated to the development team. Both users and developers made clear that gaining access to expert users is difficult and a large contributor to misunderstanding requirements. Further, users shared a reluctance to be involved in requirements determination. Consequently, requirements determination processes that rely on users to write requirements and to be highly available, even dedicated to the development team, may be flawed. Instead, observation techniques that can study key users in their environment performing their tasks while not requiring their active participation should be considered a standard tool in the development of information systems.

Quick wins for reducing misunderstanding requirements may be found by taking steps to address the most important factor to users—*Different Perspectives*, users and developers relate to each other differently—and the most important factor to developers—*Key Users*, key users who have the information for determining requirements are not available or do not stay involved during the project. The addition of user observation studies to requirements determination processes offers potential to reduce both of these factors.

The factors provide useful criteria for analyzing candidate requirements determination techniques for their appropriateness to an information systems development project. For example, if access to key users was not a problem, then HOQ or workshops would be more useful than when access is restricted. Using the factors for evaluation would result in finding one or more determination techniques that should be more productive than other techniques.

Perhaps the most significant implications of the study come from users and developers acknowledging that neither group wants to be responsible for requirements. Users shared that

they are not trained for or skilled in requirements determination while developers shared that they do not adequately understand the business of users. Consequently, any requirements determination technique that relies on the involvement of users and/or developers is fighting an uphill battle. Users and developers need to be motivated to participate, or techniques need to be used that minimize user involvement without creating additional communication difficulties.

Finally, the research shed light on many areas not treated in the scope of most techniques used for requirements determination, such as prototyping and interviews. These tools are focused on uncovering requirements, but ignore related issues such as the larger problem domain, overcoming users' and developers' lack of motivation to participate in requirements determinations, and the like. Before clearly articulating needs for an information system, or even observing those needs, the problem must be well understood in light of the organization's business objectives and strategies. Users and developers need to be told what to expect and be provided motivation for their participation.

Recommendations

Limitations of the Study

Sample Size. The sample involved three organizations from very different sectors. Although there was much agreement between the organizations about the factors that influence misunderstanding requirements, participants from other organizations in the same sectors or from other sectors may have chosen significantly different factors. Consequently, the user and developer factors discussed in this study may not be representative of information systems development in other organizations.

Software Development Process. Participants were not screened based on the software development process chosen. Although the participants indicated that they use a variety of processes, such as waterfall, iterative, incremental, and agile, the discussion in focus groups did

not focus on any one process. Repeating this study only with users and developers using a specific process, such as XP, may result in the creation of different factors.

Controlled Variables. The researcher did not seek to control variables that could impact the factors created by users and developers as well as the importance they placed on them. Organizations with high success rates of developing information systems may differ from organizations with low success rates. Users and developers with more experience may make judgments different from those with less experience. Other variables that could influence the findings would be organizational complexity, markets, size of information systems, number of users and developers involved, presence or absence of other stakeholders, using business analysts and certified facilitators, and the like.

Other Stakeholders. The study focused on users and developers as the two key groups responsible for understanding requirements for an information system. Other stakeholders are involved in an information system that may have additional or different requirements, making the creation of understood requirements even more difficult. This study was constrained to users and developers and did not consider effects from other project stakeholders.

Commercial Software Development. The factors identified were from users and developers creating information systems to be used within their organization. The development of commercial software also greatly suffers from misunderstanding requirements, but is not identical to developing information systems. The applicability of the factors in a different problem domain like commercial software development is unknown.

Self-Selection. Each focus group generated between 28 and 48 factors, but only the subset of those selected as most important by the participants were used in Phase II as the UPFs and DPFs. The self-selection process to identify important factors may have ignored other critical factors that, when minimized, could dramatically decrease misunderstanding requirements.

AHP Pairwise Comparisons. Although AHP is a very cognitively simple technique for subjectively judging the importance of a set of factors, and possibly the simplest technique to use that can create both ranking and weighting (Schniederjans & Wilson, 1991), its use presented two problems. First, comparing nine factors at one time, as users did, or ten, as developers did, is mentally challenging. This was evident in the inconsistency of responses by several participants. Another problem is that AHP is often used in an interactive environment, where the process can be modified as problems are encountered. AHP recommends validating the weightings participants provide by checking the inconsistency ratio, and if the inconsistency ratio is too high, discuss where inconsistencies have occurred and if they need to be corrected. This was not possible using web-based surveys to collect AHP information.

Limited Understanding of Causation. If this research leads to methods for reducing misunderstanding requirements, that does not necessarily mean a direct improvement in the quality or success of information systems will be achieved. Other research shows that misunderstanding requirements is a key reason, and possibly the largest reason, for failed information systems (Standish, 2002). However, eliminating misunderstood requirements may not improve the success rate of information systems as this has not yet been tested.

Suggestions for Future Research

User and Developer Shared Perspective. Users in the study only judged the importance of factors in Phase II that they have created in Phase I. Likewise, developers only judged the importance of the factors they created. The importance of the UPFs and DPFs that combined to form 14 factors should be evaluated by both users and developers. This could be accomplished via survey using a research method similar to the approach used by Havelka (1994) to examine user and developer rankings of critical success factors for information systems.

Method Testing. The initial objective for the research was to determine means for improving the quality (as perceived by users) of information systems. Prior studies associated poor quality with misunderstood requirements. Consequently, future research is needed in two key areas. First, one must determine if minimizing the impact of the UPFs and DPFs on an information systems project does decrease misunderstood requirements, resulting in better understood requirements. Second, one must verify that better understood requirements results in higher quality information systems. In both cases, requirements determination methods need to be improved and tested for their ability to minimize the negative impacts of the factors identified in the present study.

Minimizing UPFs and DPFs. The study presented ideas for minimizing UPFs and DPFs and a means of analyzing requirements determination techniques in light of the UPFs and DPFs. However, testing must be conducted to validate if and how the factors can be minimized. Methods could take several forms that require further research, such as improvements to existing requirements determination techniques, using trained facilitators, combining several requirements determination techniques together, as well as others.

Generalization. As an initial theory building study, research needs to be conducted to test the generalization of the factors. An appropriate means of doing this would be to broadly survey a random sample of users and developers of information management systems. Further, the importance of each UPF and DPF was determined by a few participants who were inconsistent with their own responses and often disagreed with each other. A larger sample is necessary to verify the true perceived importance of each factor.

Impact of Themes. The themes synthesized in the study are a reflection of all the factors participants discussed, not only those judged to be sufficiently important to become a UPF or DPF. The themes have many points of interaction between them. For example, the theme “Users are not equipped for requirements work” impacts the theme “Key users are not available or not

identified.” The interactions between themes suggest that a decrease in the negative influence of one theme may ripple through several other themes. Consequently, there may be factors not represented in the UPFs and DPFs that could have a large impact on misunderstanding requirements. The themes are rich in information that needs further examination and study.

Business Analysts. The study uncovered a need to question the effectiveness of business analysts, who often fulfill the role of translator between users and developers. They are also frequently ultimately responsible for requirements. Further research is necessary to determine the effectiveness of business analysts and the quality of information systems created with business analysts compared to information systems created without using them.

Closing Observations

An analogy to John Gray’s (1992) book “Men are from Mars, Women are from Venus” was used as an introduction to the present study. After collecting data, performing analyses, and drawing conclusions, it remains an aptly chosen parallel. The difficulties users and developers face to understand the requirements for an information system are similar to the difficulties described by Gray. The work a successful marriage counselor performs to bridge the culture gap between men and women may be similar to the work needed to bridge the understanding gap between users and developers.

As was introduced in Chapter 3, the original design to uncover reasons for misunderstanding requirements was focused on three overlapping dimensions of the problem space, as shown on Figure 27. Extending the analogy to Gray’s work (1992), suggests that perhaps success will be found not in employing a counselor for information systems teams, but more appropriately, by engaging a skilled facilitator who recognizes and minimizes problems from communication issues to personality differences, and different mental frameworks. Many of the UPFs and DPFs illustrated aspects of these three dimensions of the problem, and can

provide a foundation for further research. The UPFs and DPFs also shed light on motivational issues, process problems and basic skill deficiencies. Effectively controlling these aspects of the problem across the three dimensions of Figure 27 may minimize misunderstanding requirements and lead to higher quality information systems.

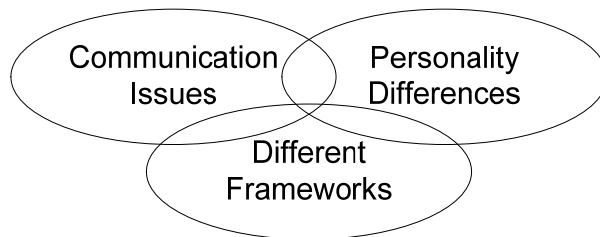


Figure 27. Three dimensions involved in misunderstanding requirements for information systems.

REFERENCES

- Abrahamsson, P., & Koskela, J. (2004). *Extreme programming: A survey of empirical data from a controlled case study*. Paper presented at the 2004 International Symposium on Empirical Software Engineering (ISESE'04), Redondo Beach, CA.
- Addison, T. (2003). E-commerce project development risks: Evidence from a Delphi survey [Electronic version]. *International Journal of Information Management*, 23(1), 25.
- Al-Rawas, A., & Easterbrook, S. (1996). *Communication problems in requirements engineering: A field study*. Paper presented at the Westminster Conference on Professional Awareness in Software Engineering, London.
- Badri, M. A. (2001). A combined AHP-GP model for quality control systems. *International Journal of Production Economics*, 72(1), 27-40.
- Baker, D., Bridges, D., Hunter, R., Johnson, G., Krupa, J., Murphy, J., et al. (2001). *Guidebook to decision-making methods* (No. WSRC-IM-2002-00002): Department of Energy.
- Beck, K. (2000). *Extreme programming explained: Embrace change*. Boston: Addison-Wesley.
- Beck, K., Beedle, M., Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., et al. (2001). *Manifesto for Agile Software Development*. Retrieved November 11, 2004, from <http://www.agilemanifesto.org/>
- Berry, D. M. (2002). The importance of ignorance in requirements engineering: An earlier sighting and a revisitation [Electronic version]. *Journal of Systems & Software*, 60(1), 83-85.
- Boehm, B. W. (1981). *Software engineering economics*. Englewood Cliffs, NJ: Prentice-Hall.
- Borenstein, S. (1991). *Programming as if people mattered*. Princeton, NJ: Princeton University Press.
- Bostrom, R. P., & Kaiser, K. M. (1982). Personality characteristics of MIS project teams: An empirical study and action-research design. *MIS Quarterly*, 6(4), 43-60.
- Bostrom, R. P., & Thomas, B. D. (1983). Achieving excellence in communications: A key to developing complete, accurate and shared information requirements. In *The Proceedings of the Twentieth Annual Computer Personnel on Research Conference* (pp. 1-13). Charlottesville, Virginia, United States: ACM Press.

- Brooks, F. P. (1987). No silver bullet: Essence and accidents of software engineering. *Computer*, 20(4), 10-19.
- Browne, G. J., & Rogich, M. B. (2001). An empirical investigation of user requirements elicitation: Comparing the effectiveness of prompting techniques [Electronic version]. *Journal of Management Information Systems*, 17(4), 223-249.
- Brugha, C. M. (2004). Phased multicriteria preference finding. *European Journal of Operational Research*, 158(2), 308-316.
- Butler, T., & Fitzgerald, B. (1997). A case study of user participation in the information systems development process. In *Proceedings of the eighteenth international conference on Information systems* (pp. 411-426). Atlanta, Georgia, United States: Association for Information Systems.
- Capella. (2005). *Institutional Review Board*. Retrieved 3/14/2005, from <https://www.capella.edu/Portal/Learner/SContent/centers/learner/policies/irb.pdf>
- Carlsson, S. (2000). *Lärande systemutveckling och samarbetsformer – från kommunikation till samförstånd genom lärande och undervisning*. Doctoral dissertation, Karlstad University Studies, Sweden.
- Cooper, D. R., & Schindler, P. S. (2003). *Business research methods* (8th ed.). New York: McGraw-Hill.
- Coughlan, J., Lycett, M., & Macredie, R. D. (2003). Communication issues in requirements elicitation: A content analysis of stakeholder experiences. *Information & Software Technology*, 45(8), 525-536.
- Creswell, J. W. (2003). *Research design: Qualitative, quantitative, and mixed methods approaches* (2nd ed.). Thousand Oaks, CA: Sage.
- Curtis, B. (1990). *Empirical studies of the software design process*. Paper presented at the Human-Computer Interaction, Cambridge, UK.
- Cushing, B. E. (1990). Frameworks, paradigms, and scientific research in management information systems. *The Journal of Information Systems*, 2(2), 38-59.
- Dana, W. H. (1993). *The X-15 lessons learned [Electronic version]*: NASA Dryden Research Facility.
- Darlington, M. J. Å., & Culley, S. J. Å. (2002). Current research in the engineering design requirement. *Proceedings of the Institution of Mechanical Engineers, Part B, Engineering Manufacture*, 216(3), 375-388.

- Davis, A. (1993). *Software requirements: Objects, functions, and states*. Englewood Cliffs, NJ: Prentice-Hall.
- Davis, A. (1995). *201 principles of software development*. New York: McGraw-Hill.
- Davis, A., Overmyer, S., Jordon, K., Caruso, J., Dandashi, F., Dinh, A., et al. (1997). Identifying and measuring quality in a software requirement specification. In R. H. Thayer & M. Dorfman (Eds.), *Software Requirements Engineering* (pp. 164-175). Los Alamitos, CA: IEEE Computer Society Press.
- Davis, A. M. (2005). *Just enough requirements management: Where software development meets marketing*. New York: Dorset House.
- Davis, G. B. (1982). Strategies for information requirements determination. *IBM System Journal*, 21(1), 4-30.
- Davis, G. B., & Monroe, M. C. (1987). Commentary on the problem of information requirements for computer applications. *Accounting Horizons*, December, 105-109.
- DeBellis, M., & Haapala, C. (1995). User-centric software engineering. *IEEE Expert*, 34-41.
- DoD. (1991). Software technology strategy. December, 1991.
- Dodgson, J., Spackman, M., Pearman, P. A., & Phillips, P. L. (2000). *DTLR multi-criteria analysis manual*. London: Office of the Deputy Prime Minister.
- Duggan, E. W. (2003). Generating systems requirements with facilitated group techniques. *Human-Computer Interaction*, 18, 373-394.
- Duggan, E. W., & Thachenkary, C. S. (2003). Higher quality requirements: Supporting joint application development with the nominal group technique. *Information Technology and Management*, 4(4), 391-408.
- Dunham, R. B. (1998). *Nominal Group Technique: A users' guide*. Retrieved 4/5/2005, from <http://instruction.bus.wisc.edu/obdemo/readings/ngt.html>
- Eldin, N. (2002). A promising planning tool: quality function deployment. *Cost Engineering*, 44(3), 28-37.
- Elliott, J. J. (2000). Design of a product-focused customer-oriented process. *Information and Software Technology*, 42(14), 973-981.
- Eriksson, H., & Penker, M. (1998). *UML toolkit*. New York: John Wiley & Sons.
- ExpertChoice. (2005). *Analytic hierarchy process*. Retrieved 3/2/2005, from <http://www.expertchoice.com/customerservice/ahp.htm>

- ExpertChoice. (2006). *EC Software*. Retrieved 4/15/2006, from <http://www.expertchoice.com/>
- ExpressScribe. (2006). *Express Scribe transcription software*. Retrieved 5/9/2006, from <http://nch.com.au/scribe/index.html>
- Finnie, G. R., & Wittig, G. E. (1999). *An intelligent web tool for collection of comparative survey data*. Paper presented at the Proc. 10th Australasian Conference on Information Systems, Wellington, New Zealand.
- Fisher, J. (1999). The value of the technical communicator's role in the development of information systems, *IEEE Transactions on Professional Communication* (Vol. 42, pp. 145).
- Foreman, E. H., & Gass, S. I. (2001). The AHP: An exposition. *Operations Research*, 49, 469-486.
- Forman, E., & Peniwati, K. (1998). Aggregating individual judgments and priorities with the analytic hierarchy process. *European Journal of Operational Research*, 108(1), 165-169.
- Forman, E. H. (1993). Facts and fictions about the analytic hierarchy process. *Mathematical and Computer Modeling*, 17(4-5), 19-26.
- Fowler, M., & Highsmith, J. (2001). The agile manifesto. *Software Development*, August, 28-32.
- Gallivan, M. J., & Keil, M. (2003). The user-developer communication process: a critical case study. *Information Systems Journal*, 13(1), 37-68.
- Gottesdiener, E. (2002). *Requirements by collaboration: Workshops for defining needs*. Boston, MA: Addison-Wesley Longman Publishing.
- Gray, J. (1992). *Men are from Mars, women are from Venus: A practical guide for improving communication and getting what you want in your relationships*. New York: HarperCollins.
- Griffin, A., & Hauser, J. R. (1993). The voice of the customer. *Marketing Science*, 12(1), 1-27.
- Guinan, P. J., & Bostrom, R. P. (1984). Development of computer-based information systems: a communication perspective. *SIGCPR Comput. Pers.*, 9(4), 17-25.
- Gulliksen, J., & Lantz, A. (2003). Design versus design: From the shaping of products to the creation of user experiences, *International Journal of Human-Computer Interaction* (Vol. 15, pp. 5-20): Lawrence Erlbaum Associates.
- Hales, R., Lyman, D., & Norman, R. (1994). QFD and the expanded house of quality. *Quality Digest*, 36-45.

- Hartwick, J., & Barki, H. (1994). Explaining the role of user participation in information system use. *Management Science*, 40, 440-465.
- Havelka, D. (1994). *Antecedents to systems development: Beliefs of information systems specialists and users*. (Doctoral dissertation, Texas Tech University, 1994).
- Havelka, D. (2003). A user-oriented model of factors that affect information requirements determination process quality. *Information Resources Management Journal*, 16(4), 15-32.
- Havelka, D., & Lee, S. (2002). Critical success factors for information requirements gathering. *Information Strategy*, 18(4), 36.
- Havelka, D., Sutton, S. G., & Arnold, V. (1998). A methodology for developing measurement criteria for assurance services: An application in information systems assurance. *Auditing*, 17, 73-92.
- Havelka, D., Sutton, S. G., & Arnold, V. (2001). Information systems quality assurance: The effects of users' experiences on quality factor perceptions. *Review of Business Information Systems*, 52(2), 49-62.
- Herlea, D. E. (1999). User participation in requirements negotiation. *SIGGROUP Bull.*, 20(1), 30-35.
- Herlea, D. E., Jonker, C. M., Treur, J., & Wijngaards, N. J. E. (2002). A compositional knowledge level process model of requirements engineering. *International Journal of Software Engineering & Knowledge Engineering*, 12(1), 41-75.
- Hevner, A. R., & Harlan, D. M. (1995). Box-structured requirements determination methods. *Decision Support Systems*, 13, 223-239.
- Hickey, A. M., & Davis, A. M. (2004). A unified model of requirements elicitation. *Journal of Management Information Systems*, 20(Issue 4), 65-84.
- Ibanez, M., & Rempp, H. (1996). *European User Survey Analysis*. Retrieved June 6, 2004, from <http://www.esi.es/VASIE/Reports/All/11000/Download.html>
- Idea Works, I. (2006). *Qualrus: The intelligent qualitative analysis program*. Retrieved 5/15/2006, from <http://www.qualrus.com/Qualrus.shtml>
- IEEE. (1990a). *IEEE standard computer dictionary: A compilation of IEEE standard computer glossaries*. New York: The Institute of Electrical and Electronics Engineers.
- IEEE. (1990b). *IEEE standard glossary of software engineering terminology*. New York: The Institute of Electrical and Electronics Engineers.

- Jin, Z., Bell, D. A., Wilkie, F. G., & Leahy, D. G. (2003). Automated requirements elicitation: Combining a model-driven approach with concept reuse. *International Journal of Software Engineering & Knowledge Engineering*, 13(1), 53-82.
- Jirotko, M., & Goguen, J. A. (1994). *Requirements engineering: social and technical issues*: Academic Press Professional, Inc.
- Johansson, E., Wesslén, A., Bratthall, L., & Höst, M. (2001). *The importance of quality requirements in software platform development: A survey*. Paper presented at the 34th Annual Hawaii International Conference on System Sciences (HICSS-34), Maui, Hawaii.
- Johnson, J. C. (1990). *Selecting ethnographic informants*. Newbury Park, CA: Sage.
- Joshi, K. (1992). Interpersonal skills for cooperative user analyst relationships: Some research issues. *Database, Winter*, 23-25.
- Katz, G. M. (2004). The voice of the customer. In P. Belliveau, A. Griffin & S. M. Somermeyer (Eds.), *The PDMA toolbook for new product development*. Hoboken, New Jersey: John Wiley & Sons.
- Kazmierczak, E., Dart, P., Sterling, L., & Winikoff, M. (2000). Verifying requirements through mathematical modeling and animation. *International Journal of Software Engineering & Knowledge Engineering*, 10(2), 251-272.
- Keil, M., & Carmel, E. (1995). Customer-developer links in software development. *Communications of the ACM*, 38(5), 33-44.
- Keil, M., Cule, P. E., Lyytinen, K., & Schmidt, R. C. (1998). A framework for identifying software project risks. *Commun. ACM*, 41(11), 76-83.
- Keil, M., Tiwana, A., & Bush, A. (2002). Reconciling user and project manager perceptions of IT project risk: a Delphi study, *Information Systems Journal* (Vol. 12, pp. 103-119): Blackwell Publishing Limited.
- Kenney, C., & Leggiere, P. (2003). Across the chasm: This lesson from an outside discipline may prevent miscommunication that can kill your strategic IT project [Electronic version]. *Intelligent Enterprise*, 6(5).
- Klockner, K., Pankoke-Babatz, U., & Prinz, W. (1999). Experiences with a cooperative design process in developing a telecooperation system for collaborative document production. *Behaviour & Information Technology*, 18(5), 373-383.
- Koksal, G., & Egitman, A. (1998). Planning and design of industrial engineering education quality. *Computers and Industrial Engineering*, 35(3-4), 639-642.

- Kudikyala, U. K., & Vaughn, R. B. (2005). Software requirement understanding using Pathfinder networks: Discovering and evaluating mental models. *The Journal of Systems and Software*, 74(2005), 101-108.
- Lamsweerde, A. v. (2000). *Requirements engineering in the year 00: A research perspective*. Paper presented at the Proceedings of the 22nd International Conference on Software Engineering, Limerick, Ireland.
- Laplante, P. A., & Neill, C. (2004). "The demise of the waterfall model is imminent" and other urban myths [Electronic version]. *ACM Queue*, 1(10).
- Larman, C., & Basili, V. R. (2003). Iterative and incremental development: A brief history. *IEEE Computer*, 3, 47-56.
- Leffingwell, D., & Widrig, D. (2000). *Managing software requirements: A unified approach*. Indianapolis, IN: Addison-Wesley.
- Leung, L. C., & Cao, D. (2001). On the efficacy of modeling multi-attribute decision problems using AHP and Sinarchy. *European Journal of Operational Research*, 132(1), 39-49.
- Lindqvist, R. (2003). *Cognitive science improves communication within information systems development projects*. Retrieved 2/5/2005, from <http://w3.msi.vxu.se/~rli/374-171.pdf>
- Liu, L., & Khooshabeh, P. (2003). *Paper or interactive?: A study of prototyping techniques for ubiquitous computing environments*. Paper presented at the CHI '03 Extended Abstracts on Human Factors in Computing Systems, Fort Lauderdale, FL.
- Lu, R., Jin, Z., & Wan, R. (1995). *Requirement specification in pseudo-natural language in PROMIS*. Paper presented at the Proceedings of COMPSAC, USA.
- Makrygiannis, N., & Enquist, H. (1998). Understanding misunderstandings. In *Proceedings of the Thirty-First Annual Hawaii International Conference on System Sciences* (Vol. 6, pp. 83): IEEE Computer Society.
- Maxwell, J. A. (1992). Understanding and validity in qualitative research. *Harvard Educational Review*, 62, 279-300.
- Mazur, G. H. (1997). *9 house of quality checks*. Retrieved 2/20/2005, from <http://www.mazur.net/works/9checks.pdf>
- McConnell, S. (1998). *Software project survival guide*. Redmond, WA: Microsoft Press.
- McConnell, S. (2000). Quantifying soft factors. *IEEE Software*, 17(6), 9-11.
- Merriam, S. (1998). *Case study research in education: A qualitative approach*. San Francisco: Jossey-Bass.

- Miles, M. B., & Huberman, A. M. (1994). *Qualitative data analysis* (2nd ed.). Thousand Oaks, CA: SAGE.
- Miller, G. A. (1956). The magical number seven plus or minus two: Some limits on our capacity for processing information. *The Psychological Review*, 63, 81-97.
- Moody, D. L., & Sindre, G. (2003). Incorporating quality assurance processes into requirements analysis education. In *Proceedings of the 8th annual conference on innovation and technology in computer science education* (pp. 74-78). Thessaloniki, Greece: ACM Press.
- Moore, M. J. (2003). *Communicating requirement using end-user GUI constructions with argumentation*. Paper presented at the 18th IEEE International Conference on Automated Software Engineering (ASE'03).
- Mrenak, G. (1990). Evolving concepts, or why users often don't recognize the software they asked for. In *Proceedings of the seventh Washington Ada symposium on Ada* (pp. 17-22). McLean, Virginia, United States: ACM Press.
- Nielsen, J. (1993). *Usability engineering*. San Diego, CA: Academic Press.
- Nielsen, J. (2001). *First rule of usability: Don't listen to users*. Retrieved November 6, 2004, from <http://www.useit.com/alertbox/20010805.html>
- NIST. (2005). *Criteria for performance excellence*. Retrieved 2/20/2005, from http://baldrige.nist.gov/PDF_files/2005_Business_Criteria.pdf
- OMG. (2005). *UML resource page*. Retrieved 5/30/2005, from <http://www.uml.org>
- Osmundson, J. S., Michael, J. B., Machniak, M. J., & Grossman, M. A. (2003). Quality management metrics for software development. *Information & Management*, 40(8), 799.
- Pai, W. C. (2002). A quality-enhancing software function deployment model. *Information Systems Management*, 19(3), 20-25.
- Pan, D., & Yanp, C. M. (1999). *Effective information processing between users and developers during information system project development*. Paper presented at the Proceedings of the 32nd Hawaii International Conference on System Sciences, Hawaii.
- Robson, C. (2002). *Real World Research* (2nd ed.). Malden, MA: Blackwell Publishing.
- Royce, W. W. (1970, August). *Managing the development of large software systems: Concepts and techniques [Electronic version]*. Paper presented at the Proceedings IEEE WESCON.
- Russell, R. S., & Taylor, B. W. (2003). *Operations management* (4th ed.). Upper Saddle River, NJ: Pearson Education.

- Saaty, T. L. (1977). A scaling method for priorities in hierarchical structures. *Journal of Mathematical Psychology*, 15(3), 234-281.
- Saaty, T. L. (1999). *Decision making for leaders: The analytic hierarchy process for decisions in a complex world*. Pittsburg, PA: RWS.
- Saaty, T. L. (2003). Decision-making with the AHP: Why is the principal eigenvector necessary. *European Journal of Operational Research*, 145, 85-91.
- Saiedian, H., & Dale, R. (2000). Requirements engineering: Making the connection between the software developer and customer. *Information and Software Technology*, 42(6), 419-428.
- Scacchi, W. (2001). Process models in software engineering. In J. J. Marciniak (Ed.), *Encyclopedia of Software Engineering* (2nd ed.). New York: John Wiley and Sons, Inc.
- Schmidt, R., Lyytinen, K., Keil, M., & Cule, P. (2001). Identifying software project risks: An international Delphi. *Journal of Management Information Systems*, 17(4), 5-36.
- Schniederjans, M. J., & Wilson, R. L. (1991). Using the analytic hierarchy process and goal programming for information system project selection. *Information Management*, 20(5), 333-342.
- Schwaber, K. (2004). *Agile project management with SCRUM*. Redmond, WA: Microsoft Press.
- SEI. (2004). *Software Engineering Institute Glossary*. Retrieved 5/12/2005, from <http://www.sei.cmu.edu/str/indexes/glossary>
- SEI. (2006). *What is CMMI*. Retrieved 06/24/2006, from <http://www.sei.cmu.edu/cmmi/general/general.html>
- Spillers, F. (2004a). *Electoral ethnography*. Retrieved November 6, 2004, 2004, from http://experiencedynamics.blogs.com/site_search_usability/2004/10/electoral_ethno.html
- Spillers, F. (2004b). Task analysis through cognitive archeology. In D. Diaper & N. Stanton (Eds.), *The Handbook of Task Analysis for HCI*. Mahway, New Jersey: Laurence Erlbaum Associates.
- Standish. (2002). *Chaos chronicles*. Retrieved 5/23/2004, from <http://www.standishgroup.com/chaos/intro1.php>
- Standish. (2005). *Chaos chronicles*. Retrieved 3/11/2005, from <http://www.standishgroup.com/chaos/intro1.php>
- Stary, C. (2002). Shifting knowledge from analysis to design: Requirements for contextual user interface development. *Behaviour & Information Technology*, 21(Issue 6), 425-440.

- Steuer, R. E., & Na, P. (2003). Multiple criteria decision making combined with finance: A categorized bibliographic study. *European Journal of Operational Research*, 150(3), 496-515.
- SurveyMonkey. (2006). *Web-based survey service*. Retrieved 5/9/2006, from www.surveymonkey.com
- Sutcliffe, A. (2003, September 8-12). *Scenario-based requirements engineering [Electronic version]*. Paper presented at the 11th IEEE International Requirements Engineering Conference, Monterey Bay, CA.
- Tiwana, A., & Keil, M. (2004). The one-minute risk assessment tool. *Communications of the ACM*, 47(11), 73-77.
- Vaidya, O. S., & Kumar, S. (2004). Analytic hierarchy process: An overview of applications. *European Journal of Operational Research*, in press, 1-29.
- Valacich, J. S., Dennis, A. R., & Connolly, T. (1994). Idea generation in computer-based groups: A new ending to an old story. *Organization Behavior and Human Decision Processes*, 57(3), 448-467.
- Valenti, S., Panti, M., & Cucchiarelli, A. (1998). Overcoming communication obstacles in user-analyst interaction for functional requirements elicitation. *SIGSOFT Softw. Eng. Notes*, 23(1), 50-55.
- Van de Yen, A. H., & Delbecq, A. L. (1974). The effectiveness of nominal, delphi, and interacting group decision making processes. *Academy of Management*, 17(4), 605-621.
- Walsh, P. (2003). Everything counts: Insurers may find that a holistic, collaborative approach to business helps ensure that all parts are working in harmony [Electronic version]. *Best's Review*, 104(3), 45.
- Wieggers, K. E. (2003). *Software requirements* (2nd ed.). Redmond, WA: Microsoft Press.
- Williams, A. (2002). *Assessing prototypes' role in design*. Paper presented at the Proceedings of the 20th Annual International Conference on Computer Documentation, Toronto, Ontario, Canada.
- Williams, L., & Cockburn, A. (2003). Agile software development: It's about feedback and change. *Computer*, June, 39-43.
- Williamson, K., & Healy, M. (2000). Deriving engineering software from requirements. *Journal of Intelligent Manufacturing*, 11(1), 3.
- Xia, W., & Lee, G. (2004). Grasping the complexity of IS development projects. *Commun. ACM*, 47(5), 68-74.

- Yang, Y. S., Jang, B. S., Yeun, Y. S., Lee, K. H., & Lee, K. Y. (2003). Quality function deployment-based optimization and exploration for ambiguity. *Journal of Engineering Design, 14*(1), 83-113.
- Yeh, Q.-J., & Tsai, C.-L. (2001). Two conflict potentials during IS development. *Information & Management, 39*(2), 135-149.
- Zrymiak, D. (2003). *Software quality function deployment: modifying the "house of quality" for software*. Retrieved 2/20/2005, from <http://software.isixsigma.com/library/content/c030709a.asp>

APPENDIX A: TEMPLATE FOR REQUESTING PARTICIPATION

Date: _____

Dear: _____

(Initial paragraph may be modified to reflect prior communications by phone or a referral from someone else.)

Permit me to introduce myself. I am a student at Capella University with extensive experience in the development of information systems. As part of my PhD work, I am researching and writing my dissertation on "Requirements Determination of Information Systems: User and Developer Perceptions of Factors Contributing to Misunderstandings." As an information systems professional (or, as appropriate, user involved in creating requirements for an information system), you are well aware of how important clearly understood requirements are. I am contacting you to ask that you participate in this research project.

I am asking that you join in a structured focus group with approximately six of your colleagues. I anticipate that the focus groups will require two hours of your time, to be scheduled during lunch, and delicious food will be provided. The focus group will be audio recorded to preserve the interaction, but I and an authorized research assistant will be the only ones with access to the recordings. A few weeks after the focus group meets, I will send you a survey that asks you to prioritize factors influencing requirement misunderstandings. The survey can be completed in less than fifteen minutes and you will receive a gift for providing your answers.

At any time, do not hesitate to ask me questions about this study. My contact information is shown below. I will contact your office later this week to discuss this project with you further.

I greatly appreciate your time and willingness to participate in this research. It is an important topic, and with your help, will benefit the future development of information systems. As a

participant, you will be among the first to see the results of this study, which I will provide your company in the form of an executive summary.

Please be aware that you are free to decide not to participate in this research, or may withdraw at anytime and request that information collected from you be destroyed. However, given the success of your company and its reliance on information systems, I hope that you will agree to be part of this project.

I look forward to talking with you soon,

Chad McAllister
PhD Candidate
719-559-1672
chad@ckmcallister.com

APPENDIX B: SURVEY INVITATION

Dear TBD Participant,

Several weeks ago you participated in a focus group with some of your peers where I asked you to discuss factors for misunderstanding requirements for information systems. As we discussed at that time, there would be a follow-up survey for you to complete, fulfilling your commitment to the research study.

I greatly appreciate you taking time to complete this in the next few days—maybe even this week if that is possible for you. Your responses are vital and the research study can not be completed until your survey results have been evaluated. For your time, you will receive a gift card to a coffee shop.

The survey is web-based. Please click on the link below to view the survey, which can be completed in approximately 15 to 30 minutes.

<http://www.surveymonkey.com/s.asp?u=874261789826&c=20D8>

Again, I and the entire research team thank you and appreciate you participating in this study. As before, your responses will remain anonymous and confidential.

Sincerely,

-Chad McAllister

Principal Researcher

719-559-1627

APPENDIX C: NOMINAL GROUP TECHNIQUE PROCEDURES

The instructions below for conducting the NGT small group sessions are adapted from the research performed by Havelka (1994) as well as discussions with Havelka (personal communication June 10, 2005) and the protocol provided by Dunham (1998). The statements for what the researcher will say to facilitate the NGT sessions are not intended to be repeated verbatim, but as an example of what should be shared.

Preparation

Before conducting an NGT session, a meeting room must be secured that can accommodate adequate space for up to ten participants and the facilitator. A table large enough to seat all the participants and allow them to view a flip chart is required.

The following supplies are needed:

1. A flip chart.
2. Masking tape, to hang flip chart pages on walls of the meeting room.
3. Pen and writing pad for each participant.
4. Completed flip chart page for the Step 1 discussion.
5. Participant sign-in form, as shown in Appendix D.
6. Handouts for the nominal question, as shown in Appendix E.
7. Handouts for voting for the factors, as shown in Appendix F.
8. Participant consent form, which is part of Appendix Q.

Opening Statement

An opening statement should include a cordial welcome, emphasize the importance of the group's task, the importance of each member's contribution, and a clear indication of the purpose of the meeting's output. Also, participants will be asked to sign an attendance sheet, as shown in Appendix D, to ensure the researcher has current contact information for sending the follow-on surveys. The following statement will be used:

My name is Chad McAllister. I know you have busy schedules and I appreciate the sacrifice you have made to be here. Your time will not be wasted because our objective is an important one. Together, we will identify factors contributing to why requirements for an information system are misunderstood.

Your participation in this meeting is part of a research study I am conducting that is overseen by a group of experts in requirements engineering, organizational improvement, marketing, and management. A hallmark of good research is ensuring participants understand the purpose of the study, what to expect, possible risks, how confidentiality will be protected, and where to ask questions—which is the purpose of an informed consent form. If you have not already signed the consent form, I'll be glad to answer any questions you have and take your signed forms now (researcher provides copies of consent form to those who need them and reviews consent form content with the participants).

Before we start exploring reasons for misunderstanding requirements, let's take a few minutes to learn more about each other. I'll start and then we'll go around the room. Please share your name, a description of your role in this organization, and something personally interesting about you that others are unlikely to know (researcher introduces himself).

We have all been involved in creating and using information systems, and from those experiences, we likely learned that the requirements for a system can easily be misunderstood. In fact, misunderstood requirements is frequently cited as the primary reason for information system projects falling behind schedule, exceeding budget, and failing to do what users needed. Our task is to think about why—why requirements are misunderstood.

In a few weeks, you will receive a survey by mail asking you to judge the importance of the factors. To make sure I know how to contact you, please write your name, phone number, email address, and mailing address on this sheet (researcher hands out sheet).

An overview of the activities to be completed during the group session should be presented at this time.

The Group Process

Step 1: Introduction of Participants and General Discussion of the Definition Stage. The group process will begin with a general discussion of the subject to be addressed by the group. The researcher will begin the discussion by giving the following description of creating requirements for an information system:

We should come to some understanding of the topic we are examining. Our focus is on requirements determination for an information system. This is typically the first phase in the development of an information system and is focused on creating the requirements for the system—what it needs to do. We will define requirements determination as (the researcher will already have definitions written on a flip chart page in view of all participants):

Requirements Determination – the eliciting, organizing, and understanding of requirements for an information system.

And we will define requirements as:

Requirements – user-related needs and objectives for the information system.

The researcher will then ask the participants if they have a different understanding of these terms. After discussing and clarifying the definitions used, the researcher will emphasize the importance of the topic without debating the specifics of the definitions or the overall objective of the research.

Step 2: Silent Generation of Ideas. The researcher will present the nominal question for the group to address. The question will be written on a worksheet with ample white space for note taking, as shown in Appendix C, and given to each participant. The researcher will begin

this step by making the statement: "Would each of you please look carefully at the question at the top of the worksheet that I am now handing out."

The researcher will then read aloud the nominal question, which is: "What factors do you feel influence misunderstanding requirements for an information system?"

The researcher will then make the following statement:

In responding to this question, I encourage you to share your insights based on your experiences with information systems in this organization.

For the next few minutes, please list your responses to this question, using brief phrases or a few words, on the worksheet in front of you. It is important that during this activity we work independently and quietly. After each of us has listed several factors, we will discuss them. For now, please list as many factors as you can. Are there any questions?

If there are no questions, the researcher will turn to his worksheet and begin to write, recording factors gained from literature and previous NGT focus sessions. The researcher will avoid questions that interpret the nominal question. Only questions regarding the process will be answered. The appropriate answer to questions suggesting certain answers to the nominal question is as follows: "Any factor that you believe influences the misunderstanding of requirements for an information system during requirements determination should be written on your worksheet."

Step 3: Round-Robin Factor Recording. After the silent recording of ideas, the researcher will ask each participant, in turn, for one factor. These factors will be listed on the flip chart, in bullet format, so all participants can see them. The researcher will continue asking for items in a round-robin fashion until all participants exhaust their lists. Some important guidelines for this step include: (a) a clear statement of the step's objective (i.e. the mapping of the group's thinking), (b) the ideas recorded should be in the form of short phrases or single words, (c)

identical duplicates should be omitted, but variations of themes is desirable, and (d) the mechanical recording of the ideas should be done clearly, quickly, and legibly.

As each item is listed it should be discussed. The objective in this step is to clarify the definition of the factor, not to win arguments. However, the discussion should include the logic behind the idea and help promote a clear understanding of the factor. The evaluation of the ideas is handled in the final NGT step.

To begin step 3, the researcher should make this statement:

Now we will discuss the factors we each listed. The purpose of the discussion is to clarify the meaning of each factor and understand the logic behind the factor. You should feel free to express varying points of view, agree, or disagree.

We will, however, want to pace ourselves so that each factor we share receives the opportunity for some attention, so we will only spend a few minutes discussing each factor.

Finally, let me point out that the author of a factor need not feel obligated to clarify or explain it. Any member of the group can share clarifying ideas.

To allow everyone to participate equally, we'll go around the table, with each of us taking a turn to share a factor from our list. If someone else in the group lists a factor that you have on your worksheet, cross it off of your list so we do not repeat the factor. If, however, in your judgment the factor on your worksheet contains a different emphasis or variation, please keep it on your list and share it when it is your turn. At anytime during the process you think of additional factors, please add them to your list.

(Turning to the first participant) Please share one factor from your list.

(Researcher numbers and records the factor verbatim on the flip chart page)
What does this factor mean and what is the logic behind it?

The researcher would then proceed to each participant, one at a time, asking for a factor from the participant's list. Each factor will be numbered and listed on the flip chart, recording clarifying remarks before proceeding to the next factor. Eventually, a participant will respond

that they have exhausted their list. The researcher will state: "That is fine, however, feel free to share again if another factor occurs to you."

The recording of factors on the flip chart should be done as quickly as possible, as the group may become impatient and disruptive if the process drags on. It is also important to list the factors in the exact words used by the participant. If factors are wordy, the researcher will ask the participant to shorten its description. Should disruptive behaviors occur during the listing of factors (i.e. attempts to discuss ideas, side conversations, etc.), the researcher should firmly ask for their cooperation. Should arguments arise, the researcher will intervene by saying "I think we understand both points of view at this time. Perhaps we should move on to the next factor" or "Perhaps we have identified different variations of the same factor, or different factors, and should consider them both separately."

After participants have shared all of their factors, the researcher will share additional factors not yet discussed that were discussed in previous focus groups and found in literature. This provides a way to carry on ideas from literature and other focus groups to allow each group of participants to build on previous ideas. The researcher will keep track of which factors he shared and asked the group to discuss versus those that the group provided.

Step 4. Individual Evaluation of Factors. The final step in the group process is an evaluation of the factors generated by each participant. This is done by asking the participants to vote for the most influential factors. To begin, each participant will be told to "vote" for the five most important factors that have been listed, using the voting worksheet shown in Appendix D. Since each factor is numbered, participants merely need to mark the five factors they feel are most influential in misunderstanding requirements. To begin this step, the researcher will make this statement:

Now we need to determine the most important factors involved in misunderstanding requirements. To accomplish this, please use the Voting Worksheet (researcher provides voting worksheet). Since each factor is numbered on the flip charts, simply mark the

number next to the factors you believe are most influential. Please choose up to five factors. You may choose fewer factors if you don't believe five are influential, but please don't choose more than five. Make your choices now and we'll tally the results in a few minutes.

The researcher does not participate in voting for factors, and waits until participants have made their choices. When the participants are done voting, the researcher will start with the first factor on the list and ask participants to raise their hand if they chose the factor. The researcher will record the number of votes next to the factor on the flip chart. Then, the researcher proceeds to the next factor, continuing through the entire list. After all votes have been recorded, the researcher will highlight the most important factors, stopping when a natural break in the votes occurs (e.g., a drop from 3 votes to 1 vote) or when the ten most-voted factors have been highlighted. If a natural break in the votes does not occur, the researcher will facilitate a discussion to identify the ten most influential factors, asking participants to revote for specific factors if necessary. If a natural break does occur, the researcher will facilitate a discussion to identify if additional factors should be considered, asking participants to vote again for specific factors if necessary.

To tally the votes, the researcher will say:

Does anyone want additional time to make their choices? (If not, proceed) To tally our votes and determine the most influential factors, let's go through each factor at a time, recording the number of times the factor was chosen. Raise your hand if you chose factor number 1. (The researcher counts the number of raised hands and records the number on the flip chart page next to the factor). Let's move on to factor number 2—please raise your hand if you chose factor number 2. (The tallying continues until votes for all factors have been recorded) Now, let's examine what we have done. It appears

The researcher identifies the occurrence or lack of a natural break in the factors that were selected as most influential and facilitates the discussion appropriately to select not more than ten factors as most influential, recording votes if re-voting is used.

To conclude the focus group, the researcher will thank the participants for their work and contributions that will help improve requirements determination processes and better understanding requirements for information systems. The participants will be reminded that they will receive a survey by mail in a few weeks, asking them to prioritize a list of aggregated factors created from several focus groups. The researcher will also ask for their commitment to complete the survey. To end the meeting, the researcher will say:

We have accomplished very important work together and I wish to thank each of you for your participation and staying with the process to the end. Our work may very well create the foundation for creating more successful information systems in the future.

The factors we defined today will be combined with the results from other organizations. In a few weeks you will receive by mail a survey asking you to prioritize the factors combined from all organizations. Consequently, you may see on the survey factors we did not discuss or factors that we discussed that are missing—this is because the most important factors from all organizations will be used. If you have any questions regarding the definition of a factor, my contact information will be on the survey and I encourage you to send me an email or call me.

I expect you will be able to complete the survey in 15 minutes or less. Does anyone expect to have difficulty getting the completed survey returned to me within a few days of receiving it? (If there is an objection, the researcher will determine the reason for the concern and work to eliminate it)

Just as a reminder, all information you provide will be kept completely anonymous and both the identity of your organization and your identify will remain anonymous in the findings of this research study. Again, thank you for your wonderful participation and expect to receive the survey in a few weeks.

APPENDIX F: FACTOR VOTING WORKSHEET

Instructions: Mark the numbers that corresponds to the factors you believe are most influential in misunderstanding requirements. Mark no more than 5 factors.

1. <input type="checkbox"/>	23. <input type="checkbox"/>
2. <input type="checkbox"/>	24. <input type="checkbox"/>
3. <input type="checkbox"/>	25. <input type="checkbox"/>
4. <input type="checkbox"/>	26. <input type="checkbox"/>
5. <input type="checkbox"/>	27. <input type="checkbox"/>
6. <input type="checkbox"/>	28. <input type="checkbox"/>
7. <input type="checkbox"/>	29. <input type="checkbox"/>
8. <input type="checkbox"/>	30. <input type="checkbox"/>
9. <input type="checkbox"/>	31. <input type="checkbox"/>
10. <input type="checkbox"/>	32. <input type="checkbox"/>
11. <input type="checkbox"/>	33. <input type="checkbox"/>
12. <input type="checkbox"/>	34. <input type="checkbox"/>
13. <input type="checkbox"/>	35. <input type="checkbox"/>
14. <input type="checkbox"/>	36. <input type="checkbox"/>
15. <input type="checkbox"/>	37. <input type="checkbox"/>
16. <input type="checkbox"/>	38. <input type="checkbox"/>
17. <input type="checkbox"/>	39. <input type="checkbox"/>
18. <input type="checkbox"/>	40. <input type="checkbox"/>
19. <input type="checkbox"/>	41. <input type="checkbox"/>
20. <input type="checkbox"/>	42. <input type="checkbox"/>
21. <input type="checkbox"/>	43. <input type="checkbox"/>
22. <input type="checkbox"/>	44. <input type="checkbox"/>

APPENDIX G: USER SURVEY

Overview

The following survey is part of a research study investigating ways to better understand requirements for information systems. You are being asked to complete this survey because of your prior participation in a focus group that helped to determine the most influential factors that cause requirements to be misunderstood. This final portion of the study will prioritize the importance of the factors.

Time

Most people will complete this survey in 15 minutes to 30 minutes. Please take time to complete it now.

Voluntary Participation and Risks

Your participation in this study is voluntary and you may terminate your participation at any time without any consequence to you or your organization. You will not be at physical or psychological risk during the study procedures. The information you provide via this survey will be kept strictly confidential by the research team.

Benefits:

Your participation will benefit your organization and the information systems community by aiding our understanding of requirement issues. You will receive a gift card as a token of appreciation for completing and returning the survey.

Confidentiality:

Your survey responses will be associated with your name to aid in collecting surveys from all participants, but this association will only be known by the research team and not made available in any published results. Your responses will be aggregated with those of the other participants in this study, further protecting your anonymity. This is done to ensure your responses remain confidential so you can respond as freely as possible.

Questions:

If you have any questions about this study, please contact the principal researcher and PhD student, Chad McAllister, at his office in Colorado by phone at 719-559-1627 or by email at chad@ckmcallister.com. You may also contact the supervising researcher, Dr. John Latham, by email at john@drjohnlatham.com. If you have any questions or concerns about your rights as a research participant, you may contact the Capella University Institutional Review Board Director, Kurt Linberg, at 1-888-227-2736.

Instructions

Please supply the following general information and then follow the directions given later for prioritizing the factors. All questions should be answered.

A. Title of your current position or role: _____

B. How many information systems have you helped to create:

- 1 2 to 3 4 to 6 7 or more

C. How long have you been involved in creating information systems? ____ years ____ months

D. Which development approach was most often used for creating these information systems:

- Unknown Waterfall Incremental Iterative Agile XP

_____ Other:

E. How would you characterize your relationship with the developers of your information systems:

- Very good Good Neutral Poor Very poor I don't have a relationship

Instructions for the Remaining Questions

In the focus group you attended, you were asked, "What factors do you feel influence misunderstanding requirements for an information system?" Now, your objective is to prioritize some of the factors you discussed. You will be prioritizing 10 factors, two at a time.

The remaining 36 questions ask you to choose between two factors at a time. For example, you will be asked to choose between Factor 1 and Factor 2, then Factor 1 again and Factor 3, then Factor 1 again and Factor 4, etc.

Select the factor you feel has more influence on misunderstanding requirements for an information system—the factor that is more important. You will then be asked to indicate how

much more important this factor is compared to the other one.

If you have a question about the definition of a factor, please contact Chad McAllister by phone at 719-559-1627 or by email at chad@ckmcallister.com.

The Factors

Details of the factors are described below and on the next page. You will be able to view these details anytime you need to, but please read them now.

Factor 1. Key users who have the information for determining requirements are not appropriately involved in requirements determination. This was expressed as:

- Correct people are not involved in requirements
- People involved in requirements determination may not be the right people
- Key people start late on the project
- Key people are not invited
- Universe of affected users is not adequately identified or understood

Factor 2. Users don't know what is possible. This was expressed as:

- Users don't understand available technology options
- Users do not realize what can be done and can't be done

Factor 3. Deadlines drive projects, leaving inadequate time for requirements determination with the resources available. This was expressed as:

- Not having enough requirements defined before development starts
- Time/schedule drives projects
- Lack of appropriate analyst resources

Factor 4. People do not understand their role and the roles of others in requirements determination. This was expressed as:

- Unclear roles and responsibilities

- Managers make decisions without understanding users' needs
- Some users have the belief that requirements determination is a waste of time

Factor 5. Requirements change during the creation of an information system and the changes are not adequately addressed. This was expressed as:

- Needs change during the course of a project; 2 years
- Business needs can change over course of development

Factor 6. Developers/IT lack knowledge of the business. This was expressed as:

- Business analysts/developers don't know business
- Lack of broad knowledge of organization to make decisions
- Not understanding interaction between systems/business processes

Factor 7. Users are unclear about their needs and the priorities of those needs. This was expressed as:

- Users poorly articulate requirements
- Users don't begin with the end in mind
- Necessity versus nicety
- Unclear system scope
- Users have conflicting needs; users use the system differently

Factor 8. Users' experience with current systems limits their ability to create requirements for a new system. This was expressed as:

- User confusion between business process and requirements; how versus need
- Users tend to be unable to separate what is currently in the system from what they need
- Users get stuck on interim solution thinking it is long term solution

Factor 9. Users and developers relate to each other differently. This was expressed as:

- Terminology is difficult—user and developer jargon
- IT and users communicate differently
- User/IT frame of reference
- No communication between correct user and correct developer

Question 1: Select the factor you feel is more important.

Factor 1. Key users who have the information for determining requirements are not appropriately involved in requirements determination.

Factor 2. Users don't know what is possible.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 2: Select the factor you feel is more important.

Factor 1. Key users who have the information for determining requirements are not appropriately involved in requirements determination.

Factor 3. Deadlines drive projects, leaving inadequate time for requirements determination with the resources available.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 3: Select the factor you feel is more important.

Factor 1. Key users who have the information for determining requirements are not appropriately involved in requirements determination.

Factor 4. People do not understand their role and the roles of others in requirements determination.

- How much more importance does the selected factor have:
- Equal importance
 - Moderate importance
 - Strong importance
 - Very strong importance
 - Extreme importance

Question 4: Select the factor you feel is more important.

Factor 1. Key users who have the information for determining requirements are not appropriately involved in requirements determination.

Factor 5. Requirements change during the creation of an information system and the changes are not adequately addressed.

- How much more importance does the selected factor have:
- Equal importance
 - Moderate importance
 - Strong importance
 - Very strong importance
 - Extreme importance

Question 5: Select the factor you feel is more important.

Factor 1. Key users who have the information for determining requirements are not appropriately involved in requirements determination.

Factor 6. Developers/IT lack knowledge of the business.

- How much more importance does the selected factor have:
- Equal importance
 - Moderate importance
 - Strong importance
 - Very strong importance
 - Extreme importance

Question 6: Select the factor you feel is more important.

Factor 1. Key users who have the information for determining requirements are not appropriately involved in requirements determination.

Factor 7. Users are unclear about their needs and the priorities of those needs.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 7: Select the factor you feel is more important.

Factor 1. Key users who have the information for determining requirements are not appropriately involved in requirements determination.

Factor 8. Users' experience with current systems limits their ability to create requirements for a new system.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 8: Select the factor you feel is more important.

Factor 1. Key users who have the information for determining requirements are not appropriately involved in requirements determination.

Factor 9. Users and developers relate to each other differently.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 9: Select the factor you feel is more important.

Factor 2. Users don't know what is possible.

Factor 3. Deadlines drive projects, leaving inadequate time for requirements determination with the resources available.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 10: Select the factor you feel is more important.

Factor 2. Users don't know what is possible.

Factor 4. People do not understand their role and the roles of others in requirements determination.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 11: Select the factor you feel is more important.

Factor 2. Users don't know what is possible.

Factor 5. Requirements change during the creation of an information system and the changes are not adequately addressed.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 12: Select the factor you feel is more important.

Factor 2. Users don't know what is possible.

Factor 6. Developers/IT lack knowledge of the business.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 13: Select the factor you feel is more important.

Factor 2. Users don't know what is possible.

Factor 7. Users are unclear about their needs and the priorities of those needs.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 14: Select the factor you feel is more important.

Factor 2. Users don't know what is possible.

Factor 8. Users' experience with current systems limits their ability to create requirements for a new system.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 15: Select the factor you feel is more important.

Factor 2. Users don't know what is possible.

Factor 9. Users and developers relate to each other differently.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 16: Select the factor you feel is more important.

Factor 3. Deadlines drive projects, leaving inadequate time for requirements determination with the resources available.

Factor 4. People do not understand their role and the roles of others in requirements determination.

How much more importance does the selected factor have:

- Equal Moderate Strong Very strong Extreme

importance importance importance importance importance

Question 17: Select the factor you feel is more important.

Factor 3. Deadlines drive projects, leaving inadequate time for requirements determination with the resources available.

Factor 5. Requirements change during the creation of an information system and the changes are not adequately addressed.

How much more importance does the selected factor have:

- Equal importance
- Moderate importance
- Strong importance
- Very strong importance
- Extreme importance

Question 18: Select the factor you feel is more important.

Factor 3. Deadlines drive projects, leaving inadequate time for requirements determination with the resources available.

Factor 6. Developers/IT lack knowledge of the business.

How much more importance does the selected factor have:

- Equal importance
- Moderate importance
- Strong importance
- Very strong importance
- Extreme importance

Question 19: Select the factor you feel is more important.

Factor 3. Deadlines drive projects, leaving inadequate time for requirements determination with the resources available.

Factor 7. Users are unclear about their needs and the priorities of those needs.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 20: Select the factor you feel is more important.

Factor 3. Deadlines drive projects, leaving inadequate time for requirements determination with the resources available.

Factor 8. Users' experience with current systems limits their ability to create requirements for a new system.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 21: Select the factor you feel is more important.

Factor 3. Deadlines drive projects, leaving inadequate time for requirements determination with the resources available.

Factor 9. Users and developers relate to each other differently.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 22: Select the factor you feel is more important.

Factor 4. People do not understand their role and the roles of others in requirements determination.

Factor 5. Requirements change during the creation of an information system and the changes are not adequately addressed.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 23: Select the factor you feel is more important.

Factor 4. People do not understand their role and the roles of others in requirements determination.

Factor 6. Developers/IT lack knowledge of the business.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 24: Select the factor you feel is more important.

Factor 4. People do not understand their role and the roles of others in requirements determination.

Factor 7. Users are unclear about their needs and the priorities of those needs.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 25: Select the factor you feel is more important.

Factor 4. People do not understand their role and the roles of others in requirements determination.

Factor 8. Users' experience with current systems limits their ability to create requirements for a new system.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 26: Select the factor you feel is more important.

Factor 4. People do not understand their role and the roles of others in requirements determination.

Factor 9. Users and developers relate to each other differently.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 27: Select the factor you feel is more important.

Factor 5. Requirements change during the creation of an information system and the changes are not adequately addressed.

Factor 6. Developers/IT lack knowledge of the business.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 28: Select the factor you feel is more important.

Factor 5. Requirements change during the creation of an information system and the changes are not adequately addressed.

Factor 7. Users are unclear about their needs and the priorities of those needs.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 29: Select the factor you feel is more important.

Factor 5. Requirements change during the creation of an information system and the changes are not adequately addressed.

Factor 8. Users' experience with current systems limits their ability to create requirements for a new system.

How much more importance does the selected factor have:

- Equal Moderate Strong Very strong Extreme

importance importance importance importance importance

Question 30: Select the factor you feel is more important.

Factor 5. Requirements change during the creation of an information system and the changes are not adequately addressed.

Factor 9. Users and developers relate to each other differently.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 31: Select the factor you feel is more important.

Factor 6. Developers/IT lack knowledge of the business.

Factor 7. Users are unclear about their needs and the priorities of those needs.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 32: Select the factor you feel is more important.

Factor 6. Developers/IT lack knowledge of the business.

Factor 8. Users' experience with current systems limits their ability to create requirements for a new system.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 33: Select the factor you feel is more important.

Factor 6. Developers/IT lack knowledge of the business.

Factor 9. Users and developers relate to each other differently.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 34: Select the factor you feel is more important.

Factor 7. Users are unclear about their needs and the priorities of those needs.

Factor 8. Users' experience with current systems limits their ability to create requirements for a new system.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 35: Select the factor you feel is more important.

Factor 7. Users are unclear about their needs and the priorities of those needs.

Factor 9. Users and developers relate to each other differently.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 36: Select the factor you feel is more important.

Factor 8. Users' experience with current systems limits their ability to create requirements for a new system.

Factor 9. Users and developers relate to each other differently.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Thank you for participating in this research study. Your involvement has made valuable contributions to our knowledge of misunderstanding requirements.

APPENDIX H: DEVELOPER SURVEY

Overview

The following survey is part of a research study investigating ways to better understand requirements for information systems. You are being asked to complete this survey because of your prior participation in a focus group that helped to determine the most influential factors that cause requirements to be misunderstood. This final portion of the study will prioritize the importance of the factors.

Time

Most people will complete this survey in 15 minutes to 30 minutes. Please take time to complete it now.

Voluntary Participation and Risks

Your participation in this study is voluntary and you may terminate your participation at any time without any consequence to you or your organization. You will not be at physical or psychological risk during the study procedures. The information you provide via this survey will be kept strictly confidential by the research team.

Benefits:

Your participation will benefit your organization and the information systems community by aiding our understanding of requirement issues. You will receive a \$5 gift card as a token of appreciation for completing and returning the survey.

Confidentiality:

Your survey responses will be associated with your name to aid in collecting surveys from all participants, but this association will only be known by the research team and not made available in any published results. Your responses will be aggregated with those of the other participants in this study, further protecting your anonymity. This is done to ensure your responses remain confidential so you can respond as freely as possible.

Questions:

If you have any questions about this study, please contact the principal researcher and PhD student, Chad McAllister, at his office in Colorado by phone at 719-559-1627 or by email at chad@ckmcallister.com. You may also contact the supervising researcher, Dr. John Latham, by email at john@drjohnlatham.com. If you have any questions or concerns about your rights as a research participant, you may contact the Capella University Institutional Review Board Director, Kurt Linberg, at 1-888-227-2736.

Instructions

Please supply the following general information and then follow the directions given later for prioritizing the factors. All questions should be answered.

A. Title of your current position or role: _____

B. How many information systems have you helped to create:

- 1 2 to 3 4 to 6 7 or more

C. How long have you been involved in creating information systems? ____ years ____ months

D. Which development approach was most often used for creating these information systems:

- Unknown Waterfall Incremental Iterative Agile XP

_____ Other:

E. How would you characterize your relationship with the users of your information systems:

- Very good Good Neutral Poor Very poor I don't have a relationship

Instructions for the Remaining Questions

In the focus group you attended, you were asked, "What factors do you feel influence misunderstanding requirements for an information system?" Now, your objective is to prioritize some of the factors you discussed. You will be prioritizing 10 factors, two at a time.

The remaining 45 questions ask you to choose between two factors at a time. For example, you will be asked to choose between Factor 1 and Factor 2, then Factor 1 again and Factor 3, then Factor 1 again and Factor 4, etc.

Select the factor you feel has more influence on misunderstanding requirements for an information system—the factor that is more important. You will then be asked to indicate how much more important this factor is compared to the other one.

If you have a question about the definition of a factor, please contact Chad McAllister by phone at 719-559-1627 or by email at chad@ckmcallister.com.

The Factors

Details of the factors are described below and on the next page. You will be able to view these details anytime you need to, but please read them now.

Factor 1. Users are uncertain about what they want and have difficulty articulating the problem and their needs. This was expressed as:

- Users don't know what they want
- Users are not sure of what they want; are vague about needs; don't understand the problem
- Users don't understand requirements; can't articulate requirements
- Users over simplify requirements
- Users begin with a preconceived notion of the solution

Factor 2. Developers and users communicate poorly. This was expressed as:

- Assumptions are made by developers and users
- Poor communication
- The telephone game: users to business analysts to developers communications
- IT and users lack of listening skills
- Users'/Managers' requirements leave out detail
- English is not the users' first language

Factor 3. Requirements prepared by developers are inadequately reviewed by users. This was expressed as:

- Lack of requirements review
- Lack of prototypes; users need a visual representation

Factor 4. Users and developers have different perspectives and translation is necessary for one group to understand the other. This was expressed as:

- Differences exist between users' and developers' viewpoint
- Translation between IT and Business is needed
- Developers - business analysts - user relationships can help or hinder requirements
- Team dynamics can help or hinder requirements determination
- Technology is complicated

Factor 5. Key users who have the information for determining requirements are not available or do not stay involved during the project. This was expressed as:

- Not getting input from key users
- Key players not always involved in requirements determination
- Lack of ongoing user involvement during design
- Appropriate people are not included; insufficient proxy for users
- Managers are not enabling users to have time for requirements determination
- Creating requirements is not a priority for customers; time is not allocated
- Business units do not create business plans
- Organizational laziness with requirements determination

Factor 6. Users impose unreasonable schedules and rush requirements determination. This was expressed as:

- Users rush requirements definitions; creates a quality problem

- Fixed deadlines; deadlines impose change in requirements

Factor 7. Users lack an understanding of how their need for an information system fits into their organizations' objectives and strategy. This was expressed as:

- Users lack an understanding of the big picture
- Lack of strategic plans for the organization
- Lack of enterprise planning; being able to understanding the whole

Factor 8. Users and developers do not maintain consistent and useful requirements documentation. This was expressed as:

- Lack of requirement documents; verbal versus written requirements
- Formal versus informal requirements
- Adequate requirements documentation

Factor 9. Developers do not have sufficient knowledge of the organization's business. This was expressed as:

- IT lacks business knowledge
- Developers lack of domain knowledge

Factor 10. A predictable requirements determination process is not used or is not understood by users and developers. This was expressed as:

- No formal organization-wide software development process
- Scope creep

Question 1: Select the factor you feel is more important.

Factor 1. Users are uncertain about what they want and have difficulty articulating the problem and their needs.

Factor 2. Developers and users communicate poorly.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 2: Select the factor you feel is more important.

Factor 1. Users are uncertain about what they want and have difficulty articulating the problem and their needs.

Factor 3. Requirements prepared by developers are inadequately reviewed by users.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 3: Select the factor you feel is more important.

Factor 1. Users are uncertain about what they want and have difficulty articulating the problem and their needs.

Factor 4. Users and developers have different perspectives and translation is necessary for one group to understand the other.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 4: Select the factor you feel is more important.

Factor 1. Users are uncertain about what they want and have difficulty articulating the problem and their needs.

Factor 5. Key users who have the information for determining requirements are not available or do not stay involved during the project.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 5: Select the factor you feel is more important.

Factor 1. Users are uncertain about what they want and have difficulty articulating the problem and their needs.

Factor 6. Users impose unreasonable schedules and rush requirements determination.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 6: Select the factor you feel is more important.

Factor 1. Users are uncertain about what they want and have difficulty articulating the problem and their needs.

Factor 7. Users lack an understanding of how their need for an information system fits into their organizations' objectives and strategy.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 7: Select the factor you feel is more important.

Factor 1. Users are uncertain about what they want and have difficulty articulating the problem and their needs.

Factor 8. Users and developers do not maintain consistent and useful requirements documentation.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 8: Select the factor you feel is more important.

Factor 1. Users are uncertain about what they want and have difficulty articulating the problem and their needs.

Factor 9. Developers do not have sufficient knowledge of the organization's business.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 9: Select the factor you feel is more important.

Factor 1. Users are uncertain about what they want and have difficulty articulating the problem and their needs.

Factor 10. A predictable requirements determination process is not used or is not understood by users and developers.

- How much more importance does the selected factor have:
- Equal importance
 - Moderate importance
 - Strong importance
 - Very strong importance
 - Extreme importance

Question 10: Select the factor you feel is more important.

Factor 2. Developers and users communicate poorly.

Factor 3. Requirements prepared by developers are inadequately reviewed by users.

- How much more importance does the selected factor have:
- Equal importance
 - Moderate importance
 - Strong importance
 - Very strong importance
 - Extreme importance

Question 11: Select the factor you feel is more important.

Factor 2. Developers and users communicate poorly.

Factor 4. Users and developers have different perspectives and translation is necessary for one group to understand the other.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 12: Select the factor you feel is more important.

Factor 2. Developers and users communicate poorly.

Factor 5. Key users who have the information for determining requirements are not available or do not stay involved during the project.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 13: Select the factor you feel is more important.

Factor 2. Developers and users communicate poorly.

Factor 6. Users impose unreasonable schedules and rush requirements determination.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 14: Select the factor you feel is more important.

Factor 2. Developers and users communicate poorly.

Factor 7. Users lack an understanding of how their need for an information system fits into their organizations' objectives and strategy.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 15: Select the factor you feel is more important.

Factor 2. Developers and users communicate poorly.

Factor 8. Users and developers do not maintain consistent and useful requirements documentation.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 16: Select the factor you feel is more important.

Factor 2. Developers and users communicate poorly.

Factor 9. Developers do not have sufficient knowledge of the organization's business.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 17: Select the factor you feel is more important.

Factor 2. Developers and users communicate poorly.

Factor 10. A predictable requirements determination process is not used or is not understood by users and developers.

- How much more importance does the selected factor have:
- Equal importance
 - Moderate importance
 - Strong importance
 - Very strong importance
 - Extreme importance

Question 18: Select the factor you feel is more important.

Factor 3. Requirements prepared by developers are inadequately reviewed by users.

Factor 4. Users and developers have different perspectives and translation is necessary for one group to understand the other.

- How much more importance does the selected factor have:
- Equal importance
 - Moderate importance
 - Strong importance
 - Very strong importance
 - Extreme importance

Question 19: Select the factor you feel is more important.

Factor 3. Requirements prepared by developers are inadequately reviewed by users.

Factor 5. Key users who have the information for determining requirements are not available or do not stay involved during the project.

- How much more importance does the selected factor have:
- Equal importance
 - Moderate importance
 - Strong importance
 - Very strong importance
 - Extreme importance

Question 20: Select the factor you feel is more important.

Factor 3. Requirements prepared by developers are inadequately reviewed by users.

Factor 6. Users impose unreasonable schedules and rush requirements determination.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 21: Select the factor you feel is more important.

Factor 3. Requirements prepared by developers are inadequately reviewed by users.

Factor 7. Users lack an understanding of how their need for an information system fits into their organizations' objectives and strategy.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 22: Select the factor you feel is more important.

Factor 3. Requirements prepared by developers are inadequately reviewed by users.

Factor 8. Users and developers do not maintain consistent and useful requirements documentation.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 23: Select the factor you feel is more important.

Factor 3. Requirements prepared by developers are inadequately reviewed by users.

Factor 9. Developers do not have sufficient knowledge of the organization's business.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 24: Select the factor you feel is more important.

Factor 3. Requirements prepared by developers are inadequately reviewed by users.

Factor 10. A predictable requirements determination process is not used or is not understood by users and developers.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 25: Select the factor you feel is more important.

Factor 4. Users and developers have different perspectives and translation is necessary for one group to understand the other.

Factor 5. Key users who have the information for determining requirements are not available or do not stay involved during the project.

- How much more importance does the selected factor have:
- Equal importance
 - Moderate importance
 - Strong importance
 - Very strong importance
 - Extreme importance

Question 26: Select the factor you feel is more important.

Factor 4. Users and developers have different perspectives and translation is necessary for one group to understand the other.

Factor 6. Users impose unreasonable schedules and rush requirements determination.

- How much more importance does the selected factor have:
- Equal importance
 - Moderate importance
 - Strong importance
 - Very strong importance
 - Extreme importance

Question 27: Select the factor you feel is more important.

Factor 4. Users and developers have different perspectives and translation is necessary for one group to understand the other.

Factor 7. Users lack an understanding of how their need for an information system fits into their organizations' objectives and strategy.

- How much more importance does the selected factor have:
- Equal importance
 - Moderate importance
 - Strong importance
 - Very strong importance
 - Extreme importance

Question 28: Select the factor you feel is more important.

Factor 4. Users and developers have different perspectives and translation is necessary for one group to understand the other.

Factor 8. Users and developers do not maintain consistent and useful requirements documentation.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 29: Select the factor you feel is more important.

Factor 4. Users and developers have different perspectives and translation is necessary for one group to understand the other.

Factor 9. Developers do not have sufficient knowledge of the organization's business.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 30: Select the factor you feel is more important.

Factor 4. Users and developers have different perspectives and translation is necessary for one group to understand the other.

Factor 10. A predictable requirements determination process is not used or is not understood by users and developers.

- How much more importance does the selected factor have:
- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 31: Select the factor you feel is more important.

Factor 5. Key users who have the information for determining requirements are not available or do not stay involved during the project.

Factor 6. Users impose unreasonable schedules and rush requirements determination.

- How much more importance does the selected factor have:
- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 32: Select the factor you feel is more important.

Factor 5. Key users who have the information for determining requirements are not available or do not stay involved during the project.

Factor 7. Users lack an understanding of how their need for an information system fits into their organizations' objectives and strategy.

- How much more importance does the selected factor have:
- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 33: Select the factor you feel is more important.

Factor 5. Key users who have the information for determining requirements are not available or do not stay involved during the project.

Factor 8. Users and developers do not maintain consistent and useful requirements documentation.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 34: Select the factor you feel is more important.

Factor 5. Key users who have the information for determining requirements are not available or do not stay involved during the project.

Factor 9. Developers do not have sufficient knowledge of the organization's business.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 35: Select the factor you feel is more important.

Factor 5. Key users who have the information for determining requirements are not available or do not stay involved during the project.

Factor 10. A predictable requirements determination process is not used or is not understood by users and developers.

- How much more importance does the selected factor have:
- Equal importance
 - Moderate importance
 - Strong importance
 - Very strong importance
 - Extreme importance

Question 36: Select the factor you feel is more important.

Factor 6. Users impose unreasonable schedules and rush requirements determination.

Factor 7. Users lack an understanding of how their need for an information system fits into their organizations' objectives and strategy.

- How much more importance does the selected factor have:
- Equal importance
 - Moderate importance
 - Strong importance
 - Very strong importance
 - Extreme importance

Question 37: Select the factor you feel is more important.

Factor 6. Users impose unreasonable schedules and rush requirements determination.

Factor 8. Users and developers do not maintain consistent and useful requirements documentation.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 38: Select the factor you feel is more important.

Factor 6. Users impose unreasonable schedules and rush requirements determination.

Factor 9. Developers do not have sufficient knowledge of the organization's business.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 39: Select the factor you feel is more important.

Factor 6. Users impose unreasonable schedules and rush requirements determination.

Factor 10. A predictable requirements determination process is not used or is not understood by users and developers.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 40: Select the factor you feel is more important.

Factor 7. Users lack an understanding of how their need for an information system fits into their organizations' objectives and strategy.

Factor 8. Users and developers do not maintain consistent and useful requirements documentation.

- How much more importance does the selected factor have:
- Equal importance
 - Moderate importance
 - Strong importance
 - Very strong importance
 - Extreme importance

Question 41: Select the factor you feel is more important.

Factor 7. Users lack an understanding of how their need for an information system fits into their organizations' objectives and strategy.

Factor 9. Developers do not have sufficient knowledge of the organization's business.

- How much more importance does the selected factor have:
- Equal importance
 - Moderate importance
 - Strong importance
 - Very strong importance
 - Extreme importance

Question 42: Select the factor you feel is more important.

Factor 7. Users lack an understanding of how their need for an information system fits into their organizations' objectives and strategy.

Factor 10. A predictable requirements determination process is not used or is not understood by users and developers.

- How much more importance does the selected factor have:
- Equal importance
 - Moderate importance
 - Strong importance
 - Very strong importance
 - Extreme importance

Question 43: Select the factor you feel is more important.

Factor 8. Users and developers do not maintain consistent and useful requirements documentation.

Factor 9. Developers do not have sufficient knowledge of the organization's business.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 44: Select the factor you feel is more important.

Factor 8. Users and developers do not maintain consistent and useful requirements documentation.

Factor 10. A predictable requirements determination process is not used or is not understood by users and developers.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Question 45: Select the factor you feel is more important.

Factor 9. Developers do not have sufficient knowledge of the organization's business.

Factor 10. A predictable requirements determination process is not used or is not understood by users and developers.

How much more importance does the selected factor have:

- Equal importance Moderate importance Strong importance Very strong importance Extreme importance

Thank you for participating in this research study. Your involvement has made valuable contributions to our knowledge of misunderstanding requirements.

APPENDIX I: ALL FACTORS CREATED BY USERS

Vote	Factor Description
100%	Lack of appropriate analyst resources.
71%	Users poorly articulate requirements.
50%	Correct people are not involved in requirements
50%	Time/schedule drives projects
50%	Users don't understand available technology options
44%	Key people are not invited
44%	Key people start late on the project
43%	Terminology is difficult—user and developer jargon.
43%	Users tend to be unable to separate what is currently in the system from what they need.
33%	Business analysts/developers don't know business
33%	Business needs can change over course of development
33%	IT and users communicate differently
33%	Managers make decisions without understanding users' needs
33%	Needs change during the course of a project—2 years
33%	Not having enough requirements defined before development starts
33%	Not understanding interaction between systems/business processes
33%	Unclear roles and responsibilities
33%	Universe of affected users is not adequately identified or understood
33%	Users don't begin with the end in mind
33%	Users get stuck on interim solution thinking it is long term solution
33%	Users have conflicting needs; users use the system differently

Vote	Factor Description
29%	No communication between correct user and correct developer.
29%	People involved in requirements determination may not be the right people.
29%	Some users have the belief that requirements determination is a waste of time.
29%	Unclear system scope.
29%	User/IT frame of reference.
29%	Users do not realize what can be done and can't be done.
22%	Lack of broad knowledge of organization to make decisions
22%	Necessity versus nicety
22%	User confusion between business process and requirements; how versus need
17%	Details are not identified
17%	Developers make assumptions
17%	Global workforce makes communications difficult
17%	IT is not sufficiently investigating software
17%	No standard process for requirements determination
17%	Requirements documentation is not robust
17%	Users and business analysts are not trained in requirements determination
17%	Users expect developers/business analysts will guide requirements
17%	Users, business analysts and developers speak differently based on perspective
14%	Developers, business analysts, or whoever does requirements determination have their own bias.
14%	ITs fails to observe users at work.
14%	Needs change over course of project.
14%	Predetermination of technology to use not based on requirements.
14%	Users not considering related areas—the exceptions.
11%	Communications prior to collecting requirements
11%	False assumptions are made by users and IT

Factors Contributing to Misunderstanding Requirements

245

Vote	Factor Description
11%	IT is unfamiliar with business terms
11%	IT/Vendors don't understand business
11%	Key people are not available
11%	Lack of a business case
11%	Lack of requirements review
11%	Lack of understanding and communicating between user groups
11%	Lack of user knowledge of best tools (technology) available.
11%	Needs are lost in translation
11%	Priorities are made by upper management without consulting the project team
11%	Upper management alters requirements
11%	Users do not know what developers need to hear
11%	Users use terms differently
0%	Business analysts/developers are not approachable
0%	Changes in requirements are hard to re-review
0%	Conflicting needs of users.
0%	Design work is attempted during requirements determination.
0%	Developers are not involved early enough
0%	Exceptions to the standard business process are not identified
0%	Failure to get the users time—allocation of resources.
0%	Getting consensus from users officially part of a project.
0%	Getting users involved who should be involved.
0%	IT chooses software for selfish reasons at times
0%	IT does not see a need for change or understand what is wrong with current system
0%	IT has been told they don't need to understand the users' business.
0%	IT hides capabilities; don't have time to handle

Vote	Factor Description
0%	IT is unrealistic; over promises
0%	Lack of cooperation between groups of users
0%	Lack of willingness to change
0%	Outside agents influence project.
0%	People transition during the course of a project
0%	Poor communication within IT staff (intra communication)
0%	Requirement reviews don't always happen
0%	Requirements are communicated informally
0%	Supervisors may not be the best people to provide requirements.
0%	Terminology differences
0%	Trying to fit new requirements into an existing system.
0%	Users are not given enough time to think through requirements
0%	Users believe it's IT's job to develop requirements.
0%	Users don't have sufficient time for requirements work
0%	Users don't know what tools IT has.
0%	Users don't understand ultimate goal
0%	Users have difficulty articulating needs
0%	Users neglect / are ignorant of behind the scenes requirements.
0%	Wrong/bad/incomplete requirements

APPENDIX J: USER THEMES BASED ON DISCUSSION IN FOCUS GROUPS

ID	Description of Theme	Number of Orgs	Support
UT1	User-Developer Translation: Users and developers speak differently; Business analysts do not ask enough questions to help users uncover details the developers will need; Developers are asking users to be analysts (developers do not want to do it); Users are not qualified to be analysts—users do not understand what developers need; Someone who can translate between users and developers is needed.	3	60%
UT2	Developers Lack Understanding of the Business: Business analysts and developers do not understand the business; Developers who know the business well understand users' needs better; Developers who know the business well may be too biased and believe they know better than the user; Developers and technology vendors do not understand what a group of users do.	3	59%
UT3	Effect of Time: Project time-constraints limit functionality delivered to users; Requirements are gathered too quickly in order to meet schedule deadlines; Enhancements take too long for users to see; Users' needs change over the course of a project; Requirements grow stale if they are not worked on immediately; During a project changes are not effectively incorporated—treated as enhancement requests after the project will complete; Developers blame users for changing their mind when it is not their fault—changes occurred over the course of the project; Business needs can change while an information system is being developed; Requirements are not dynamically changing as business needs change during the course of a project.	3	58%
UT4	Key Users Are Not Involved in Requirements: Key people are not available when they are needed; The necessary expertise who know what is needed is missing; Failing to get consensus from users who should be involved (groups that were not considered); Managers, who are often involved in requirements determination, may not be representative of what is needed—line people need to be involved; Represented users lack big-picture understanding; Not all groups affected by the system are properly identified and involved in the project; There is a lack of broad knowledge of the organization to know who will be impacted by an information system.	3	50%
UT5	Articulation Difficulties: Users do not provide enough detail to adequately describe their needs; Users do not begin with the end in mind—do not clarify articulate the goal; Users understand current process but do not understand the totality of their knowledge when describing needs to a developer; Users say one thing but are really thinking another; Users have problems conveying what they are visualizing.	2	50%
UT6	User versus Developer Frame of Reference: Developers need details while users want an overview; Users have a totally different way of looking at a	2	41%

Factors Contributing to Misunderstanding Requirements 248

ID	Description of Theme	Number of Orgs	Support
	problem than a developer does; Need to have an analyst to be a middle-man, a translator; Users do not understand developers; Developers do not understand users.		
UT7	Users Do not Understand Technology: Users do not fully understand the options available to meet their needs; Users do not understand what can and cannot be done; Users are not inclined to learn more about technology; Users lack knowledge of what would be the best tools or technology to help solve their problem; Users do not know what questions to ask.	3	35%
UT8	Big Picture Understanding of the Problem: Users do not always fully understand the problem or need; Users do not know how one system interacts with another, but expect developers to; Users do not create business case to justify and explain problem and requirements; Users do not understand or are not given the ultimate goal for a new system; Users focus on requirements, not the actual business need.	2	34%
UT9	Terminology Difficulties: Developers do not know business terminology and users do not understand developers' terminology; Users and developers have different meanings for same terms; Users and developers use the same terms to talk about different things; Requirement gatherers assume they understand the terminology; Use of acronyms complicates terminology issues; Developers are unfamiliar with business terms; Users use terms differently.	3	33%
UT10	Assumptions Are Made by Both Developers and Users: Developers make assumptions about requirements instead of asking users questions; Users neglect to consider or are ignorant of behind-the-scenes functions that must be addressed in new requirements; Users make assumptions about the design or provide a design during requirements determination; Users make assumptions about what the system will do—ask for a TV remote and get something that only changes channels; Developers make assumptions about what users tell them.	3	31%
UT11	Box Thinking by Users: Users cannot ask for functionality if they do not know it exists; Experience with existing systems constrains ability to think about a new system; Users have difficulty starting with a blank page to create a system; Users do not consider impacts to or by adjacent areas—other aspects of the problem or the bigger picture; Users lack a willingness to learn something new—a new information system; Users are reluctant to change, more comfortable with the familiar; Users think of requirements that merely extend their current processes instead of examining the real business need that must be accomplished.	3	24%
UT12	Developers Know Better: Developers choose technology for selfish reasons, not because it is the best fit for a problem; Developers can be difficult to approach because they are arrogant and intimidating; Developers have a bias—they will create what they think users need, not what users ask for; Developers do not want to know "why", only "what"; Developers presume they already know what the user needs; Developers believe they know everything about the system and	2	22%

Factors Contributing to Misunderstanding Requirements 249

ID	Description of Theme	Number of Orgs	Support
	what users need.		
UT13	Unclear Who Is Responsible for Requirements: Users do not want to be bothered with requirements—they want to say what they need and expect developers to read their minds for the details; Developers should remind users to consider capabilities they may need; Developers should help guide the creation of requirements; No formal process for requirements determination; Users do not want to be bothered again by developers until the system works; Users can not be held responsible for system not meeting business needs if they do not participate.	2	22%
UT14	Poor Requirements Reviews: There is seldom a "is this what you meant" review of the requirements with users; No formal walk-through of the requirements; Users do not know when requirements change—poor change control procedures; Users are not given requirements to review; Requirements documentation is poor and difficult to review; If users are given requirements documentation, they do not understand it; As requirements change over the course of a project, reviews need to continue; A history for why decisions were made is not captured in the requirements documentation; Requirements documentation is not clear or understandable.	2	19%
UT15	Requirements Documentation is Poor: Requirements are communicated informally and may not be written down; Requirements documentation is not robust; A history for why decisions were made is not captured; Requirements documentation is not clear or understandable.	1	17%
UT16	Schedule Drives Projects: Deadlines drive projects and the functionality users receive; Requirements are gathered too quickly and detail is not addressed.	1	17%
UT17	Users Lose Hope: Users find work-arounds to problems because they do not believe IT is ever going to fix the issue; Some users believe requirements determination work is a waste of time, and consequently do not work very hard at it; Needs change over time and IT can not respond quickly enough; Past user dissatisfaction with information systems causes users to be discouraged to do requirements work.	2	16%
UT18	Users are not Equipped for Requirements Work: Users do not have the skills to do requirements work; Developers expect users to create perfect requirements, but users do not have any training in requirements determination; Users do not know what developers need to hear to create requirements; Users are asked to attend a requirements meeting without first having time to think about their needs; Users do not know what is expected from them to help define requirements.	2	16%
UT19	Conflicting User Needs: Users in the same department can have different and conflicting objectives; Different groups will use the system in different ways, creating conflicts or missing requirements; One or more users may provide contradictory requirements; Not all users may have the same understanding of the problem or the big picture; Some users may have a different agenda; Some users will not be satisfied; Users tend to do what is best for themselves, not	3	13%

Factors Contributing to Misunderstanding Requirements 250

ID	Description of Theme	Number of Orgs	Support
	other users or groups of users.		
UT20	Box Thinking by Developers: Developers have difficulty thinking of how to extend an existing system to be what the users want; Developers' existing knowledge/experience constrains what they consider possible; Developers try to create a new system based on work they did on an old system; Developers do not appreciate why a system needs to be changed—why the users need a change.	2	13%
UT21	Telephone Game: Communication breaks down as message travels from users to business analysts to developers; No communication exists between the correct user and the correct developer—intermediary people in the way distort or lose information.	2	12%
UT22	Requirements are Unexpectedly Changed: Upper management changes priorities of requirements without consulting users; Managers make decisions without understanding the needs of the users—the people in the trenches most affected by an information system.	1	12%
UT23	Design Causes Constraints: Developers can influence the whole system and what is possible in the future by their technology choices; Developers may make technology choices for other reasons than being the best fit for the requirements.	1	7%
UT24	Development Starts Before Requirements Are Complete: Developers start coding before the users have adequately discussed their needs; Requirements are not reviewed before development starts.	1	7%
UT25	Information Gets Lost: Poor communication among IT staff; Requirements agreed to be one person may not added to the project plan and acted upon; Developers may believe they know better and ignore information; If a requirements review is conducted, users learn developers have changed the requirements without first discussing the changes; The requirement may be relayed between multiple people, being confused in the end.	2	6%
UT26	Users Do Not Prioritize Requirements Work: Key users who need to participate do not attend requirements meetings; Managers fail to provide the right people and make their time available; Actual needs are not well represented—key users have already found ways to work successfully and do not invest time in requirements work; Users do not clarify or are confused themselves about their priorities.	2	5%

APPENDIX K: QUALRUS SCRIPT FOR LISTING CODES FOR EACH FACTOR

```
string(myCode).
ForEach code in Project.Codes
  {display(code)} {Use if you want to see all code detail}
  myCode=code.name
  writeln("-----")
  writeln(myCode)
  writeln()
  ForEach segment in CurrentSource
    if (segment.HasCode(mycode)) Then
      {display(segment)} {Use if you want to see all segment detail}
      writeln(segment.text)
      writeln()
    endif
  EndFor
EndFor

      EndFor
```

APPENDIX L: AHP INCONSISTENCY VALUES FOR USER PARTICIPANTS

User	Inconsistency
10U1	0.36
10U2	0.98
10U3	0.22
10U4	0.52
10U5	0.39
10U6	1.45
20U1	0.81
20U2	0.44
20U4	0.63
20U5	0.39
20U6	0.87
20U7	0.37
30U1	0.37
30U2	0.54
30U3	0.35
30U4	0.8
30U5	0.83
30U6	1.26
30U8	0.71
30U9	0.71
30U7	0.38

APPENDIX M: ALL FACTORS CREATED BY DEVELOPERS

Vote	Factor Description
100%	Key players not always involved in requirements determination
57%	Developers - business analysts - user relationships can help or hinder requirements determination
57%	Lack of requirements review
57%	The telephone game: users to business analysts to developers communications
57%	Users are not sure of what they want; are vague about needs; do not understand the problem
50%	Assumptions made by developers and users
50%	Lack of ongoing user involvement during design
44%	Appropriate people are not included; insufficient proxy for users
44%	Users do not understand requirements—can't articulate requirements
43%	English is not the users' first language
38%	Formal versus informal requirements
33%	Technology is complicated
31%	Lack of enterprise planning; being able to understanding the whole
29%	Business units do not create business plans
29%	Lack of strategic plans for the organization
29%	Managers are not enabling users to have time for requirements determination
29%	Not getting input from key users
29%	Team dynamics can help or hinder requirements determination
29%	Translation between IT and Business is needed
29%	Users begin with a preconceived notion of the solution
27%	Developers lack of domain knowledge

Vote	Factor Description
27%	No formal organization-wide software development process
27%	Poor communication
27%	Users over simplify requirements
25%	Fixed deadlines; deadlines impose change in requirements
25%	Lack of prototypes—users need a visual representation
25%	Users do not know what they want
25%	Users rush requirements definitions; creates a quality problem
25%	Users'/Managers' requirements leave out detail
24%	Adequate requirements documentation
24%	Organizational laziness with requirements determination
21%	Creating requirements is not a priority for customers; time is not allocated
21%	IT and users lack of listening skills
18%	Developers do not understand business
18%	Fear and miscommunications of project goals: users' fear of change
18%	Key users are the least accessible
18%	Lack of organization in documentation: hard to find information
18%	Users not understanding end goal or management's objective
14%	Both sides (business and IT) may not care to learn about the other
14%	Identifying correct sample of users
14%	Unfounded assumptions about the knowledge of the audience (regarding documentation)
13%	A change in project leadership during the project
13%	Developers confidence versus arrogance
13%	Differences exist between users' and developers' viewpoint
13%	Differences in users and developers worldview, perspectives, or frame of reference
13%	IT difficulty knowing what questions to ask

Vote	Factor Description
13%	IT lacks business knowledge
13%	Lack of requirement documents; verbal versus written requirements
13%	No business analyst—no translator
13%	Not sufficient time to focus on requirements
13%	Requirements get stale over life of project
13%	Requirements must be testable
13%	Scope creep
13%	User versus developer terminology
13%	Users change during project
13%	Users lack an understanding of the big picture
11%	Assumptions
11%	Control of the Process
11%	Idealistic Scope Creep
11%	Keep Users Engaged
11%	Lack of Tactical Authority
11%	No Business Analysts
11%	No Measurements—Lack of Accountability
11%	Not Enough Requirement Specificity
11%	Organization Culture (related to 19)
11%	Spiritual—God is On Our Side
11%	Team Members Withhold Opinions or Information
11%	Unrealistic Deadlines
11%	Vague Objectives
0%	Business does not look at capacity
0%	Business objectives change mid-stream (not a scope creep issue)

Vote	Factor Description
0%	Business people use different terms themselves for the same thing.
0%	Communicating with Respect
0%	Conflicting Requirements
0%	Contention between existing work and requirements work
0%	Customers wait too long to think about requirements
0%	Developers not part of initial business meeting
0%	Different worlds (factor seeded by researcher and agreed upon by group)
0%	External Constraints
0%	Forcing a technology to a problem
0%	Geographically scattered teams can strain requirements determination
0%	IT arrogance—analysts know better than users (emphasized as a key issue by BA, but not voted for by anyone)
0%	IT lacks interest in requirement gathering
0%	IT/User Frame of Reference [seeded by researcher]
0%	Lack of Change Control
0%	Lack of Clear Role Definitions
0%	Lack of cross-training
0%	Lack of Examples to Clarify Requirements
0%	Lack of standardization in the documentation
0%	Language barriers: users and IT use different terms
0%	Management May Not See Value in Formal Processes
0%	Maturity Differences
0%	No Project Priorities
0%	Not Prioritizing Requirements
0%	Outside contractors' lack of knowledge of in-house systems
0%	Past relationships make users lazy

Vote	Factor Description
0%	Perceived notions of developers and users
0%	Personality Differences
0%	Politics and Sabotage
0%	Prematurely jumping to a solution (factor seeded by researcher and rephrased by group)
0%	Priorities Change
0%	Reliance on Intuition
0%	Resistance to Change
0%	Signoff signatures does not ensure accuracy
0%	Spending Other's Money
0%	System does not get used even though it was built as requested
0%	Technology Constrains Solution
0%	Technology is Intimidating
0%	Terminology
0%	Too much documentation: exact same information across documents
0%	Tune Message to Audience
0%	Users are afraid to give ideas to IT
0%	Users do not understand IT or care to start
0%	Users extrapolate from their past system experience
0%	Users give different levels of detail
0%	Users make invalid or unstated assumptions
0%	Users' rigid thinking
0%	Users try to be too technical

APPENDIX N: DEVELOPER THEMES BASED ON DISCUSSION IN FOCUS GROUPS

ID	Description of Theme	Number of Orgs	Support
DT1	<p>Key Users Are Not Available or Are Not Identified: Most valuable users to help with requirements determination are the least available because they are too valuable doing their existing work; Most input comes from non-expert users; The right users are not identified; Managers do not prioritize requirements work; Individual users may not represent the needs of the business or the larger user group; Key users are not available to participate in requirements; Managers participate, but are not the work-a-bees who will be using the system or have the best knowledge for helping with requirements; Key users are too busy with other work and managers do not make them available; Key people can not get together—conflicting schedules and other demands make group meetings nearly impossible; People providing requirements do not have enough knowledge of what is needed—managers send wrong people to participate; Managers who provide requirements do not have the business process knowledge to do so; Users do not stay involved throughout project—they get too busy with their daily work.</p>	3	95%
DT2	<p>Users Do not Know What They Want: Users do not know what they want; Users are uncertain about what they want; Users know what they do not want when they see a prototype; Different users provide various levels of specificity and detail about their needs; Users make assumptions about how things work or will work; Once they see the system or a prototype, they know what they do not want or more of what they want; Developers need to give users something to see—users need something concrete before they can better describe their requirements; Users rely on past experience to describe what they want.</p>	3	64%
DT3	<p>Big Picture Understanding: Users do not have a full grasp of the problem, and how a solution fits into the organization's objectives and strategies; Users lack knowledge about what the business needs to achieve; Users lack insight into the direction of the organization and how an information system fits with existing plans; Solutions are accepted before the problem is clearly understood—prematurely reaching a solution; Strategic plans are not in place to provide boundaries to IS projects; Users have vague business objectives; Project planning does not account for what the entire organization is doing and how it fits into the "whole"; Users lack understanding of what the whole system is going to be or where it is going in the future; Users do not consider the entire task—just automating part of the process, such as filling in a form; Users base their understanding on past experiences.</p>	3	40%
DT4	<p>Developers Lack Business Knowledge: Developers' lack of knowledge about the users' domain hinders effective communication; Developers do not know what questions to ask; Users have little patience and time for ignorant discussions; Developers are not interested in gaining domain knowledge; More business knowledge allows developers to ask better questions and fill in details; Developers are not becoming experts on the business; Developers are not offering value to the business because they do not understand what users do.</p>	3	36%

Factors Contributing to Misunderstanding Requirements 259

ID	Description of Theme	Number of Orgs	Support
DT5	<p>Poor Requirements Documentation: Same and possibly conflicting information exists in multiple documents; Too much or too little detail—lack of knowing how much detail is needed; Poorly organized documents make it difficult to find specific information when needed; Lack of a standardized way to organize requirements documents; Not all requirements are documented; Requirements shared in "informal" meetings may be assumed to be accepted, but are not recorded and tracked by developers; Documents are not updated when requirements change; The documentation is inadequate to understand requirements; Documentation provides too much or too little information for the needs of the audience.</p>	3	32%
DT6	<p>Requirements Are Not Adequately Reviewed: Users do not actively review requirements created by developers; Too much dependence on email and document routing instead of face-to-face or phone conferences between those providing requirements and those programming the system; Developers are not involved in requirements reviews; Developers accept documented requirements as-is without understanding the problem from users' point of view; An accepted requirements document does not mean users and developers understand the requirements.</p>	2	32%
DT7	<p>Terminology: Developers are unfamiliar with business terminology; Developers and users may not be interested in learning the other's terminology; Users use different terms to refer to the same concept (e.g., part number, order number, etc.); Words can mean different things to different people; You interpret words through your experience; Common words may even mean different things to IT and users; Developers need to understand the users' business processes before they can understand them; Users call user interface elements different things; Developers and users speak two different languages; Translation is needed between users and developers.</p>	3	31%
DT8	<p>Users Are Intimidated by Developers: Some users are insecure, afraid to share their thoughts in meetings because they may be judged; Intimidated users may agree to one thing in a meeting with others but say something completely different one-on-one with someone they trust; Users are more comfortable talking with developers than with other developers; Users do not want to look stupid, so they do not participate in meetings.</p>	2	29%
DT9	<p>Translation is Needed: Developers need a translator (business analyst) who understands users and developers; Developers have to be their own business analysts, and that is a skill they do not have; A business analyst can help insulate developers from users and be a single point of contact for requirements; Users have difficulty understanding developers and developers have difficulty understanding users; Developers do not know what questions to ask, or do not ask enough questions, to understand requirements; Not having a business analyst wastes developers' time.</p>	2	28%
DT10	<p>Insufficient Detail: Requirements provided by users leave out detail and contain many unanswered questions—questions that may not appear until design is performed; Missing detail is a result of not having key users involved; Users need to think about the problem more before describing requirements—what they really need; Developers always find conflicting requirements or missing detail that requires additional information from the users; Lack of examples are provided by users to clarify requirements—no use cases; Requirements are too simple; Requirements are not prioritized.</p>	2	26%

Factors Contributing to Misunderstanding Requirements 260

ID	Description of Theme	Number of Orgs	Support
DT11	Telephone Game (Chinese Whispers): Information is lost or distorted as it is passed, both verbally and in documentation, between users, business analysts, and developers; Too many non-critical people are involved in requirements determination; Developers are isolated from users; Developers are not involved in initial user discussions about the problem and need for an information system; Developers understand the problem and needs better when involved early.	1	24%
DT12	No Standard Development Process: No formal consistent development process is used, including requirements determination; No accountability for projects exists—people are not fired when projects go bad; Management may not see value in formal process—they view it as slowing down progress; No one, such as a Business Analyst, is in control of the processes that are used; There is a lack of clear role definition—who does requirements determination, business plan, resource allocation, ...; No change control process—changes create confusion about the requirements.	1	21%
DT13	Past Relationships: Developer-user relationships and past experience with users are more valuable than a requirements document or domain knowledge; Personal relationships are used by developers to get users to participate in requirements determination; The better relationships developers have with users, the more developers learn from the users; developers are more likely to be involved earlier in projects if they have positive relationships with users involved; Users can become accustomed to working with the same developer and become lazy—making assumptions and not fully communicating; Developers may make assumptions because of close working relationships with users.	2	18%
DT14	Users Do Not Prioritize Requirements Work: Users wait until the last minute to work on requirements, leaving no time for reviews with development; Requirements work is not important to users and consequently poor quality requirements can be expected; Users have their regular work to perform and are not allocated time by their managers to work on requirements; Users' resource allocation is not considered when asked to work on requirements; Developers expect the requirements to be of poor quality.	1	18%
DT15	Team Members Withhold Opinions: Users do not want to look stupid, so they do not participate in meetings; Some team members feel intimidated to share information in meetings, but may one-on-one; As people mature, they are more likely to freely share information.	1	17%
DT16	Users Do Not Understand Technology: Users do not know development terminology, constraints of system development, programming, etc.; Users are not interested in gaining development knowledge; Users complain that they do not want to understand technology, just to do their job; Users may attend meetings, but they are not following the discussion; Technology is intimidating for users.	3	16%
DT17	Assumptions: Developers assume they know the users' business and users assume they know what developers are talking about; Unstated assumptions are made by both developers and users; Developers are focused on programming, and in haste make assumptions about the requirements;	2	13%
DT18	Different Perspectives: There is a difference in perspective between one who is asking for something and one who is delivering something; Developers that must create	2	13%

Factors Contributing to Misunderstanding Requirements 261

ID	Description of Theme	Number of Orgs	Support
	detailed code must have a different perspective than users; Developers have difficulty not designing, at least mentally, when discussing requirements; User's do not want to be involved in requirements and expect developers to provide what they need; Developers and users communicate differently; A line should exist between what developers and users do, but the line is blurred; Developers create requirements because users will not do the work.		
DT19	The Effect of Time: Business objectives can change during the course of an IS project, impacting the requirements; Information systems may be abandoned, even if completed by developers, because of changing objectives; The problem, needs, and possible solutions are better understood by users as more experience is gained with each prototype; Requirements change over time, especially when they are not immediately acted upon; Users change during a project and new users may introduce or change requirements; Developers never have sufficient time to get the requirements right—to focus on the requirements; Project leadership may change over the project, which impacts requirements that have already been determined.	2	12%
DT20	Box Thinking By Users: Users extrapolate from what they know—their past experiences with information systems; Users do not realize there are other ways things can be done; Users are focused on their own needs and do not recognize how a system may impact other groups; Users and developers constrain themselves by having a preconceived notion of the solution.	2	12%
DT21	Personality Differences: People process information differently—some need to listen during meetings and share ideas later, others need meetings to generate ideas, others need visuals for ideas, etc.; It is not consistent with a developer's personality to figure out business needs; Users and developers think differently—different levels of detail, process information differently, different concerns, etc.	1	12%
DT22	Organizational Culture: Users insist on doing something, even if developers say it is not feasible; People tend to be too nice—not wanting to hurt another's feelings; "Our organization is unique"-attitude so learning from outside organizations is ignored.	1	11%
DT23	Language Barrier: Non-native English speaking team members make effective communications challenging.	1	9%
DT24	Project Deadlines: Requirements change as deadlines are approached, de-scoping work to fit the schedule; Mid-course changes cause confusion with the development team and work may not be clearly prioritized; Project deadlines are set before project begins and requirements are developed; Resources are moved away from projects to those projects with the tightest deadline; Lack of project prioritization results in overly stretched resources and projects that are halted part-way through.	2	8%
DT25	Developers Are Arrogant: Developers to not prioritize time to talk with users; Developers believe they know what users need better than the users; Developers may ignore users or not really listen to them; Developers may falsify requirements based on their experience, discounting the experience of users; Most developers do not enjoy working with users, doing "soft" requirements work—they would rather do "meaty" coding and configuring; Developers do not like being told what to do by the users; Developers may choose a technology primarily because they want to learn it, not	2	6%

Factors Contributing to Misunderstanding Requirements 262

ID	Description of Theme	Number of Orgs	Support
	because it is most appropriate for the problem; Developers make users defensive either by showing apathy or arrogance; Users expect developers will look down on them; Developers are frustrated when they do not control the schedule—when a deadline is imposed and the "engineering" is not considered; As soon as developers have a "we know better" tone with developers, it turns users off.		
DT26	Users Are Resistant to Change: Users may be hostile to developing a new information system because it means a change to their job, or even the elimination of their job; Users fear how the information system will change their job; Users tend to do things the way they always have; Users have a conservative nature.	2	6%
DT27	Team Dynamics: People want to spend time discussing the work with other people they like—they are less likely to talk with someone about a question when the team dynamics are dysfunctional; Team dynamics are more important than domain knowledge—if you like the people involved, you can get the information you need; Good team dynamics improve requirements determination; Geographically scattered teams strain requirements determination.	1	6%
DT28	Developers Lack Listening Skills: Developers do not listen to users effectively; Developers do not know how to listen—they lack communication skills to be effective listeners.	1	3%

APPENDIX O: AHP INCONSISTENCY VALUES FOR DEVELOPER PARTICIPANTS

Developer	Inconsistency
10D1	0.19
10D2	0.18
10D3	0.2
10D4	0.41
10D5	0.08
10D6	0.48
10D7	0.08
20D1	0.25
20D2	0.83
20D3	0.53
20D4	0.09
20D5	0.3
20D6	0.02
20D7	0.23
20D8	0.4
30D1	0.17
30D2	0.13
30D3	0.28
30D4	0.32
30D5	0.13
30D6	0.1
30D7	0.4
30D8	0.2
30D9	0.96

APPENDIX P: INTERDEPENDENCIES BETWEEN THEMES

In addition to the most important factors selected by the participants in the present study, the discussion of all factors was synthesized to create themes. Consequently, the themes reflect all factors discussed by participants. From the users' focus groups, 26 themes were synthesized, while 28 themes were synthesized from the developers' focus groups. Support for each theme was implied as a percentage from zero to 100 based on the number of votes factors received that were related to the theme.

Several scenarios can be envisioned for relating user themes, listed in Appendix J, and developer themes, listed in Appendix N, to show how one theme influences another. Exploring scenarios is an important exercise because the factors judged to be most important, which relates to themes with the most support, may be meaningfully influenced by a seemingly unimportant theme. Analyzing the relationships between themes uncovers other dimensions of the problem with misunderstanding requirements. Many of the factors are viewed as related to motivation issues, others are tied to process problems, and a few require skill improvements.

Consistent with the theory-building nature of the research study, the relationships highlighted illustrate areas that may improve requirements understanding, but have not been tested and cannot be stated with authority. Further, the relationships are not exhaustive, but focused only on a few interactions to develop the scenario.

Scenarios centered on four themes are considered: (a) key users are not available, (b) users do not know what they want, (c) users and developers operate from different frames of reference, and (d) users are intimidated by developers. These four were chosen because the first two were the most important factors to developers and users respectively; the third is a common

belief among developers; and the forth was expressed as a core issue by developers but received little support when it came time to select the most important factors.

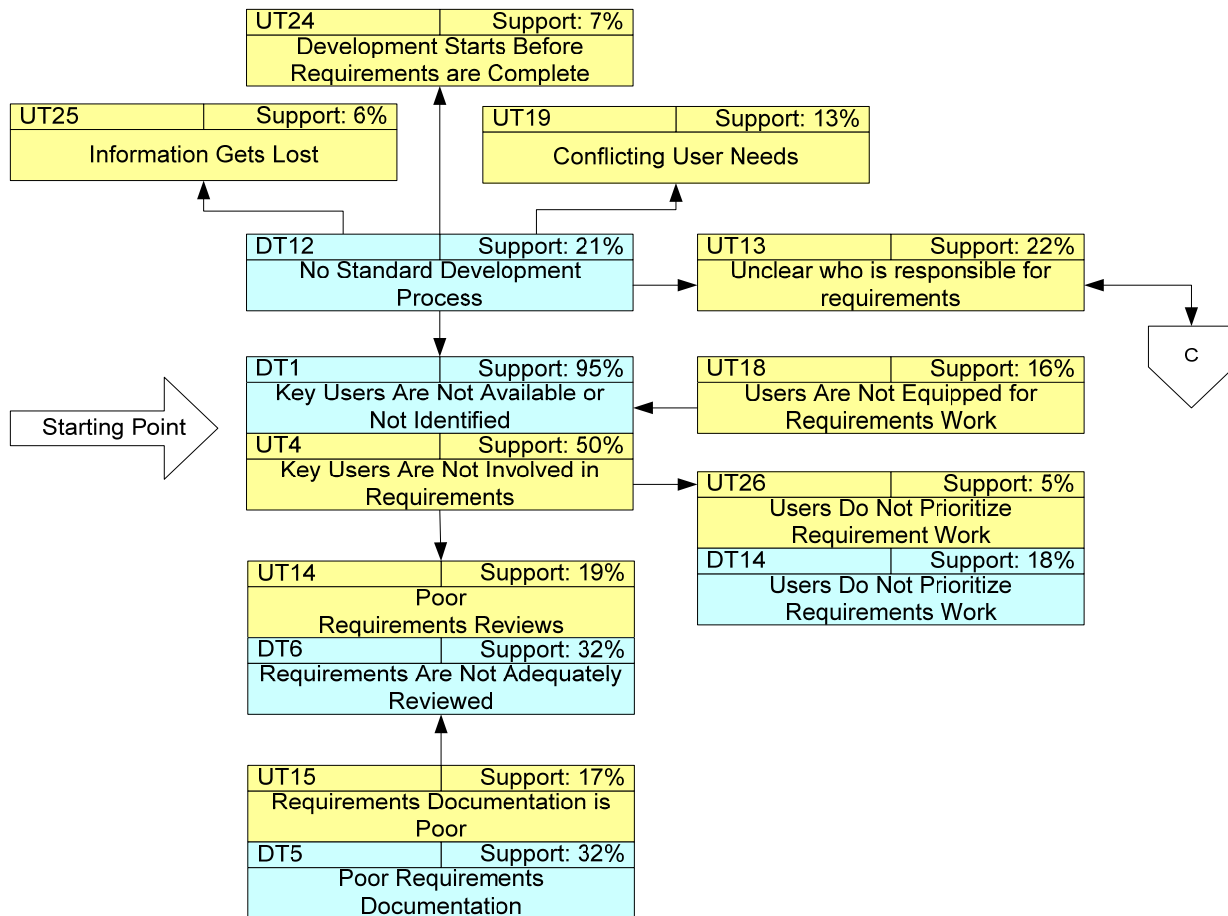


Figure P-1. Themes related to Key Users are Not Available.

User themes are colored in yellow (light gray when printed) and developer themes are colored in blue (darker gray when printed).

Key Users Are Not Available (DT1 / UT4). Several reasons were identified in the focus groups for why key users are not available, such as their time is too valuable, they have their own procedures for being successful and do not need another information system, and that they have had poor experiences helping with past information systems. One organization shared that users

were asked to volunteer to aid developing an information system, but that their employment would be terminated at the end of the project because the new system would make them obsolete. Consequently, a dimension of key users' reluctance to participate in requirements determination is motivation—they are not personally or professionally motivated to do so.

Several themes are related to key users not being available, as shown in Figure P-1. An input to this theme is UT18: Users are not equipped for requirements work. Users who knew what to expect and how to contribute to requirements determination would be better prepared to participate. Failure to prepare users for requirements work is a process problem. Another input is DT12: No standard development process, which also causes users to not know what to expect. The lack of a standard development process leads to other issues, including that information provided by users is lost and not acted upon, which further erodes users motivation, trust in the process, and trust in developers (UT25); that software development begins before requirements are complete, which may be a process problem a communication problem (UT24); that users have conflicting needs, which are not perceived as being fairly resolved and is a process problem (UT19); and that users are not clear who is responsible for requirements and expects developers to complete them, which involves both motivation and process problems (UT13).

An output from DT1 / UT4 is that users do not prioritize requirements work because it is not important to them or their managers, a problem with motivation (UT26 / DT14). Another output is poor requirements reviews because users want little involvement in requirements determination, do not understand requirements written by developers, and do not want to take responsibility for requirements they expect will not be delivered but for which they will still receive blame for (UT14 / DT6). Reviews are complicated because requirements documentation is poor (UT15 / DT5).

This scenario is connected with the next one through UT13 because users who do not know who is responsible for requirements is related to users not knowing what they want, denoted with the “C” connector.

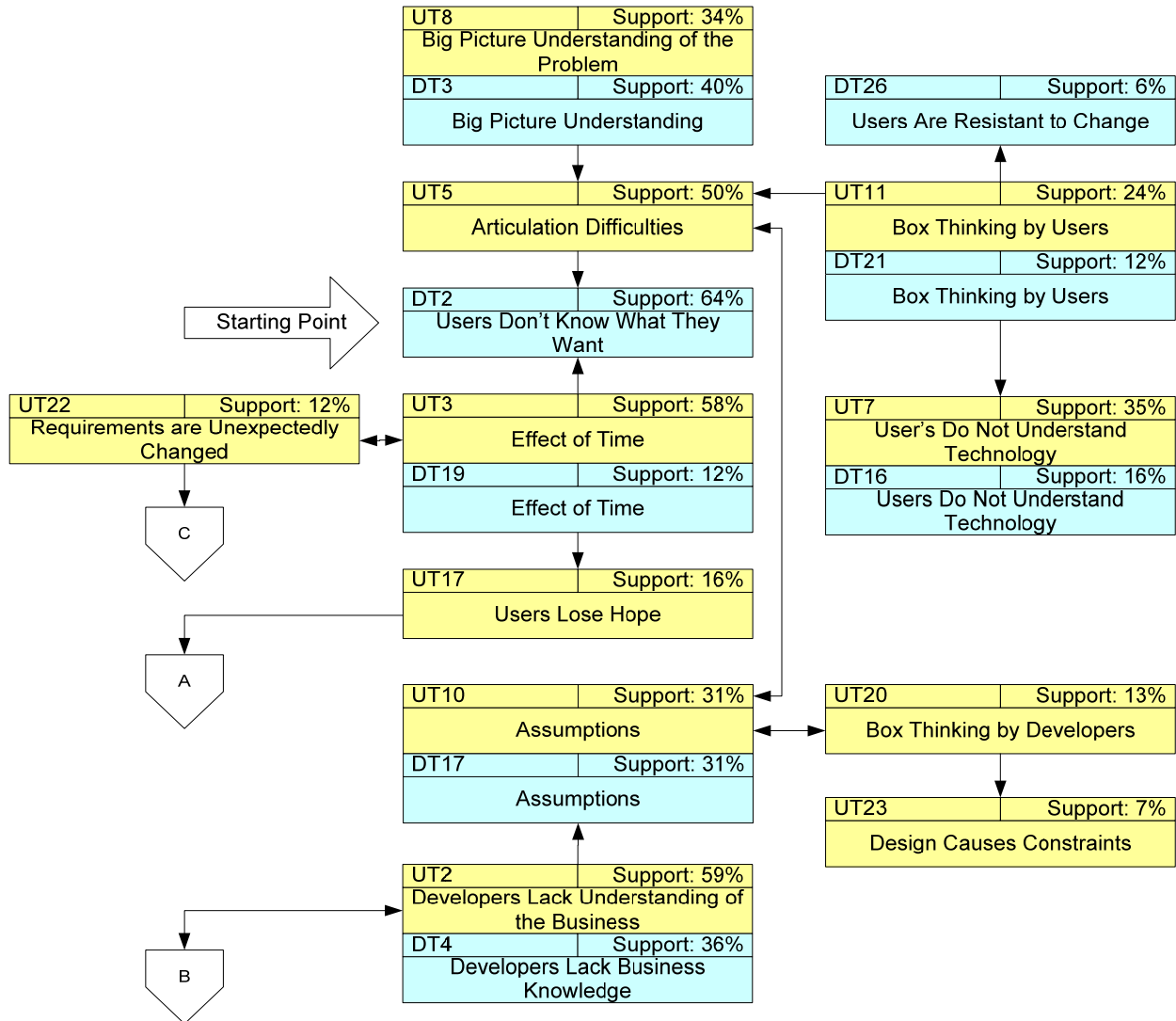


Figure P-2. Themes related to Users Do Not Know What They Want

User themes are colored in yellow (light gray when printed) and developer themes are colored in blue (darker gray when printed).

Users Do Not Know What They Want (DT2). This theme was the second most supported theme by developers. It relates to the several other themes, as depicted in Figure P-2. Two key inputs to this theme are articulation difficulties and the effect of time. Articulation difficulties (UT5) involves users not having a clear grasp of the larger problem domain their needs relate to, which could be process related (UT8 / DT3); users who make assumptions when discussing their needs, which is a skill issue (UT10 / DT17); and users whose past experience and unfamiliarity with what is possible constrains their thinking, which is both a skill and process problem (UT11 / DT21).

Box thinking by users (UT11 / DT21) is related to users resisting change, a motivation issue (DT26); and users not understanding technology, a skills issue (UT7 / DT16). Assumptions are made by developers based on their past experience and understanding of what is possible (UT20); and their lack of knowledge of the business, a skills and motivation issue (UT2 / DT4). Box thinking by developers (UT20) is viewed by users as constraining technical designs (UT23).

Another significant input to users not knowing what they want is the effect of time (UT3 / DT19). When users change requirements, developers view them as changing their mind because they did not really know what they wanted to begin with. The effect of time is not considered as a possible cause, but is a likely culprit, as needs legitimately change with changing business objectives. Another source of change is related to users who better understand the problem as time goes on, and should be expected and accounted for in requirements determination processes. The effect of time also opens the door to requirements being unexpectedly changed by stakeholders who are not users and who do not communicate the changes back to users, which is a process issue (UT22).

This scenario has links to the other three scenarios, shown in Figure P-2, indicated by the “A”, “B”, and “C” connectors.

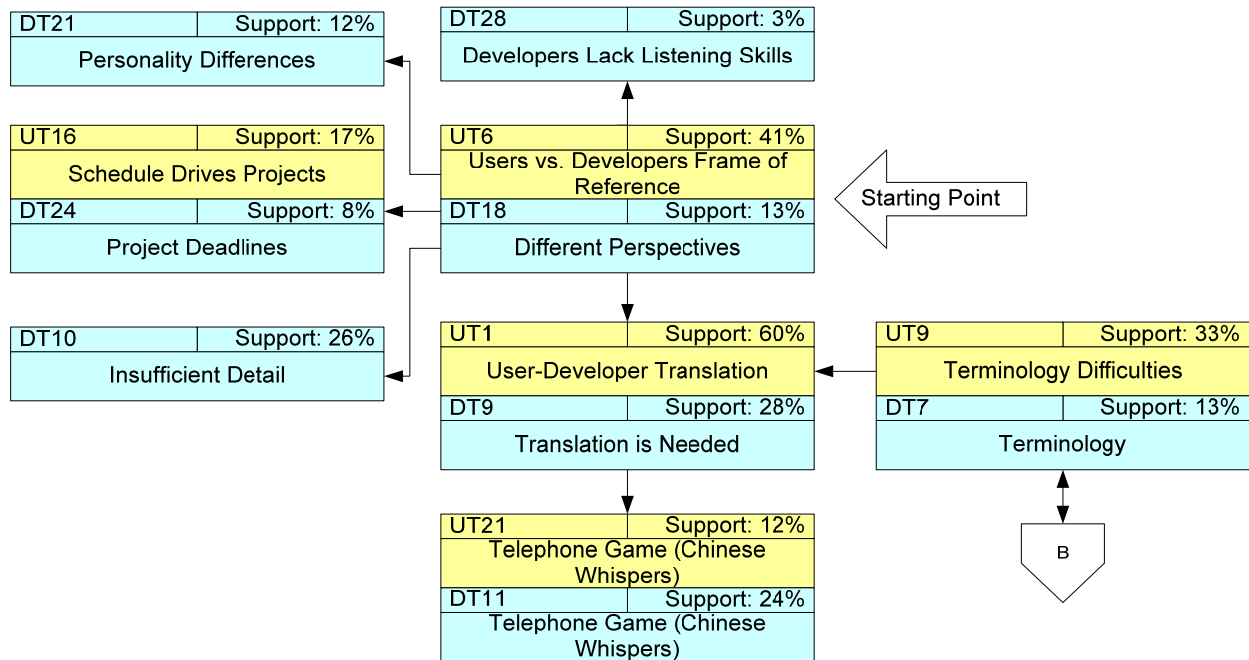


Figure P-3. Themes related to Users and Developers Operate from Different Frames of Reference.

User themes are colored in yellow (light gray when printed) and developer themes are colored in blue (darker gray when printed).

Users and Developers Operate from Different Frames of Reference (UT6 / DT18). Figure P-3 depicts the related themes. Outputs from this theme include acknowledgement that developers are poor listeners, which is both a skills and motivation issue (DT28); that developers require greater detail than users are accustomed to, which is related to process (DT10); that users and developers tend to have different personalities (DT21); that users and developers think differently about project deadlines, another process issue (UT16 / DT24); and that users and developers need someone to help translate between them, another process issue (UT1 / DT9). Another theme driving the need for translation is difficulties with the terminology organic to users versus developers, a skills issue (UT9 / DT7). However, translation is often the role of a

business analyst or project manager, which introduces another party in the communication chain that can introduce distorted views of requirements (UT21 / DT11).

Differences in users and developers frame of reference are related to users not knowing what they want via terminology as indicated by the “B” connector.

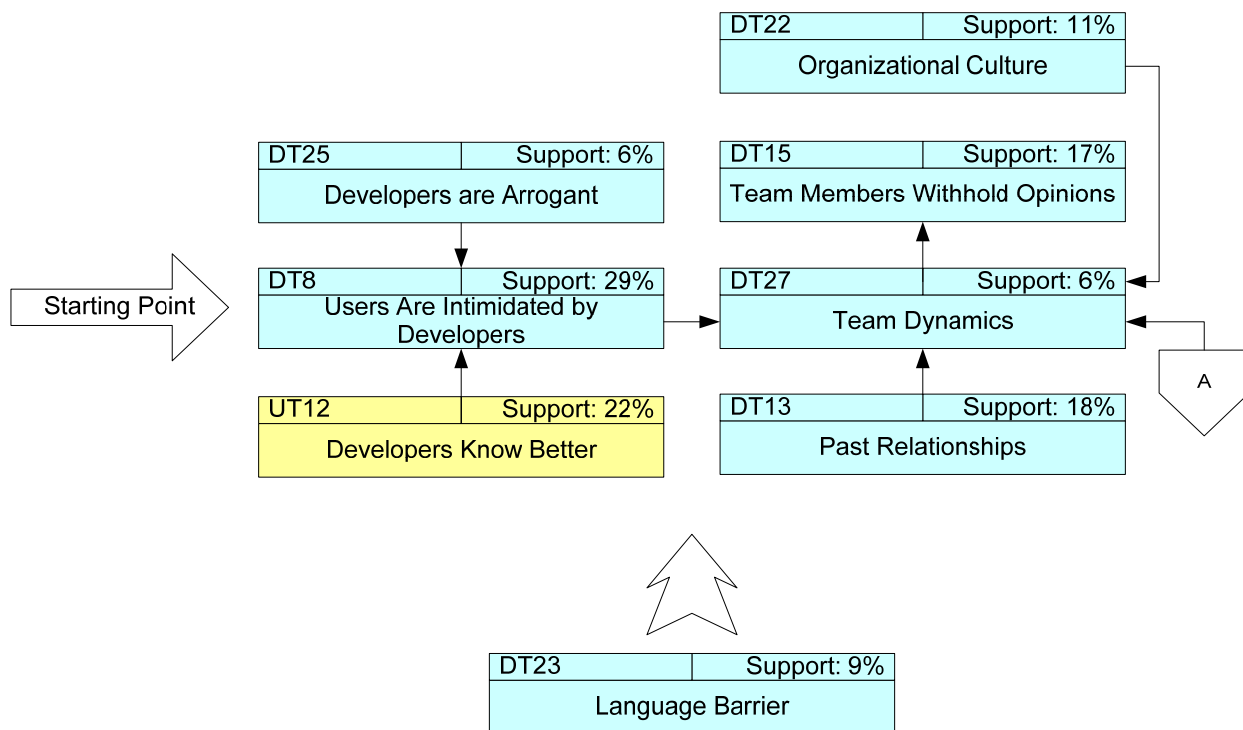


Figure P-4. Themes related to Users are Intimidated by Developers.

User themes are colored in yellow (light gray when printed) and developer themes are colored in blue (darker gray when printed).

Users are Intimidated by Developers (DT8). This theme was voiced in all of the developer focus groups, either directly or indirectly, but was surprisingly absent from the user groups until specifically asked about by the researcher.

Figure P-4 depicts the related themes, showing two themes as inputs: developers tend to be arrogant given the creativity and intellectually challenging work they do and their desire to be

left alone and not bothered by users (DT25); and that developers believe they know more about what a system must do than users do (UT12).

The output of DT8 is team dynamics that are eroded because users feel they can not freely approach developers (DT27). Team dynamics is effected either positively or negatively by past relationships between users and developers as well as within each group (DT13); and team dynamics effects team members willingness to share information with each other, fearing their ideas may be ridiculed if team dynamics are poor (DT15). Other influences on team dynamics come from the organizations culture that may take many forms, such as a developer-led organization versus a business-led organization (DT22). It is also related to users losing hope based on their past experiences or future expectations of the development team, depicted with the “A” connector in the figure.

A complicating issue across all of the themes is team members that are not native speakers of the same language (DT23). When this occurs, which is customary in international organizations that have decentralized users but centralized information systems development, communication between users and developers becomes more challenging, exacerbating problems with misunderstanding requirements.

APPENDIX Q: INSTITUTIONAL REVIEW BOARD EXCERPTS

This appendix includes relevant excerpts of the Institutional Review Board application. The complete packet is controlled by Capella University. The following is included:

- (1) Request for Organization to Participate
- (2) Research Participant Informed Consent Form
- (3) Email Confirmation from Participating Organizations

Request for Your Organization to Participate in an Information Systems Research Study

As a PhD student attending Capella University and living in Colorado, I am researching and writing a dissertation on "Requirements Determination of Information Systems: User and Developer Perceptions of Factors Contributing to Misunderstandings."

My research is expected to benefit organizations developing information systems (IS). Since your organization relies on IS development, you are already aware of how important clearly understood requirements are to the success of IS projects. The results of my research will provide details on why requirements are misunderstood and in turn, provide you with information to improve the success of your IS projects.

For your organization to participate in the research study, I am asking that you form two focus groups. The first focus group needs to consist of seven to ten IS developers. The second focus group needs to consist of seven to ten IS users who have been involved in specifying requirements for IS projects. Each group will be asked to discuss the factors they believe contribute to misunderstanding requirements. A structured group facilitation method called Nominal Group Technique will be used. To minimize disrupting your work, each group will meet for a maximum of 2 hours over lunch. A few weeks after the focus groups meet, each participant will need to complete a survey, which will require no more than 30 minutes of their time.

Your organization's identity and the identity of your employees participating in this study will remain strictly confidential and anonymous.

Given the success of your organization and its reliance on IS, I hope that you will agree to be part of this project. It is an important topic, and with your help, will improve our ability to create successful information systems. As a participating organization, you will be among the first to see the results of this study, which I will provide in the form of an executive summary.

I look forward to talking with you soon to answer your questions about this research.

Chad McAllister
PhD Candidate
chad@ckmcallister.com

Capella University, 225 South 6th Street, 9th Floor, Minneapolis, MN 55402

RESEARCH PARTICIPANT CONSENT FORM

Requirements Determination of Information Systems: User and Developer Perceptions of Factors Contributing to Misunderstandings

Invitation To Participate:

You have been invited to participate in a study on requirements determination because of your experience with information systems in your organization.

Purpose:

The objective of this research is to better understand why requirements for information systems are misunderstood.

Description of Study Procedures:

You will participate in a facilitated focus group with others from your organization in similar roles. The focus group will meet for two hours to identify reasons for misunderstanding requirements. The discussion will be audio taped and only the facilitator and his assistant will have access to the recordings. A few weeks after the focus group meets, you will be asked to complete a survey. The survey can be completed in approximately 15 minutes or less. The focus group will take place in a private conference room located in your facilities.

Voluntary Participation and Risks:

Your participation in this study is voluntary and you may terminate your participation at any time without any consequence to you or your organization. You will not be at physical or psychological risk during the study procedures. A minimal risk, which is no greater than that which is encountered daily in your work life, is the breach of confidentiality of information shared during the study procedures. Although the research team is using established methods to protect the confidentiality of any information shared, there is a minimal risk that information shared during the focus group could be attributed to you by another participant.

Benefits:

Although there is no direct benefit to you for participating in this study, your participation will benefit your organization and the information systems community by aiding our understanding of requirement issues.

Confidentiality:

Your discussion during the focus group and survey responses will be kept confidential and available only to the research team for analysis purposes. Interview tapes will be stored in a fire-resistant safe

controlled by the principal researcher. Only the research team will listen to and read the transcribed the information you give us. Published results from your discussion during the focus group will not be linked to your organization, your name, address, or any other identifying information. Your survey responses will be associated with your name to aid in collecting surveys from all participants, but this association will only be known by the research team and not made available in any published results. Your responses will be aggregated with those of the other participants in this study, further protecting your anonymity. This is done to ensure your responses remain confidential so you can respond as freely as possible.

Questions:

If you have any questions about this study I will be happy to answer them, now or in the future. Please contact the principal researcher and PhD student, Chad McAllister, at his office in Colorado by phone at 719-559-1627 or by email at chad@ckmcallister.com. You may also contact the supervising researcher, Dr. John Latham, by email at john@drjohnlatham.com.

If you have any questions or concerns about your rights as a research participant, you may contact the Capella University Institutional Review Board Director, Kurt Linberg, at 1-888-227-2736.

I agree to participate in this research study. The researcher has answered any questions I had.

- Check here to receive a signed copy of this consent form.

Participant's Printed Name

Date

Participant's Signature

Researcher's Signature

Confirmations from Participating Organizations

The following email text was sent to the three organizations participating in the research study:

Dear _____,

I wish to thank you for representing to your organization my dissertation research study entitled "Requirements Determination of Information Systems: User and Developer Perceptions of Factors Contributing to Misunderstandings." As we have already discussed, the research will involve two focus groups, each requiring two hours, and a follow-up survey.

Please reply to this email to confirm your permission to conduct this research study with your organization.

Thank you,
-Chad McAllister

PhD Candidate

Universal: (719) 559-1627

E-mail: chad@ckmcallister.com

Email confirmations were received from each organization.