INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700  800/521-0600
INTEGRATED SCENARIO AND PROCESS MODELING SUPPORT
FOR COLLABORATIVE REQUIREMENTS ELICITATION

by
Ann Marie Hickey

__________________________
Copyright © Ann Marie Hickey 1999

A Dissertation Submitted to the Faculty of the
COMMITTEE ON BUSINESS ADMINISTRATION
In Partial Fulfillment of the Requirements
For the Degree of
DOCTOR OF PHILOSOPHY
WITH A MAJOR IN MANAGEMENT

In the Graduate College
THE UNIVERSITY OF ARIZONA

1999
As members of the Final Examination Committee, we certify that we have read the dissertation prepared by Ann Marie Hickey entitled Integrated Scenario and Process Modeling Support for Collaborative Requirements Elicitation and recommend that it be accepted as fulfilling the dissertation requirement for the Degree of Doctor of Philosophy.

I hereby certify that I have read this dissertation prepared under my direction and recommend that it be accepted as fulfilling the dissertation requirement.

Dissertation Director  Jay F. Nunamaker, Jr.  April 15, 1999
STATEMENT BY AUTHOR

This dissertation has been submitted in partial fulfillment of requirements for an advanced degree at The University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

Brief quotations from this dissertation are allowable without special permission, provided that accurate acknowledgment of source is made. Requests for permission for extended quotations from or reproduction of this manuscript in whole or in part may be granted by the copyright holder.

SIGNED:  Ann M. O'Dickey
ACKNOWLEDGEMENTS

Numerous individuals have contributed in many different ways to the successful completion of this dissertation and my doctoral degree. I greatly appreciate everyone's contributions, but have only enough space to individually acknowledge a few. My sincerest thanks and gratitude go out to all, and in particular:

To my dissertation committee, Jay Nunamaker, Doug Vogel, Jim Lee, and Dave Meader for their patience and support as I struggled to complete this dissertation.

- To Jay, for volunteering to take over as my advisor and chair of my committee despite his many other commitments, and for all his time and assistance during this last semester even as he faced his own challenges.
- To Doug, for encouraging me to return to school and helping me with the transition, and for sticking with me to the end, even when his career took him elsewhere.
- To Jim, for being a great friend and mentor, for providing continuous research guidance, technical support and personal encouragement, and for convincing me that I really could finish this on time.
- To Dave, for helping out at the last minute so that I could defend as scheduled.

To the Scenario and Process Analyzer (SPA) development team, Eric Santanen, Bill Saints, Conan Albrecht, Jian (Jimmy) Xu, Dongsong Zhang, Lina Zhou, and Duffy Gillman, for their many contributions, hard work and dedication to the success of the SPA prototype, and for keeping on going after Bill's devastating death.

To the researchers, administrators, staff, and sponsors of the University of Arizona's Center for the Management of Information (CMI), for the many research opportunities, assistance, and support during my research assistantship with CMI, and especially,

- To Doug Dean, for the opportunity to work in partnership with him on our research projects and publications, and for his many years of great advice and assistance.
- To Melissa Glynn and Mari Heltne, for being good friends and caring Associate Directors, watching out for me to make sure I had time for my own work too.
- To Betty Albert, Jeannette Reinig, Lynn Thacker, and Terry McKenna and his team, for their exceptional support, and for always being there when I needed them.
- To the many fine civilian and military personnel of the Department of Defense who have supported my research and participated in the meetings essential to its success.

To the faculty and staff of the MIS Department, for making my Ph.D. student experience an incredible learning opportunity, and for all the support they've provided me.

And finally, to my fellow doctoral students, for sharing the pleasures and the frustrations of surviving the challenges we all face, especially, Tom Rodgers, D. Liginlal, Mick McQuaid, Chienting Lin, Janna Crews, John Kruse, Andrea Houston, and Rosie Hauck.
DEDICATION

I am truly blessed to have the opportunity to dedicate this dissertation to my husband

   Thomas John Hickey

who has allowed me turn our lives upside down for the last four years and without
whose unwavering support I never would have achieved my dream of earning a Ph.D.

I would also like to dedicate this dissertation and my education to my parents

   Arthur and Beatrice Lucchini

who have always encouraged me to pursue my dreams and supported my start down
this road to educational achievement at great sacrifice to themselves and our family.

Finally, it is with great sadness that I dedicate this work to CMI's Programming Director

   William "Bill" Saints

whose tireless efforts made the Scenario and Process Analyzer prototype a reality, but
who tragically died before he could enjoy the success of our efforts.
## TABLE OF CONTENTS

| LIST OF FIGURES | 15 |
| LIST OF TABLES | 17 |
| ABSTRACT | 18 |
| CHAPTER 1 - INTRODUCTION | 20 |
| 1.1 Background | 20 |
| 1.2 The Research Problem | 21 |
| 1.3 Research Questions | 23 |
| 1.4 Research Approach | 26 |
| 1.5 Importance of the Research | 28 |
| 1.6 Overview of the Dissertation | 29 |
| CHAPTER 2 - LITERATURE REVIEW | 30 |
| 2.1 Information Systems Development | 30 |
| 2.1.1 Software Development Problems | 30 |
| 2.1.2 The Software Development Process | 31 |
| 2.1.2.1 The Waterfall SDLC | 32 |
| 2.1.2.2 Evolutionary Development | 32 |
| 2.1.2.3 Incremental Development | 33 |
| 2.1.2.4 The Spiral Model | 34 |
| 2.1.2.5 Prototyping | 35 |
| 2.1.2.6 Rapid Application Development (RAD) | 35 |
TABLE OF CONTENTS – Continued

2.1.3 Summary ........................................................................................................... 36

2.2 Requirements Engineering .................................................................................. 36
   2.2.1 Requirements Overview ................................................................................. 36
   2.2.2 User Participation .......................................................................................... 38
   2.2.3 Requirements Elicitation .............................................................................. 40
   2.2.4 Requirements Analysis .................................................................................. 42
   2.2.5 Requirements Specification .......................................................................... 43
   2.2.6 Requirements Summary ............................................................................... 44

2.3 Group Support Systems for Collaborative Requirements Elicitation ............... 46
   2.3.1 History ........................................................................................................... 47
   2.3.2 Requirements Elicitation Using a General-Purpose GSS ......................... 48
   2.3.3 Collaborative Modeling Tools ....................................................................... 49
   2.3.4 Collaborative Software Engineering Methodology ..................................... 52
      2.3.4.1 User Participation in CSEM ................................................................. 52
      2.3.4.2 Collaborative Processes, Tools, and Facilitation Techniques . 54
      2.3.4.3 CSEM Phases ..................................................................................... 54
      2.3.4.4 CSEM Open Issues ............................................................................ 56

2.4 Scenarios ............................................................................................................. 57
   2.4.1 Introduction .................................................................................................... 58
   2.4.2 Why Scenarios? ............................................................................................ 61
TABLE OF CONTENTS – Continued

2.4.3 Scenarios in Software Engineering ........................................ 63
2.4.4 Scenario Classification Framework ........................................ 66
2.4.5 Scenarios in Practice ............................................................ 70
2.4.6 Scenario Issues ................................................................. 72

2.5 Summary of Literature Review .................................................. 74

CHAPTER 3 – RESEARCH APPROACH ............................................. 75

3.1 Overview – Multi-Methodological/Multi-Phase Approach ................. 75
3.2 GSS Research Model .............................................................. 79
3.3 Phase I Theory Building .......................................................... 81
3.4 Phase I Observation: Exploratory Case Study Design ...................... 83
  3.4.1 Overview ................................................................. 83
  3.4.2 Study Questions .......................................................... 85
  3.4.3 Propositions ............................................................... 86
  3.4.4 Units of Analysis ......................................................... 86
  3.4.5 Linking Data to Propositions ......................................... 88
  3.4.6 Criteria for Interpreting Findings .................................. 89
3.5 Phase I Experimentation: Laboratory Experiment Design ............... 91
  3.5.1 Research Question and Propositions ................................ 91
  3.5.2 Research Constructs and Hypotheses ............................... 93
  3.5.3 Experimental Design .................................................. 95
TABLE OF CONTENTS – Continued

3.5.4 Questionnaire Development.................................................................99
3.5.5 Analytical Techniques ........................................................................100

3.6 Phase II Systems Development: Research Design .................................102
3.6.1 Construct a Conceptual Framework......................................................103
3.6.2 Develop a System Architecture..............................................................105
3.6.3 Analyze and Design the System.............................................................105
3.6.4 Build the System ..................................................................................106
3.6.5 Observe and Evaluate the System ..........................................................107

3.7 Phase II Observation: Case Study Design ..............................................107

3.8 Phase II Theory Building .........................................................................108

CHAPTER 4 – SCENARIO USAGE IN CSEM ..................................................109

4.1 A Conceptual Framework for CSEM Scenario Usage ...........................109
  4.1.1 Requirements Phase ...........................................................................110
  4.1.2 Planning Phase ..................................................................................111
  4.1.3 Design Phase .....................................................................................112
  4.1.4 Implementation Phase ........................................................................112
  4.1.5 Summary of Scenario Usage and Relationships ..............................113

4.2 Preliminary Collaborative Scenario Elicitation Methodology .................115

4.3 Initial Contextual Scenario Data Model ..................................................120
TABLE OF CONTENTS – Continued

CHAPTER 5 – RESULTS ................................................................................................................. 126

PHASE I: Collaborative Scenario Elicitation Using GroupSystems .................. 126

5.1 Case Study Results: Unstructured Group Systems Scenarios ............... 126
  5.1.1 Background ................................................................................................................. 126
  5.1.2 Cultural and Natural Resources Management Meeting ......................... 127

5.2 Experimental Results: The Impact of Scenario Format ....................... 130
  5.2.1 Participant Demographics and Motivation ......................................................... 131
  5.2.2 Assessing Overall Scenario Quality and Productivity ............................... 131
  5.2.3 Comparing Scenario Quality and Productivity ............................................... 134
  5.2.4 Analysis of Post-Session Questionnaire Results ............................................ 136

5.3 Case Study Results: Structured Group Systems Scenarios .................. 138
  5.3.1 Background ................................................................................................................. 139
  5.3.2 Waste Management Meetings ............................................................................. 139
    5.3.2.1 Waste Management Meeting 1 ..................................................................... 139
    5.3.2.2 Waste Management Meeting 2 ..................................................................... 141
  5.3.3 Hazardous Material Management Meetings ................................................... 142
    5.3.3.1 Hazardous Material Management Meeting 1 .......................................... 142
    5.3.3.2 Hazardous Material Management Meeting 2 .......................................... 144
    5.3.3.3 Hazardous Material Management Meeting 3 .......................................... 145

5.4 Phase I Results Summary ......................................................................................... 146
TABLE OF CONTENTS – Continued

5.4.1 Unstructured Group Systems Scenarios Results Summary .......... 146
5.4.2 Experimental Results Summary .................................. 147
5.4.3 Structured Group Systems Scenarios Results Summary .......... 148
5.4.4 Motivation for Phase II Research ................................ 149

PHASE II: Integrated Scenario and Process Modeling Using SPA .......... 150

5.5 SPA Systems Development ........................................... 150

5.5.1 Construction of a Conceptual Framework ....................... 150
  5.5.1.1 Core SPA Prototype Functionality .......................... 151
  5.5.1.2 Business Process Modeling .................................. 152
  5.5.1.3 Integrated Scenario and Process Modeling Framework ...... 157
  5.5.1.4 SPA System Building Process ................................ 159

5.5.2 Development of a SPA System Architecture ...................... 160
  5.5.2.1 Constraints .................................................. 160
  5.5.2.2 Development Objectives .................................... 161
  5.5.2.3 Measurable Requirements ................................... 162
  5.5.2.4 Architecture Overview .................................... 163

5.5.3 Analysis and Design of SPA ....................................... 165
  5.5.3.1 Logical Database Design .................................... 165
  5.5.3.2 User Interface Design ....................................... 166

5.5.4 Building SPA ...................................................... 168
TABLE OF CONTENTS – Continued

5.5.5 Observing and Evaluating SPA .................................................. 169

5.6 Case Study Results: Evaluating SPA ................................................. 169
  5.6.1 Background ............................................................................. 170
  5.6.2 Air Force Meeting 1: Same Time, Same Place ........................... 171
  5.6.3 Air Force Meeting 2: Same Time, Different Place ..................... 178
  5.6.4 Air Force Meeting 3: Different Time, Different Place ............... 184
  5.6.5 Summary ................................................................................ 187

CHAPTER 6 – DISCUSSION .................................................................. 188

PHASE I: Collaborative Scenario Elicitation Using GroupSystems ........ 188

6.1 Phase I Observations .................................................................... 188
  6.1.1 Interpretation of Case Study Results ...................................... 189
  6.1.2 Limitations of the Case Studies ............................................. 192

6.2 Phase I Experiment ....................................................................... 193
  6.2.1 Interpretation of Experimental Results ................................... 194
  6.2.2 Limitations of the Experiment ............................................... 195

6.3 Phase I Theory Building ............................................................... 196

PHASE II: Integrated Scenario and Process Modeling Using SPA .......... 198

6.4 Phase II Systems Development ..................................................... 198

6.5 Phase II Observations .................................................................. 204
  6.5.1 Interpretation of the Case Study Results ............................... 204
TABLE OF CONTENTS – Continued

6.5.2 Limitations of the Case Studies .................................................. 206

6.6 Phase II Theory Building .................................................................. 207

CHAPTER 7 – CONCLUSIONS AND FUTURE RESEARCH ..................... 210

7.1 Conclusions ...................................................................................... 210

7.2 Limitations ....................................................................................... 215

7.3 Contributions ..................................................................................... 217

7.4 Future Research ................................................................................ 218

APPENDIX A – COMPLETE SAMPLE SCENARIO ................................... 222

APPENDIX B – UNSTRUCTURED SCENARIO INSTRUCTIONS .................. 225

B.1 Generating Scenarios ........................................................................ 225

B.2 Scenario Examples ............................................................................ 225

B.2.1 Outline Scenario (Example 1) ....................................................... 225

B.2.2 Outline Scenario (Example 2) ....................................................... 226

B.2.3 Narrative Scenario (Example 3) ..................................................... 226

APPENDIX C – EXPERIMENT DOCUMENTATION ................................ 227

C.1 Experiment Outline/Script .................................................................. 227

C.2 Treatment 1 Instructions and Example ............................................... 229

C.2.1 Scenario Description and Instructions ........................................... 229

C.2.2 Scenario Example 1: Student Checks Grades using UA Info .......... 230

C.3 Treatment 2 Instructions and Example ............................................... 231
TABLE OF CONTENTS – Continued

C.3.1 Scenario Description and Instructions ........................................... 231
C.3.2 Scenario Example 2: Student Checks Grades using UA Info ....... 232
C.4 Treatment 3 Instructions and Example ............................................. 233
C.4.1 Scenario Description and Instructions ........................................... 233
C.4.2 Scenario Example 3: Student Checks Grades using UA Info ....... 234
C.5 Post-Session Questionnaire ............................................................... 236

APPENDIX D – EXPERIMENT RESULTS ....................................................... 238

APPENDIX E – SPA REQUIREMENTS DOCUMENTS ........................................ 239
E.1 SPA Functional Requirements List .................................................. 239
E.2 SPA Logical Data Model ................................................................. 248
E.3 SPA Data Entity and Attribute Definitions ....................................... 249

APPENDIX F – SPA EVALUATION RESULTS ............................................... 258
F.1 Air Force Meeting 1 Evaluation Results ........................................... 258
F.2 Air Force Meeting 2 Evaluation Results ........................................... 269

APPENDIX G – METHODOLOGY FOR COLLABORATIVE SCENARIO
ELICITATION USING GROUPSYSTEMS .................................................. 281

APPENDIX H – METHODOLOGY FOR COLLABORATIVE SCENARIO
ELICITATION USING SPA ................................................................. 289

REFERENCES ......................................................................................... 298
LIST OF FIGURES

Figure 1 – Relative cost to fix errors................................................................. 44
Figure 2 – Collaborative Software Engineering Methodology ............................ 53
Figure 3 – Types of scenarios ........................................................................ 60
Figure 4 – Multi-methodological research approach ......................................... 76
Figure 5 – Simplified GSS research model ........................................................ 79
Figure 6 – Scenario experiment instructions ..................................................... 96
Figure 7 – Unstructured scenario format ............................................................ 97
Figure 8 – Numbered step format .................................................................... 98
Figure 9 – Structured step format ..................................................................... 98
Figure 10 – CSEM scenario usage .................................................................. 109
Figure 11 – Test scenarios .............................................................................. 113
Figure 12 – Scenario relationships ................................................................. 114
Figure 13 – CSEM requirements elicitation steps ............................................. 116
Figure 14 – Action completeness .................................................................... 132
Figure 15 – IDEF3 graphical symbols .............................................................. 154
Figure 16 – UML Activity Diagram graphical symbols ...................................... 156
Figure 17 – SPA process hierarchy and information panels ............................. 167
Figure 18 – SPA process hierarchy and graphical view .................................... 168
Figure 19 – Group decomposes business activities (Step 1) ........................... 283
Figure 20 – Facilitator adds scenario template outline (Step 2) .................... 283
LIST OF FIGURES – Continued

Figure 21 – Users add scenario information as comments (Step 3) ................................. 285
Figure 22 – Facilitator copies group review results to template (Step 4) ............................. 285
Figure 23 – Users identify scenario actions (Step 5) .......................................................... 286
Figure 24 – Group reviews and agrees on scenario actions (Step 6) .................................... 286
Figure 25 – Users identify exceptions and related issues (Step 7) ....................................... 288
LIST OF TABLES

Table 1 – Comparison of scenario and traditional perspectives ........................................ 61
Table 2 – A chronological overview of research tasks ...................................................... 78
Table 3 – Types and sources of evidence ........................................................................ 88
Table 4 – Example of scenario-level information .............................................................. 123
Table 5 – Example of action-level information ................................................................. 124
Table 6 – Scenario statistics ........................................................................................... 132
Table 7 – Comparison of the three treatment formats ....................................................... 134
Table 8 – Comparison of the unstructured vs. structured formats .................................... 134
Table 9 – Questionnaire factor analysis summary ............................................................ 137
Table 10 – Comparison of questionnaire standardized factor scores ............................... 137
Table 11 – IDEF3 process description constructs ............................................................ 153
Table 12 – UML Activity Diagram constructs .................................................................. 156
Table 13 – Summary of Phase I case study results ......................................................... 188
Table 14 – Experimental results summary ...................................................................... 193
Table 15 – Task/technology fit and GSS effectiveness ..................................................... 208
Table 16 – Factor loadings .............................................................................................. 238
Information systems development research has documented the importance and the difficulty of eliciting requirements from users. Research on the use of Group Support Systems (GSS) for requirements elicitation led to development of the Collaborative Software Engineering Methodology (CSEM) and identified the need for collaborative methods and tools to provide a dynamic picture of the business processes that a system must support. Recent research suggests that scenarios can fill this need. A review of the scenario literature showed that although there is widespread agreement on the usefulness of scenarios, there are many questions on how to implement a user-focused, scenario-based systems development process. The purpose of this research was to advance understanding in this area and to determine: What are the collaborative modeling processes, tools, and facilitation techniques needed to effectively elicit scenarios from users in a group environment?

A two-phase, multi-method systems development research approach was used. The first phase focused on use of a general-purpose GSS for collaborative scenario elicitation. A conceptual framework and initial methodology were developed and then evaluated during exploratory case studies and a laboratory experiment. The second phase focused on development and evaluation of a special-purpose GSS and methodology.

Phase I results showed that: users can easily define scenarios which provide rich pictures of the problem domain; an iterative, collaborative methodology with scenario and action prompts is needed to ensure scenario completeness; and limitations of
general-purpose GSS negatively impacted productivity. The Collaborative Distributed Scenario and Process Analyzer (SPA) provides integrated textual scenario and graphical process modeling capabilities which successfully overcame these limitations.

This research made several contributions. CSEM was extended to define scenario usage opportunities throughout development. Scenario content, form, group process and facilitation techniques were defined for collaborative scenario elicitation using a general-purpose GSS, which can be used now by practitioners. A special-purpose GSS tool (SPA) was developed and integrated into a comprehensive methodology which allows user groups to rapidly define and analyze scenarios in face-to-face and distributed settings. Finally, flexibility designed into SPA opens up opportunities for many other uses for SPA and serves as a first-step towards a build-your-own GSS tool.
CHAPTER 1 – INTRODUCTION

The hardest single part of building a software system is deciding precisely what to build. No other part of the conceptual work is so 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. (Brooks, 1986)

1.1 Background

Several decades of information systems development research support Frederick Brooks’ claim and have clearly shown both the importance and the difficulty of developing complete and accurate information systems requirements. Many of the seemingly endless reports of spectacular system failures have been attributed to requirements problems (Standish, 1995). Other studies have shown that problems are 200 times more expensive to correct during testing than during the requirements phase (Boehm, 1981). Most researchers and practitioners agree that user involvement is critical to the success of the requirements process (Carmel, Whitaker, & George, 1993). A special challenge is in determining how to effectively involve groups of users in that process.

Traditional requirements elicitation techniques depend on user interviews or group meetings as the primary mechanisms for involving users. However, these techniques are notoriously inefficient, especially when dealing with large, highly diverse user groups. The problem of gathering accurate requirements and the inefficiencies of user interviews and group meetings were some of the early driving forces in Group Support Systems (GSS) research. GSS have been highly successful in improving group meeting
productivity and outcomes in real-world settings (Nunamaker, Briggs, Mittleman, Vogel, & Balthazard, 1996-97). On-going research on the use of GSS for collaborative requirements elicitation has led to the development of special-purpose GSS tools for activity and data modeling and an over-arching Collaborative Software Engineering Methodology (CSEM) (Dean, Lee, Pendergast, Hickey, & Nunamaker, 1997-98). However, development of CSEM identified a need for additional collaborative tools to support key aspects of the requirements elicitation process. The Activity Modeler provides a structured mechanism for establishing the scope of the system and identifying the functions that it should support (Dean, Lee, Orwig, & Vogel, 1994-95). The Group Data Modeler has proven effective for capturing data requirements (Lee, Dean, & Vogel, 1997). Neither provides a dynamic picture of the business processes that a system must support. Recent research has suggested that scenarios may be an effective mechanism for eliciting behavioral requirements.

1.2 The Research Problem

Scenarios are narrative descriptions of human work processes that specifically identify the sequence of actions taken to accomplish a goal (Carroll, 1995). They are being increasingly used throughout all phases of information systems development (Carroll, 1995; Rolland et al., 1998a) including requirements elicitation (Potts, Takahashi, & Anton, 1994; Weidenhaupt, Pohl, Jarke, & Haumer, 1998). Research indicates that this increased use is because concrete scenarios are more compatible with user's situational problem solving tendencies than traditional abstract requirements (Carroll, 1995). They also help reduce complexity (Weidenhaupt et al., 1998). However,
although there is widespread agreement on the usefulness of scenarios, there is also an extremely wide diversity of approaches for using scenarios to support systems development.

Rolland et al. (1998a) surveyed scenario-based approaches identified in the research literature and found significant variation in: (1) why scenarios were used (purpose), (2) whether scenarios were static or dynamic (life cycle), (3) the information included in scenarios (content), and (4) the mechanism for expressing scenarios (form). Weidenhaupt et al. (1998) used this same classification framework to survey scenario use in industry. Comparisons of scenario research and industrial practice found that, in practice: (1) scenarios are used for a much wider range of purposes than identified in the research literature, (2) scenario life-cycles were much more complex than addressed in the research literature, (3) scenario content focused on the human-computer interaction but also addressed the broader organization context or narrower internal system interaction which was similar to the research approaches, and (4) scenario form was primarily textual and did not take advantage of the richer graphical and animated forms focused on by research approaches (Jarke, Bui, & Carroll, 1999). These disconnects between research and practice provide a rich source of research opportunities. In particular, practitioners are seeking much more specific process guidance and improved tool support (Weidenhaupt et al., 1998). These problems become even more complex when attempting to integrate scenarios into a user-focused, collaborative requirements elicitation process such as the one proposed by the Collaborative Software Engineering Methodology (CSEM).
1.3 Research Questions

The primary objective of this research is to provide the guidance currently lacking in the literature by determining how scenarios can be elicited directly from large, diverse user groups so that the resulting scenarios can be used to support user collaboration during the requirements and other information systems development phases. More specifically, the purpose of this research is to determine:

What are the collaborative modeling processes, tools, and facilitation techniques needed to effectively elicit scenarios from users in a group environment?

Preliminary analysis of this question in conjunction with a review of scenario research and practice highlighted a number of additional, more detailed questions that should also be addressed. The scenario classification framework proposed by Rolland et al. (1998a) helped to both identify and to organize these questions.

**Purpose.** Scenarios were initially included in CSEM to support collaborative requirements elicitation. While the primary focus of this research continues to be on that use, both research and practice have identified a number of other potential scenario uses which could be of benefit in CSEM. These uses must be more clearly understood to ensure that the scenario elicitation methodology developed as part of this dissertation supports (or can be easily modified to support) these additional uses. This need gives rise to the following question:

How will scenarios be used during all CSEM development phases and how are the scenarios developed for these different purposes related?

**Lifecycle.** Researchers generally take into account a static view of scenarios while practitioners express a need for guidance for managing scenarios as they change and
evolve during the systems development process. In addition to considering the various purposes scenarios can be used in CSEM, it is also essential to understand:

How will scenarios change and evolve during development and how can that change be effectively supported?

**Content.** Required scenario information content may vary significantly based on the proposed use of scenarios. In addition, while users may be able to easily provide some types of scenario information, other information may require more technical knowledge than users possess and should be provided by developers. The focus of this research is on elicitation of scenarios directly from users to support requirements determination. Rich, contextual scenario descriptions that describe work processes in a broad organizational context can easily be documented by users and provide the type of information needed for requirements determination. Several researchers identify information that should be included in a contextual scenario description (e.g., Benner, Feather, Johnson, & Zorman, 1993; Carroll, 1995; Leite et al., 1997; Potts et al., 1994). The most comprehensive analysis is Pohl and Haumer’s (1997) survey of contextual scenario modeling techniques with recommended scenario and interaction context models. These proposals must be evaluated in terms of CSEM requirements to determine:

What information should be included in the contextual scenarios elicited directly from user groups?

**Format.** Researchers have also proposed many alternative scenario representations including: unstructured natural language (e.g., Carroll, 1995); structured natural language such as tabular scripts (Potts et al., 1994) or grammars (Leite et al., 1997);
various modeling techniques such as message sequence charts (Regnell, Andersson, & Bergstrand, 1996) or scenario trees (Hsia et al., 1994); or other graphical representations (e.g., Benner et al., 1993). However, limited research has been done on which representation is easiest for users or results in the highest quality scenario descriptions, which leads to the question:

What format should be used for contextual scenarios and how does that format impact scenario quality and productivity of user definition of scenarios?

In addition to the four areas identified above, industrial survey results highlighted the need for improved process guidance and tool support (Weidenhaupt et al., 1998). Lessons learned from GSS research also indicate that group process, facilitation techniques, and GSS tools must be considered to ensure effective group meetings (Nunamaker et al., 1996-97). Therefore, these areas must also be considered when developing a comprehensive collaborative scenario elicitation methodology.

**Group Process and Facilitation.** Scenario researchers have generally not provided detailed process guidance (Jarke et al., 1999). Practitioners have had to significantly extend existing scenario approaches as part of their implementation process (Weidenhaupt et al., 1998). Even less guidance is available on how to collaboratively elicit scenarios from users; especially on how those techniques should vary based on the characteristics of the specific tool used. A primary goal of this research is to determine:

What group process and facilitation techniques are appropriate for each collaborative tool used to elicit contextual scenarios from user groups?
Collaborative Tool Support. Researchers are actively developing tools to support scenario-based approaches (e.g., Maiden, Minocha, Manning, & Ryan, 1998; Sutcliffe, Maiden, Minocha, & Manuel, 1998). However, these tools have not yet transitioned into practice. In fact, Weidenhaupt et al. (1998) reported that the most commonly used tool in industry is the word processor. In addition, none of these tools provide the simultaneous user support needed for a collaborative process such as CSEM. GSS provide the necessary collaborative support, but it is not clear whether general-purpose GSS tools can provide the desired level of support for an effective scenario elicitation process.

Specifically, the question is:

Can general-purpose GSS tools be used for collaborative scenario elicitation or is a special-purpose tool needed?

If a special-purpose collaborative tool is needed, then research is needed to determine:

What are the functional requirements for a collaborative scenario modeling tool?

1.4 Research Approach

A multi-methodological systems development research approach incorporating theory building, systems development, experimentation, and observation was used to address the research questions identified above (Nunamaker, 1992; Nunamaker, Chen, & Purdin, 1991a). Because one of the criteria for using a systems development research methodology is that a new system must provide a better solution than existing systems (Nunamaker et al., 1991a), the research was conducted in two phases. The purpose of the first phase was to determine how existing GSS tools could be used to support
collaborative scenario elicitation. Results of the first phase provided justification for development of a new system which was the focus of the second research phase.

During the first phase, results of the scenario literature review and prior GSS research guided theory building to develop an initial methodology for collaborative elicitation of scenarios using a general-purpose GSS. Exploratory case studies of real-world scenario meetings and a laboratory experiment were used to increase understanding of the issues involved in collaborative scenario elicitation and to refine the proposed methodology. Results from the first phase identified significant problems and limitations associated with using a general-purpose GSS to support collaborative scenario elicitation. Therefore, the second phase of the research focused on development and evaluation of a special-purpose GSS designed to overcome these problems.

The systems development research process outlined in (Nunamaker et al., 1991a) was used during the second phase to guide development of the Collaborative Distributed Scenario and Process Analyzer (ColD SPA) prototype. Exploratory case studies were conducted to observe and evaluate use of the prototype during real-world meetings. Results of these studies were used to revise the collaborative scenario elicitation methodology developed during the first phase and to identify needed prototype improvements and other areas requiring additional research.

The theoretical GSS research model proposed in (Dennis, George, Jessup, Nunamaker, & Vogel, 1988) which has guided GSS research throughout the years (Nunamaker, Dennis, Valacich, Vogel, & George, 1991c) was used as the underlying model for the experiment and case studies conducted during both research phases. This
model emphasizes that characteristics of the group, task, context, and technology influence the process used, which influences the meeting outcome (Dennis et al., 1988). These characteristics were either controlled during research design or documented as part of the research results for all methods. This common research model was used to improve comparability between phases and methods and to support an integrated assessment of the complete research effort.

1.5 Importance of the Research

This research is needed to increase understanding of the issues involved in collaborative scenario elicitation. This increased understanding will enable modification of CSEM to take advantage of the benefits of scenarios throughout all development phases, providing a comprehensive life cycle perspective not addressed in the current literature. Developing specific recommendations for scenario content, form, group process and facilitation techniques will enable practitioners to implement a user-focused, scenario-based development process now and will represent an improvement of the limited process guidance currently available in the literature. The research will also define how a general-purpose GSS can be used to support the recommended scenario process. This will allow practitioners to rapidly gather requirements directly from large groups of users, in terms users understand, via collaborative definition and analysis of business scenarios. A general-purpose GSS will provide better support for collaborative scenario elicitation that the most commonly used tool, the word processor. However, even better support could be provided by a special-purpose GSS which integrates a textual scenario and graphical process modeling capability into a single collaborative
tool specifically designed for user scenario definition. Integrating such a tool into a comprehensive methodology for collaborative scenario elicitation will allow user groups to rapidly define, analyze, and achieve consensus on the large number of scenarios necessary to define requirements for today's complex information systems. The resulting improvements in user participation and the requirements process will address two of the major causes of systems development problems identified by the Standish Group. As a result, the chance of successful systems development success should greatly increase, and users will be rewarded with systems that meet or exceed their expectations.

1.6 Overview of the Dissertation

The remainder of this dissertation is organized as follows. Chapter 2 provides a detailed discussion of the literature on scenarios, supported by a review of the systems development literature on the development process, requirements elicitation, user participation and GSS usage. Chapter 3 defines the research approach. Chapter 4 frames the research by providing a comprehensive view of scenario usage in CSEM. Research results are presented in Chapter 5 with implications of those results discussed in Chapter 6. The dissertation concludes with Chapter 7 which summarizes the contributions and limitations of the research and identifies opportunities for future research.
CHAPTER 2 – LITERATURE REVIEW

2.1 Information Systems Development

Information systems development represents both the general research problem area and the research methodology for this dissertation. Therefore, this review of the literature begins with an overview of the importance of research in this area, followed of a summary of alternative approaches to software development. The review continues to drill down to the dissertation’s specific focus on collaborative requirements elicitation using scenarios through discussions of the literature on requirements engineering (section 2.2), GSS for collaborative information systems development (section 2.3), and scenarios (section 2.4).

2.1.1 Software Development Problems

In 1995, the Standish Group estimated that more than $250 billion per year was being spent on software development in the United States (Standish, 1995). The performance record of the IS industry, however, is nowhere near as impressive as this investment. For example, the news is full of stories about the Y2K computer problem and spectacular software development failures. The Government Accounting Office has written numerous reports documenting the seemingly endless problems the Internal Revenue Service is encountering as it attempts to modernize its computer systems.

To determine the scope of the software development problem, the Standish Group (1995) surveyed 365 IT executives representing 8,380 application development projects. The overall project success rate was only 16.2%, with another 52.7% projects delivered
either late, over budget, or with reduced functionality. Over 30% of projects would never be implemented, costing U.S. firms an estimated $81 billion. Survey participants identified requirements problems and lack of user involvement as the top two causes of project problems or cancellations. Resolving these two problems were two of the top three criteria for increasing the potential for project success. These issues are discussed in section 2.2. Standish Group research also indicated that moving towards more rapid evolutionary and incremental software development processes, such as those discussed in the next section, should also increase the success rate. (Standish, 1995)

2.1.2 The Software Development Process

Software engineering can be viewed as a combination of a software process with specific software methods supported by appropriate software tools. The software process defines the general activities, tasks, milestones, and deliverables of the development effort (Pressman, 1997). This is also often referred to as a systems development life cycle (SDLC) (Schach, 1996). Most software development texts, such as those by Conger (1994), DeGrace and Stahl (1990), and Schach (1996), provide excellent summaries of software development methodologies and life cycles. McConnell (1996) also provides a very good comparison and evaluation of the various life cycles in a rapid development environment. Drawing on the material from these references, abbreviated descriptions of the major life cycles options with their advantages and disadvantages are provided in the following sections.
2.1.2.1 The Waterfall SDLC

The Waterfall SDLC was developed in the 1960s to provide defense contractors with a project life cycle and documentation standard. It is also called the Sequential SDLC since phases occur in sequence with the output of each phase feeding the next phase. Although the exact number and names of phases vary depending on the source, typical phases include: Analysis, Design, Programming, Testing, and Implementation.

The principal advantages of this SDLC is its clear definition of phases with specific milestones and reviews required to transition to the next phase and its emphasis that all activities be fully documented. These advantages, however, also represent its primary disadvantages as well. Although its rigid structure provides excellent project control, it does not permit adaptation to the needs of the specific project. In addition, documentation oftentimes became the end rather than the means with massive amounts ofunread documentation being produced. The other major problems of the waterfall SDLC include incomplete and changing requirements definitions, excessive costs and time, and communication problems with users.

2.1.2.2 Evolutionary Development

Evolutionary SDLCs are often used today to overcome the problems of the Waterfall SDLC. Under an evolutionary SDLC, a subset of the user's requirements are analyzed, designed, developed, and implemented. Since only a subset of the requirements is addressed, these systems can obviously be delivered more rapidly than the complete system under the Waterfall SDLC. Additional requirements are then identified and added to the system in an evolutionary manner with each delivery
representing a greater portion of the user’s requirements. This approach allows requirements analysis on a “just-in-time” basis, eliminating many of the requirements problems of the Waterfall SDLC. Also, because of the many, short, evolutionary development cycles, closer contact is maintained with users, limiting potential communication gaps. The primary problem with an evolutionary SDLC is that it can easily devolve into a “build-and-fix” approach unless a strong, flexible architecture is implemented during initial development. The other major problem is that neither users nor managers clearly understand when requirements will be met and oftentimes even when systems will be delivered.

2.1.2.3 Incremental Development

The Incremental SDLC is very similar to an evolutionary SDLC in that it delivers the system in increments which meet subsets of the user’s requirements. Some versions of the incremental SDLC assume that a comprehensive requirements analysis and architectural design is completed up-front and only the detailed design and implementation phases are performed incrementally. Other approaches to incremental development simply call for high-level requirements analysis and design initially with detailed requirements analysis and design accomplished during each increment’s development process. Regardless of the level of detail for the up-front analysis and design, the goal of an incremental SDLC is to deliver the system in pre-planned increments. As a result, users understand when their requirements will be met.

Alan Davis, in his book on 201 Principles of Software Development, strongly supports the use of incremental development. One of his key development principles is:
Grow systems incrementally. One of the most effective techniques to reduce risk in building software is to grow it incrementally. Start small, with a working system that implements only a few functions.... The advantages are (1) lower risk with each build, and (2) seeing a version of the product often helps users envision other functions they would like. The disadvantage is that, if an inappropriate architecture is selected early, a complete redesign may be necessary to accommodate later changes. (Davis, 1995, p. 21)

As Davis warns, the primary problem with an incremental SDLC, like that of an evolutionary SDLC, is that it can degenerate into “build-and-fix” development. However, because of the incremental approach’s emphasis on up-front requirements analysis and architectural design, this problem is less likely to occur with incremental development.

2.1.2.4 The Spiral Model

The Spiral SDLC is Boehm’s evolutionary development SDLC which focuses on risk management (Boehm, 1981). His focus on controlling project, management, and other systems development risks alleviates many of the management problems encountered during other evolutionary SDLCs. Each evolutionary cycle includes analysis of objectives for the cycle, risk management, development, and then planning for the next cycle. Prototypes are used to help manage risk. The system evolves until the last cycle when a structured development cycle occurs to produce the final, tested system. While the Spiral Model is extremely strong in the area of risk management, it is much less clear about how or when developers should advance to the next evolutionary cycle. It is also not clear whether incremental fielding occurs or whether system delivery occurs only after the final, structured evolutionary cycle.
2.1.2.5 Prototyping

Prototyping can be used to replace individual phases of the Waterfall SDLC for proof of concept or to assist in clarifying specific user requirements. Prototypes can also be used for complete iterative development as part of an evolutionary or incremental software development process when requirements are not well-understood. They are also used for rapid development of portions of the system needed to meet urgent user requirements. Despite their obvious power, prototypes are subject to misuse. Conger identifies the major prototyping problems as: “misuse to circumvent proper analysis and design, and never completing prototypes as proper applications” (Conger, 1994). Other advantages and disadvantages of a prototype SDLC are similar to those for incremental and evolutionary development. The most critical of these is the tendency to degenerate into a “build-and-fix” approach to software development.

2.1.2.6 Rapid Application Development (RAD)

Most RAD methodologies focus on delivering systems in an extremely short time-frame, often as rapidly as 3 - 6 months (Kerr, 1994). While RAD methodologies use many techniques such as prototyping to speed up the application development process, they also rigidly restrict the scope of the system to one that can be delivered within the specified time-frame. On this basis, RAD methodologies can be considered a special case of an evolutionary or incremental SDLC with the same advantages and disadvantages.
2.1.3 Summary

In a roundtable discussion of software engineering experts on whether Internet applications could be engineered, one of the experts said that "the waterfall model is dead, the spiral model is dying, and the rapid prototype is becoming the product" (Pressman, 1998, p. 106). Although this statement may be a little extreme, it is true that the demands for rapid development are increasing for all types of applications. As a result, most software developers are using or moving towards evolutionary and incremental prototype-based processes. Regardless of the process used, requirements must still be defined with users involved in the process. These topics are discussed next.

2.2 Requirements Engineering

This phase of the literature review explored the requirements engineering process. The first section provides an overview of the requirements process. Next, the need for user participation in the requirements process is explored. This is followed by a review of techniques for requirements elicitation, analysis and specification. The session concludes with a discussion of requirements importance, problems, and trends.

2.2.1 Requirements Overview

One of the universal truths of requirements engineering is that requirements describe what the system should do and not how it should do it (Siddiqi, 1994). The problem with this traditional definition, as Alan Davis explains in detail, is that "one person’s how is another person’s what" (Davis, 1993, p. 17). To counter this problem, Thayer and Dorfman (1997) defines a requirement as "(1) a condition or capability needed by a user to solve a problem or achieve an objective; (2) a condition or capability
that must be met or possessed by a system... to satisfy a contract, standard, specification of other formally imposed document." Mil-Std 498 includes a similar definition of a requirement as "a characteristic that a system... must possess in order to be acceptable to the acquirer."

There is also a similar needs vs. wants misconception. The objective of requirements engineering is to determine what the customer needs, not what the customer wants as is commonly stated (Davis, 1993; Schach, 1996). This focus on the customer leads to an assumption that the customer must be involved in the requirements engineering process. The need for user participation is discussed further in section 2.2.2.

There are a wide variety of methods for accomplishing the above-stated objective, most of which are related by common underlying principles. Pressman (1997) states that all requirements engineering methods must define the problem's information domain, the software's functions, and the external behavior of the software. Davis (1993) agrees, but characterizes the three essential components which must be addressed as objects, functions, and states. Several others argue that requirements methods must go beyond this focus on the software and take a wider view of the problem domain. For example, Jackson believes that requirements are concerned with the purpose of software, which is found in the problem context (Jackson, 1995; Siddiqi & Shekaran, 1996). Butler et al. (1999) maintain that since software implements a work process, design of those work processes must be considered in conjunction with the design of software. Many participatory design approaches such as ETHICS (Mumford, 1995) and the Soft Systems Methodology (Checkland, 1990; Checkland & Scholes, 1991) implement this approach.
Combining these views, it can now be said that requirements engineering methods should address the objects, functions, and states of both the software and the problem domain.

Requirements engineering methods also implement a requirements life cycle within the framework of the systems development life cycle (SDLC). However, just as occurs for SDLCs, there is no universal agreement on how to divide and name the activities of the requirements life cycle (Siddiqi & Shekaran, 1996). Davis (1993) divides the life cycle into two activities: problem analysis and product description. Graham (1998) identifies two aspects: requirements elicitation and requirements analysis. Zave (1997) propose three phases: elicitation, validation, and specification. Jarke and Pohl (1994) identifies three different, but similar tasks: elicitation, expression, and validation. Later, Pohl (1996) expanded his list to define four primary tasks: elicitation, negotiation, specification/documentation, and validation/verification.

Extracting the essence of these approaches, this dissertation will consider three primary activities: elicitation, analysis, and specification. An overview of methods in each of these areas is provided in sections 2.2.3 - 2.2.5, following the more general discussion of user participation.

2.2.2 User Participation

The information systems community believes “the success of a system is proportional to the degree to which the ‘users’ of that system are ‘involved’ in its design and development” (Carmel et al., 1993, p. 40). The need for active user involvement is often listed as an essential principle for effective systems development (e.g., Graham,
1998). A 1995 survey of IS executives specifically identified user involvement as the number one factor impacting a software development project's success (Standish, 1995).

Reviews of the empirical research on the impact of user involvement on MIS success, however, have found little to only moderate support for this relationship (e.g., Ives & Olson, 1984; Pettingell, T., & Remington, 1988). These findings may be due in part to differences in what is meant by user participation and involvement. Barki and Hartwick (1989) argue for a clear distinction between user participation, "referring to the behaviours and activities that the users perform during IS development" (El Emam, Quintin, & Madhavji, 1996, p. 6), and user involvement as a psychological state related to the importance and relevance of the system to the user (Barki & Hartwick, 1989; El Emam et al., 1996). Mumford (1995) discusses three different types of user participation: consultative, representative, and consensus participation. The findings may have also been caused by differences in the way user participation was implemented, the moderating factors considered in the studies, and a host of other reasons. Not surprisingly, therefore, the results of individual studies have been mixed.

Even given the limited empirical support for user participation, researchers have identified several important reasons for involving users in information systems development projects. Many argue that users have the right to determine their own destinies and that, since IS impact their destinies, they have a moral right to be involved in development of those IS (Mumford, 1995). Others maintain that IS are a form or organizational change and that user participation is essential to acceptance of that change and the IS (El Emam et al., 1996). More pragmatically, as organizations and IS become
more complex, user participation is the only effective mechanism for gathering and reconciling all the information required to define requirements since users typically understand their requirements best (Curtis, Krasner, & Iscoe, 1988).

There are multiple methods of implementing user participation. Several are discussed in the following section of requirements elicitation techniques. Whatever the method chosen, however, the Standish Group identifies five critical factors for ensuring the success of that participation: (1) find the right users, (2) involve users early and often, (3) maintain communications throughout the life of the project, (4) make involvement easy, and, most importantly, (5) talk to users to find out what they need. (Standish, 1996).

2.2.3 Requirements Elicitation

The primary requirements elicitation technique is still the interview where the analyst asks the user to define his or her requirements (Schach, 1996). However, serial interviews with multiple users can be extremely time consuming and do not provide an efficient mechanism for resolving conflicting information (Conger, 1994). Other traditional elicitation techniques include: group meetings, observation, review of software, questionnaires, temporary assignments, review of internal documents, and review of external documents. Conger (1994) provides descriptions of each of these techniques and then summarizes their strengths and weaknesses.

To overcome the weaknesses of traditional techniques, many organizations are moving towards workshop, group, or other team-oriented methods. RAD-like techniques include users as part of the development team (Kerr, 1994). Workshop-based techniques
generally involve joint user – developer teams working together in facilitated workshops to identify and analyze problems, negotiate differences, and propose solutions (Pressman, 1997). In Europe, socio-technical methods such as participatory design (Carmel et al., 1993) and ETHICS (Mumford, 1995) are popular examples of workshop approaches. In the United States, more technically-oriented Joint Application Development (JAD) workshops are preferred (Carmel et al., 1993). JAD was originally developed as an innovative way for users and developers to work together to identify requirements, but it is now used for almost any kind of group problem-solving task. Traditionally, JAD consists of five phases: (1) defining the JAD project, (2) researching the project background by the JAD facilitator, (3) preparing for the session, (4) conducting the JAD facilitated structured workshop, and (5) producing the final document (Wood & Silver, 1995). The heart of JAD is the structured and facilitated meeting. Carmel et al. (1993) identify four key elements on which a successful JAD meeting is built: facilitation, agenda setting and structure, documentation, and group dynamics. Traditionally, JAD meetings have been “low-tech,” but are now becoming increasingly dependent on single-user CASE tools or other computer support (Wood & Silver, 1995). Benefits of JAD and other similar workshop-based approaches include: improved communication and feedback between users and developers, reduced user-versus-developer conflicts, improved meeting and group productivity, less-contentious resolution of conflicting views, reduced requirements development time and possibly cost, and establishment of a rapid development tempo (Graham, 1998; Wood & Silver, 1995).
The above discussion summarizes the methods for eliciting requirements from users. Once those requirements are elicited, they must be analyzed. The next section identifies some of the most common requirements analysis methods.

2.2.4 Requirements Analysis

Davis (1993) and Wieringa (1996) provide excellent overviews of the various requirements analysis techniques currently in use. Davis divides his discussion into those techniques focusing on objects (Coad's Object-Oriented Analysis, Jackson System Development, Entity Relationship Modeling), functions (Data flow diagrams, Data dictionaries, Structured requirements definition, Structured analysis and design technique (SADT), Structured analysis and system specification), and states (Finite state machines, State charts, Requirements Engineering Validation System, Petri nets) to name just a few of the available techniques. Wieringa's list adds some European-based techniques such as the Information Systems Work and Analysis of Changes and the Soft Systems Methodology. Conger (1994) compares process-oriented and data-oriented techniques such as Information Engineering.

Virtually all of these techniques have been designed for use by a systems analyst or requirements engineer. Although many include the user to a greater or lesser degree, it is still the trained analyst or engineer documenting the requirements using the technique's specific methods and models.
2.2.5 Requirements Specification

The literature review of requirements specification focused on two areas: (1) identifying the characteristics of a good software requirements specification and (2) reviewing existing software requirements specification standards.

One laundry list of the characteristics of a good software requirements specification includes: correct, unambiguous, complete, verifiable, consistent, understandable by customers, modifiable, traced, traceable, design independent, annotated, concise, and organized (Davis, 1993). Bahill et al. (1996) provide a similar list of characteristics from a systems requirements perspective. Wieringa (1996) restates these characteristics as: communicability, truth, completeness, verifiability, and maintainability. Most software development texts include similar laundry lists. Whatever list is used, it is clear that the great majority of requirements specifications do not satisfy all these characteristics. In addition, it may nor be feasible or possibly even desirable to even attempt to satisfy them all. However, these lists still serve as valuable goals during the specification development process.

Existing Software Requirements Specification (SRS) standards such as Mil-Std 498 and ANSI/IEEE 830 provide guidelines on what should be included in an SRS and how it should be organized. These standards do not dictate particular formats, but rather serve as guidelines and checklists to ensure that all required information is included. To assist in the selection of an appropriate standard, several SRS standards are reviewed and compared in (Giakoumakis & Xylomenos, 1996).
2.2.6 Requirements Summary

It's obvious from the previous discussion that requirements engineering takes time and money. Many managers and developers simply do not understand why they need to "waste" their scarce resources on this process. Research has clearly shown the importance of requirements in the software development process. As previously mentioned, the Standish Group survey identified requirements problems and lack of user involvement as the top two causes of project problems or cancellations (Standish, 1995). Other potential customer impacts of requirements problems include: the software may not satisfy customer's needs, different interpretations of the requirements may cause problems between customers and developers, it may not be possible to test if the software satisfies the requirements, and "both time and money may be wasted building the wrong system" (Davis, 1993).

![Figure 1 - Relative cost to fix errors](image-url)
Cost and timing are also critical because the relative cost of fixing requirements errors increases exponentially in later life cycle phases. Figure 1, developed based on data from Boehm’s survey of IBM, GTE and TRW software development (Boehm, 1976), shows this dramatic relationship. Although this chart provides a convincing argument for development of error-free requirements, a variety of potential requirements problems make this goal extremely difficult to achieve.

Requirements problems may result from the customers’ lack of understanding or an inability to clearly define their requirements (Schach, 1996). Many developers are integrating prototypes into their requirements process to aid users in visualizing their requirements (Andriole, 1994; Davis, 1993; McConnell, 1996; Schach, 1996).

Often developers may skip, rush through, or otherwise improperly conduct the requirements analysis (Schach, 1996). A recent survey of requirements engineering practices shows that even when developers attempt to properly conduct the requirements analysis, they face major challenges in deciding: how users should participate in the process, whether the existing system should be examined, if software packages should be evaluated as part of the requirements analysis, what level of detail is required for models and requirements, and other key project management issues (El Eman & Madhavji, 1995). The increasing emphasis on requirements engineering as a discipline may help deal with some of these problems.

The most overwhelming difficulty, however, is the nature of requirements themselves. Requirements are constantly changing. So, even if all the above problems are resolved and requirements are accurately specified, the requirements will often
change before the system can be implemented. Developers try to limit the impact of changing requirements by shortening the development life cycle through the use of evolutionary or incremental development processes.

Emerging trends in requirements engineering will also assist developers in dealing with these problems. Siddiqi and Shekaran (1996) conclude that "the next wave of requirements techniques and tools will account for the problem and development context, accommodate incompleteness, and recognize the evolutionary nature of requirements engineering." As discussed in the next section, user-focused tools such as Group Support Systems and GSS-enabled techniques are one possible move in this direction. Scenarios, discussed in section 2.4, are another.

2.3 Group Support Systems for Collaborative Requirements Elicitation

The problem of gathering accurate requirements and the inefficiencies of user interviews and group meetings were some of the early driving forces in Group Support Systems (GSS) research at the University of Arizona. An overview of this early work is presented in section 2.3.1. While GSS have been highly successful in improving general group meetings (Nunamaker et al., 1996-97), they've had mixed results in supporting electronic-JAD (E-JAD) meetings as discussed in section 2.3.2. On-going research on the use of GSS for collaborative requirements elicitation has led to the development of special-purpose GSS tools for activity and data modeling, discussed in section 2.3.3, and an over-arching Collaborative Software Engineering Methodology (CSEM), an overview of which is included in section 2.3.4.
2.3.1 History

The foundations of GSS research at the University of Arizona can be directly traced back to Problem Statement Language/Problem Statement Analyzer (PSL/PSA), developed as a part of the ISDOS (Information System Design and Optimization System) project (Teichrow & Hershey, 1982). PSL/PSA is a tool for formally describing and analyzing the requirements of a system. Dennis et al. (Dennis et al., 1988) provided a brief history of PSL/PSA development and highlighted some key factors which drove its evolution into Plexsys, the predecessor of today’s GroupSystems. PSL/PSA development started with an “assumption of correct requirements” (Dennis et al., 1988, p. 620). As a result, PSL/PSA focused on requirements specification and analysis rather than the elicitation of requirements from users. It was the experiences from using an early PSL/PSA prototype to support development of a Navy financial management system that began to change the direction of this research (Nunamaker & Konsynski, 1976). Navy users’ refusal to write the structured requirements specifications and the recognition that users groups often consisted of 10 – 20 people drove development of the Plexsys system. Plexsys was designed as an “analyst’s and user’s workbench” consisting of a methodology (Konsynski & Nunamaker, 1982), software, and a supporting meeting room (Dennis et al., 1988). The Plexsys software extended PSL/PSA by adding a collection of integrated tools to aid users in the requirements determination process. Over time, use of Plexsys changed from this original purpose to the discussion of general issues and problems (Dennis et al., 1988). It was this shift in usage that caused the redirection of research to Arizona’s development of a general-purpose GSS,
GroupSystems (Nunamaker et al., 1991c). During the 1990s, one stream of GSS research at the University of Arizona has shifted back to systems development.

2.3.2 Requirements Elicitation Using a General-Purpose GSS

General-purpose GSS have proven extremely useful in supporting requirements elicitation and analysis activities (Dean et al., 1997-98). The University of Arizona has successfully supported a wide variety of DOD and other user groups in requirements activities such as brainstorming on current problems, identifying potential solutions, developing high-level lists of functional requirements, and evaluating existing and prototype systems using GroupSystems tools.

In contrast, results of using GroupSystems to support electronic-JAD (E-JAD) meetings have been mixed. Carmel compared five E-JAD meetings using GroupSystems to five traditional JAD meetings (Carmel, 1991; Carmel, George, & Nunamaker, 1995; Carmel, George, & Nunamaker, 1992). He concluded that participation was greater and more equal in E-JAD meetings, but that conflict resolution and discipline were better in traditional meetings. One problem with this study was that trained JAD facilitators were only used during the traditional JAD meetings. E-JAD meetings relied on GroupSystems facilitators with limited to no JAD experience. Wood and Silver (1995) hypothesized that it was this lack of trained facilitators and traditional JAD preparation which caused the differences rather than the electronic support. Results of other studies (Chen & Liou, 1991; Liou & Nunamaker, 1993) have been more positive, but have also pointed out the limitations associated with using general-purpose GSS tools to support requirements elicitation (Chen, Nunamaker, & Weber, 1989). Many of these limitations are directly
caused by the lack of modeling support in general-purpose GSS. These tools do not provide many of the key functions required to support modeling including structured information and relationship capture capabilities, semantic checking, and complexity management mechanisms such as abstraction and information hiding (Dean et al., 1997-98). These limitations led to development of modeling GSS as described next.

### 2.3.3 Collaborative Modeling Tools

Researchers at the University of Arizona have developed two collaborative modeling tools, Activity Modeler and Group Data Modeler, designed to overcome the problems of traditional modeling meetings and single-user CASE tools. Descriptions of both these tools are provided in (Dean et al., 1997-98). The purpose of this discussion is to focus on the lessons learned from the development and use of these tools and to identify the key features which led to their successful support of collaborative modeling.

Activity Modeler supports collaborative development of business process models based on the IDEF0 standard. The history of the evolution of Activity Modeler is described in detail in (Dean, 1995; Dean et al., 1994-95; Dean, Orwig, Lee, & Vogel, 1994; Pendergast, Dean, Lee, Nevstrujev, & Katic, 1996). Key features of the tool and modeling process identified during evolution and use of the tool include the following:

- Interfaces must be simplified and explicitly geared to non-analyst users to limit the amount of training required and improve ease-of-use.
- Effective concurrency controls must be implemented to support parallel input of model data by users.
• Semantic checking is essential for development of high-quality models. The tool should check for errors and provide users recommendations for correcting the errors whenever possible.

• Repository support is required to avoid duplication of key model elements (e.g., duplicate inputs, outputs, controls, or mechanisms (ICOMs) in the activity model). Activity Modeler requires that users check the glossary before adding new ICOMs to the model.

• Text and graphical views should be integrated into a single tool. The ability for users to easily switch between views aids model comprehension. Early experiences with using a single-user CASE tool for graphical modeling support were problematic.

• Storing model data semantically versus graphically provided increased flexibility to interface with other tools. It did, however, require development of a Viewer to generate the graphical views from the semantic data.

• Skilled facilitators and analysts using tailored meeting processes are required to produce high-quality models. In particular, the method used to reconcile activity models significantly impacted model quality. (Dean, 1995; Dean et al., 1994-95; Dean et al., 1994; Dean, Orwig, & Vogel, 1996)

Group Data Modeler allows multiple participants to contribute and review data model content including entity names and definitions as well as attribute names, definitions, and meta-data. The history of the development, evolution, and use of Group
Data Modeler is described in (Dean, Lee, & Vogel, 1997; Lee, 1995; Lee et al., 1997).

Critical lessons learned about the tool and process include the following:

- The interface must be simplified and designed for the non-analyst user/novice modeler to avoid unnecessary complexity and overly restrictive modeling conventions. Screens should be tailored to the specific task to reduce complexity. For example, meta-data screens should only prompt for information that is directly applicable to the attribute’s data type.

- The tool should support tasks non-analysts can perform, e.g., entity and attribute definition, and avoid complex modeling issues like cardinality.

- Skilled facilitators and modelers using tailored meeting processes will improve model quality. Iterative processes which engage the user in stages (e.g., entities first, then attributes, and finally meta-data) are preferred. Starting the meeting with a preliminary model is also recommended.

Many of the lessons learned from the development and use of general-purpose software are equally applicable to collaborative modeling tools and are restated below.

- Build GSS software as independent special-purpose modules.
- Subtle differences in user interface can make big differences in group dynamics.
- Keep the user learning curve short; use simple interfaces.
- Provide easy import and export capabilities both between modules and with external tools.
- Provide for both task and process support. (Nunamaker et al., 1996-97, p. 178)

Successful experiences with Activity Modeler and Group Data Modeler have proven the effectiveness of special-purpose GSS in overcoming the limitations of general-purpose GSS for requirements modeling. These same experiences have shown,
however, that although general-purpose and modeling GSS are critical to the information systems development process, they do not by themselves provide all the requirements information needed. Nor do the specific meeting processes developed for the modeling GSS tools define a complete systems development methodology. Current University of Arizona research has therefore been extended to integrate the strengths of general and modeling GSS tools with the best systems development practices to develop the Collaborative Software Engineering Methodology (CSEM).

2.3.4 Collaborative Software Engineering Methodology

The Collaborative Software Engineering Methodology (CSEM) (Dean et al., 1997-98) provides a comprehensive framework that is guiding related research towards the goal of enabling participation of multiple users throughout the systems development process. CSEM was specifically designed to support incremental development, but can also be used for other development approaches. The CSEM life cycle consists of four loosely separated phases: planning, requirements, design, and implementation. As shown in Figure 2, CSEM phases are linked in a continuous cycle to emphasize the iterative and ongoing nature of systems development in today's organizations. The primary outputs of each phase are displayed as links between successive phases.

2.3.4.1 User Participation in CSEM

Because the goal of CSEM is to enable user participation, the maximization of user, analyst, and developer collaboration is a critical element of each phase. As a result, the Collaborative Software Engineering Life Cycle (Figure 2) is portrayed with this
user/analyst/developer collaboration at its center. Building out from this core are three different levels of user participation: selected users who represent the user groups; groups of users who represent the user community; and the entire user community.

![Collaborative Software Engineering Methodology](image)

**Figure 2 – Collaborative Software Engineering Methodology**  
(Dean et al., 1997-98)

This layering of user participation is required because CSEM is designed to be scaleable to large, complex information systems supporting extensive and diverse user populations. Although it is possible to survey these users periodically to identify their needs or provide them with status reports (e.g., during the planning phase), it is virtually impossible to involve all users on a regular basis. Therefore, a representative group of subject matter experts is drawn from the various user constituencies. User groups meet
regularly to generate requirements (e.g., business scenarios), resolve conflicting viewpoints (e.g., varying business processes), enhance proposed models (e.g., integrated data models) or prototypes, and validate key deliverables (e.g., prioritized requirements). Between meetings, selected group members work with analysts and developers either individually or in small group meetings to answer questions and develop initial models (e.g., system use cases). These initial models can be provided as read-aheads to the user group, greatly increasing the productivity of the large group meetings.

2.3.4.2 Collaborative Processes, Tools, and Facilitation Techniques

The productivity of the large group meetings is also enhanced by the methodology's emphasis on the use of collaborative processes, tools, and facilitation techniques which enable synchronous input by groups of non-analyst users without requiring them to become methodology experts. A combination of general-purpose GSS, collaborative modeling tools, and sophisticated CASE tools are used to support CSEM. The methodology also emphasizes the need for skilled facilitators applying well-designed meeting methods. Depending on the maturity of the group, the complexity of the task, and the specific development activity, these group processes may occur multiple times in a single group meeting or may be spread across several meetings. Skilled facilitators are needed to recognize these differences and to guide the user group towards efficient accomplishment of the its goals.

2.3.4.3 CSEM Phases

**Planning Phase.** The purpose of the planning phase is to identify high-level user needs and to map them into Information System (IS) development plans. An incremental
system development life cycle is used to support timely fielding of information system increments to rapidly satisfy customer requirements and to get real-world user feedback for future increments. Because of the need for rapid systems development, IS plans are established collaboratively by management and senior analysts/developers based on user input (e.g., surveys) instead of by a time-consuming strategic planning process. IS plans and proposed schedules are provided to development teams and the user community. The plans are updated based on feedback from the users and as understanding of user needs increases with development and fielding of successive information systems increments.

**Requirements Phase.** The purpose of the requirements phase is to guide users’ transition from analysis of business needs to identification of systems requirements. The process begins with top-down decomposition of business activities followed by development of detailed business scenarios and data requirements for those activities. The scenarios and data requirements are subsequently translated into alternative system use cases which are implemented in a prototype. The development and evaluation of system use cases and prototypes increases the specificity of the requirements. This results in higher quality requirements than those provided by more traditional methods.

**Design and Implementation Phases.** CSEM seeks to improve speed and quality of the design and implementation phases by continuing its emphasis on user participation and prototype development and evaluation. The methodology implements a RAD-like approach, with analysts and developers working with the user group and its representatives to refine user interfaces and define required design specifications.
Because of their involvement throughout the development process, the user groups are also ideal participants for the implementation phase's alpha testing. Beta testing generally expands outward to include additional members of the user community at large. Upon completion of the implementation phase, the new system, along with its associated models and requirements/design specifications, serve as inputs to the next iteration of the life cycle. In this manner, the systematic re-use of requirements, designs, and subsystems is specifically incorporated into CSEM, thereby also accruing the benefits of this effective development practices.

2.3.4.4 CSEM Open Issues

Development of CSEM identified a need for additional collaborative tools to support key aspects of the requirements elicitation process. As stated previously, requirements must describe the objects, functions, and states of a system and its environment. Group Data Modeler can be used to capture the object view. Activity Modeler can provide the function view. But neither provides a dynamic picture of the business processes that a system must support, i.e., the state view. Recent research has suggested that scenarios may be an effective mechanism for eliciting behavioral requirements. Although business scenarios and system use cases (interaction scenarios) have been included in CSEM's requirements phase, further research is needed to on how to fully integrate scenarios into CSEM and how those scenarios should be elicited from user groups. The next section includes an extended discussion of the scenario literature which will be used as the basis for resolving these questions.
2.4 Scenarios

Interest in scenarios has grown dramatically as researchers and practitioners try to deal with the software and requirements engineering problems discussed in the previous sections. This popularity has proven to be both a strength and a weakness for scenario-based approaches. Their strength rests in that fact that everyone seems to agree on the current importance and future potential of scenarios (Jarke et al., 1999). However, this diversity of use also leads to their weakness – scenarios mean very different things to different people, even within the same discipline. It is very difficult to integrate the scenario research if researchers cannot even agree on a common definition for a scenario.

The purpose of the next section is twofold: (1) to provide a brief interdisciplinary view of the history and scope of scenario usage, and then, (2) to focus on scenario usage in software engineering to provide some common definitions that will be used through the remainder of the discussion. Section 2.4.2 will identify some of the key characteristics of scenarios which have driven their popularity. Section 2.4.3 will provide an overview of scenario research in the human-computer interaction (HCI) and software engineering (SE) communities. Section 2.4.4 continues the scenario research survey. It introduces a scenario classification framework proposed by researchers at the European Cooperative Requirements Engineering with Scenarios (CREWS) project and uses that framework to point out key aspects of scenario research within the HCI and SE communities. Section 2.4.5 reports results of the CREWS project's survey of scenario usage in practice. The final section concludes the scenario literature review by
identifying some of the many open issues which have been identified based on these surveys of research and practice.

2.4.1 Introduction

In their broadest sense, scenarios can be considered as descriptions of a set of events (Jarke et al., 1999). More specifically, from a software engineering perspective, scenarios are viewed as concrete descriptions of the sequence of activities that a user engages in when performing a specific task (Carroll, 1995). Examples, scenes, narrative descriptions of contexts, and prototypes have also been loosely characterized as scenario-based approaches (Rolland et al., 1998a). During the last decade, scenario-based approaches have been increasingly used in both the human-computer interaction and software engineering communities throughout all phases of development (Carroll, 1995) including requirements engineering (Benner et al., 1993; Hsia et al., 1994; Leite et al., 1997; Potts et al., 1994; Weidenhaupt et al., 1998). A brief history of scenario usage, however, shows a much longer and richer history.

The scenario concept originated in the theater, which can be clearly seen from most dictionary definitions of the term. Research in scenarios can be traced back to its origins in military and strategic gaming. Since that time, economists have used scenarios for long range planning, management scientists for strategic planning and decision making, and policy makers for assessing policy impacts. The HCI community became interested in scenarios to support interface specification during the late 1980s, but it was the mid-1990s before the software engineering community began to seriously look at scenarios (Jarke et al., 1999). Recently, scenarios have begun to emerge as a tool for
managing change in several disciplines including management science, HCI, and software engineering (Jarke & Kurki-Suonio, 1998). Because of the diversity of scenario history and use, there is no unified scenario research framework or community (Jarke et al., 1999; Jarke & Kurki-Suonio, 1998). Initiatives such as the European CREWS project and a 1998 workshop held at Dagstuhl Castle, Germany, have begun to address this issue by bringing together researchers from the different disciplines. Results of the 1998 workshop are reported in (Jarke et al., 1999). One of the most interesting results was development of the following interdisciplinary definition of a scenario:

> A scenario is a description of the world, in a context and for a purpose, focusing on task interaction. It is intended as a means of communication among stakeholders, and to constrain requirements engineering from one or more viewpoints (usually not complete, not consistent, and not formal). (Jarke et al., 1999)

From the workshop participants’ perspective, the main purpose of developing these sorts of scenarios is “to stimulate thinking about possible occurrences, assumptions relating to those occurrences, possible opportunities and risks, and courses of action” (Jarke et al., 1999). Other results of the workshop will be addressed as applicable during the following discussion.

Given this basic understanding of the interdisciplinary history of scenarios, the goal of the remainder of the scenario literature review is to focus on scenario research and practice in human-computer interaction and software engineering. These fields have also experienced wide variations in scenario use and terminology. An introduction to the three most commonly used types of scenarios is presented next to reduce some of the confusion caused by these variations (Rolland et al., 1998a).
The software engineering community is probably most familiar with the interaction scenario, more often referred to as a "use case" (Jacobson, 1995; Jacobson, Christerson, Jonsson, & Overgaard, 1992; Jacobson, Ericsson, & Jacobson, 1995). As graphically shown in Figure 3, this type of scenario focuses on the interaction of a user and the system necessary to accomplish a specific system-supported transaction (Kuutti, 1995). Software engineers also analyze internal scenarios that define the sequence of activities that the computer system performs to execute a specific system task.

Looking at a broader perspective, business scenarios are designed to capture a larger organizational context by describing all activities the user engages in to accomplish a business task (not just interactions with the system), as well as information about the user's business goals, resources required, and the social setting (Kuutti, 1995). These business scenarios have been alternatively referred to as "rich" (Kuutti, 1995).
"contextual" (Pohl & Haumer, 1997), or "environmental" (Jarke et al., 1999) scenarios. Business scenarios are especially useful during requirements elicitation when user groups seek to develop common definitions of business processes, assess opportunities for process improvement, and evaluate the business impacts of alternative system solutions.

2.4.2 Why Scenarios?

Researchers have identified several reasons why scenario-based approaches are so popular. Many stem from the characteristics of the scenarios themselves. These characteristics are highlighted in the comparison between the scenario and traditional perspectives to systems development summarized in Table 1.

<table>
<thead>
<tr>
<th>the scenario perspective</th>
<th>the establishment view</th>
</tr>
</thead>
<tbody>
<tr>
<td>concrete descriptions</td>
<td>abstract descriptions</td>
</tr>
<tr>
<td>focus on particular instances</td>
<td>focus on generic types</td>
</tr>
<tr>
<td>work driven</td>
<td>technology driven</td>
</tr>
<tr>
<td>open-ended, fragmentary</td>
<td>complete, exhaustive</td>
</tr>
<tr>
<td>informal, rough, colloquial</td>
<td>formal, rigorous</td>
</tr>
<tr>
<td>envisioned outcomes</td>
<td>specified outcomes</td>
</tr>
</tbody>
</table>

Table 1 — Comparison of scenario and traditional perspectives (Carroll, 1995, p. 5)

A variation of this list, presented in (Carroll, 1999; Jarke et al., 1999), identifies five key properties of scenarios which have motivated use in HCI. Scenarios: (1) focus designers on the planned system use, (2) are concrete but support flexible design processes, (3) provide multiple task-oriented views which can be used in many ways, (4) support a "middle-level" abstraction of design, and (5) contribute to user-developer communication using a common, work-oriented language.
From a more theoretical perspective, cognitive science provides some insight into why these properties of scenarios make them so popular with users. Carroll focused on three specific cognitive psychology principles to explain this popularity:

- Concrete material is cognitively accessed and interpreted more easily and thoroughly.
- Incomplete material is elaborated with respect to one's own knowledge when it is encountered. This process of elaboration creates more robust and accessible memories.
- Narrative structures appear to be universally understood and employed by people from all cultures. (Jarke et al., 1999)

Benner et al. (1993) make a similar observation from cognitive science, highlighting that people respond better to specific situations which allow them to use their natural situated problem-solving techniques (i.e., responding to the details of the situation versus carrying out more abstract plans). Kyng (1995) emphasizes that users are experts at working in specific contexts, not in evaluating abstract specifications. From a practical perspective, the CREWS survey of industrial practice found that scenarios can help to reduce complexity. (Weidenhaupt et al., 1998).

Scenario popularity can also be attributed to the many valuable uses of scenarios. First, scenarios can be considered as “middle-level abstractions” between models and the real-world because they are more concrete than the abstract models while at the same time more abstract (by eliminating unnecessary detail) than the real world (Carroll, 1995; Jarke et al., 1999; Jarke & Kurki-Suonio, 1998). These “down-to-earth middle-level abstractions” can be used to “promote shared understanding of the current system, and joint creativity towards the future” (Jarke et al., 1999). This use is directly applicable to many activities in requirements engineering, HCI, and software development. More
generally, scenarios can be used to assist virtually any model-based change management approach. Jarke and Pohl (1994) describe a four steps in their change management approach: (1) reverse engineer the existing system to create an initial model, (2) define the desired change to create a new model, (3) implement the change in a new system, and (4) integrate the legacy system into the new system. Current scenarios can serve as a middle-level abstraction between the existing system and the initial model and aid in step 1. Similarly, future scenarios can be used between the new model and the new system to assist in step 3. And finally, envisioning the change directly from the current scenario to the future scenario can improve step 2’s change specification process. The resulting scenario-based change management process provides a framework for many scenario techniques (Jarke et al., 1999). The next section provides an overview of HCI and software engineering scenario-based approaches in research.

2.4.3 Scenarios in Software Engineering

Scenarios can be used during all phases of the systems development life cycle to support a wide range of specific requirements engineering, HCI, design, implementation, and evaluation activities as well as aiding general activities like user-developer communication and team-building (Carroll, 1995). The following discussion provides examples of scenario research in these areas.

Some of the most active research on the use of scenarios to support requirements engineering is sponsored as part of the CREWS project. An overview of three of the main streams of CREWS scenario research follows.
Researchers at RWTH Aachen (Aachen University of Technology) are exploring methods and tools for using real world scenes (RWS) captured in multimedia to capture and analyze the current system usage. They propose that these RWS can be used to identify high-level goals of the current system which drive development of goals for the new system (Haumer, Pohl, & Weidenhaupt, 1998a; Haumer, Pohl, & Weidenhaupt, 1998b). Using RWS to support requirements traceability is discussed in (Haumer, Pohl, Weidenhaupt, & Jarke, 1999b). In (Haumer, Heymans, Jarke, & Pohl, 1999a), they integrate the earlier research into an environment which support RWS capture, agent-oriented modeling, interactive animation using message trace diagrams, goal modeling, and a prototype tool to support traceability.

CREWS researchers at the University of Paris (Sorbonne) are also looking at goal modeling, but are addressing the problem from a linguistics-oriented perspective. They have proposed a structured approach for using scenarios defined using specific authoring guidelines to discover goals (Rolland, Souveyet, & Ben Achour, 1998b; Si-Said & Rolland, 1998). Those authoring guidelines have evolved over time from general use-case authoring guidance (Nurcan, Grosz, & Souveyet, 1998; Rolland & Ben Achour, 1998) to much more specific scenario writing and management guidelines (Ben Achour, 1998a; Ben Achour, 1998b; Ben Achour, Rolland, & Souveyet, 1998) with tools to support automated natural language processing (Ben Achour, 1997).

City University of London researchers are developing the CREWS-SAVRE (Scenarios for Acquisition and Validation of Requirements) method and software assistant tool (Sutcliffe et al., 1998). The tool has been designed to support semi-
automatic generation of interaction scenario instances based on input from the
requirements engineer, general domain models, and detailed specifications of the types
of exceptions which occur in information systems (Maiden, Minocha, Manning, & Ryan,
1997a; Maiden, Minocha, Ryan, Hutching, & Manning, 1997b; Maiden et al., 1998).
Sutcliffe has led the effort focusing on integrating the tool into comprehensive scenario-
based requirements and socio-technical analysis methods (Sutcliffe, 1998; Sutcliffe et
al., 1998; Sutcliffe & Minocha, 1998; Sutcliffe & Minocha, 1999).

Scenario research is not limited to requirements engineering and the CREWS
projects. For example, researchers at Virginia Polytechnic Institute have an active
research program which began with an HCI emphasis, expanded to design, and is now
moving to the requirements phase as well. They have defined a scenario-based
framework, called the task-artifact framework, to guide participatory design. Teams
consisting of users, developers, and HCI specialists generate scenarios, analyze claims
(i.e., causal relationships), and envision system features and future work scenarios. The
goal of this process is to ensure that design is grounded in the work tasks and that design
rationale is captured (Carroll, 1994; Carroll & Rosson, 1992; Carroll, Rosson, Chin, &
Koenemann, 1998; Chin & Rosson, 1998; Chin, Rosson, & Carroll, 1997). In (Rosson &
Carroll, 1995), Rosson and Carroll focus on the issues involved in the specification of
user tasks and describe a prototype tool developed to capture those specifications and
related design rationale. Adapting this work to the OO world, Rosson (1999) proposes a
specific approach for merging OO analysis and design by “interleaving analysis of user
and object interaction scenarios” (p. 49).
Scenario research in the OO community has its primary foundation in Jacobson's work on use cases (Jacobson, 1995; Jacobson et al., 1992; Jacobson et al., 1995). Use cases (i.e., interaction scenarios) focus on the interaction between actors and the system required to achieve some desired goal and are generally represented using a high-level diagram showing this interaction. Rebecca Wirfs-Brock's work on responsibility-centered design was an early example of scenario usage in the OO community (Wirfs-Brock, 1995). The use case concept has been adopted by most OO methodologies and has been included as part of the standard object modeling language, UML (Booch, Rumbaugh, & Jacobson, 1999; Fowler & Scott, 1997; Rational, 1997).

Other potential uses of scenarios can be seen in case studies such as Karat's (1995) discussion of development of a speech recognition system and Johnson's (1995) report on requirements gathering with medical professionals. Nielsen (1995) describes how scenarios can be used to support usability engineering. Other scenario research initiatives are discussed in the next section to illustrate specific aspects of the CREWS scenario classification framework.

2.4.4 Scenario Classification Framework

Scenarios are used in a wide variety of ways and contexts. Rolland et al. (1998a) propose a framework for classifying the different scenario approaches described in the research literature. This framework classifies the product (i.e., the scenario) only. It does not attempt to capture information about the process used to produce that product. The framework classifies scenarios based on four primary views: purpose, lifecycle, content, and form. Specific facets are described for each view to provide a more detailed
understanding of the breadth of research into scenario-based approaches. Examples of specific approaches are used to illustrate specific views and facets.

**Purpose.** The purpose view defines why developers are using scenarios. As described in the previous section, scenarios may be used to support different phases of software development, e.g., requirements elicitation or user interface design. More generally, the purpose is defined as descriptive, exploratory, or explanatory. Scenario approaches designed for capturing requirements, such as the Inquiry-based Requirements Analysis (Potts et al., 1994), are classified as descriptive. Exploratory scenarios are useful for exploring alternatives. For example, Bodker (1999) describes how scenarios can be used to explore alternative product designs. Explanatory scenarios are useful for capturing explanations of issues such as design rationale (MacLean & McKerlie, 1995). The goal of scenario usage in CSEM is to use descriptive, exploratory, and explanatory scenarios as appropriate throughout the development process.

**Lifecycle.** The lifecycle view describes how scenarios change and evolve during the systems development process. A scenario's lifecycle may be limited to static scenario definition or focus on evolution of scenarios throughout the project. The lifecycle view captures both the life-span of the scenario as well as the operations performed on it. A scenario's life-span can be temporary (i.e., transient) or persist for the life of the project. Scenarios used to validate a requirements specification or evaluate a system may be transient and not last beyond the specific validation or evaluation task. In contrast, scenarios which are included as part of the requirements specification persist through the life of the project. CSEM scenarios are designed to be persistent.
Operations which can be performed on scenarios include the initial capture (i.e., generation) of scenarios, refinement to improve readability or support reuse, integration with other scenarios, expansion to include additional information, or permanent deletion. The initial focus of CSEM scenario research is on the capture operation, but will eventually need to support all these operations in addition to between scenario operations like collection of scenarios representing the same concept but differing in viewpoint, elaboration into different scenarios, disjunction into alternative courses of action, and composition of multiple scenarios into a combined scenario (Zorman, 1995).

Content. The content view defines the type of knowledge included in a scenario. As discussed in section 2.4.1, scenario content may focus on the user-system interaction, internal system operations, or describe the broad organizational context. The context facet describes four types of contextual information: organization environment, organization context, system interaction, system internal. CSEM business scenarios focus on capturing the first two types of information. Use cases describe the system interaction, while internal scenarios include the system internal type of information.

The abstraction facet captures whether scenarios represent a concrete instance, an abstract type, or a mixture of the two. Regnell et al. (1996) refer to this as the granularity of the scenario and recommend additional research to determine appropriate levels of granularity.

The argumentation and coverage facets define additional information that may be included in a scenario description. Data models provide this same sort of information. Several researchers have proposed data models for scenarios (e.g., Leite et al., 1997;
Regnell et al., 1996). Pohl and Haumer (1997) analyzed existing scenario approaches and identified eleven approaches which addressed contextual scenarios. They then compared the data collected by each approach and used the results of the analysis to create scenario and interaction data models. Information from this analysis and several of the source approaches were used to develop the initial data model for CSEM business scenarios.

**Form.** The form view describes the format used to represent the scenario. This view has probably received the most attention in the scenario literature. The description facet of the form view defines the medium (text, graphics, image, video, software prototype) and the notations (formal, semi-formal, informal) of the scenario representation. The presentation facet describes if the representation is animated and its level of interactivity (none, hypertext-like, advanced).

Formats of scenarios-based approaches in the research literature vary widely. Textual formats are the most common, ranging from unstructured natural language (e.g., Carroll, 1995) to structured natural language such as tabular scripts (Potts et al., 1994) to formal grammars (Leite et al., 1997) and relation-based specifications (Desharnais, Frappier, Khedri, & Mili, 1998). Various modeling techniques such as scenario trees (Hsia et al., 1994), use case diagrams (Jacobson et al., 1992), message sequence charts (Regnell et al., 1996), interaction diagrams (Beringer, 1997; Koskimies, Systa, & Tuomi, 1998), Petri nets (Lee, Cha, & Kwon, 1998), other graphical representations (e.g., Benner et al., 1993; Zorman, 1995), or even multimedia (Haumer et al., 1998b) may be
used instead of or in addition to textual descriptions. CSEM business scenarios will be represented using a combination of structured text and a graphical process model.

This same classification framework was used to structure the survey of scenario usage in industry described in the next section.

2.4.5 Scenarios in Practice

CREWS project team members conducted a survey of 15 European projects using scenario-based approaches (Jarke et al., 1997; Weidenhaupt et al., 1998). The CREWS scenario classification framework (Rolland et al., 1998a) was used to structure the survey and guide interviews (Weidenhaupt et al., 1998). Researchers concluded that:

Scenario-based approaches are becoming ubiquitous in systems analysis and design but remain vague in definition and in scope. A survey of current practices indicates we must offer better means for structuring, managing, and developing their use in diverse contexts. (Weidenhaupt et al., 1998, p. 34)

Analyzing the results using the scenario classification framework, Weidenhaupt et al. (1998) observed the following:

- **Purpose.** In addition to the expected use for requirements elicitation and validation, scenarios were used to make requirements more concrete and reduce complexity. They were also useful in helping to achieve at least partial consensus, facilitating interdisciplinary development. Two-thirds of the projects used scenarios in conjunction with prototypes as part of a rapid development approach.

- **Lifecycle.** Lifecycle and other management issues included the need to support partial views, scenario evolution, and distributed scenario development.
Scenarios evolved over time on all projects using one or more of the following types of evolution: top-down decomposition, from black-box to white-box scenarios, from informal to formal scenario definitions and through incremental development. Most projects also used the original scenarios as the basis for system test cases.

- **Content.** All three types of scenarios were used. Approximately half the projects used contextual and/or internal scenarios, while virtually all used the interaction scenario.

- **Form.** Free-format and structured text (e.g., a template or a table) were the most commonly used representations, especially for contextual and interaction scenarios. Images (e.g., screen dumps or layouts) and interaction diagrams were commonly used for internal scenarios.

Lack of process guidance and tool support was a critical problem. All scenario users had to significantly extend existing scenario/use case guidance to effectively incorporate scenarios into their development process. Most only used a word processor to support the scenario process. Even with this problems, scenarios were identified as effective bridges: to business processes, between users and developers, between architectural and implementation components, between developers, between development phases, and between structure and behavior (Weidenhaupt et al., 1998, p. 43-44). The need for this last bridge was what drove inclusion of scenarios in CSEM, although they are also used to provide many of the other bridges as well.
Comparing the results of the industry to the state of scenario research, CREWS researchers (Jarke, 1999; Jarke et al., 1999) reached several important conclusions. The purposes for which scenarios were used were much broader and the lifecycles more complex than the research would suggest. Practitioners did consider all three types of scenarios addressed in the literature but rarely used the more formal representations that are the focus of much of the research. At best, they used structured templates or tables.

Results from the industry survey and the comparison of research and practice highlighted several areas for future research. These issues are discussed next.

2.4.6 Scenario Issues

Participants at the Dagstuhl workshop summarized the main research issue as “how to get the best value for the money invested in scenario-based techniques” (Jarke et al., 1999). They also discussed four more specific issues at the conference: systematic capture and generation of scenarios, individual scenario representation issues, how to incorporate scenarios into existing requirements and software development methods, and how to manage scenarios through evolution and specification (Jarke et al., 1999).

Conclusions from the industry survey were similar. The need for additional research on relationships with other techniques and scenario evolution and management directly mapped to two of the workshop issues. Survey results also identified a need for identification of traceability methods, increased process guidance, and improved tool support (Weidenhaupt et al., 1998). The need for better process guidance is a recurring theme in the literature as well (Mack, 1995; Regnell et al., 1996; Rolland et al., 1998a). Nardi (1995) reemphasizes the need for effective scenario management and processes
which support reusability. Mack (1995) highlights the need for procedures for establishing the completeness of scenarios as design specifications, and, more importantly, for specific guidance on how to elicit scenarios from users in a team-based environment. Scenario usage in CSEM is facing many of the same issues, especially the need for collaborative process guidance.

Overall, the findings and observations from the scenario literature review and industrial survey were simultaneously encouraging and discouraging. They were encouraging in that they strongly support the inclusion of scenarios as a key component of the CSEM requirements elicitation and analysis process. However, they were discouraging in that they indicate that neither industry nor the research literature provide explicit guidance on how best to elicit scenarios from groups of users. Although there are many good ideas in the literature, the scenario literature differs from CSEM needs in two main ways. First, the majority of scenario-based approaches focus on supporting individual systems analysts, requirements engineers, HCI specialists, or developers. Few focus on collaborative processes or providing direct support for scenario definition and analysis by users. As a result, these approaches simply do not provide the simplified user-comprehensible interface required by non-analyst users which is currently available for general-purpose and modeling GSS. Second, the scenario-based approaches generally focus on the requirements analysis, specification, or design phases. However, scenario usage in CSEM begins during requirements elicitation with a focus on the business preceding the traditional system-focused requirements analysis phase.
Therefore, recommendations from the literature must be carefully evaluated for the applicability to the CSEM requirements elicitation phase.

2.5 Summary of Literature Review

The discussion of information systems development and requirements engineering clearly showed that an effective, user-focused, collaborative development process is essential to MIS success. Discussions of general-purpose and modeling GSS highlighted their potential for supporting such a process. However, development of CSEM, a methodology designed to provide that capability, identified a need for new methods and tools to capture behavioral (i.e., state) requirements. The review of the scenario research and practice showed that scenarios could satisfy that need. Scenarios are designed to account for the problem context and, by their very nature, can accommodate incompleteness - two of the key characteristics hypothesized to be essential for emerging requirements engineering methods (Siddiqi & Shekaran, 1996). When incorporated into CSEM, scenarios also provide support for the last criteria – recognition of the evolutionary nature of requirements. However, there are many unanswered questions relating to scenarios in general. Even more arise when trying to use scenarios for collaborative requirements elicitation from user groups. These open issues drove development of the primary research question for this dissertation:

What are the collaborative modeling processes, tools, and facilitation techniques needed to effectively elicit scenarios from users in a group environment?

The research approach used to address this question is described in the next chapter.
CHAPTER 3 – RESEARCH APPROACH

3.1 Overview – Multi-Methodological/Multi-Phase Approach

Selection of an appropriate research methodology can make the difference between success and failure of a research project. To address this problem, IS researchers have developed research classification frameworks, identified appropriate strategies for IS research, and debated the strengths and weaknesses of various approaches (e.g., Benbasat, 1984; Galliers & Land, 1987; Ives, Hamilton, & Davis, 1980; Nolan & Wetherbe, 1980). Similar discussions are taking place in the field of GSS research. The emerging consensus is that multi-methodological approaches provide the greatest insight into the complex social and technical issues involved in GSS research (Vogel & Nunamaker, 1990; Zigurs, 1993). Differences between GSS field and experimental research results (e.g., Dennis, Nunamaker, & Vogel, 1990-1991) emphasize the importance of using multiple methods to achieve a balanced research perspective. The tradition for GSS research at the University of Arizona is this multi-methodological approach with systems development as its core (Nunamaker et al., 1996-97). Although systems development is not a universally accepted IS research approach, Nunamaker (1992) argues convincingly that IS (and GSS) researchers must build systems to develop new theories. The IS field simply does not yet have the accumulated experience required to develop overarching theories from which new theories can be synthesized. Rather, systems must be built to serve as proofs-of-concept for new theory. As can be seen from the literature review in Chapter 2, this is certainly the case for
research in collaborative scenario elicitation. Therefore, a multi-methodological systems development research approach incorporating theory building, systems development, experimentation, and observation (Figure 4) was used to address the research questions identified in Chapter 1.

![Figure 4 - Multi-methodological research approach](Adapted from (Nunamaker et al., 1991a))

Nunamaker et al. (1991a) identify five criteria for the use of systems development as a research methodology:

1. the purpose is to study an important phenomenon in areas of information systems through system building,
2. the results make significant contribution to the domain,
3. the system is testable against all stated objectives and requirements,
4. the new system can provide better solutions to IS problems than existing systems, and
(5) experience and design expertise gained from building the system can be generalized for future use. (p. 101)

All five criteria were fully met, either by the nature of the research problem or through the design of the research. The importance of this research to the field of information systems and the need for systems to support collaborative scenario elicitation are clearly presented in Chapter 1. The significant contributions made by this research will be summarized in the final chapter. The systems development research design, described in section 3.6, requires that criteria three and five be satisfied. To ensure that the fourth criteria was satisfied and that the new system did provide better support for collaborative scenario elicitation than existing systems, a two-phase research approach was used.

The purpose of the first research phase was to determine if an existing GSS, GroupSystems, could be used to support collaborative scenario elicitation. The problems and limitations of using GroupSystems identified during phase one served as justification for development of a new system. The purpose of the second research phase was to develop and evaluate a new collaborative tool which provided improved support for collaborative scenario elicitation by overcoming the shortcomings of existing tools.

Research during each phase was conducted using the methods shown in Figure 4 in an iterative fashion with results from individual research tasks driving the research conducted in the next task. A chronological summary of the specific research tasks in each phase is shown in Table 2. Detailed research designs for each research method and phase are described in sections 3.3 – 3.8. Research results are presented in Chapters 4 and 5, with a discussion of results following in Chapter 6.
### Research Phase/Tasks

<table>
<thead>
<tr>
<th>Phase I: Collaborative Scenario Elicitation Using GroupSystems</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Theory Building: Develop a framework for scenario usage in CSEM and propose initial collaborative scenario elicitation methodology.</td>
<td>1995-96</td>
</tr>
<tr>
<td>• Observation: Perform case study of DOD group developing unstructured scenarios using GroupSystems.</td>
<td>Fall, 1996</td>
</tr>
<tr>
<td>• Experiment: Conduct laboratory experiment to evaluate impact of different textual scenario formats using GroupSystems.</td>
<td>Fall, 1997</td>
</tr>
<tr>
<td>• Observation: Perform case study of DOD groups defining scenarios using structured templates with GroupSystems.</td>
<td>1997-98</td>
</tr>
<tr>
<td>• Theory Building: Update collaborative scenario elicitation methodology for use with GroupSystems based on lessons learned.</td>
<td>1998</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Phase II: Integrated Scenario &amp; Process Modeling Using SPA</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Systems Development: Design and develop the Collaborative Distributed Scenario and Process Analyzer (ColD SPA) prototype.</td>
<td>1998-99</td>
</tr>
<tr>
<td>• Observation: Perform case studies of DOD groups using SPA.</td>
<td>Fall, 1998</td>
</tr>
<tr>
<td>• Theory Building: Revise collaborative scenario elicitation methodology for use with SPA based on lessons learned.</td>
<td>1999</td>
</tr>
</tbody>
</table>

Table 2 — A chronological overview of research tasks

One potential problem with using multiple research methods is that it can significantly complicate comparison of research results and limit an integrated assessment of the entire research effort. Although this problem cannot be totally avoided, use of a common underlying research model for all research tasks can limit its impact. The theoretical GSS research model (Figure 5) proposed in (Dennis et al., 1988) has guided GSS research throughout the years (Nunamaker et al., 1991c). It was used during this research as the underlying model required to ensure comparability of results between research methods and phases. An overview of this model is presented in the next section.
3.2 GSS Research Model

Dennis et al. (1988) integrated concepts from several early GSS researchers into a high-level theoretical model that identifies some of the factors that impact the outcome of a GSS meeting. A simplified version of this model is shown in Figure 5. The model emphasizes that characteristics of the group, task, context, and GSS technology jointly influence the meeting process, which in turn influences the outcome of a GSS meeting (Dennis et al., 1988).

![Figure 5 - Simplified GSS research model](image)

Specific characteristics of these factors which have been identified in the literature (Dennis et al., 1988; Nunamaker et al., 1991c; Nunamaker et al., 1993) are listed below.

- **Group.** Characteristics of both the group and individual group members should be considered. Group characteristics include group size, proximity (i.e., same place or distributed), history (e.g., one-time vs. ongoing), group charter (formal vs. informal group), experience, and cohesiveness.
• **Task.** The nature of the task has a critical influence on the meeting process and outcome. Research has shown that while GSS greatly improves efficiency for some task types, it is much less effective for others (Dennis & Gallupe, 1993). Some key characteristics of the task include its type, complexity, activities required to complete the task, and whether it’s rational or political.

• **Context.** Organizational culture and environment, time pressure, and group and individual incentive and reward systems are some of the context characteristics which should be considered when conducting and evaluating GSS research.

• **GSS.** Characteristics for this factor include whether or not a GSS was used, the design and specific features (e.g., anonymity) of the GSS used, GSS facility design, and level of process and task support provided by the GSS.

• **Process.** Some of the characteristics of the process, which should be closely monitored during GSS meetings, include the degree of process and task structure, number of sessions, types and levels of leadership and participation, and amount of conflict and non-task behavior.

• **Outcome.** Many outcome measures have been used in GSS research. Dennis et al. (1988) identified: “decision/outcome quality, participant satisfaction with the outcomes and the process, participant confidence in the outcomes, process time required, level of group consensus, number of comments during the meeting, and the number of alternatives or issues considered” (p. 596-7). Generally, these outcomes have been summarized as (1) effectiveness measures of the group’s actual performance, (2) efficiency measures of the
relationships between the inputs and outputs, and (3) participant satisfaction with the both the process and outcomes (Nunamaker, Dennis, Valacich, & Vogel, 1991b; Nunamaker et al., 1991c).

This model has been repeatedly used for GSS research and is critical for the accurate evaluation of research results because GSS experiences have clearly shown that:

It is inappropriate to say that GSS use “improves group task performance” or “reduces member satisfaction”; all statements must be qualified by the situation – the group, task, context, and GSS to which they apply. (Nunamaker et al., 1993, p. 127)

To avoid this pitfall, the research design, results, and discussion presented in this dissertation will use this GSS research model and will include descriptions of the group, task, context, and GSS. Detailed research designs with appropriate identification of this information are presented next for each research task and phase.

3.3 Phase I Theory Building

Theory building may include “development of new ideas and concepts, and construction of conceptual frameworks, new methods or models (e.g., mathematical models, simulation models, and data models)” (Nunamaker, 1992, p. 3). The first step in theory building for this research was to develop an initial conceptual framework for scenario usage in CSEM that responded to the research question:

How will scenarios be used during all CSEM development phases and how are the scenarios developed for these different purposes related?

Results of the literature review presented in Chapter 2 as well as prior GSS experiences at the University of Arizona and personal experiences as a system analyst were used to develop this initial framework.
The next step in theory building was to propose a method for collaborative elicitation of scenarios using an existing GSS, GroupSystems. The goal was to develop a preliminary GroupSystems response to the question:

What group process and facilitation techniques are appropriate for each collaborative tool used to elicit contextual scenarios from user groups?

A data model was also developed as part of the methodology to respond to the question:

What information should be included in the contextual scenarios elicited directly from user groups?

Information from the literature review and prior GSS experiences were used to define the preliminary collaborative scenario elicitation method and contextual scenario data model. Results from these first two steps in theory building are presented in Chapter 4.

The proposed framework and initial collaborative scenario elicitation methodology were used to facilitate the early scenario sessions with real-world user groups. Case studies of these groups and results from the related Phase I experiment significantly increased understanding of the issues involved in collaborative scenario elicitation. Lessons learned from the case studies and experiment were used to refine the proposed GroupSystems scenario elicitation methodology and to enhance the initial CSEM scenario framework by adding information required to respond to the question:

How will scenarios change and evolve during development and how can that change be effectively supported?

Refinements of the framework and methodology made based on Phase I research results are discussed in Chapter 6.
3.4 Phase I Observation: Exploratory Case Study Design

The goals of the Phase I exploratory case studies were: (1) to increase understanding of collaborative scenario elicitation using GroupSystems, (2) to evaluate the proposed methodology, and (3) to help define questions and directions for follow-on research. A case study research design was chosen to provide a structured process for accomplishing these goals. The research design for the Phase I case studies is described in the following sections. In the first section, an overview of the design is presented which describes why a case study approach was selected and summarizes the main characteristics of the case study. The detailed description of the research design has been structured based on Yin's five primary components of a research design:

1. The study's questions
2. Its propositions
3. Its unit(s) of analysis
4. The logic linking the data to the propositions
5. The criteria for interpreting the findings (Yin, 1994, p. 20)

An analysis of the appropriate design for each of these components is presented in subsections 3.4.2 through 3.4.6.

3.4.1 Overview

The selection of the case study research methodology was based on Yin's comparison of methodologies. Yin identifies three conditions to guide the selection of the appropriate research methodology: (1) the type of the research question, (2) the degree of experimental control possible, and (3) the focus on contemporary vs. historical events (Yin, 1994). Since this research focuses on how to elicit scenarios from user groups with no or limited behavioral control of contemporary events in real-life
contexts, Yin identifies the case study as the most appropriate research strategy (Yin, 1994, p. 6).

The next design issue was the classification of the case study as exploratory, descriptive, or explanatory. In this case, an exploratory case study best supported the study's goal to explore and evaluate a methodology for collaborative scenario elicitation. A descriptive design is inappropriate because, although it provides descriptions of what is happening, it does not respond to the question of what methods are most effective for scenario elicitation. An explanatory design is also inappropriate because not enough is known at this point to begin developing or testing explanations of the scenario elicitation meeting results.

The final issue involved the selection of the type of case study design. Because the goal of the research is to develop a general-purpose collaborative scenario elicitation methodology, a multiple case study design was chosen. Multiple cases supported evaluation of successive versions of the methodology and collaborative tools used by different groups working on various tasks. They also provided some support for analytical generalization, the primary process used by case studies to ensure external validity (see section 3.4.6). As discussed in more detail in section 3.4.4, the case study also included multiple units of analysis. The detailed components of this exploratory, multiple-case, embedded case study research design are described in the following sections.
3.4.2 Study Questions

Clear definition of the study’s research questions “is probably the most important step to be taken in the research design” (Yin, 1994, p. 7). These questions are essential not only to the selection of the case study research design, but also to the detailed design of the case study. Many of the problems with case study design stem from the inaccurate or unclear specification of these questions. As previously discussed in Chapter 1, the study’s primary research question is:

What are the collaborative modeling processes, tools, and facilitation techniques needed to effectively elicit scenarios from users in a group environment?

The primary purpose of the Phase I case studies was to gather the information needed to refine the preliminary responses to the following detailed questions developed during the initial Phase I theory building.

What group process and facilitation techniques are appropriate for each collaborative tool used to elicit contextual scenarios from user groups?

What information should be included in the contextual scenarios elicited directly from user groups?

The case studies were also used to increase understanding of collaborative scenario elicitation so that initial responses to the following questions could be developed.

What format should be used for contextual scenarios and how does that format impact scenario quality and productivity of user definition of scenarios?

Can general-purpose GSS tools be used for collaborative scenario elicitation or is a special-purpose tool needed?
3.4.3 Propositions

The second component of the case study research design is the propositions. "Each proposition directs attention to something that should be examined within the scope of the study" (Yin, 1994, p. 21). Propositions can highlight critical theoretical issues and can also focus the search for supporting evidence. Although not required for exploratory case studies such as this one, they give additional, more detailed purpose and direction for the study. Yin therefore recommends that propositions be stated whenever possible, even for exploratory studies (Yin, 1994). The guiding proposition for this case study was:

**Proposition.** Use of the 'right' combination of group process, facilitation techniques, and collaborative tools will increase the efficiency and effectiveness of scenario elicitation by user groups.

During the early exploratory case studies, this proposition principally served to provide purpose and scope for the study. However, in the future, as the research transitions from exploratory to explanatory, both theoretical and empirical evidence will be sought to support this proposition.

This proposition, in combination with the study questions, was used to identify the units of analysis for this case study as described in the following section.

3.4.4 Units of Analysis

There were several different options for the appropriate unit of analysis for this case study, each with its own advantages and disadvantages. Potential units of analysis included: the complete scenario elicitation process for a system which may consist of multiple activities both during and between meetings, a single scenario elicitation
session, the users involved in scenario elicitation, or the facilitators guiding the scenario elicitation process. Three criteria were used to resolve this dilemma:

- Yin (1994) states that the selection of the appropriate unit of analysis should be directly guided by the study's research questions; therefore this was the primary criterion in the selection process.

- The other two criteria were related to time. Yin states that "specific time boundaries are needed to define the beginning and end of the case" (Yin, 1994, p. 24). In addition, the duration of each individual case must be relatively short to allow for the completion of multiple cases during the research project.

A complete scenario elicitation process for a system was eliminated as a possible unit of analysis because it occurred over too long of a time frame. Users or facilitators were also eliminated as the primary unit of analysis because they would give a potential biased perception of the study's questions that must be viewed from multiple perspectives. A single scenario elicitation session was selected as the primary unit of analysis for these case studies because it provided the most complete view of the study's questions within a realistic time frame.

The final question related to the unit of analysis is whether a holistic or embedded case study will be conducted. Although users and facilitators were eliminated as the primary unit of analysis, they each still can provide valuable insight into the study's questions. Therefore, an embedded case study design was selected which considers the perspectives of each of these participants in the process within the context of the scenario elicitation session unit of analysis. The next section discusses various
approaches for collecting data from both the primary and embedded units of analysis and linking that data to the propositions.

3.4.5 Linking Data to Propositions

The fourth component of a case study research design is linking the data to the propositions. However, before identifying these linkages, the types and sources of data must be identified. The basic concept underlying the proposed data collection strategy is to emphasize a multi-method, multi-source approach to increase the amount of information available, to improve construct validity, and to allow for triangulation of findings. The basic types and sources of evidence that were collected during the case studies are summarized in the following table.

<table>
<thead>
<tr>
<th>Type of Evidence</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Participant – Observation</td>
<td>Case Notes</td>
</tr>
<tr>
<td>Interviews</td>
<td>Facilitators</td>
</tr>
<tr>
<td>Survey</td>
<td>Users</td>
</tr>
<tr>
<td>Documentation</td>
<td>Session Output</td>
</tr>
<tr>
<td>Archival Records</td>
<td>Existing Models</td>
</tr>
</tbody>
</table>

Table 3 – Types and sources of evidence

When acting as participant - observers, the author took extensive case notes throughout the scenario elicitation meeting. The GSS research model was used to guide note taking to ensure that key characteristics of the group, task, context, GSS technology, and process were fully documented. Meeting facilitators were interviewed to gather their perceptions of the efficiency and effectiveness of the proposed methodology, recommendations for improvement, and subjective judgments of the improvements to the quality and completeness of the requirements. Users were surveyed following the
Phase II sessions to gather feedback on the process, tool, and facilitation techniques and to request their assessment of the quality of the output and productivity of the process used. The documentation of the session outputs was reviewed to determine the number, quality, and completeness of the new and/or modified models so that they could be compared from session to session. The archival records of the existing models developed prior to the meeting were used to evaluate the changes between the existing model and the session output.

These multiple sources of data provided preliminary support for the main proposition. The improved efficiency and effectiveness of the scenario elicitation and process was evaluated based on the researcher’s participant - observation, interviews with facilitators, and the analysis of the session output. The archival records were used to increase the researcher’s understanding of the problem domain and support the other analyses. In the next section, the criteria for interpreting the results of these analyses are described.

3.4.6 Criteria for Interpreting Findings

The criteria for interpreting the case study findings were based on a direct application of Yin’s mapping of the validity and reliability concepts from empirical research in the social sciences to appropriate case study tactics (Yin, 1994, pp. 32 - 38). In addition to tactics for ensuring reliability, Yin describes tactics for three of the four types of validity described in Cook and Campbell (1979). He does not address the fourth type, statistical conclusion validity, since it does not apply to case studies that depend on
analytical vs. statistical analysis. The tactics that were used as criteria for interpreting the findings from the case studies are briefly described in the following paragraphs.

**Construct Validity** addresses whether the constructs that are measured or observed actually represent the hypothesized theoretical concepts (Cook & Campbell, 1979, p. 59). Yin recommends the use of multiple sources of evidence as one possible way of dealing with this issue (Yin, 1994, p. 34). As can be observed from the discussion in the previous section on the many sources of data that were collected, this tactic was directly implemented in the case study.

**Internal Validity** is concerned with establishing causal relationships and therefore has limited applicability to exploratory research that does not posit causal relationships. However, Yin’s tactic of explanation building to ensure internal validity will be used to inform future research efforts (Yin, 1994, p. 35). Yin’s other tactics relating to internal validity are less applicable to this research study.

**External Validity** deals with the generalizability of findings to other persons, places, or settings (Cook & Campbell, 1979, p. 71). Yin recommends the use of replication through a multiple case study design to ensure external validity based on analytical generalization. The use of multiple cases representing scenario elicitation for different systems by different user groups should improve the external validity of these case studies.

**Reliability.** To demonstrate that the study could be repeated with the same results, Yin recommends and this study implements the use of a documented case study protocol and procedures (Yin, 1994, p. 37).
3.5 Phase I Experimentation: Laboratory Experiment Design

3.5.1 Research Question and Propositions

The laboratory experiment was designed to respond to the need for increased understanding of the impact of scenario format by focusing on the initial individual user definition of contextual scenarios using the three most common textual scenario formats. Its goal was to provide a detailed response to the research question:

What format should be used for contextual scenarios and how does that format impact scenario quality and productivity of user definition of scenarios?

The scenario literature summarized in Chapter 2 showed the wide variety of formats used to describe scenarios. The three most common textual scenario descriptions are: (1) unstructured free-format narrative text, (2) separately numbered steps with free-format descriptions, and (3) more-structured formats such as tables with separate columns for each scenario data item (e.g., actor, action name, description) and numbered rows for each scenario action. However, the literature review also showed that little information is provided on why these formats were chosen or on the impacts of these formats. Because of the lack of research in this area, the analysis was primarily exploratory in nature.

Previous experiences provide some insight into what impacts might be realistically expected. Specifically, increasing the structure of scenarios by dividing the scenario description into numbered steps (as is the case in format two) increases the user's emphasis on separately defining each action in a scenario and therefore increases the quality of the scenario by increasing the probability of including all actions in the
scenario description. Increasing the structure even more by separating the action identification from its descriptive information (as in format three) even further increases the focus on the actions and therefore should further increase scenario quality. This analysis led to the proposition:

Proposition 1: Increasing the degree of scenario structure will increase the quality of the scenario descriptions.

Increasing the focus on individual actions should also make it easier to provide complete descriptive information about those actions. For example, once actions are identified, the user could easily review each of the separately defined actions in formats two and three and add the necessary descriptive information that directly relates to those actions. In contrast, adding this information to an unstructured paragraph (format one) requires a much more extensive process (e.g., searching for each individual action, determining where to add the information into the narration, editing the paragraph to add the information, and possibly rewriting some sentences to maintain readability of the modified section). Therefore, the next proposition was:

Proposition 2: Increasing the degree of scenario structure will increase the quality of the action descriptions.

However, structure may make it harder to include non-sequential activities such as exceptions or alternative courses of action, because it may be harder for the user to determine where to include this information in a strictly sequential listing such as that imposed by the structured formats (formats two and three). Therefore, it was proposed that:
Proposition 3: Increasing the degree of scenario structure will decrease the quality of non-sequential scenario information (e.g., exceptions and alternatives).

Finally, the potential impact of productivity is somewhat clearer. Human-computer interaction studies have shown that the time required to perform a task is a function of the amount of thinking time required for each ‘chunk’ of information plus the time required for each mouse movement and keystroke (Card & Moran, 1995; Card, Moran, & Newell, 1983). Because users will be constrained by and have to think more about the structure as well as the content of scenarios and will be required to perform more keystrokes to enter actions (format two) and descriptions (format three) as separate items, it was proposed that:

Proposition 4: Increasing the degree of scenario structure will decrease the productivity of user definition of scenarios.

The research constructs and specific hypotheses developed to assess the impacts of scenario format and analyze these propositions are described next.

3.5.2 Research Constructs and Hypotheses

The scenario literature is essentially silent on how to measure scenario quality. Since the primary purpose of using scenarios is to aid the requirements definition process, the requirements literature was reviewed to determine how it measures quality. This literature identifies several measures for the quality of a software requirements specification (SRS) (e.g., see summary in Davis et al., 1997). The most commonly mentioned is completeness.

Davis et al. (1993) state that an SRS is complete if “everything that the software is supposed to do is included in the SRS.” Completeness is generally measured as percent
of total requirements included in the SRS ((# requirements in SRS)/(total # requirements)). Comparably, scenario completeness can be measured to assess whether all actions have been included in the scenario as the percent of total actions included in the scenario description ((# actions)/(total # action)). Using this information, proposition 1 can be reformulated to hypothesize that:

**H1:** Increasing the degree of scenario structure will increase the percent of total actions included in the scenario description.

Davis et al. (1993) evaluate the completeness of each individual requirement by measuring whether all information required to describe the requirement is provided. For scenarios, individual action completeness can be assessed in a similar manner. But, in this case, there are two types of descriptive information for actions: (1) information directly related to the action as part of the regular action sequence, and (2) information such as exceptions which may occur for an action or alternative ways of performing an action which are non-sequential in nature. Therefore to assess the first type of action completeness, the completeness for each type of sequential action information (e.g., actor, description) will be calculated and added together to determine the average sequential information per action. Based on proposition 2, it can be hypothesized that:

**H2:** Increasing the degree of scenario structure will increase the average sequential information per action included in the scenario description.

To assess the second type of action completeness, the completeness for each type of non-sequential action information (e.g., exception, alternatives) will be calculated and added together to determine the average non-sequential information per action. It is hypothesized from proposition 3 that:
H3: Increasing the degree of scenario structure will decrease average non-sequential information included in the scenario description.

Productivity is traditionally measured as the quantity produced divided by the time or resources taken to produce that quantity. For scenarios, quantity has generally been measured as the total number of scenarios or the length of scenarios in lines or pages (e.g., Weidenhaupt et al., 1998). Since all subjects will have identical, but limited, time to define just a few scenarios, total length of scenario descriptions was used to measure productivity. (Note: If subjects had worked different lengths of time, these totals would have been divided by each subject’s time.) Based on proposition 4, it can be hypothesized that:

H4: Increasing the degree of scenario structure will decrease the total length of scenario descriptions.

3.5.3 Experimental Design

The four research hypotheses were evaluated using results from a laboratory experiment. During the experiment, a convenience sample of undergraduate MIS students used the same GSS tool (GroupSystems Group Outliner) used in the DOD sessions. Group Outliner also provides the same sort of functionality as word processors, the tool most commonly used for scenario definition in industry (Weidenhaupt et al., 1998). The students defined scenarios for their university’s current course registration system using one of three scenario textual formats. Cross-referencing to the GSS research model: the group was MIS undergraduates; the task was definition of course registration scenarios; the context was a laboratory experiment conducted in a face-to-
face mode with students receiving extra class credit for participating; and the technology was Group Outliner. A detailed process description is provided next.

The experiment lasted approximately two hours and included a short Group Outliner training session, a practice exercise, definition of the course registration scenarios, and completion of a post-session questionnaire. A standard script (see Appendix C.1) was used to ensure that common instructions and time frames were used for each experimental session. Subjects spent 45 minutes defining up to four course registration scenarios, specifically describing the steps students take to: (1) Register for spring semester classes, (2) Add a class after initial registration, (3) Drop a class, and (4) Change section for a class. They were told to concentrate on developing the best possible description for the first scenario and that they did not need to complete all scenarios. If they finished the first scenario, they worked on the other scenarios. All subjects were given a handout (see Appendix C.2 – C.4) which provided (1) a common definition of what information should be included in a scenario description (Figure 6), (2) instructions on how to input the scenario description for their specific treatment format, and (3) a sample scenario using their treatment format.

To develop a scenario description, you must specifically identify each action that must be accomplished. For each action or step in the scenario, you must identify who performs that step (e.g., a person or a system) and what specific action is done. You should also identify any data/information that is needed to complete that step, the reason the step is done, any major error checking or exceptions that may occur, as well as alternative ways of accomplishing that step.

Figure 6 – Scenario experiment instructions
In addition, all subjects, regardless of treatment, where provided with an initial outline listing scenario categories (e.g., UA Course Registration Scenarios) and specific scenarios (e.g., Student Registers for Spring Semester Classes) (see top portions of Figure 7 – Figure 9). The treatments varied based on what format subjects used to input the descriptions for those scenarios. Subjects assigned to the first treatment entered their scenario descriptions as *unstructured* text into Group Outliner’s comment window (Figure 7).

Subjects assigned to treatment 2 listed the scenario actions as *numbered steps* by adding the actions as individual outline items in Group Outliner (Figure 8). Group Outliner automatically generated step numbers.
1 Group Outliner Practice
2 UA Info Scenarios
-2.1 Student Checks Semester Grades using UA info
  -2.1.1 Student Starts Internet Browser
  -2.1.2 System Displays Browser's Main Screen
  -2.1.3 Student Requests UA's Web Site
  -2.1.4 System Displays UA's Web Page
  -2.1.5 Student Selects "Student Information"
  -2.1.6 The system displays the Student Information web page.
  -2.1.7 The student then selects the "Student Link" link, again based on its description.
  -2.1.8 Temporarily, the system displays a web page which asks the student to select Student Link Version 1 or Version 2 which are each briefly described.
  -2.1.9 Assuming the student selects Version 2 since the system says it has a newer, simpler user interface, the system will then display the secure login screen which asks the student to enter his/her Student ID Number and Personal Identification Number (PIN) to ensure that students only access their own information.
  -2.1.10 The student enters the ID and PIN and then clicks the Login button.
  -2.1.11 The system then validates the ID and PIN and requests the student to re-enter them if they are not valid. If the student does not enter a valid ID and PIN or encounters any error in the previous steps that he/she cannot fix, the student cannot look-up grades using UA info's web access from that computer. The student can try another computer or use an alternative method for checking grades (e.g., RSVP, check with professor, wait for grades in mail, or check at Registrar's Office).
  -2.1.12 If the ID and PIN were entered correctly, the system displays the main Student Link Screen.
  -2.1.13 The student then clicks on the icon for Grades.

Figure 8 – Numbered step format

1 Group Outliner Practice
2 UA Info Scenarios
-2.1 Student Checks Semester Grades using UA info
  -2.1.1 Student Starts Internet Browser
  -2.1.2 System Displays Browser's Main Screen
  -2.1.3 Student Requests UA's Web Site
  -2.1.4 System Displays UA's Web Page
  -2.1.5 Student Selects "Student Information"
  -2.1.6 The system displays the Student Information web page.
  -2.1.7 The student then selects the "Student Link" link, again based on its description.
  -2.1.8 Temporarily, the system displays a web page which asks the student to select Student Link Version 1 or Version 2 which are each briefly described.
  -2.1.9 Assuming the student selects Version 2 since the system says it has a newer, simpler user interface, the system will then display the secure login screen which asks the student to enter his/her Student ID Number and Personal Identification Number (PIN) to ensure that students only access their own information.
  -2.1.10 The student enters the ID and PIN and then clicks the Login button.
  -2.1.11 The system then validates the ID and PIN and requests the student to re-enter them if they are not valid. If the student does not enter a valid ID and PIN or encounters any error in the previous steps that he/she cannot fix, the student cannot look-up grades using UA info's web access from that computer. The student can try another computer or use an alternative method for checking grades (e.g., RSVP, check with professor, wait for grades in mail, or check at Registrar's Office).
  -2.1.12 If the ID and PIN were entered correctly, the system displays the main Student Link Screen.
  -2.1.13 The student then clicks on the icon for Grades.

Figure 9 – Structured step format
Finally, subjects assigned to treatment 3 defined *structured steps* by entering structured action names (who + verb + object, e.g., Student Starts Internet Browser) as individual outline items in Group Outliner and then adding detailed action information as comments (Figure 9). Group Outliner automatically generated step numbers.

The three formats served as the treatment levels in a single factor randomized complete block design. Experimental session was used as the blocking factor to control for (1) possible differences in experimental conditions between sessions and (2) the non-random assignment of subjects to sessions. Within each session, subjects were randomly assigned (without replacement) to one of the three treatments, resulting in each treatment being assigned to 3-4 subjects per experimental session. Seven experimental sessions with a total of 23 observations per treatment were conducted. The total sample size of 69 exceeded the 63 conservatively estimated as needed to achieve a power of .80 for \( \alpha=0.05 \).

### 3.5.4 Questionnaire Development

As stated above, each student completed a post-session questionnaire at the end of each experiment. The post-session questionnaire was designed to collect:

- **Subject’s assessment of the quality of their scenarios and process productivity.**
  The thirteen quality and six productivity questions directly paralleled the proposed quality and productivity constructs.

- **Ease of use of the tool used to define scenarios (GroupSystems Group Outliner) to ensure that it did not negatively impact scenario definition.** These seven questions were directly taken from Fred Davis’ widely used perceived ease of use instrument (Davis, 1989).
• Demographic data so that homogeneity of subjects across treatments could be assessed.

• Self-reports on motivation to ensure that there were no significant differences between treatments.

A text copy of the questionnaire is included in Appendix C.5. The actual questionnaire was administered to participants on-line using the GroupSystems Survey tool.

3.5.5 Analytical Techniques

Analysis of the experimental results was based on quantitative analysis of scenario quality and productivity measures and post-session questionnaires plus qualitative assessment of the scenario descriptions and the researcher’s observations from the experimental sessions.

Analysis of scenario content focused on the first scenario since that was the only one defined by all subjects. To ensure accurate and consistent counts of the scenario content, a master spreadsheet was developed which consolidated all scenario information provided by the subjects for scenario one. Scenario information included actors, actions, descriptions, data requirements, exceptions and alternatives, based on the definition of a scenario description provided in the experiment instructions (Figure 6). The spreadsheet was compared to the university’s course registration instructions to ensure that it accurately reflected the current process. A total of 66 possible actions were identified in the master spreadsheet description of scenario one. Each subject’s scenario definition was then compared against the master spreadsheet to count the number of actions, actors,
descriptions, data requirements, exceptions, and alternatives identified for that scenario.

Measures of scenario quality and productivity were calculated as follows:

- **Percent of total actions** was calculated by dividing the number of actions in the scenario by 66 (the total possible number of actions).

- **Average sequential information per action** was calculated by adding the number of actions, actors, descriptions, and data requirements together (i.e., the sequential information) and dividing that sum by the number of actions.

- **Average non-sequential information per action** was calculated by adding the number of exceptions and alternatives together (i.e., the non-sequential information) and dividing that sum by the number of actions.

- **Total scenario length** was calculated by counting the number of words included in all scenario descriptions. Length was measured in words to ensure a consistent count because of the variations in other traditional measures of length such as lines or pages. Although length is the only productivity measure commonly used for scenarios, questions could be raised on whether it is a valid measure. For example, does a scenario description with 1000 words provide twice the information and therefore twice the quality as one with 500 words or is it just more verbose? Also, does it take more words to express the same amount of scenario information in any of the formats? Therefore, before using word count to measure productivity, two analyses were performed. First, the correlation between the length of scenario one and total scenario one information was evaluated. Second, the ratio of word count to total scenario
information was analyzed to assess format conciseness. Since correlation was high (.80) and there were no statistically significant differences in conciseness between treatments, total scenario length in words was used to measure overall productivity.

Preliminary analysis focused on the demographic and motivation questionnaire results. A common factor analysis of the individual scenario quality, productivity, and ease of use questions was conducted to ensure the questions loaded on the appropriate factors. Factor scores were used to compare treatments on the resulting factors.

Quantitative analyses of the scenario quality and productivity measures began with evaluation of descriptive statistics followed by comparisons between treatments. Since experimental session was not significant as a blocking factor for any measure, it was eliminated from the analysis. All quality and productivity measures were evaluated together using MANOVA to ensure control of the experiment-wide error rate (Hair, Anderson, Tatham, & Black, 1995). Tests for differences between means for individual measures were then conducted using ANOVA with pair-wise comparisons of means evaluated using Tukey's method of multiple comparisons to ensure control of the error rate (Hair et al., 1995; Neter, Wasserman, & Kutner, 1990).

3.6 Phase II Systems Development: Research Design

The primary purpose of this research task was to develop and evaluate a new system that provided better support for collaborative scenario elicitation than existing systems. The literature review in Chapter 2 documented the many systems development methodologies that could have been used to accomplish this task. However, those
methodologies view the development process from an operational view vs. a research perspective. To ensure the appropriate research perspective, the systems development research methodology defined in (Nunamaker et al., 1991a) was used to guide this task. This methodology consists of five stages which are described in the following sections.

3.6.1 Construct a Conceptual Framework

The research perspective of this methodology is firmly established in this first stage which requires the researcher to clearly state and justify the significance of his or her research question. The question should be discussed in terms of an appropriate conceptual framework to set the stage for theory building.

The goal of this research task was to continue to refine the response to this dissertation's primary research question:

What are the collaborative modeling processes, tools, and facilitation techniques needed to effectively elicit scenarios from users in a group environment?

The significance of this question was previously discussed in Chapter 1. The CSEM scenario framework and collaborative scenario elicitation methodology developed during Phase I in response to this question served as the conceptual framework for this task. The Phase I research also served as the initial investigation of required functionality for the new system. During this task, system functionalities and requirements were explored in more depth by re-evaluating the earlier responses to the questions:

What are the functional requirements for a collaborative scenario modeling tool?
What information should be included in the contextual scenarios elicited directly from user groups?

What format should be used for contextual scenarios and how does that format impact scenario quality and productivity of user definition of scenarios?

What group process and facilitation techniques are appropriate for each collaborative tool used to elicit contextual scenarios from user groups?

Two other issues traditionally addressed during this stage are (1) to develop an understanding the system building processes and procedures and (2) to study relevant disciplines for new approaches and ideas. Both issues were directly applicable to this research.

The first was important because the systems development was a funded team project with the author serving as primary researcher, requirements engineer, and project leader. Other team members served as system architect, interface designer, and programmers. Understanding the strengths and weaknesses of the prototyping process used was critical to success as a project leader and to the on-time delivery of the first version of the prototype to the research sponsors.

The second issue was important because the emphasis of the research expanded from scenarios in Phase I to an integrated scenario and process modeling perspective in Phase II. This change occurred because the Phase I results showed that users needed a graphical process modeling capability in addition to the textual scenario format. This expansion required investigation of process modeling approaches in addition to the scenario techniques previously explored as part of the Chapter 2 literature review.
3.6.2 Develop a System Architecture

Development of a system architecture not only requires definition of a unique architecture and functionalities of system components. It also requires identification of constraints imposed by the environment, statement of development objectives, and specification of measurable requirements. This information is necessary to ensure that the system functionalities satisfy the known constraints, objectives and requirements. Definition of measurable requirements and an appropriate architecture also ensure that the “system is testable against all the stated objectives and requirements” and “experience and design expertise gained from building the system can be generalized for future use” as required when using systems development as a research methodology (Nunamaker et al., 1991a, p. 101).

Development of the system architecture for the Collaborative Distributed Scenario and Process Analyzer (ColD SPA) prototype was truly a team effort. The author developed the initial list of constraints, development objectives, and system functional requirements. The system architect and programmers then proposed a system architecture and functionalities to satisfy that list. The entire team worked together under the author’s leadership to identify additional constraints, objectives, and requirements and to negotiate and finalize the system architecture.

3.6.3 Analyze and Design the System

During this stage of the systems development research methodology, requirements are analyzed, databases and user interfaces designed, and, for object-oriented systems, system functions and behaviors allocated to classes.
As with the previous stage, analysis and design of SPA was a team effort. The author conducted the primary requirements analysis, developed the logical database design, and specified user interface requirements. The user interface specialist developed alternative interface designs to meet those requirements and then worked in collaboration with the author and other team members to finalize the design. Detailed definition of classes and allocations of system functions and behaviors to those classes were done by the programmers under the direction of the system architect. Therefore, results of this research stage presented in Chapter 5 will be limited to those architectural issues in which the author played a direct role.

3.6.4 Build the System

System building is essential for proof-of-concept and to provide an artifact which can be evaluated to provide feedback to the theory building process. It allows researchers to "learn about the concepts, frameworks, and design" and "gain insights about the problems and complexity of the system" (Nunamaker et al., 1991a, p. 98).

Teamwork continued during the systems building stage of the SPA prototype, but with a definite workload shift from the author to the systems architect and programmers who were responsible for all coding. The author's role during this stage was limited to project leadership and requirements engineering including leading discussions required to clarify requirements, identifying new requirements, and establishing development priorities. The entire systems building effort was an incredible learning experience for the team because of the newness of the SPA concept and the development environment. Insights gained from the systems building process are documented in Chapter 5.
3.6.5 Observe and Evaluate the System

The final stage in the systems development research methodology is to observe and evaluate the system to determine compliance with the stated requirements, assess its impact on users and groups, and identify desired improvements. Use of the SPA prototype was observed during case studies as described in section 3.7. Lessons learned during systems development and the case studies were consolidated and used to revise the CSEM scenario framework and collaborative scenario elicitation methodology as described in section 3.8.

3.7 Phase II Observation: Case Study Design

Design of the case studies for Phase II was very similar to those conducted during Phase I. The goals of the Phase II exploratory case studies were: (1) to increase understanding of collaborative scenario and process elicitation using the SPA prototype, (2) to evaluate the proposed methodology, and (3) to help define questions and directions for follow-on research. The major difference here were that the case studies used SPA instead of GroupSystems for scenario or process elicitation (vs. just scenarios in Phase I). The Phase I justification for an exploratory, multiple-case, embedded case study research design also applied to Phase II. As previously discussed in Chapter 1 and Phase I, the study's primary research question was:

What are the collaborative modeling processes, tools, and facilitation techniques needed to effectively elicit scenarios from users in a group environment?

The primary purpose of the Phase II case studies was to gather the information needed to adapt the responses to the above question developed during Phase I for GroupSystems as
required to support SPA. The case studies were also used to increase understanding of
collaborative scenario and process elicitation using the SPA prototype so that a more
detailed response to the following question could be developed.

What are the functional requirements for a collaborative scenario
modeling tool?

The Phase II case studies’ proposition, units of analysis, logic linking the data to the
proposition, and criteria for interpreting the findings were the same as for Phase I.

3.8 Phase II Theory Building

The primary goal of theory building during Phase II was to use the results of the
research conducted in both phases to refine and improve the CSEM scenario framework
and collaborative scenario elicitation methodology developed in Phase I. Comprehensive
responses to the research’s primary question,

What are the collaborative modeling processes, tools, and facilitation
techniques needed to effectively elicit scenarios from users in a group
environment?

were developed for both GroupSystems and SPA. However, because this research was
exploratory in nature and only the first version of SPA was evaluated, further research is
needed to validate those responses. A secondary goal of the Phase II theory building
task, therefore, was to identify those elements of the theory and other open issues
requiring additional research.
CHAPTER 4 – SCENARIO USAGE IN CSEM

4.1 A Conceptual Framework for CSEM Scenario Usage

An overview of the Collaborative Software Engineering Methodology (CSEM) was presented in Chapter 2. A more complete description of CSEM is provided in (Dean et al., 1997-98) and supplemental training materials developed for individual clients. While these materials provide an initial picture of the role of scenarios in CSEM, they do not provide a comprehensive picture of scenario usage throughout development. This gap resulted in the research question:

How will scenarios be used during all CSEM development phases and how are the scenarios developed for these different purposes related?

The purpose of this section is to respond to this question by defining an initial conceptual framework for scenario usage in CSEM. Results of the literature review and prior GSS and systems analysis experiences were used to develop this framework.

The CSEM defines four primary development phases: planning, requirements, design, and implementation. Analysis of the scenario literature highlighted potential roles for scenarios throughout all four phases. Figure 10 provides an overview of how the different types of scenarios are used during each of the CSEM development phases.

![Figure 10 - CSEM scenario usage](image-url)
Detailed descriptions for each scenario type by development phase are presented in the following sections. Scenario usage during the requirements phase is presented first since requirements are the focus of this research. It also serves as the basis for usage during the other phases, which are described after the requirements phase.

4.1.1 Requirements Phase

During the early stages of the requirements phase, user groups describe how they do their jobs by defining business scenarios, often called contextual scenarios in the research literature. Starting the information systems requirements determination process with business scenarios ensures that the process is focused on the business needs of the users. It also leverages on users' expertise by asking them to provide the information they know (how they do their jobs) versus what they may not (what an IS should do). Other critical roles for business scenarios during the requirements phase include:

- Provide a mechanism for users to: (1) develop a common understanding of key business processes, (2) reconcile differences between how different organizations and users currently perform those processes, (3) manage multiple process views, (4) identify opportunities for process improvement, and (5) brainstorm ideas on what they would like the new system to do.

- Document a rich picture of the business which analysts and developers can use to quickly learn about the problem domain, analyze to determine areas needing additional clarification, enhance with information gained during follow-on user interviews or meetings held to get that clarification, and use as a baseline from which system use cases can be developed.
• Assist users and data modelers in identification of data requirements by identifying what data is required to support a business scenario, how that data is used, and by whom.

• Support assessment of IS alternatives based on how they impact the business during evaluation of requirements prototypes and analysis of proposed integrated data models.

As the emphasis moves from the business to the information system during the later stages of the requirements phase, scenario usage transitions from the business scenario to the interaction scenario (i.e., system use case). System use cases help analysts explore alternative IS solutions with users, support documentation of detailed business processing logic, aid developers in determining what to prototype, and provide a mechanism for bridging the gap between business and IS requirements. Each system use case should be traceable back to the business scenario(s) it was designed to satisfy, allowing users to directly evaluate its impact on those scenarios.

4.1.2 Planning Phase

Business scenarios can also be useful during the planning phase. Planning is particularly important in CSEM because of its focus on an incremental development life cycle. As a result, the overall systems development project must be planned, systems divided into increments, and then each individual increment planned. High-level business scenarios can assist developers and customers in prioritizing systems/increment development efforts based on the projected impact on the business. They can also be
useful in dividing systems into increments based on business scenario priorities or logical business groupings of scenarios.

4.1.3 Design Phase

System use cases can be used to support many of the design phase’s activities. Analysts and developers can use them during discussions with users to clarify requirements, analyze additional exceptions or alternatives, and refine the user interface. Developers use them to explore alternative designs, document design rationale, and identify opportunities for reuse of code. System use cases continue to drive prototypes developed to evaluate alternative designs or the user interface. They are also useful for allocating work between development teams or individual programmers.

Internal scenarios can be used during detailed design by modifying the “black-box” system use cases to include specifications of internal system processing.

4.1.4 Implementation Phase

The CSEM implementation phase includes programming, testing, and fielding activities. Scenarios can be used during each of these activities.

Internal scenarios can be used during system coding as detailed programming specifications. The design rationale captured in system use cases can also aid developers in understanding the intended design when internal scenarios are incomplete.

Using scenarios during testing will significantly increase the traceability between the development phases and from systems to business requirements. Test scenarios represent detailed instances of normal and exceptional courses of action that the system is required to support and can be developed at three levels as shown in Figure 11.
Program test scenarios, developed from the internal scenarios, are useful for testing system internals during unit and integration testing by developers. System test scenarios, developed from the system use cases, can be used during system testing to verify that the system satisfies the IS requirements. Finally, user test scenarios, developed from the business scenarios, are useful during acceptance testing by users to ensure that the system fully satisfies their business requirements.

Scenarios could also be used for planning and implementation of system fielding, but these scenarios should be separately developed by the fielding team and would not be directly linked to the business, interaction, internal, or test scenarios described during the other CSEM phases.

4.1.5 Summary of Scenario Usage and Relationships

The previous sections defined the roles business, interaction, internal, and test scenarios play during CSEM development. They also provided an overview of how those scenarios are related (Figure 11). In the simplest case, one business scenario is
supported by a single system use case which is linked to a single internal scenario.

Details of the user – system interaction are added to the business scenario to create the system use case. Additional details about internal system processing are added to the system use case to create the internal scenario. Test scenarios evolve from their associated scenarios, generally in a one-to-many relationship, by defining specific normal and exceptional test cases for each scenario.

Although some scenarios may be related in this simple manner, often more complex many-to-many relationships will be required. For example, one business scenario may be supported by multiple system use cases, and one use case may support multiple business scenarios. Figure 12 provides a limited example of these more complex relationships where portions of two business scenarios are supported by a common system use case (#2). Other portions of the business scenarios are supported by unique system use cases. In addition, some portions of the business scenarios may not require system support and therefore are not associated with any system use case. It is

Figure 12 – Scenario relationships
easy to imagine that as the number of business scenarios increase, the relationships to system use cases could become very complex. Relationships between system use cases and internal scenarios could also be complex, especially in object-oriented environments that emphasize code reusability.

The preceding discussion on CSEM scenario usage and the relationships between the different types of scenarios provides a conceptual framework for this research. Now that the goals for scenario usage throughout development are understood, the discussion can turn to the first step in that process and the focus of this research – collaborative elicitation of business scenarios from users during the requirements phase.

4.2 Preliminary Collaborative Scenario Elicitation Methodology

The next step in theory building was to propose a method for collaborative elicitation of scenarios using an existing GSS, GroupSystems. The goal was to develop a preliminary GroupSystems response to the question:

What group process and facilitation techniques are appropriate for each collaborative tool used to elicit contextual scenarios from user groups?

The proposed methodology describes how GroupSystems can be used to elicit scenarios during the requirements phase in an evolutionary process that starts with rich descriptions of business processes, then evolves those business (contextual) scenarios into specific system use cases, prototypes, and finally into requirements specifications. This is accomplished through an iterative process of individual user scenario definition, group refinement of scenarios, development team analysis and implementation, and group review and feedback until requirements are agreed upon. Both the type and format
of scenarios evolve as shown in the following overview of the six primary CSEM Requirements Phase activities (Figure 13), focusing on the role of scenarios in the overall requirements process.

Figure 13 – CSEM requirements elicitation steps

Step 1. Identify Business Activities

The first step in the requirements phase is the identification of business activities. GroupSystems Group Outliner is commonly used to support user groups in creating informal activity hierarchies. The Activity Modeler is used to develop structured activity models when more formalism is warranted. In this case, the activity hierarchy is exported to Group Outliner after the activity model is completed. The user group analyzes the activity hierarchy to identify the activities having the greatest need or promising the greatest potential benefit from new or improved information systems support. The output of this step is a preliminary definition of the scope of the IS and a Group Outliner hierarchy of discrete business activities and processes within that scope for which business scenarios will be generated.
**Step 2. Generate Business Scenarios**

The purpose of this step is to capture the dynamic view of business processing requirements through the definition of business scenarios. During this step, user groups generate business scenarios for each of the discrete business activities included in the current increment as identified during business activity modeling (step 1). GroupSystems Group Outliner is used to add scenario definitions to the activity hierarchy.

Initially, users work in parallel to develop rich, contextual business scenarios to describe current work processes. Scenarios are generally defined for activities at the lowest level of the activity hierarchy (the leaf nodes of the tree). The goal is to define at least one scenario describing the normal course of action for each leaf node activity included in the current increment. Additional scenarios may be defined to document variations in the normal course of action for different users or organizations or to describe alternative methods for performing the activity. Minor exceptions that occur during the normal course of actions can be captured as part of those scenario descriptions. However, if exceptions cause major variations in the process, separate exceptional scenarios should be defined to describe those variations.

Next, the user group is led to converge on a common definition of work processes through the iterative refinement of the scenario descriptions. When there are multiple methods for performing an activity (i.e., multiple normal scenarios), the user group should attempt to agree on a common process. If they cannot, they should justify why the different processes are required since supporting multiple processes for the same activity may complicate the IS development effort.
User group members also brainstorm on how the scenario might be improved and on how information systems might support the business process. In this way improvements from business process changes and information systems enhancements can be identified. Business scenarios are updated accordingly.

Data required to support these scenarios are also identified so they can be incorporated into the data model. Upon completion of the initial integrated data model (step 3), the user group evaluates potential impacts of the integrated data on the business, i.e., how data integration would change the desired business scenarios (Hickey, Dean, & Vogel, 1997). If these impacts are acceptable, the business scenarios are updated to reflect those changes. Otherwise, the user group recommends revisions to the initial integrated data model that are acceptable from a business processing perspective.

The integrated business scenarios developed during this step identify how an information system can support the business and provide a rich context for evaluating the impacts of alternative information systems solutions on the business. The outputs of this step are the data requirements input to step 3 and the integrated business scenarios used during step 4 to identify alternative system use cases.

**Step 3. Develop Data Model**

In parallel with business scenario modeling, data modelers work with user groups to develop an integrated data model to capture the business data requirements and associated business rules. The data modeler develops an initial model from existing systems and data models. Users then analyze the business scenarios to identify additional data requirements. Data modelers simultaneously work with users to identify potential
data integration opportunities, which in turn are incorporated into the initial integrated data model. User groups assess the impact of integration on the business, using scenarios as described in step 3. Feedback from that analysis is used to update the integrated data model, which is validated by the users. The output of this activity is the integrated data model which is used extensively during the next two steps and is included in the final requirements specification.

**Step 4. Define System Use Cases**

In this step, the focus shifts from the business to the information system. Systems analysts, working with small groups of users, analyze the integrated business scenarios, data model, and IS ideas to define alternative system use cases. CSEM use cases consist of a narrative description of the action sequence supplemented by menu, screen, and report mock-ups and logical models as required to fully specify the use case for the developers. These use cases are used during step 5 to develop the initial requirements prototypes for user evaluation.

**Step 5. Evaluate Prototypes**

A requirements prototype is developed based on the proposed system use cases to help users identify and validate their requirements. The business scenarios are used to guide the prototype evaluation and provide a rich context for evaluating the impacts of alternative information systems solutions on the business. Feedback from the evaluation of the prototype is used to update the business scenarios, data models, system use cases, and the prototype in an iterative development and evaluation process. The outputs of this
Step 6. Prioritize and Document Requirements

The goal of this step is to consolidate and prioritize requirements identified during user group sessions. Documentation of these requirements in combination with the activity, data, and business scenario models, the system use cases, and the prototype provide a complete and useful requirements specification for the development team. The succinct documentation generated by this step is supplemented by the knowledge participants have gained from taking part in the process. This critical step summarizes the results of the entire requirements process and documents user priorities and agreements.

4.3 Initial Contextual Scenario Data Model

A preliminary data model was also developed as part of the collaborative scenario elicitation methodology to respond to the question:

What information should be included in the contextual scenarios elicited directly from user groups?

Information from the literature review and prior GSS experiences were used to define the preliminary collaborative scenario elicitation method and contextual scenario data model.

A review of the literature on contextual scenarios indicated that two types of information must be gathered about business scenarios. The first type is information about the scenario as a whole (e.g., what is the goal or objective of the scenario). The
second type is information about each individual action within a scenario (e.g., who
performs the action). Pohl and Haumer (1997) emphasized that some information
traditional viewed as relating to the scenario, e.g., resource requirements, in actuality
represented a roll-up of those requirements for each individual action. They also
developed a very comprehensive contextual data model, which served as the basis for
this model. However, some information in their model was not applicable or appropriate
for CSEM business scenarios. For example, the identification of scenario location (e.g.,
Army installation) could serve to further polarize organizations and inhibit the ability to
reach consensus on multi-organizational processes. Therefore, location was not included
in the scenario data model. There was also some information that Pohl and Haumer
(1997) did not identify which could aid users in identifying scenarios. For example,
(Whitten & Bentley, 1997) describe how triggers can be used to help users identify
events. Prompting users to think about what inputs, events, time, or state changes
triggered them to action should be a very useful mechanism for identifying scenarios.
Therefore, trigger was included in the scenario data model. The scenario and action data
items included in the model are described in the following paragraphs.

A complete contextual business scenario description should include information
about the scenario as a whole and detailed information about each individual action or
step taken to accomplish that scenario. Scenario-level information should include:

- Name – Short descriptive name of the scenario, generally of the form Verb +
  Object.
- **Description** – Used to capture a brief description of the scenario, i.e., what’s happening. Not to be confused with objective, described separately.

- **Objective** – Used to capture the objective, purpose, or reason for doing the scenario, i.e., why are you doing the process, what do you expect to get.

- **Trigger** – Input, event, time, or state change that triggers the start of a scenario.

- **Preconditions** – Used to capture the preconditions, prerequisites, and/or starting constraints for a scenario. May also be used to list key assumptions.

- **Viewpoint** – Role name of individual or groups of individuals from whose perspective this scenario is being described. For example, a manager and a worker may have very different views of a business process, so it is important to identify which view is being used.

- **Results** – Used to capture the result, output, post condition, and/or ending constraints for a scenario, generally in the form of system outcomes. This data item will normally be identified by the analyst during system use case development based on the users’ definition of the scenario objective.

- **Exceptions** – Used to identify problems, error conditions, or constraints on continuing which may impact on the successful completion of a scenario. These exceptions generally impact major portions of the scenario and change the expected normal course of action. Exceptions impacting one or two individual actions are documented as part of the action description.
• Alternatives – Used to capture alternative ways of accomplishing the entire scenario or major portions of the scenario. Alternative ways of completing an individual action are included in the description for that action.

• Related Issues – Used to capture other factors, constraints, states, pending changes, improvement ideas, disagreements, or any other information that is not captured elsewhere that may influence or impact the scenario.

An example of scenario-level information is shown in Table 4.

<table>
<thead>
<tr>
<th>Scenario Information</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Generate HM Authorization Request</td>
</tr>
<tr>
<td>Description/Objective</td>
<td>Complete a request for authorization (form) to use HM</td>
</tr>
<tr>
<td>Preconditions</td>
<td>Requestor must know what HM and have legitimate requirement to use</td>
</tr>
<tr>
<td>Viewpoint/Who Does This?</td>
<td>New HM User</td>
</tr>
<tr>
<td>Trigger</td>
<td>Mission need to perform task using HM</td>
</tr>
<tr>
<td>Result</td>
<td>Form accepted &amp; control # assigned</td>
</tr>
<tr>
<td>Exception</td>
<td>In emergency, complete after the fact</td>
</tr>
<tr>
<td>Related Issues</td>
<td>Authorized use, quantity limits</td>
</tr>
</tbody>
</table>

Table 4 – Example of scenario-level information

Action-level information which should be included in the scenario description includes:

• Actor – List of who or what does this action. Can be an individual, group of individuals, or a system. Individuals are usually identified by role name vs. person name.

• Name – Short descriptive name of the action, generally of the form Verb + Object.

• Description – Used to capture a brief description of the action, i.e., what's happening.
• Information Resources – Used to capture objects, data attributes or entities needed to accomplish the action. These could be either data attributes (fields) from specific entities (tables) or just the entity (table) names depending on the level of detail necessary.

• Other Resources – Used to capture any resources, other than information (captured as Information Resources), needed to accomplish the action. Could be other people, systems, materials, products, production technology (equipment or machines), communication media, etc., that are consumed or needed during the scenario action.

• Exceptions – Used to identify problems, error conditions, or constraints on continuing which may impact on the successful completion of the action.

• Alternatives – Used to capture alternative ways of accomplishing the action.

An example of action-level information is shown in Table 5.

<table>
<thead>
<tr>
<th>Action Information</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actor</td>
<td>Requestor</td>
</tr>
<tr>
<td>Action</td>
<td>Completes Form</td>
</tr>
<tr>
<td>Description</td>
<td>Enters requestor, material, process, and authorization information</td>
</tr>
<tr>
<td>Exceptions</td>
<td>If do not know shop/process code, get from Hazmart/Hazmincen or ESOHF Team</td>
</tr>
<tr>
<td>Alternatives</td>
<td>Use manual or automated form</td>
</tr>
<tr>
<td>Data/Other Resources Required</td>
<td>EPA ID</td>
</tr>
</tbody>
</table>

Table 5 - Example of action-level information

A complete description of the “Generate Hazardous Material Authorization Request” scenario is included in Appendix A. This scenario is used as a sample throughout the remainder of this dissertation.
The proposed contextual scenario data model defines information generally required in a business scenario description. It should be used as a starting point for planning the scenario elicitation process. The data model may need to be revised to include more or less information based on the information already gathered, the status of the current system, or the existence of legacy systems. The focus should always be on rapidly capturing the essential business knowledge of the users to provide systems analysts and developers the information they need to develop IS solutions.
PHASE I: Collaborative Scenario Elicitation Using GroupSystems

5.1 Case Study Results: Unstructured Group Systems Scenarios

The first case study of a collaborative scenario elicitation meeting was conducted in December 1996, shortly after development of the Collaborative Software Engineering Methodology (CSEM). The initial framework and methodology for scenario usage during the CSEM requirements phase (sections 4.1 - 4.2) had been proposed, but had not been evaluated in a real-world setting. The contextual scenario data model (section 4.3) had not yet been defined. The purpose of this case study was (1) to begin evaluation of the proposed methodology and (2) to determine what additional questions needed to be addressed to develop a comprehensive collaborative scenario elicitation methodology. A general-purpose GSS, GroupSystems, was used to support the scenario meetings because it provided the required collaborative support, appeared to "fit" the task, and had been successfully used for similar meetings and tasks.

5.1.1 Background

All the Phase I case studies analyzed meetings sponsored by the Defense Environmental Security Corporate Information Management (DESCIM) Program Management Office (PMO). The DESCIM PMO is responsible for development of integrated information systems to support environmental security management functions at Army, Navy, Air Force, Marine, and DOD installations world-wide. The University of
Arizona’s Center for the Management of Information (CMI) has provided research, process management, and facilitation support for the DESCIM projects since 1992.

The first case study focused on the cultural and natural resources management integrated system. An initial list of cultural and natural functional requirements and an integrated data model had been developed for the DESCIM PMO prior to the meeting. No activity model was available although the requirements list provided an initial view of cultural and natural resources management activities.

5.1.2 Cultural and Natural Resources Management Meeting

The DESCIM cultural and natural resources management meeting was a same time, same place meeting held at the University of Arizona on December 9 – 13, 1996. The following description of the meeting is structured to describe the primary components of the GSS research model including the meeting tasks, group, context, technology, process and outcome.

Task. The goals for the cultural and natural resources meeting were (1) create an activity model based on the previously defined functional requirements and (2) develop scenarios to define how cultural and natural personnel perform their jobs.

Group. Sixteen cultural and natural resources subject matter experts representing the Army, Navy, Air Force, Defense Logistics Agency (DLA), Department of Defense (DOD), and National Guard Bureau (NGB) actively participated in the meeting. Several DESCIM PMO representatives also attended, primarily as observers. Two CMI researchers facilitated the meeting. The author observed the scenario elicitation and review activities.
**Context.** The meeting was conducted as a face-to-face meeting using a large permanent GroupSystems meeting facility at the University of Arizona. All DOD participants traveled to attend the meeting and would be directly impacted by the meeting outcome, either as users or members of the project team for the proposed system.

**Technology.** A general-purpose GSS, GroupSystems, was the primary software used to support the meeting. The facilitators and all meeting participants used individual personal computers to access GroupSystems. Each participant's GroupSystems workstation was directly connected via a local area network (LAN) to the meeting room's local GroupSystems server. Three public screens were connected to the facilitators' workstations and other display equipment.

**Process.** Meeting participants developed a preliminary hierarchical list and definitions of cultural and natural resource management activities using GroupSystems Group Outliner. Next, participants created a high-level activity model to refine the activity hierarchy and define relationships between activities using GroupSystems Activity Modeler. The activity hierarchy was shifted back to Group Outliner so participants could define business scenarios, generally for the lowest level nodes in the activity hierarchy (i.e., the leaf nodes). All participants were provided written instructions describing what a scenario is, how to define a scenario, and three sample scenarios in free-format text and numbered step formats (see Appendix B). Participants documented the scenario name, viewpoint, and steps required to accomplish the scenario as comments for each scenario's parent activity in the outline.
Group review of the activity list and scenarios showed that two activities needed further decomposition and analysis: (1) Sustain resources and (2) Conduct resource studies. These portions of the model were exported to separate Group Outliner activities. Sustain resources was decomposed two more activity levels with scenarios provided for these lowest levels. Conduct resource studies was decomposed into specific actions required to accomplish the studies. Actions were added to the outline with definitions added as comments. These two models were then merged back into the original model to create a consolidate view of cultural and natural resources activities, activity definitions, and scenarios.

**Outcome.** Users quickly and easily defined scenarios for a wide range of business activities based on the general instructions provided. Most scenarios were associated with the leaf-node activities or the next level up, except when these activities were further decomposed at a later time (e.g., as occurred for Sustain resources). In those cases, scenarios occurred at multiple levels throughout the activity hierarchy. These differences, however, did not cause any serious problems other than some minor confusion when reviewing the final report on the merged model. The main problem was that the resulting scenarios varied greatly in the type of information, format, and level of detail included in the descriptions. For example, some descriptions included only a high-level definition of the goal of the scenario. Other scenarios included the goal, viewpoint, and detailed actions required to achieve that goal as requested in the handout. Still others provided a general scenario description and then focused on how a system should support that scenario versus providing the requested scenario actions. Finally, some
scenarios were written as free-format textual descriptions while others were clearly broken down into numbered steps.

End-of-session feedback from the users indicated they found the GroupSystems tools easy to use. Users did express a need for a clearer, up-front statement of the meeting goals and more detailed instructions for the activity modeling and scenario definition processes. Facilitators and observers agreed that users seemed to find the GroupSystems tools very easy to use. Some users did appear to have difficulty understanding how to logically decompose activities and develop the activity model. Most seemed to have much less difficulty in quickly grasping the scenario concept, although they were less clear on how best to document scenarios. This confusion and the lack of definitive content and format guidelines were, in all likelihood, the primary causes of the scenario differences discussed above. Regardless of these differences, however, the final scenario descriptions did provide a very rich, if sometimes incomplete, picture of cultural and natural resources management.

5.2 Experimental Results: The Impact of Scenario Format

The purpose of this experiment was to increase understanding of the impact of scenario format so that the detailed scenario guidelines users requested (section 5.1.2) could be developed. The experiment focused on the initial individual user definition of contextual scenarios using the three most common textual scenario formats. Its goal was to provide a detailed response to the research question:

What format should be used for contextual scenarios and how does that format impact scenario quality and productivity of user definition of scenarios?
The experimental design presented in section 3.5 describes the task, group, context, technology, and process components of the GSS research model. Additional information about the group is provided in section 5.2.1. The remainder of this section provides detailed information about the outcome of the experiment. In general, all participants easily defined the main actions for each scenario. However, many did not provide all the requested information for each action (e.g., data requirements, exceptions, and alternatives). Problem areas and quality and productivity differences between the treatments are highlighted in sections 5.2.2 – 5.2.3.

5.2.1 Participant Demographics and Motivation

Analysis of the demographic questions showed that no significant differences existed between treatment groups on these characteristics. Subjects were primarily senior MIS majors with multiple MIS courses, but only limited analytical expertise. On average, they used computers one or more times a day and had very good computer and word processing expertise with better than average typing skills. They also were somewhat familiar with GroupSystems. The other experimental check was for motivation and effort. On average, motivation and effort were very high (4.28 and 4.17 on a 5-point scale) with no significant differences between treatment levels.

5.2.2 Assessing Overall Scenario Quality and Productivity

To develop a better understanding of subjects' ability to define scenarios, descriptive statistics for scenario quality were analyzed. All statistics showed a high degree of individual variability and identified important quality problems. For example, as shown in Table 6, while the average number of actions per scenario definition was
18.62, the individual counts varied from a low of 6 to a high of 38. Most other measures showed similar variability. These results were consistent with observations during the experiments where some subjects seemed to be rapidly defining scenarios while others seemed to be struggling.

<table>
<thead>
<tr>
<th>Scenario Measures</th>
<th>Mean</th>
<th>Std Dev</th>
<th>Median</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td># Actions</td>
<td>18.62</td>
<td>5.06</td>
<td>18</td>
<td>6</td>
<td>38</td>
</tr>
<tr>
<td># Actors</td>
<td>17.77</td>
<td>5.40</td>
<td>18</td>
<td>5</td>
<td>38</td>
</tr>
<tr>
<td># Descriptions</td>
<td>9.17</td>
<td>4.54</td>
<td>9</td>
<td>2</td>
<td>22</td>
</tr>
<tr>
<td># Data Requirements</td>
<td>4.13</td>
<td>2.58</td>
<td>4</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td><strong>Total Sequential Information</strong></td>
<td><strong>49.70</strong></td>
<td><strong>15.06</strong></td>
<td><strong>49</strong></td>
<td><strong>15</strong></td>
<td><strong>106</strong></td>
</tr>
<tr>
<td># Exceptions</td>
<td>2.64</td>
<td>2.18</td>
<td>2</td>
<td>0</td>
<td>11</td>
</tr>
<tr>
<td># Alternatives</td>
<td>0.74</td>
<td>0.98</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td><strong>Total Non-Sequential Information</strong></td>
<td><strong>3.38</strong></td>
<td><strong>2.46</strong></td>
<td><strong>3</strong></td>
<td><strong>0</strong></td>
<td><strong>13</strong></td>
</tr>
<tr>
<td>Total Scenario Information</td>
<td>53.07</td>
<td>16.39</td>
<td>53</td>
<td>15</td>
<td>115</td>
</tr>
<tr>
<td>% Total Possible Actions</td>
<td>0.28</td>
<td>0.08</td>
<td>0.27</td>
<td>0.09</td>
<td>0.58</td>
</tr>
</tbody>
</table>

Table 6 – Scenario statistics

Statistics on action completeness were also interesting and surfaced some potential quality problems. For example, while 95% of action specifications included the actor,

![Figure 14 - Action completeness](image-url)
less than 50% provided a description of the action with even fewer identifying data requirements (22%), exceptions (17%), or alternatives (4%). Figure 14 shows similar problems for all treatments, therefore, the scenario format, process, or tool must change to improve action completeness.

Scenario descriptions were also evaluated qualitatively to analyze the types of errors and to identify major ambiguities in the descriptions. The majority of errors were errors of omission whereby subjects did not identify all actions for a scenario or all information requested for individual actions. These types of errors are accounted for in the scenario and action completeness quantitative measures. There were a few other errors in the scenario descriptions which were primarily caused by incorrect action sequence. However, these errors often identified viable alternative action sequences or highlighted areas where the action sequence was not very logical, so they should be considered sources of design information vs. errors to be ignored.

Although text is inherently ambiguous, there were some glaring examples of ambiguities in the scenario descriptions which should be considered when attempting to improve scenario quality. The most common problem was the vague specification of data requirements. For example, many subjects used course number and course call number interchangeably in their scenario definitions when in fact they are distinctly different data items with totally different formats. Another common problem area was ambiguous specification of actors as he/she/it. For example, sometimes it was not clear whether the actor was the student, the instructor, or the registrar.
5.2.3 Comparing Scenario Quality and Productivity

To respond to the research question on the impact of scenario format and to evaluate the four experimental hypotheses, this section compares the scenario quality and productivity measures for the three treatment formats and the unstructured (treatment 1) vs. structured formats (treatments 2 and 3).

**Multivariate Analysis.** A multivariate analysis of variance (MANOVA) was conducted using all four quality and productivity measures to assess the overall fit of the model. Wilks’ lamda was used to assess this fit since it is the most commonly used test statistic for the overall significance in MANOVA (Hair et al., 1995). As shown in Table 7, the overall model fit when comparing the three treatment formats was weakly significant (p=0.0573). Statistical significance of the model improved when comparing the unstructured format to the structured formats to p=0.0286 as shown in Table 8. Given that these results showed at least weak significance for the overall model, analysis of the individual quality and productivity measured continued next.

<table>
<thead>
<tr>
<th>Model/Variable</th>
<th>Treat 1</th>
<th>Treat 2</th>
<th>Treat 3</th>
<th>F Value</th>
<th>p value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall Model (Wilks’ Lamda)</td>
<td>F(8,126)=1.96</td>
<td>0.0573</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Percent of Total Actions</td>
<td>0.3017</td>
<td>0.28</td>
<td>0.27</td>
<td>F(2,66)=1.16</td>
<td>0.3187</td>
</tr>
<tr>
<td>Average Seq Info per Action</td>
<td>1.7359*</td>
<td>1.6809</td>
<td>1.5588*</td>
<td>F(2,66)=3.42</td>
<td>0.0386</td>
</tr>
<tr>
<td>Average Non-Seq Info per Action</td>
<td>0.2208*</td>
<td>0.1768</td>
<td>0.1372*</td>
<td>F(2,66)=2.99</td>
<td>0.0573</td>
</tr>
<tr>
<td>Total Scenario Length</td>
<td>800.30*</td>
<td>632.48</td>
<td>567.30*</td>
<td>F(2,66)=3.86</td>
<td>0.0260</td>
</tr>
</tbody>
</table>

* Significant difference between means (p<.05)

Table 7 – Comparison of the three treatment formats

<table>
<thead>
<tr>
<th>Model/Variable</th>
<th>Unstructured (Treat 1)</th>
<th>Structured (Treat 2 &amp; 3)</th>
<th>F Value</th>
<th>p value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall Model (Wilks’ Lamda)</td>
<td>F(4,64)=2.90</td>
<td>0.0286</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Percent of Total Actions</td>
<td>0.3017</td>
<td>0.2724</td>
<td>F(1.67)=2.29</td>
<td>0.1349</td>
</tr>
<tr>
<td>Average Seq Info per Action</td>
<td>1.7359</td>
<td>1.6198</td>
<td>F(1.67)=3.63</td>
<td>0.0611</td>
</tr>
<tr>
<td>Average Non-Seq Info per Action</td>
<td>0.2208*</td>
<td>0.1570*</td>
<td>F(1.67)=4.61</td>
<td>0.0354</td>
</tr>
<tr>
<td>Total Scenario Length</td>
<td>800.30*</td>
<td>599.89*</td>
<td>F(1.67)=7.19</td>
<td>0.0092</td>
</tr>
</tbody>
</table>

* Significant difference between means (p<.05)

Table 8 – Comparison of the unstructured vs. structured formats
Scenario Completeness. Hypothesis 1 states that increasing the degree of scenario structure will increase the percent of total actions included in the scenario description. However, comparisons of the Percent of Total Actions show no statistically significant differences between treatments (Table 7) or between the structured and unstructured formats (Table 8). In fact, although not significant, the treatment means seem to indicate the opposite effect with the unstructured treatment means higher than the structured means. In summary, there is no support for Hypothesis 1.

Sequential Action Completeness. Hypothesis 2 states that increasing the degree of scenario structure will increase the average sequential information per action included in the scenario description. Results of the analysis show that the means of Average Sequential Information Per Action are statistically different (p=.0386) when comparing all three treatments (Table 7) and weakly statistically different (p=.0611) when comparing the unstructured and structured treatments (Table 8). However, pair-wise comparison of the treatment means indicates that the unstructured treatment 1 is statistically greater than the structured treatment 3 (p<.05). This result directly contradicts Hypothesis 2.

Non-Sequential Action Completeness. Hypotheses 3 states that increasing the degree of scenario structure will decrease average non-sequential information included in the scenario description. Comparisons of the Average Non-Sequential Information Per Action show a weak statistical difference (p=.0573) when comparing the three treatments (Table 7) and a significant difference (p=.0354) when comparing the unstructured and structured treatments (Table 8). Pair-wise comparisons of the treatment means show that
treatment 1 is statistically greater than treatment 3 (p<.05) with the same result for the unstructured format (treatment 1) vs. structured formats (treatments 2 and 3). These results provide direct support for Hypothesis 3.

**Productivity.** Hypothesis 4 states that increasing the degree of scenario structure will decrease the total length of scenario descriptions. Comparison of *Total Scenario Length* shows a statistically significant difference (p=.0260) when comparing treatments (Table 7) and a strong statistically significant difference (p=.0092) when comparing the unstructured and structured formats (Table 8). Pair-wise comparisons shows a statistically significant difference with treatment 1 greater than treatment 3 and the unstructured format greater than the structured formats. These results provide strong support for Hypothesis 4.

5.2.4 Analysis of Post-Session Questionnaire Results

To further explore the research question, subject's perceptions of scenario quality, productivity, and ease of use, collected as part of the post-session questionnaire, were analyzed. The analysis showed that subjects generally rated scenario quality, productivity, and GroupSystems ease of use very high. There were no significant differences between treatments for scenario quality or GroupSystems ease of use. In contrast, subjects rated the structured treatments significantly higher on (1) the ease of the scenario definition method, and (2) whether the method allowed them to do what they needed.

Although the questionnaire results showed few treatment differences, the results did seem to indicate consistency between related questions. A common factor analysis
was performed to explore this commonality. Significance tests reported using the maximum likelihood method of factor analysis with Varimax rotation indicated that three factors were sufficient. Detailed factor loadings are shown in Appendix D with the factor analysis results summarized in Table 9. The three factors directly map to the quality, ease of use, and productivity concepts. In addition, Table 9 shows that the Cronbach's alpha for all factors met or exceeded the recommended .80 standard for business research.

<table>
<thead>
<tr>
<th>#</th>
<th>Factor Description</th>
<th>Proposed Items</th>
<th>Actual Items</th>
<th>Cronbach's Alpha</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Quality</td>
<td>13</td>
<td>11*</td>
<td>.89</td>
</tr>
<tr>
<td>2</td>
<td>Ease of Use</td>
<td>7</td>
<td>8**</td>
<td>.90</td>
</tr>
<tr>
<td>3</td>
<td>Productivity</td>
<td>6</td>
<td>5**</td>
<td>.80</td>
</tr>
</tbody>
</table>

* Questions on alternatives and exceptions dropped due to low commonality with other variables.
** Because of vague question wording, productivity question loaded slightly higher on ease of use.

Table 9 - Questionnaire factor analysis summary

Means of standardized factor scores for each treatment and the unstructured versus structured treatments are summarized in Table 10. As expected from the individual question analysis, the only statistically significant difference is between the unstructured and structured treatments with the structured groups rating productivity higher than the free-format group.

<table>
<thead>
<tr>
<th>#</th>
<th>Factor Description</th>
<th>Treat 1</th>
<th>Treat 2</th>
<th>Treat 3</th>
<th>Unstructured (Treat 1)</th>
<th>Structured (Treat 2&amp;3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Quality</td>
<td>0.10</td>
<td>0.14</td>
<td>-0.25</td>
<td>0.10</td>
<td>-0.05</td>
</tr>
<tr>
<td>2</td>
<td>Ease of Use</td>
<td>-0.17</td>
<td>0.16</td>
<td>0.00</td>
<td>-0.17</td>
<td>0.08</td>
</tr>
<tr>
<td>3</td>
<td>Productivity</td>
<td>-0.28</td>
<td>0.08</td>
<td>0.20</td>
<td>-0.28*</td>
<td>0.14*</td>
</tr>
</tbody>
</table>

* Significant (p<.05)

Table 10 - Comparison of questionnaire standardized factor scores
5.3 Case Study Results: Structured Group Systems Scenarios

The primary purpose of these case studies was to gather the information needed to refine the preliminary collaborative scenario elicitation methodology and contextual scenario data model (sections 4.2 – 4.3) developed to respond to the research questions:

What group process and facilitation techniques are appropriate for each collaborative tool used to elicit contextual scenarios from user groups?

What information should be included in the contextual scenarios elicited directly from user groups?

The collaborative tool used during these case studies was GroupSystems Group Outliner. Therefore, an additional goal of these case studies was to respond to the question:

Can general-purpose GSS tools be used for collaborative scenario elicitation or is a special-purpose tool needed?

These case studies all used a template to prompt for the required scenario information because of the completeness problems identified during the Phase I experiment (section 5.2). As a result, the case studies were also used to refine the response to the question:

What format should be used for contextual scenarios and how does that format impact scenario quality and productivity of user definition of scenarios?

The proposed collaborative scenario elicitation methodology, contextual scenario data model, and Group Outliner use with a scenario template were evaluated during several DESCIM scenario and use case sessions facilitated by University of Arizona researchers. The evaluation specifically focused on the Waste and Hazardous Material Management use case sessions because each were representative of key approaches.
evaluated during these case studies and had follow-on sessions where the revised
template and elicitation methodology could be further evaluated.

5.3.1 Background

The Waste and Hazardous Material Management meetings were sponsored by the Defense Environmental Security Corporate Information Management (DESCIM) Program Management Office (PMO) as part of its responsibility to develop a Hazardous Substance Management (HSM) target system. A comprehensive, but somewhat outdated, activity model had been previously developed. The Waste and Hazardous Material Management functional areas represented separate portions of that model.

5.3.2 Waste Management Meetings

Two face-to-face meetings were sponsored by DESCIM in January and March of 1998 to develop scenarios for the waste management portion of the HSM target system. The following meeting descriptions are structured based on the primary components of the GSS research model including the meeting tasks, group, context, technology, process and outcome. The second meeting description is abbreviated and only addresses differences with the first meeting.

5.3.2.1 Waste Management Meeting 1

Task. The primary meeting task was to define scenarios for waste management activities A3.1 – A3.10 and their children.

Group. Twelve DOD employees including environmental and waste management specialists representing the Army, Navy, Air Force, Department of Defense (DOD), and
National Guard Bureau (NGB), DESCIM PMO representatives, and HSM programmers participated in the meeting. The meeting was facilitated by a CMI researcher.

**Context.** The face-to-face meeting was conducted using a permanent GroupSystems meeting facility owned by the Navy.

**Technology.** A general-purpose GSS, GroupSystems, was the primary software used to support the meeting. The facilitators and all meeting participants used individual personal computers to access GroupSystems. Each participant's GroupSystems workstation was directly connected via a local area network (LAN) to the meeting room's local GroupSystems server.

**Process.** The existing waste management activity hierarchy was reviewed and then used as the starting point for definition of the scenarios. The activity hierarchy was imported to Group Systems Group Outliner. Scenario template items were added as children for each leaf-node activity. Template items were defined by HSM technical personnel and reflected a use case orientation. Items included: objective, preconditions, successful end conditions, failed end conditions, primary users, secondary users, trigger, scenario (numbered steps), and exceptions. Users provided input by adding comments for each template item in the outline. The group then reviewed these comments with the facilitator modifying the template item in the outline to include the group result.

**Outcome.** The group did not achieve all of its meeting goals. Scenarios for activities A3.1 – A3.5 and their children were completed and reviewed by the group. Users also provided input for scenarios for the remaining activities, but did not have time to review them to reach consensus. End-of-session feedback from the users indicated
they were somewhat frustrated with the slow pace of the scenario definition and review. They also questioned the benefit of some of the more technical information included in the template. The facilitator agreed that some template items were more technically-oriented than desirable for a user scenario elicitation meeting. Feedback on GroupSystems was generally positive, but the process of adding template items and group results to the outline was tedious for the facilitator. In addition, one user expressed concern that the basic waste management process had never really been “flowcharted.”

5.3.2.2 Waste Management Meeting 2

**Task.** The purpose of the second meeting was to complete definition of scenarios for waste management activities A3.1 – A3.8 and their children.

**Group.** Fourteen DOD employees including environmental and waste management specialists representing the Army, Navy, Air Force, Department of Defense (DOD), and National Guard Bureau (NGB), and DESCIM PMO representatives participated in the meeting. HSM programmers did not attend this meeting. The same CMI researcher facilitated the meeting.

**Context.** The meeting was conducted face-to-face in a permanent GroupSystems meeting facility owned by the Navy.

**Technology.** GroupSystems Group Outliner was the primary software used.

**Process.** The scenario definition process was basically the same as the one used in the earlier meeting. The only major changes were to a user-focused template more closely aligned to the contextual scenario data model proposed in section 4.3. Template items included: objective, precondition, ‘who does this?’, trigger, scenario, exceptions,
and related issues. Successful and failed end conditions were dropped. Primary and secondary users were replaced by ‘who does this?’ ‘Related issues’ was added to capture pending changes or other important user information related to the scenario.

**Outcome.** The group achieved all goals and completed scenarios for waste management activities A3.1 – A3.8 and their children. End-of-session feedback was much more positive because of this and the improved scenario template.

### 5.3.3 Hazardous Material Management Meetings

Three face-to-face meetings were sponsored by DESCIM in January, March, and May of 1998 to develop scenarios for the hazardous material management portion of the HSM target system. The following meeting descriptions are structured based on the primary components of the GSS research model including the meeting tasks, group, context, technology, process and outcome. The second and third meeting descriptions are abbreviated and only address differences with the preceding meetings.

#### 5.3.3.1 Hazardous Material Management Meeting 1

**Task.** The purpose of the first meeting was (1) to revise the activity hierarchy for activity A1, Management Hazardous Material, and (2) to define scenarios for activity A1 and its children.

**Group.** Twenty DOD employees including environmental and hazardous material management specialists representing the Army, Navy, Air Force, Department of Defense (DOD), and National Guard Bureau (NGB) and DESCIM PMO representatives participated in the meeting. The meeting was facilitated by a CMI researcher.
**Context.** The face-to-face meeting was conducted using a permanent GroupSystems meeting facility owned by the Navy.

**Technology.** A general-purpose GSS, GroupSystems, was the primary software used to support the meeting. The facilitators and all meeting participants used individual personal computers to access GroupSystems. Each participant’s GroupSystems workstation was directly connected via a local area network (LAN) to the meeting room’s local GroupSystems server.

**Process.** The majority of the meeting was spent reviewing and revising the existing hazardous material management activity hierarchy. GroupSystems Group Outliner was used to support this process. The first comment for each activity in the hierarchy was a definition of the activity. The facilitator then added a blank scenario template as a second comment for all leaf-node activities. As with the first waste management meeting held at approximately the same time, the template items reflected a technical, system use case perspective. Items included: objective, module, preconditions, successful end conditions, failed end conditions, primary users, secondary users, trigger, scenario (numbered steps), extensions and variations. Users added comments for each leaf-node activity to describe the different template items. Results of group review were consolidated from those comments and recorded immediately following the blank template in the comments for each leaf-node activity.

**Outcome.** As stated previously, the group spent the majority of the meeting revising the activity hierarchy. Scenarios were completed for activities A1.1.1 – A1.1.6. Only limited input was provided for the other hazardous material management activities.
End-of-session feedback from the users indicated they were somewhat confused about the development process being used and how scenarios contributed to that process. As with the first waste management session, the facilitator felt that some template items were more technically-oriented than desirable for a user scenario elicitation meeting.

5.3.3.2 Hazardous Material Management Meeting 2

**Task.** The purpose of the second meeting was to complete definition of scenarios for activity A1, Management Hazardous Material, and its children.

**Group.** Fifteen DOD employees including environmental and hazardous material management specialists representing the Army, Navy, Air Force, Department of Defense (DOD), and National Guard Bureau (NGB) and DESCIM PMO representatives participated in the meeting. The meeting was facilitated by the same CMI researcher.

**Context.** The face-to-face meeting was conducted using a permanent GroupSystems meeting facility owned by the Navy.

**Technology.** GroupSystems Group Outliner was the primary software used.

**Process.** The basic scenario definition process was similar to the one used during the first meeting. The scenario template was revised to drop technical items. Remaining items included: objectives, preconditions, ‘who does this?’, trigger, scenario, exceptions, and related issues. The facilitator added the revised scenario template as a comment for all leaf-node activities. Users added comments for each leaf-node activity to describe the different template items. Results of the group review were consolidated from those comments and recorded immediately following the blank template in the comments for each leaf-node activity.
**Outcome.** Scenarios were not completed for all child activities of A1. End-of-session feedback from the users indicated they were frustrated with the slow group review process caused by extensive discussions of the differences between services. Others expressed concerns that some important details were left as assumptions and not fully documented. The facilitator also observed that some users felt more comfortable drawing graphical diagrams of the sequence of scenario actions before documenting them in GroupSystems.

5.3.3.3 Hazardous Material Management Meeting 3

**Task.** The purpose of the third meeting was to complete definition of scenarios for activity A2, Hazardous Material Use, and its children.

**Group.** Seventeen DOD employees including environmental and hazardous material management specialists representing the Army, Navy, Air Force, Department of Defense (DOD), and National Guard Bureau (NGB) and DES CIM PMO representatives participated in the meeting. The meeting was facilitated by the same CMI researcher.

**Context.** The face-to-face meeting was conducted using a permanent GroupSystems meeting facility owned by the Navy.

**Technology.** GroupSystems Group Outliner was the primary software used.

**Process.** The basic scenario definition process was similar to the one used during the second waste management meeting. The scenario template was the same one used during the second waste and hazardous material management meetings. The facilitator copied the blank template items as children of the leaf-node activities in the outline. Users added their input as comments for each template item. The facilitator copied
results of group review from the comments to the appropriate template items in the outline.

**Outcome.** Scenarios were completed for all child activities of A2. Scenarios for some activities identified specific actions required to accomplish the activity while others simply included a list of possible scenarios with no detailed action listing. End-of-session feedback from the users was more positive than earlier meetings, primarily because the meeting goals were accomplished. Feedback on GroupSystems was positive, although the facilitator did feel that the process of adding the scenario template and results of group review to the outline was tedious.

5.4 Phase I Results Summary

A detailed discussion of the Phase I results is included in Chapter 6. However, the key results are summarized here as motivation for the Phase II research.

5.4.1 Unstructured Group Systems Scenarios Results Summary

Results of the unstructured GroupSystems scenarios meeting showed that users can easily define scenarios using GroupSystems which provide a rich picture of the problem domain – providing clear encouragement for continuation of this scenario research. However, the results also showed that when used in an undisciplined manner, scenarios can be incomplete, unfocused, fail to provide critical information, and extremely inefficient to define and analyze. These results supported users' request for more detailed guidelines than those provided by the collaborative scenario elicitation framework and methodology presented in sections 4.1 – 4.2. The contextual scenario data model documented in section 4.3 was a first step in that direction. The Phase I
experiment provided the next step by increasing understanding of how scenario format impacted the completeness of the data included in scenario descriptions.

5.4.2 Experimental Results Summary

The experimental results clearly showed completeness problems for all formats. The lack of on-screen information prompts for all formats (including the structured formats) may have contributed to this problem. However, the lower productivity of the structured formats indicated that any structure added to increase completeness must be carefully tailored to support scenario elicitation while limiting its impact on productivity.

Another possible cause of the completeness problems was that subjects were asked to provide all the information at once, which may have caused an information overload problem. Most subjects did well identifying the main actions, but may have done better with total action completeness if an iterative definition process had been used which requested actions, then action details, and finally exceptions and alternatives.

Although scenario completeness was a problem, usability of the collaborative tool used to support both the unstructured GroupSystems scenario sessions and the experiments was not. Users continued to rate GroupSystems Group Outliner very high on ease of use. Therefore, the next step in the research was to determine whether an appropriate level of structure could be defined which increased scenario completeness while continuing to take advantage of the usability of GroupSystems Group Outliner.

The purpose of the structured GroupSystems scenario sessions was to explore these questions and to continue the move towards a more detailed methodology for collaborative scenario elicitation.
5.4.3 Structured Group Systems Scenarios Results Summary

Results of the structured GroupSystems scenario meetings identified both the strengths and weaknesses of using GroupSystems and the proposed methodology for collaborative scenario elicitation.

Early sessions used a more system-focused scenario template including such items as successful end conditions and failed end conditions. These types of items were often difficult for users to complete or duplicated information already provided, e.g., in objective. This feedback supported the strong business/user focus (vs. system focus) of the contextual scenario data model proposed in section 4.3.

Procedures for using a structured scenario template with Group Outliner also evolved over the various sessions. To provide the prompts required to ensure scenario completeness and the visibility requested by users, facilitators were required to copy a scenario template and group review results to the outline for each scenario. Although this process was awkward and time-consuming for the facilitator, it did seem to provide users with the flexibility and visibility they were looking for during the scenario elicitation process and it did increase scenario completeness.

Users also occasionally expressed the need to graphically diagram the flow and sequence of actions before documenting scenario actions, especially for complex scenarios with parallel processing and multiple decision points. Since no graphical capability is included in Group Outliner, users sketched out action sequences on white boards as needed for complex scenarios. After the group had agreed on the action sequence, they were generally able to document the process in Group Outliner.
5.4.4 Motivation for Phase II Research

In summary, the Phase I results indicate the need for a detailed and iterative collaborative scenario elicitation methodology with appropriate prompts at the scenario and action level to improve completeness. The proposed collaborative scenario elicitation methodology was modified to the maximum extent possible to provide these capabilities (see section 6.3). The primary advantages of using this methodology with GroupSystems Group Outliner is that it provides easy-to-use collaborative support for capturing rich, textual scenario descriptions. Those advantages, however, come at a cost. Group Outliner provides only limited and awkward template support. But the Phase I results show that there are significant completeness problems when a template is not used. Group Outliner also provides only weak support for an iterative methodology, again because of its limited template support. It is difficult enough for facilitators to add the entire template to each leaf node. This process would be even worse if they had to add additional template information to each leaf node at each step of the incremental scenario elicitation process. Finally, the lack of an integrated graphical view of activity flow, sequence, and decision logic slowed users when they tried to document complex scenarios.

The purpose of the next phase of the research was to address these limitations through development of a special-purpose, integrated structured textual scenario and graphical process modeling prototype designed to support an iterative collaborative scenario elicitation methodology.
PHASE II: Integrated Scenario and Process Modeling Using SPA

5.5 SPA Systems Development

The goal of this research task was to design and develop a new system that provided better support for collaborative scenario elicitation than existing GSS. The Collaborative Distributed Scenario and Process Analyzer (ColD SPA) prototype evolved from the lessons learned in Phase I and integration of two funded research prototypes: (1) Process Modeler funded by the Air Force, and (2) Scenario Modeler funded by the DESCIM PMO. Results of this research are presented in the following sections for each of the five stages of the systems development research methodology (Nunamaker et al., 1991a) used to guide the task.

5.5.1 Construction of a Conceptual Framework

The CSEM scenario framework, collaborative scenario elicitation methodology, and contextual scenario data model developed and revised during Phase I served as the overarching conceptual framework for this task. This framework and the Phase I results served as the starting point for investigating the required functionality for the SPA prototype as discussed in section 5.5.1.1. Next, the business process modeling literature was studied to identify alternative approaches for satisfying those requirements (section 5.5.1.2). Results of these first two steps were analyzed to develop an integrated scenario and process modeling framework for the SPA prototype (section 5.5.1.3). Finally, section 5.5.1.4 provides a brief overview of the processes and procedures that were used to control the SPA systems building process.
5.5.1.1 Core SPA Prototype Functionality

The lessons learned from the Phase I research were combined with previous experiences in developing general-purpose and modeling GSS to identify five core requirements for SPA system functionality.

1. SPA must include all GroupSystems Group Outliner functionality. Users consistently rated Group Outliner very high on easy-to-use, collaborative support for hierarchically decomposing and commenting on outline elements. Therefore, Group Outliner served as the baseline for those capabilities.

2. SPA must implement the essential features of modeling GSS including a user-comprehensible interface that focuses on information the user can provide. Lessons learned from development and evaluation of Activity Modeler and Group Data Modeler were used to guide implementation of this capability.

3. SPA must be application-specific and include prompts for scenario and action data included in the contextual scenario data model defined in section 4.3.

4. SPA must provide flexible support for an iterative methodology for collaborative scenario elicitation similar to the one described in this research. Support must be flexible because this methodology is still being refined and is generally adapted for the specific domain or project under study.

5. SPA must include an integrated graphical process modeling capability. Since this area was new to the SPA project team and had not been investigated during the Chapter 2 literature review, the team did not know how to
implement this capability. Therefore, business process modeling approaches were studied to identify alternative methods for providing this capability.

5.5.1.2 Business Process Modeling

A process model can be described as "an abstract description of an actual or proposed process that represents selected process elements that are considered important to the purpose of the model and can be enacted by a human or a machine" (Curtis, Kellner, & Over, 1992, p. 76). Process models can be developed from different perspectives such as functional, informational, behavioral, or organizational perspectives (Curtis et al., 1992). In MIS, some of the earliest process models took a functional perspective. For example, data flow diagrams represented data transformation processes and the information flows between those processes (Conger, 1994). Business process reengineering and other process improvement initiatives have focused on the behavioral and organizational perspectives for modeling general business processes. These business process models include information such as process sequence, decision criteria, and who performs the process — the same sort of information including in the contextual scenario data model. The purpose of this task, therefore, was to study alternative graphical business process modeling techniques which could be used to satisfy the essential SPA functional requirements defined in section 5.5.1.1.

Recently, Butler et al. (1999) argued that information systems enable, and often enforce, certain work processes and that therefore the design of those systems should be explicitly connected to the design of work processes. They go on to state that "one of the best specified languages for representing work processes in the physical world is
IDEF3" (Butler et al., 1999, p. 41). The IDEF3 Process Description Capture Method was developed by Knowledge Based Systems, Inc. (KBSI) for the U.S. Air Force Armstrong Laboratory as part of its Integrated Computer-Aided Manufacturing (ICAM) Definition (IDEF) of methods program (Mayer et al., 1992). The basic process element in IDEF3 is called a “Unit of Behavior (UOB).” The definition of a UOB and the other IDEF3 process description constructs are summarized in Table 11.

<table>
<thead>
<tr>
<th>IDEF3 Construct</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scenarios</td>
<td>Provide the scope and context for a process description.</td>
</tr>
<tr>
<td>Unit of Behavior (UOB)</td>
<td>Describe “things that happen in the world” – Function, Process, Scenario, Activity, Action, Operation, Event, Decision, Procedure</td>
</tr>
<tr>
<td>UOB Elaboration</td>
<td>Textual Description of UOB, Objects that participated in the process (agent, affected, participated, created, destroyed). Facts/Assertions, Constraints</td>
</tr>
<tr>
<td>UOB Decompositions</td>
<td>To increase level of detail or provide different views.</td>
</tr>
<tr>
<td>Junctions (split or join)</td>
<td>Specify the logic of process branching. Simplify capture of the sequencing and timing relationships between multiple process paths.</td>
</tr>
<tr>
<td>(and, or, exclusive or)</td>
<td></td>
</tr>
<tr>
<td>(synchronous, asynchronous)</td>
<td></td>
</tr>
<tr>
<td>Links</td>
<td>Denote significant constraint relationships between UOBs. Draw attention to important relationships. Types - temporal, logical, causal, natural, conventional</td>
</tr>
<tr>
<td>Link Specification Document</td>
<td>Link textual description, source, destination, objects participating in the relationship, facts &amp; constraints.</td>
</tr>
<tr>
<td>Precedence Links</td>
<td>Express simple temporal precedence or an enablement relationship.</td>
</tr>
<tr>
<td>Relational Links</td>
<td>No predefined semantics. User-defined links highlighting the existence of a relationship, e.g., before, during, after, meets, overlaps, finishes, starts, causes, enables, triggers.</td>
</tr>
<tr>
<td>Object Flow Links</td>
<td>Highlight the participation of an object in two UOB instances.</td>
</tr>
<tr>
<td>Referents (UOB, Junction, Object, Elaboration, OSTN, Scenario, Note, Go-to)</td>
<td>Allow references to other IDEF3 constructs or provides additional information to enhance model.</td>
</tr>
</tbody>
</table>

Table 11 – IDEF3 process description constructs (Mayer et al., 1992)
The graphical representations for the primary IDEF3 elements are shown in Figure 15. Junctions can be used to ‘join’ two or more links together or to ‘fork’ out to two or more processes. Synchronous junctions can be indicated by added a bar on the left side of the box for synchronous joins or on the right side for synchronous forks.

Asynchronous junctions do not include the bar. A revised version of the IDEF3 report (Mayer et al., 1995) slightly modifies these symbols, but it includes the same basic constructs.

As can be seen from the preceding discussion, the IDEF3 process description method provides a comprehensive business process modeling capability. However, personal experiences with IDEF3 indicate that some portions such as the different types of synchronous and asynchronous junctions may be too complex for users to quickly grasp. Therefore, while the capabilities of IDEF3 served as input to the SPA requirements process, several other alternatives were investigated in the search for a simplified, easy-to-use graphical process modeling technique. Petri-nets (Zurawski & Zhou, 1994) were eliminated from consideration since they were even more complex than IDEF3 (Davis, 1988). In contrast, several Unified Modeling Language diagrams seemed to have potential as a user-comprehensible representation.
The Unified Modeling Language (UML) was adopted by the Object Management Group as the standard object-oriented modeling language (Booch et al., 1999). The UML standard defines several diagrams for behavior modeling that could possibly be used to graphically portray business processes. UML's Sequence Diagram is commonly used to document use case scenarios. It is a type of interaction diagram which shows the objects participating in a scenario and a time-sequenced list of messages the objects exchange to accomplish the scenario goal (Booch et al., 1999). The Sequence Diagram's focus on objects and messages increases its complexity beyond what is desirable for a simplified, graphical process modeling technique targeted to non-analyst users. UML's State Diagrams are also too complicated for non-analyst users to independently create without significant training and assistance from analysts or modelers. These technically oriented diagrams are more suitable for use by analysts and developers than by untrained users.

In contrast, UML's Activity Diagram is a greatly simplified version of the UML state diagram which can be easily used for business process modeling (Fowler & Scott, 1997). It is designed to provide a dynamic view of a system or process which shows the flow from activity to activity (Booch et al., 1999). Officially it is defined as "a special case of a state diagram in which all or most of the states are action states and in which all or most of the transitions are triggered by completion of actions in the source states" (Rational, 1997). The basic constructs in the Activity Diagram are defined in Table 12. These definitions seem rather technical, however, they are relatively simple in practice. Activity and action states are directly comparable to IDEF3's UOB and can be used to represent a business process and the steps in the process, respectively.
### Activity Diagram Construct

<table>
<thead>
<tr>
<th>Activity Diagram Construct</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action State</td>
<td>Represents the execution of an atomic action. It's a simple state with an entry action whose only exit transition is triggered by the implicit event of completing the entry action.</td>
</tr>
<tr>
<td>Activity State</td>
<td>Represents the execution of a non-atomic sequence of steps that has some duration. It's a hierarchical action where an associated sub-activity model is executed.</td>
</tr>
<tr>
<td>Pseudo State</td>
<td>Abstraction of different types of nodes in a state machine graph which function as transient points in transitions from one state to another, such as branching and forking, simple transitions, or object flows.</td>
</tr>
</tbody>
</table>

Table 12 - UML Activity Diagram constructs
(Rational, 1997)

The user does not even have to be concerned with the difference between activity and action states since they are both represented with the same graphical symbol (Figure 16). The graphical symbols for pseudo-states are also much simpler than its definition would imply. The decision, fork, and join pseudo-states are simply different types of branches that can occur in process flows and are comparable to IDEF3 junctions.

Transitions and object flows represent two types of links between activity or action states.

![Figure 16 - UML Activity Diagram graphical symbols](Rational, 1997)
In summary, both IDEF3 and the UML Activity Diagram satisfy SPA's requirement for a graphical business process modeling capability. The basic elements of each are directly comparable. However, because the Activity Diagram has a simpler representation and can be linked to the other UML diagrams, it was selected as the initial graphical process modeling representation. The goal for future versions of SPA is to support both representations and to allow facilitators to select the Activity Diagram or the IDEF3 representations as part of the session configuration process.

5.5.1.3 Integrated Scenario and Process Modeling Framework

The conceptual framework, methodology, and data model developed during Phase I were specifically designed to support scenario elicitation. As can be seen from the preceding discussion on business process modeling, scenario and process modeling have very different constructs and terminology. These differences had to be reconciled before an integrated scenario and process modeling capability could be included in SPA. One of the first major challenges was to define what the basic components that are successively decomposed to create a SPA model are and what they should be called.

The CSEM specifies that business activities and functions should be decomposed into sub-activities/functions. For each of the sub-activities/functions with system implications, users identify business scenarios which provide concrete examples of how users currently accomplish those functions. Business scenarios are then decomposed into the specific steps or actions that users perform during those scenarios. Using CSEM as a guideline, the basic components could be called activities, functions, sub-activities, scenarios, steps or actions — depending on the level of the decomposition.
From a graphical process modeling perspective, processes are decomposed into sub-processes, which are further decomposed into tasks or actions. The UML Activity Diagram decomposes activity states into action states. Again, the name of the component depends on the level of decomposition. The IDEF3 Process Description Capture Method eliminates this dependency by calling all the basic components at all levels "Units of Behavior" which represent a function, process, scenario, activity, operation, decision, action, event, or procedure (Mayer et al., 1992).

While the IDEF3 concept provided the simplicity desired for the SPA prototype, it was not clear whether all these different components could be consolidated into one generic element. For example, the contextual scenario data model defined in section 4.3 requires different data requirements for scenarios and actions. A comparison of data requirements for all levels for both scenarios and processes showed that there was enough overlap to support consolidation into a single basic component with standardized data elements. The other advantages of consolidating into a single component are:

- The abstraction problem (e.g., where one person’s scenario is another’s step, or where a scenario step in the requirements phase becomes a use case during design) does not have to be dealt with – all components will be the same.
- The user does not have to understand the differences between the different concepts and names.

These advantages, when combined with the results of the data analysis, resulted in the decision to combine all SPA components at all levels of decomposition into a single generic component. The next question was what to call this generic component. The
IDEF3 term, "unit of behavior," seemed too awkward. After extensive discussion with SPA team members of all alternatives from both CSEM and IDEF3, the term "process" was selected as the generic term to represent all components in the SPA model, regardless of decomposition level. Therefore, the term "process" is used whenever the component being referred to could be an activity, function, sub-activity/function, process, sub-process, scenario, task, action, or step. The more specific term will be used when appropriate.

The decision to define a single generic "process" for the SPA system provided the framework required to continue development of an integrated scenario and process modeling capability. The overview of the development process presented next completes the discussion of the first stage of the systems development research methodology.

5.5.1.4 SPA System Building Process

The SPA prototype was developed using a collaborative, incremental, rapid prototyping approach to systems building. Development of SPA was a team effort, with each team member having clearly defined roles and responsibilities. The author served as primary researcher, project leader, and requirements engineer. Other team members served as system architect, interface designer, and programmers. Critical SPA requirements, priority, design, and development decisions were jointly discussed during weekly project team meetings. All team members participated in the discussion, although the individual responsible for a particular area had final decision authority. Increments were time-constrained to coincide with research contract deliverables. Understanding the strengths and weaknesses of the prototyping process used was critical
to success as a project leader and to the on-time delivery of the first versions of the prototype to the research sponsors.

5.5.2 Development of a SPA System Architecture

Development of the SPA system architecture was a team effort. The author developed the initial list of constraints, development objectives, and system requirements. The system architect and programmers then proposed a system architecture to satisfy that list. The entire team worked together under the author's leadership to identify additional constraints, objectives, and requirements and to negotiate and finalize the system architecture. The results of these deliberations are summarized in the following sections.

5.5.2.1 Constraints

The primary constraint was that the SPA system architecture fully support all core SPA functional requirements defined in section 5.5.1.1. Since SPA is an interactive system, an additional constraint was that the system must be designed to maximize usability from the user's (vs. facilitator's or developer's) perspective. The system must be easy-to-use and require minimal user training. In addition, system response times must be kept to a minimum so that they do not negatively impact the user's ability or willingness to use the SPA prototype. Once the decision was made to develop SPA as a web-based distributed system, the system response time constraint was extended to include minimizing network response time which in turn required that the system be designed to minimize network traffic.
The proposed system architecture must also not require any new hardware or software (e.g., browsers) purchases for the clients. Purchases for servers should be low, but could be negotiated. Assumptions on the capabilities of client machines were guided by a combination of current technology and prices and available systems in CMI and its GroupSystems meeting rooms. In general, it was assumed that clients would have access to Pentium-class PCs which at least met Windows system memory requirements, had Internet access, and supported a screen resolution of 800 x 600 or better.

Finally, CMI research contract requirements and deadlines for SPA must be met or exceeded. All requirements, design, and development decisions must always consider the impact on the Air Force and DESCIM contracts before being finalized.

5.5.2.2 Development Objectives

Development objectives were driven both by the requirements of the systems development research methodology and by the SPA system architect who also served as CMI’s programming director. The first objective was that the “experience and design expertise gained from building the system can be generalized for future use” (Nunamaker et al., 1991a).

The system architect also was very interested in exploring new technology and proposed a development objective of exploring the use of Java to create a web-based SPA prototype that would support distributed meetings. Since the original SPA concept was to develop SPA on top of GroupSystems as Group Data Modeler had been, this objective required significant negotiation between the functional and technical members of the project team. While all team members were excited about the potential for
supporting distributed meetings, the approval to develop the SPA client software in Java was given only after the system architect assured functional personnel that all core SPA functional requirements and constraints could be fully met by a Java client.

The final development objective was to maximize code reusability, especially for the server through reuse of the server software previously developed for the CoReview prototype developed to support distributed code reviews and other structured evaluations.

5.5.2.3 Measurable Requirements

The development of measurable requirements was an iterative process of discovery, analysis, refinement and negotiation which started with the core SPA system functionality defined in section 5.5.1.1 and ended with the detailed requirements needed to guide system design and development. The requirements process was not restricted to this stage of the research. Rather, it spanned multiple iterations through each of the stages of the systems development research methodology. A copy of the current SPA requirements list which resulted from this process is included in Appendix E. This list is a living document which continues to be used to guide SPA development and testing.

Another critical part of the requirements process was the negotiation of which requirements would be included in the first version of the prototype vs. deferred to future versions. Contract requirements and deadlines drove much of the negotiation process. Only requirements which provided core scenario and process modeling capabilities which could be implemented within the specified deadlines were included in the first version. Many important requirements were reluctantly deferred by the project leader to
future versions solely because of the limited programming resources and time available. Other requirements were automatically deferred because they were not part of the core functionality, represented desired enhancements, or were identified only after observing use of the initial SPA prototype. Results of this requirements negotiation and allocation between versions were also documented in the requirements list.

5.5.2.4 Architecture Overview

The SPA prototype uses an Internet-based client-server architecture. It eventually will run using any Java 1.1 compatible browser on any kind of client machines. However, because of the varying degree of Java 1.1 support provided by current Internet browsers, the initial SPA prototype requires use of the latest version of Microsoft's Internet Explorer browser (version 4.0 with service pack 1). There are three main components of the Collaborative Scenario Modeler: the Server component, Client component, and a supporting Administrative Module. These components are described in more detail in paragraph 4.1. An overview of the concept of execution is provided in paragraph 4.2, with a description of the database architecture included in paragraph 4.3.

Server. The server component was developed in Delphi to store and maintain data in support of several different Internet-based applications including the SPA. Data for the process model is stored on a Windows NT server as Paradox tables, and are retrieved and modified by the client through a Java applet. The server processes requests from client programs and modifies and/or retrieves data from its master database based on those requests. The server is also responsible for broadcasting messages from one client to other active clients.
**Client.** The SPA client component is implemented as a Java applet. The Java applet is downloaded from the NT server when the user logs in. The client-side Java applet implements a graphical user interface and handles data integrity checking. The Client Component can be used by administrators, session facilitators, as well as regular users. While users have only the privileges given to them by the session facilitator, the session facilitator has additional administrative capabilities including:

- Defining or changing the session configuration by defining what panels and prompts to display.
- Defining of changing user rights by defining whether users have rights to add, modify, delete, or simply view various components of the model.

**Administrative Component.** This component is primarily an administrative utility and supports two different types of users: system administrators and session owners/facilitators. Administrators are specially-designated users who have all privileges including those required to create other administrators, maintain user lists/passwords, to designate users as session owners/facilitators, as well as to create actual SPA sessions. Session owners/facilitators will use the Administrative module to:

- Create SPA sessions
- Create users & passwords
- Add users to a session
- Delete users from a session
- Add additional owners/facilitators to a session
- General session file management and clean-up
**Concept of Execution.** In the SPA prototype, data for a process model are kept both on the server and on all clients which are actively connected to the server. The server keeps the master copy of the data in the Paradox tables. When a user logs in to the server, the client receives a copy of the current Java applet. The user then selects which session to join. Data for that session is then downloaded from the server into the client’s cache memory. Whenever a client updates the process model, the update request is sent to the server. If the server permits the update, it will store the update in the server master database and will broadcast the update to all active clients (including the client requesting the update). Clients then update their copy of the data in memory and update the user’s display of the process model as required.

**5.5.3 Analysis and Design of SPA**

SPA analysis and design was a team effort. The author continued the requirements analysis, developed the logical database design, and specified user interface requirements. The user interface specialist developed alternative interface designs to meet those requirements and then worked in collaboration with the author and other team members to finalize the design. Class definitions and allocations of system functions were done by the system architect and programmers and is not discussed here.

**5.5.3.1 Logical Database Design**

The logical data model was developed by adapting the proposed contextual scenario data model (section 4.3) based on the integrated scenario and process modeling concepts described in section 5.5.1.3 and adding the new data items that were identified during development of the integrated concept. The data model was then revised further
to take advantage of the flexibility provided by the proposed user interface design (section 5.5.3.2). A copy of the final normalized logical model is included in Appendix E.2. Definitions of entities and data items follow in Appendix E.3. The primary entity in the model is the process. All other entities are either directly or indirectly related to process.

5.5.3.2 User Interface Design

SPA's user interface includes three primary data capture components: a hierarchical structure to capture process names and decompositions (e.g., scenarios and actions), a text capture area to allow descriptions of various aspects of the process, and a graphical process model for defining and displaying process sequence. A brief overview of each of these components is provided next.

- **Process Hierarchy** – A hierarchical (tree) view shows how a process is decomposed into sub-processes and then further into individual actions required to accomplish each process/sub-process. Each node in the process hierarchy is given a short action name of the form “verb + object,” e.g., “Request Travel Orders.” This component is shown on the left side of Figure 17 and Figure 18.

- **Process Information Panels** – A series of free-format text and fixed data entry panels, each panel capturing a different type of information about a process, e.g., its description, resource requirements, triggering event, or actor responsible for accomplishing the action. A set of panels is associated with each process (node) in the process hierarchy. Panels are displayed for the
process currently highlighted in the hierarchical (tree) view. They are designed so that the facilitator change the visibility or add new free-format text panels when required to support an iterative process. An example of a free-format text information panel is shown on the right side of Figure 17.

![Figure 17 - SPA process hierarchy and information panels](image)

- **Graphical Process Model** – A graphical view of the process model showing sequence and flow of the actions. Each process node in the hierarchical (tree) view is automatically shown on the diagram Participants link nodes together to reflect action sequence and decision flow. The SPA prototype automatically
redraws the diagram to reflect the action sequence identified by the participants. The graphical view is displayed on the right side of Figure 18.

![Figure 18 - SPA process hierarchy and graphical view](image)

5.5.4 Building SPA

Teamwork continued during the systems building stage of SPA, but with the systems architect and programmers carrying the majority of the workload. The author's role was limited to project leadership, requirements engineering, and testing. An initial version of the SPA prototype was developed and delivered on-time. The use of Java and the differences in how each browser supports Java did cause various delays and frustrations for all team members during development. Testing was also difficult and
incomplete because of the complexity of testing a collaborative tool with lots of built-in flexibility designed to function over the Internet. Other insights gained from the systems building process will be discussed in Chapter 6.

5.5.5 Observing and Evaluating SPA

The final stage in the SPA systems development research was to observe and evaluate the system to determine compliance with the stated requirements, assess its impact on users and groups, and identify desired improvements. Results of this evaluation are presented in section 5.6. Lessons learned during systems development and these case studies and the revised CSEM scenario framework and collaborative scenario elicitation methodology are discussed in Chapter 6.

5.6 Case Study Results: Evaluating SPA

The initial Collaborative Distributed Scenario and Process Analyzer (ColD SPA) prototype was evaluated in a series of meetings conducted as part of a technology demonstration for the Air Force's Depot Operations Modernization Environment (DOME) research project. The purpose of the DOME Technology Demonstration was to showcase how the SPA prototype could be successfully used to improve an existing Air Force process. The process selected for the technology demonstration was the AFTO 107 Request for Technical Assistance process used by Air Force units to request depot-level repair assistance. The key participants were Mountain Home wing and logistics support personnel and Warner-Robins Air Logistics Center AFTO 107 managers and engineers involved in repair of the F-15 aircraft. Representatives of two Air Force major commands (MAJCOMs) and several other F-15 units also participated during the
demonstration or background interviews. The demonstration was conducted over a series of three face-to-face and distributed meetings to explore the strengths and weaknesses of the SPA prototype in each of these environments.

A detailed description of the technology demonstration is provided in the following sections. The first section briefly describes the meetings and interviews held throughout the DOME project to gather preliminary information about the AFTO 107 process. Each of the technology demonstration meetings is then discussed individually.

5.6.1 Background

During the initial phase of the DOME project, GroupSystems was used during face-to-face meetings to gather the initial information about the AFTO 107 Request for Technical Assistance process. At Mountain Home, CMI researchers facilitated a group of wing and logistics support personnel in the development of a detailed activity model of the wing AFTO 107 process. Similarly, CMI researchers worked with AFTO 107 managers and engineers at Warner Robins Air Logistics Center to develop an activity model of the depot AFTO 107 process. Activity Modeler was used to support collaborative model development during both these meetings. Interviews were also conducted with AFTO 107 personnel at the Air Combat Command to develop an informal model of the major command (MAJCOM) process and at Seymour-Johnson and Tyndall Air Force Bases to identify local differences in the wing AFTO 107 processes. In preparation for the technology demonstration, the author consolidated these models into a strawman AFTO 107 process model which became the starting point for the technology demonstration.
5.6.2 Air Force Meeting 1: Same Time, Same Place

The first phase of the DOME Technology Demonstration was a same time, same place meeting held at the University of Arizona on October 19 – 21, 1998. This meeting is described in detail by first providing an overview of the specific meeting task, group participants, meeting context, and technology, followed by a more detailed discussion of the meeting process and outcome.

Task. The primary task was to validate and refine the strawman AFTO 107 process model so that it accurately represented the process from the depot, MAJCOM, and wing viewpoints. To accomplish this task, participants defined/validated each required action, provided a short name and a complete description for those actions, determined the event which triggered each action to start, identified who was responsible for performing the action, and highlighted critical resources required to complete the action. After completing the process model, participants briefly identified problems with the AFTO 107 process to begin the process improvement task planned as the focus of the next meeting.

Group. Six Air Force personnel actively participated in the meeting including a logistician and engineer from Mountain Home AFB, the F-15 AFTO 107 manager and a reengineering specialist from Warner-Robins Air Logistics Center, and F-15 maintenance managers from two Air Force major commands. Participants were military and civilian employees with over 20 years experience, including an average of over 12 years with the AFTO 107 process. Most participants had limited familiarity with the DOME project, with only one participating in the earlier meetings. While all participants
used computers at least several times a day and had good experience with an Internet
browser, they reported limited to no GroupSystems or modeling experience.

Three CMI researchers (including the author) facilitated and observed the meeting.
Two of these researchers had participated in at least some of the earlier information
gathering meetings so they were familiar with the basics of the AFTO 107 process.

Context. This initial meeting of the technology demonstration was conducted as a
face-to-face meeting using a large permanent GroupSystems meeting facility at the
University of Arizona. The CMI technical staff provided the support required to ensure
that the hardware, software, and communication network operated smoothly throughout
the meeting. All systems were fully tested and problems resolved well before meeting
participants arrived.

All Air Force participants traveled from their home bases in Idaho, Georgia,
Virginia, and Texas to attend the meeting.

Technology. The facilitators and all meeting participants used individual personal
computers connected to a local area network (LAN), and through the LAN, to the
Internet. Three public screens were connected to the facilitators' workstations and other
display equipment. The primary software used during the meeting was the new SPA
prototype. SPA is designed as a client-server application for use in face-to-face and
distributed environments. The SPA server is located at the University of Arizona and
communicates with both local and remote clients via the Internet. Users downloaded the
SPA client software and a copy of the AFTO 107 process model using Microsoft's
Internet browser, Internet Explorer, version 4.01. As described in the following section,
GroupSystems was also used at the beginning and end of the meeting. Each participant’s GroupSystems workstation was directly connect via the LAN to the meeting room’s local GroupSystems server.

**Process.** The meeting began with the Air Force contracting office representative and CMI researchers providing a brief overview of the DOME project and the goals of the DOME technology demonstration. CMI researchers then presented the specific goals for this meeting and general meeting guidelines, and had all meeting participants introduce themselves to the group. After the introductions, CMI researchers requested that participants sign-in using GroupSystems Topic Commenter. The purpose of this exercise was twofold: (1) to develop an accurate participant roster, and (2) to introduce GroupSystems to the participants.

The SPA prototype was introduced next using a simple model which described the steps required to “Go TDY.” CMI researchers provided brief instructions on the three main components of the SPA:

- **Process Hierarchy** – A hierarchical (tree) view shows how a process is decomposed into sub-processes and then further into individual actions required to accomplish each process/sub-process.

- **Process Information Panels** – A series of panels, each panel capturing a different type of information about each individual process in the process hierarchy. Panels are displayed for the process currently highlighted in the hierarchical (tree) view.
Graphical Process Model – A graphical view of the process model showing sequence and flow of the actions. Instructions were also provided on how to add/modify nodes in the process hierarchy and add/modify textual information to the process information panels. Participants practiced these actions by modifying the sample “Go TDY” process model to more accurately reflect their actual (or desired) TDY process.

CMI researchers then used the SPA and led the group through a high-level review of the strawman integrated AFTO 107 process model. As described in the background section, this model was consolidated from the individual wing, depot, and MAJCOM activity models developed during earlier meetings and interviews. All participants could view the AFTO 107 process model on their individual workstations and on the public screen.

Participants then worked individually (or in pairs with the other representative from their organization) to refine the process model by adding new processes left out of the original model, modifying process names to improve clarity, and adding process descriptions whenever they were missing. The role names of individuals responsible for each process were also identified. Although all participants reviewed the entire model, participants primarily focused on improving their portion of the model (e.g., depot personnel worked on depot processes). CMI researchers assisted participants whenever they had problems with the SPA, but most participants seemed to find the tool easy to use. There was some confusion, however, on the difference between adding/editing actors on the master actor list and adding/deleting actors from a specific process. This
confusion highlighted a need to improve the SPA's user interface for the Actor Panel to clearly distinguish between these two actions. Regardless of these problems, all participants were able to assign actors to processes after some additional instruction.

The next step in the process, group review of the AFTO 107 process model to ensure consensus, was a challenge because of the number of processes and the several different categories of information captured for each process. CMI researchers decided to review the processes and their descriptions during the first pass through the model so that the group could initially focus on identifying all process actions correctly. During this review, true integration of the model occurred as the group worked together to ensure that every action by one location had an appropriate response at another location. Also, the group checked that valid responses were identified for each feasible action outcome (e.g., accept or reject). Several processes were added during the review based on this criteria. Missing or incomplete descriptions were also corrected. Processes were then reviewed to ensure that actors were correctly identified for all sub-processes and individual actions. Next, Triggers were validated for all high-level and non-sequential processes. The group did not bother to identify triggers for purely sequential actions when the trigger was obvious – simply the completion of the previous action. The graphical view of the AFTO 107 model was not used since it would have added significantly to the modeling time and the majority of the process actions were highly sequential and triggers generally provided sufficient clarification of the non-sequential actions. Finally, a few resources were identified when they represented a recurring problem area for the AFTO 107 process. The decision to review the model a category at
a time worked very well. Although the first pass through the model was rather slow, passes to review other categories of information went very quickly since all participants were very familiar with the model. There was also very little movement between categories during the review. For example, there were only a few times the process description had to be reviewed before an actor could be assigned. The majority of the time, the process name provided sufficient information.

After the group reached consensus on the AFTO 107 process model, the process hierarchy and process descriptions were copied to a GroupSystems Group Outliner session. Participants then used Group Outliner to identify AFTO 107 process problems and possible solutions to begin the process improvement brainstorming process. Finally, participants completed a GroupSystems Survey to get their feedback on the SPA, the quality of the AFTO 107 process model, and the approach used to develop the model.

**Outcome.** The final AFTO 107 process model included 88 process nodes. Descriptions were provided for all processes. Most sub-process and action level processes had more than one actor identified as being responsible for accomplishing the process. Multiple actors were identified when (1) several offices played key roles in accomplishing the process or, (2) the responsibility for a process varied at different Air Force bases. The process of identifying actors seemed to really emphasize this latter problem. As discussed in the previous section, triggers and resources were also identified when appropriate.

Developing the AFTO 107 process model also seemed to emphasize other problems as well. During the a short GroupSystems session at the end of the meeting,
participants rapidly identified 16 problem areas and recommended possible solutions for approximately half the problems.

Results of the survey conducted at the close of the session were very positive. Detailed survey results are included in Appendix F.1 and are briefly summarized here. The average participant assessment of the AFTO 107 process model ranged from 4.0 to 4.2 (1 – poor, 5 – excellent) on all quality characteristics. The average rating of the meeting approach was even higher at 4.5 on a similar 5 point scale. Participants did suggest that both the process model and the overall meeting could have been improved with broader participation by: (1) Air Force staff and (2) other Air Force units.

Participants' assessments of the SPA prototype were also extremely encouraging. Although participants strongly emphasized a few critically needed improvements, they still rated this first version of the SPA over 4 (out of a possible 5) on all functionality, usability and ease of use questions except recovering from errors. This last rating was expected since the SPA did have a few problems which required users to completely exit the tool, then restart the browser and reload the software and model whenever they occurred. Other recommended improvements included: (1) elimination of the continual redrawing/repositioning of the process tree hierarchy each time anyone modified it, (2) addition of common editing capabilities such as cut and paste, and (3) extending the current process name limit from 40 characters to at least 60. The first recommendation was implemented prior to the second meeting of the technology demonstration, which is described next.
5.6.3 Air Force Meeting 2: Same Time, Different Place

The second phase of the DOME Technology Demonstration was a same time, different place meeting held simultaneously at Mountain Home Air Force Base in Mountain Home, Idaho, and Robins Air Force Base in Warner Robins, Georgia, on November 6, 1998. The detailed description of this meeting follows the same format used to discuss the first meeting.

**Task.** The AFTO 107 process model was further refined during the second meeting of the technology demonstration using the same basic approach used in the earlier meeting. The primary task for this meeting, however, was to complete identification of AFTO 107 process problems and possible solutions, and to consolidate those solutions into a recommended process improvement action list.

**Group.** Eighteen Air Force personnel attended the kick-off for the meeting. However, only twelve actively participated throughout the entire meeting and remained to complete the questionnaire at the end of the meeting. A few participants left early, possibly because of the technology problems which slowed the start of the meeting. Other personnel left during the day because of other mission requirements, a common problem when meetings are held on-site. The Mountain Home logistician and engineer who participated in the first meeting were joined by eleven local maintenance, quality assurance, plans and scheduling, and other operational personnel at the Mountain Home AFB meeting location. The F-15 maintenance manager from one Air Force MAJCOM and the F-15 AFTO 107 manager and reengineering specialist from Warner-Robins Air Logistics Center were joined by an engineer from the F-15 project office and the DOME
contracting office representative at the Robins AFB meeting site. New military and
civilian participants generally had less experience than the first meeting participants,
dropping the average total experience for all participants to just under 20 years
experience and average AFTO 107 experience to 8.5 years. New participants were not
familiar with the DOME project. While most participants used computers at least several
times a day and had good to very good experience with an Internet browser, they all
reported limited to no GroupSystems or modeling experience.

The three CMI researchers from the first meeting also facilitated and observed the
second meeting, two at Mountain Home and one at Robins AFB.

Context. The second meeting of the technology demonstration was conducted as a
same time, different place meeting using the new GroupSystems meeting facilities set-up
at Mountain Home and Robins AFBs as part of the DOME contract. The meeting was
scheduled for 0730 – 1230 MST (0930 – 1430 EST) to accommodate the time
differences between the two sites. Technical support was provided by a combination of
communications and computer Air Force and contractor base support personnel at both
locations. Technical specialists were supposed to test their individual components of the
system prior to the meeting. However, as discussed in the next section, many technical
problems still existed when CMI researchers arrived two days before the meeting. In
addition, access to the Robins meeting room was limited to after hours because a non-
DOME meeting was scheduled into the room for both days. This greatly complicated
trouble-shooting at Robins and resulted in several problems during the meeting.
For this meeting, only the Air Force contracting and MAJCOM representatives and CMI researchers were required to travel to attend the meeting. The elimination of the travel requirement significantly increased meeting participation, especially at Mountain Home AFB.

**Technology.** The facilitators and all meeting participants at both sites used individual personal computers connected to a local area network (LAN), and through the LAN, to the Internet. Projectors were connected to the facilitator’s workstations to provide a public display. The primary software used during this meeting was the SPA prototype. CMI’s SPA server was used again, communicating with the remote meeting locations via the Internet. CMI researchers had to download and install Microsoft’s Internet Explorer on all workstations prior to the meeting to support the SPA clients.

GroupSystems software was also used during the meeting. Local participants used Citrix clients to connect via the Internet to CMI’s GroupSystems Citrix server which was used to support the meeting. CMI researchers also had to download and install the Citrix client software on all participant’s workstations prior to the meeting. Extensive testing and coordination with technical support personnel was required at Mountain Home to establish a working Internet connection for the meeting room and to allow access to CMI’s servers through the local firewall. In addition, SPA testing between the University of Arizona, Mountain Home, and Robins AFB indicated that Robins’ poor Internet connectivity (frequent time-outs) caused unexpected missing process model data for some Robins participants.
Video teleconferencing (VTC) equipment was provided to both sites, also as part of the DOME contract, specifically to support distributed meetings such as the one planned for the technology demonstration. Mountain Home and Robins AFBs were responsible for providing the ISDN telephone lines required to operate the VTC equipment. CMI researchers arrived to find ISDN problems at both locations, indicating that the VTC equipment had not been tested as promised. Long hours of coordination with local communication personnel and the VTC vendor were required to get the VTC equipment operational. However, problems continued to plague the Robins location throughout the meeting, essentially making the VTC equipment more of a distraction than a help. The back-up audio-conferencing capability was also extremely limited due to low quality speaker phones at both meeting locations.

**Process.** The meeting began with a welcome from the CMI researchers at Mountain Home, followed by a brief presentation by the CMI researcher at Robins on the goals of the DOME technology demonstration, specific meeting goals and general meeting guidelines. Because of lack of software on Mountain Home participant’s software, this presentation could not be displayed on participants’ workstations and had to be displayed using the CMI researcher’s laptop connected to the projector. The plan was then to have all meeting participants introduce themselves using VTC, but VTC problems made this impossible. Participants did sign-in using GroupSystems Topic Commenter.
The SPA prototype was introduced next using the same training procedure and "Go TDY" model used in the first meeting. All participants at both locations then had the opportunity to practice adding and modifying information in the model.

CMI researchers at Mountain Home then facilitated a chauffeured review of the AFTO 107 process model developed during the first technology demonstration meeting. The purpose of this review was to: (1) familiarize new participants with the model, (2) have all participants recommend improvements and changes to the model, and (3) achieve group consensus on model content. Initially, all participants reviewed the model on their personal workstations with changes made by both the facilitator and participants. However, part way through the review, Mountain Home AFB lost Internet connectivity for the entire base for over an hour. CMI researchers quickly dialed into the CMI SPA server using a personal laptop and connected the laptop to a projector so that Mountain Home participants could follow the review using the public screen. Robins AFB participants maintained their Internet connection, so they could continue to review the model at their personal workstations or at their neighbor's workstation when poor connectivity caused data loss at their workstation. Throughout the review, the two meeting locations were connected via audio conferencing so that Robins participants could hear the facilitator's comments and ask questions/make comments as desired. Both locations had a very difficult time following group discussions at the other site. In addition, Robins participants requested clarification of what process node was currently being reviewed, since they had no way of viewing what node the facilitator was highlighting. The entire review was extremely time consuming because of the
technology problems, poor inter-site communication, and the length of the model. Some participants seemed to lose interest, but many important improvements were made to the model during the review.

Following the chauffeured review, participants were asked to work in parallel to identify AFTO 107 process problems and recommend improvements. A new process information panel was added by the facilitator to the SPA to specifically capture these problems and solutions. Problems and solutions identified during the first meeting were copied onto these panels for the appropriate process prior to the meeting. After participants completed their input, one CMI researcher at Mountain Home led both sites through a chauffeured review of the problems and solutions. Participants discussed and clarified solutions and then recommend specific process improvement action items needed to start implementation/evaluation of those solutions. A second CMI researcher at Mountain Home recommended these action items in a GroupSystems Categorizer session.

The meeting concluded with an overview of plans for the next meeting, a short training session on the GroupSystems tools which would be used in that meeting. Finally, the twelve remaining participants used GroupSystems Survey to complete a questionnaire similar to the one used in the first meeting.

**Outcome.** The final AFTO 107 process model increased from 88 to 93 process nodes. Descriptions, actors, triggers, and resources were added or modified when required. Approximately 40 problems and 40 proposed solutions were also documented as part of the process model, up from the 16 problems and 8 solutions identified during
the first meeting. The 40 proposed solutions were combined into 9 recommended process improvement action items which were documented using GroupSystems.

Results of the survey conducted at the close of the session were again very positive. Detailed survey results are included in Appendix F.2 and are briefly summarized here. The average participant assessment of the AFTO 107 process model ranged from 4.2 to 4.5 (up from 4.0 – 4.2) on all quality characteristics. Quality of the recommend improvements averaged 4.2. The average rating of the overall meeting approach did drop slightly from 4.5 to 4.3, but participants rating of the distributed aspects of the meeting averaged an extremely high 4.5 to 5.0. Participants did indicate some difficult in working with participants at the other meeting location, rating this item 3.5, a full point lower than the 4.5 rating for working with participants at their location.

SPA problems with the constant redrawing/repositioning of the process hierarchy identified during the first meeting were resolved prior to the second meeting. This change, possibly combined with some participant’s familiarity with the tool, resulted in participant’s assessment of the SPA prototype’s ease of use increasing from 4.1 to 4.3. The major problem identified during this meeting was the missing data caused by Robins’ poor Internet connectivity. Some participants also provided some interesting recommendations for improvement, but in general, all participants were remarkably positive about the prototype.

5.6.4 Air Force Meeting 3: Different Time, Different Place

The third and final phase of the DOME Technology Demonstration was a different time, different place meeting held from 11 – 27 January 1999 with wing, MAJCOM, and
deposit personnel participating directly from their offices at their convenience during that time-frame. Although this meeting primarily used GroupSystems instead of the SPA prototype, a description of this meeting is included because the meeting demonstrates: (1) how output from SPA can be used, and (2) important lessons learned for distributed meetings such as those SPA could be used to support in the future. The description uses the same format as the first two meetings.

Task. During the final meeting, participants were asked to comment on the recommended AFTO 107 process improvement action list developed during the second meeting, and then to vote on the importance and feasibility of each of the recommended action items.

Group. All participants from the first two technology demonstration meetings were invited to participate in the final meeting.

Context. The final meeting of the technology demonstration was conducted as a different time, different place meeting with participants using their personal computers to participate directly from their desktops at their convenience. CMI researchers set-up and monitored the meeting from the University of Arizona. No facilitators or Air Force personnel were required to travel to participate in this fully distributed meeting. The lack of travel should have significantly increased participation, however, the lack of motivation to participate during an established time-frame seemed to be a stronger factor and negatively impacted meeting participation.

Technology. The primary software used during this meeting was GroupSystems. Local participants used Citrix client software to connect via the Internet using an Internet
browser to CMI’s GroupSystems Citrix server. GroupSystems instructions were provided to participants via Email. Instructions were also provided on how to access SPA to view the final AFTO 107 process model, but no participants seemed to use this capability.

**Process.** During the first week of the meeting, participants were asked to login to GroupSystems to review and comment on the action items developed during the previous meeting. Participants could also view the final AFTO 107 process model by logging in to the SPA prototype. At the beginning of the second week, CMI researchers shifted the 10 action items to two votes: (1) the importance of each action item for improving the AFTO 107 process and (2) the feasibility of each action item being implemented by Air Force. Participants had just over one week to login to GroupSystems and record their votes.

CMI researchers monitored participation throughout the entire two and a half week meeting, sending Emails encouraging participation and offering assistance mid-way through each week with status messages provided at the end of each week. Both activities were extended because of low participation, but this had only a limited impact on the vote and no apparent impact on the initial comment activity.

**Outcome.** Participation during this final meeting was very disappointing. During the review of the recommended AFTO 107 process improvement action list, only one new action and one comment were added by participants. Seven participants did vote on the importance of these action items, with five of those also voting on their feasibility. The voting results show some interesting similarities and differences between the
importance and feasibility of the recommended action items. For example, both votes rated establishing and standardizing on a single central POC as the top two items. In contrast, Air Staff funding of a real-time system was rated as the third most important, but the least feasible improvement action.

5.6.5 Summary

In conclusion, the DOME technology demonstration was a definite success, even when considering the technology problems encountered during the second meeting. The goals for each meeting were achieved within the allotted time-frame and AFTO process improvements were identified and evaluated on importance and feasibility. Participants were very satisfied with the quality of the AFTO process model, the recommended improvements, and the meeting approach used during the technology demonstration. The majority of DOME components met or exceeded all project expectations. The GroupSystems software worked flawlessly in both the face-to-face and distributed meetings. While several improvements were identified for the SPA, this first version of the SPA prototype was sufficiently stable and provided enough functionality so that users were easily able to develop and evaluate a very comprehensive model of the AFTO process. Even the loss of Internet connectivity at Mountain Home did not unduly hamper users because of the back-up dial-in Internet connection provided by CMI. Participant feedback on the SPA confirmed its successful support of the DOME technology demonstration.
CHAPTER 6 – DISCUSSION

PHASE I: Collaborative Scenario Elicitation Using GroupSystems

A preliminary discussion of the Phase I results was included in section 5.4 as motivation for the Phase II research. The main points of that discussion are reiterated and expanded on as each of the Phase I research tasks are reviewed in the next sections.

6.1 Phase I Observations

Six scenario elicitation meetings were observed during Phase I. The task, group, context, and technology were similar for all six meetings with a scenario elicitation task conducted by DOD environmental security subject matter experts during a face-to-face meetings supported by a general-purpose GSS, GroupSystems Group Outliner.

<table>
<thead>
<tr>
<th>Case Study</th>
<th>Process</th>
<th>Outcome</th>
<th>Issues</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cultural &amp; Natural Resources Meeting</td>
<td>Unstructured content &amp; process, scenarios at leaf &amp; non-leaf nodes</td>
<td>Met goals: Yes Users: Satisfied</td>
<td>Scenario content, format &amp; detail varied; Need detailed content &amp; process guidelines</td>
</tr>
<tr>
<td>Waste Management Meeting 2</td>
<td>Business template &amp; results in outline, user input in comments</td>
<td>Met goals: Yes Users: Positive</td>
<td>Copying to outline tedious for facilitator.</td>
</tr>
<tr>
<td>Hazardous Material Management Meeting 1</td>
<td>Technical template, results &amp; user input in comments</td>
<td>Met goals: Some Users: Confused</td>
<td>Confused about meeting &amp; development process &amp; role of scenarios.</td>
</tr>
<tr>
<td>Hazardous Material Management Meeting 3</td>
<td>Business template &amp; results in outline, user input in comments</td>
<td>Met goals: Yes Users: Positive</td>
<td>Copying to outline tedious for facilitator.</td>
</tr>
</tbody>
</table>

Table 13 – Summary of Phase I case study results
The primary differences between the meetings related to the process used, meeting outcomes including whether the goals were met and users satisfied, and the specific issues highlighted during the meeting. These items are summarized in Table 13.

6.1.1 Interpretation of Case Study Results

The cultural and natural resources management meeting represented the first attempt at eliciting scenarios using the conceptual framework and methodology developed by integrating the findings from the scenario literature review into the Collaborative Software Engineering Methodology (CSEM). The scenario literature (at that time) implied that scenario elicitation was simply a matter of asking users to tell stories. As a result, users were provided only very general scenario content guidelines (Appendix B) and no scenario definition process support. Results of the meeting showed a somewhat more complicated picture. Users were able to easily define unstructured scenarios using GroupSystems Group Outliner which provided a rich picture of the problem domain. Such rich pictures would be extremely useful for systems analysts and developers trying to quickly understand a specific problem domain. However, the resulting scenarios were often incomplete and varied significantly in content, format, and level of detail portrayed – making them difficult to analyze. These results highlighted that much more was unknown about collaborative scenario elicitation than originally anticipated. Therefore, this first meeting was more important for identifying research questions and the need for a detailed collaborative scenario elicitation methodology, than it was for providing answers to those questions.
The five structured GroupSystems scenario meetings used issues raised during the first meeting and results from the Phase I experiment to guide implementation of a structured template-based scenario elicitation process. As can be seen from the summary in Table 13, both the template and the meeting process evolved based on lessons learned from the preceding meetings. Results of the structured GroupSystems scenario meetings supported use of a business-oriented scenario data model and template, identified both the strengths and weaknesses of using GroupSystems and the proposed methodology for collaborative scenario elicitation, and highlighted the negative impacts of the productivity problems that resulted from those weaknesses.

Early sessions used a more system-focused scenario template including such items as successful end conditions and failed end conditions. Users did not understand why the technical items were included in the template. They were difficult for users to complete and often duplicated information already provided (e.g., objective and successful end condition generally had similar content). Templates based on the proposed contextual scenario data model (section 4.3) were better received by the users. This feedback supported the strong business/user focus (vs. system focus) of the proposed data model.

Procedures for using a structured scenario template with GroupSystems GroupOutliner also evolved over the various sessions. (Note: Because of limited GroupOutliner support for templates, templates were only able to directly prompt for scenario-level information in the contextual scenario data model. Users were expected to include action-level information in the unstructured action descriptions.) Initially, the entire template and all scenario information were entered as comments for each individual
scenario. Only activities and scenarios were visible in the outline. Users had to double-click on the scenario to view the scenario information. When the group reviewed the information, the agreed upon response was moved to the top of the comment list and divided from other comments. Users had a great deal of difficulty with this approach because too much information was included in one comment window and they had limited visibility of that information. As a result of this feedback, templates were added to the outline so the comment window only contained information about a single template item for an individual scenario. While users felt this was an improvement, they still felt they lacked visibility of the agreed-upon scenario information. This feedback resulted in the final modification whereby the facilitator copied the agreed upon information into the template item in the outline. Although adding the template and copying group results to the template items was awkward and time-consuming for the facilitator, it did seem to provide users with the flexibility and visibility they were looking for during the scenario elicitation process.

Users also occasionally expressed the need to graphically diagram the flow and sequence of actions before documenting scenario actions, especially for complex scenarios with parallel processing and multiple decision points. Textual representations are simply not a good representation of these types of complex scenarios. Since no graphical capability is included in Group Outliner, users sketched out action sequences on white boards as needed for complex scenarios. After the group had agreed on the action sequence, they were generally able to document the actions in Group Outliner.
As described above, facilitators and users learned how to work around the GroupSystems limitations they encountered. However, these work-arounds were often slow and negatively impacted meeting productivity. As shown in Table 13, meeting goals were only partially achieved in three of the five structured GroupSystems scenario meetings. Table 13 also shows that users consistently expressed frustration whenever goals were not achieved. Limitations in GroupSystems support for structured templates directly contributed to the productivity problems encountered which, in turn, contributed to the failure to achieve meeting goals.

6.1.2 Limitations of the Case Studies

The two major limitations of the case studies were directly related. First, while users were asked to provide a large amount of information about scenarios, only limited support in the form of a structured template was provided to manage this information overload. In addition, the template only helped users manage scenario-level information. It did not provide any support for managing action-level information overload problems.

The second limitation was that the available collaborative tool, GroupSystems Group Outliner, provided only limited and awkward template support. Changes in the review procedures requested by users to increase information visibility and reduce information overload only made Group Outliner's limitations more obvious and directly slowed productivity. In addition, GroupSystems could not easily support an iterative template-based methodology or provide a graphical modeling capability, two other possible mechanisms for dealing with the information overload problem.
6.2 Phase I Experiment

The experimental results provided mixed support for the four hypotheses (Table 14). Hypotheses which claimed that the unstructured format would be better (hypotheses 3 and 4) were generally supported. However, those that stated that the structured formats would be better either were not supported (hypothesis 1) or were contradicted (hypothesis 2). Although there were differences in the degree of statistical preference, all quality and productivity measures rated higher on the unstructured formats.

<table>
<thead>
<tr>
<th>#</th>
<th>Hypothesis</th>
<th>Treatment Comparisons</th>
<th>Unstructured vs. Structured Format</th>
<th>Preferred Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Structured Scenarios More Complete</td>
<td>Not Supported</td>
<td>Not Supported</td>
<td>Unstructured</td>
</tr>
<tr>
<td>2</td>
<td>Structured Scenarios Higher Sequential Action Completeness</td>
<td>Contradiction</td>
<td>Weak Contradiction</td>
<td>Unstructured</td>
</tr>
<tr>
<td>3</td>
<td>Unstructured Scenarios Higher Non-Sequential Action Completeness</td>
<td>Weak Support</td>
<td>Supported</td>
<td>Unstructured</td>
</tr>
<tr>
<td>4</td>
<td>Unstructured Scenarios More Productive</td>
<td>Supported</td>
<td>Strong Support</td>
<td>Unstructured</td>
</tr>
</tbody>
</table>

Table 14 -- Experimental results summary

A broader analysis of the quantitative and qualitative results can be summarized into three main findings: (1) scenario completeness was low for all formats, (2) completeness was lower for structured formats than the unstructured format, and (3) results of higher productivity measures for the unstructured format contradicted subjects' perceptions of higher productivity for structured formats. Possible reasons for each of these findings is discussed next.
6.2.1 Interpretation of Experimental Results

The results clearly showed completeness problems for all formats at both the action and scenario level (Figure 14). At the action level, only half of actions included descriptions, with less than a quarter including data requirements, exceptions, and alternatives. One possible explanation is that, although the instructions requested this information, none of the formats specifically prompted for it, so subjects had no on-screen reminders as they developed the scenario descriptions. In addition, subjects were asked to provide all the information at once, which may have caused an information overload problem. Therefore, while the structured formats may have made it easier to iteratively refine the action definitions to add more detailed information, since the process did not specifically tell subjects to do this, they did not. Most subjects did well identifying the main actions, but may have done better with total action completeness if the definition process had been split into at least two steps: (1) identifying the actions and then (2) adding the remaining detailed action information. At the scenario level, completeness was also low with only an average of 28% of the possible 66 actions included per scenario. One possible cause for this problem was the fact that subjects developed scenario definitions individually and did not collaborate to capture actions identified by others.

In addition to the common completeness problems, completeness was significantly lower for the structured formats when compared to the unstructured format (see Table 7 and Table 8). This may have been caused by the structured formats’ action numbering which increased focus on the action sequence at the cost of lost attention to the other
action information. Format 3’s named and numbered steps magnified this problem by totally separating the action from the other information. Both structured treatments’ focus on sequential actions also made it more difficult to represent non-sequential information such as exceptions and alternatives. Most importantly, the structured treatments provided no specific structural support for increasing completeness, e.g., with prompts or templates.

Productivity results were somewhat contradictory, with the productivity metrics clearly indicating that the unstructured format was more productive (see Table 7 and Table 8) while subjects felt more productive using the structured formats (see Table 10). One possible explanation for this contradiction is that while the unstructured format did not hinder productivity, it also did not provide any specific support for scenario definition. So when subjects were asked if the method they used supported scenario definition, they preferred the structured formats which used more of Group Outliner’s capabilities. In addition, subjects may have focused on the higher word count of the unstructured format without recognizing the associated increased content, thereby feeling less productive because more words were required for their scenario definitions.

6.2.2 Limitations of the Experiment

Some of the variability in the scenario descriptions may have been caused by vague definition of the scope of the experiment's scenarios. For example, while some subjects felt registering for classes started with calling the university's course registration system, others began with planning their schedules. Still others began with determining the courses they needed to take based on their major. While this may be useful when
trying to develop an initial understanding of the complexities of a scenario, in general it will make reaching consensus much more difficult. In the future, the scope of scenarios should be more clearly defined before beginning definition of the actions necessary to accomplish that scenario.

A second limitation of this study was that, although subjects were told what to information to include in a scenario definition and how to document that information using Group Outliner, they were not told what process to use to develop the scenario definition. This lack of process guidance may have been a major contributor to the action completeness problems observed. For example, the action completeness results (Figure 14) seem to indicate that subjects focused on identifying the main actions only. If they had been told to identify the main actions first, and then go back to add the other action information, action completeness may have been much higher. Future research should evaluate how an improved process can increase scenario quality and productivity.

A final limitation of the study was that the combination of the selected scenario formats and the Group Outliner tool may have significantly limited subjects' ability to quickly define quality scenarios. As a minimum, neither seemed to provide any support for any of the completeness problem areas. Future research should investigate alternative scenario format and tool combinations to improve quality and productivity.

6.3 Phase I Theory Building

The Phase I results indicate the need for a detailed and iterative collaborative scenario elicitation methodology with appropriate prompts at the scenario and action level to improve scenario completeness. An iterative process should reduce the
information overload problem and may improve scenario completeness as each step focuses on a specific sub-set of the required scenario information. A collaborative process enables information sharing which should increase completeness. Finally, including specific prompts should also increase scenario completeness and reduce the information overload problem since users will have to remember what information to include in the scenario descriptions. The initial collaborative scenario elicitation methodology was modified to the maximum extent possible to provide these capabilities. A detailed description of the revised methodology, designed for use with GroupSystems Group Outliner, is included in Appendix G.

The primary advantages of using this methodology with GroupSystems Group Outliner is that it provides easy-to-use collaborative support for capturing rich, textual scenario descriptions. Those advantages, however, come at a cost. Group Outliner provides only limited and awkward template support. But the Phase I results show that there are significant completeness problems when a template is not used. Group Outliner also provides only weak support for an iterative methodology, again because of its limited template support. It is difficult enough for facilitators as it is to add the entire template to each leaf node. This process would be even worse if they had to add additional template information to each leaf node at each step of the incremental scenario elicitation process. Therefore, they currently add the entire template at one time and hope session participants are paying attention when the facilitator tells them which template information they should be providing at a particular point in the scenario
definition process. Finally, the lack of an integrated graphical view of activity flow, sequence, and decision logic slowed users when documenting complex scenarios.

The purpose of Phase II was to address these limitations through development of a special-purpose, integrated structured textual scenario and graphical process modeling prototype designed to support iterative collaborative scenario elicitation.

PHASE II: Integrated Scenario and Process Modeling Using SPA

6.4 Phase II Systems Development

Development of the initial Collaborative Distributed Scenario and Process Analyzer (ColD SPA) prototype was an exciting and frustrating process of discovery, solving problems, taking advantages of opportunities, and making compromises. Teamwork was essential, not only for successful development of SPA, but also as the source of synergistic solutions which have made SPA a flexible collaborative tool with application for a much wider variety of problem analysis tasks than originally planned. The remainder of this section discusses some of these challenges, synergistic solutions, and lessons learned during the development process.

The initial SPA concept was simply to build some additional functionality on top of GroupSystems Group Outliner, similar to what had been done for Group Data Modeler. The first change in direction came when the system architect convinced the rest of the team that SPA should be developed using Java to create a web-based SPA prototype that could support distributed meetings as well as traditional face-to-face meetings. Everyone was excited about the opportunity to add the distributed capability, but they were also somewhat leery about the potential impacts on performance and speed.
of development. Although some of these concerns proved to be valid, SPA's web-based architecture is one of its strongest selling points.

Even with this change of direction, the assumption was that the requirements process would be relatively straightforward. Five core requirements (see section 5.5.1.1) defined the essence of SPA required functionality. Existing systems such as GroupSystems Group Outliner, Activity Modeler, Group Data Modeler, and IDEF3 process modeling tools such as KBSI's ProSim provided excellent models for how to implement the five core requirements. The reality of the SPA requirements discovery process was totally different than assumed and far from straightforward. The actual requirements analysis and design process involved weekly discussion meetings, multiple iterations, and extensive documentation. Requirements and design decisions were documented on an internal web-page for the team. They also had to documented in Software Design Descriptions (SDD) for the Air Force and DECSIM PMO to satisfy contract requirements. Eventually they had to be summarized into a discrete requirements list (see Appendix E.1) so that allocation of requirements to SPA increments could be efficiently managed.

There were several key reasons why the requirements process was so much more complicated than expected. First, the initial expectations were simply unrealistic. Secondly, not all team members were as intimately involved as the author in the goals and background of the research. For example, some team members were not familiar with the GroupSystems, collaborative modeling, and IDEF3 tools which were guiding SPA development. Others had not used collaborative tools in real-world meetings so
were unaware of the criticality of simple user interfaces and flexible facilitator options. Still others did not have systems analysis experience, were not familiar with CSEM, and did not understand the role of scenarios in CSEM. Third, there were some of the features of the source tools that needed improvement. Finally, there were some difficult conceptual issues involved in the integration of the scenario and process modeling concepts as discussed next.

Integration of the scenario concepts of activities/scenarios/actions and the process modeling concepts of processes/sub-processes/tasks into a single generic concept of process had both benefits and risks. The primary benefit was simplicity. Users would not be required to understand the differences between the different levels. Levels of abstraction could easily change as required for a specific process or task, again without the user needing to be concerned with the different concepts. The primary risk was that the process concept would be too general. Users could lose focus on identifying specific scenarios and actions required to accomplish those scenarios. There were also data model implications. The contextual scenario data model recommended capturing some similar and some different information at the scenario and action level. The next challenge was to refine the data model to reflect this integrated concept.

Data requirements for both the scenario and action levels as well as traditional process modeling data were included in the data model as shown in Appendix E. Many of the specific text data items (e.g., description, objective, trigger) were identified as sub-classes of a common super-class, All_TextTYPES. Creation of this super-class provided significant flexibility in capturing other types of textual information, as will be
described in the discussion of the user interface. This entity also included attributes for storing the name of the text type, the prompt requesting that information, and the mouse text describing the purpose of the panel used to capture this information. This flexibility allows the facilitator to change from the generic "process" prompts to tailored scenario prompts if the former terminology is too generic for a specific user group or task.

The primary SPA interface is shown in Figure 17 with a process hierarchy on the left and process information panels on the right. The interface designer proposed this model to avoid the confusion often caused by using too many pop-up (modal) windows where users can lose track of which window is active. Mapping the data requirements to the process information panels, two types of panels were identified. Fixed panels captured specific data items in a structured format. For example the actor and object panels allow users to select actors (objects) from a master list and assign them to a specific process. Use of the master list reduces some of the ambiguity found during the Phase I experiment. Text panels support capture of free-format textual comments, with a separate panel for each type of textual information that needs to be captured for a process. The text type name, prompt, and mouse text for each panel are data values read from the All_Text_Type table. Because these items are data values versus hard-coded in the program, their values can be easily changed and new text panels created as required to support a specific task.

The panel concept also provided the flexibility needed to support an iterative scenario elicitation methodology. The facilitator determines whether each possible panel is visible or invisible to the user. This allows the facilitator to limit information overload
by only displaying a few panels at any given time. It also provides simple support for an iterative collaborative scenario elicitation methodology by allowing the facilitator to tailor which panels are visible for each step in the methodology. The facilitator can change panel visibility on-the-fly as required during the scenario elicitation meeting.

The previous discussion provides an overview of how some of the key analysis and design issues were resolved to ensure SPA provided all its required core scenario and process modeling functionality while providing significant flexibility to support other tasks. These analysis and design decisions, however, also placed a major burden on the system building task which is discussed next.

The primary operating concept during systems building was one of trade-offs. Functional requirements had to be deferred so that contract deadlines could be met. However, the core functionality still had to be delivered, so negotiations were required to separate required from desired capabilities for the initial prototype. This negotiation process was complicated by the fact that the team did not want to degenerate into a build-and-fix development paradigm. So, requirements that impacted the essential architecture of SPA had to be included in the initial prototype. Only requirements without major database or architectural impacts could be deferred to future prototype versions. As mentioned previously, results of these negotiations were documented on the internal web-site and as part of the functional requirements list (see Appendix E.1).

The selection of Java and the inconsistent Java support built-in to current Internet browsers also caused problems during system building. It continually frustrated developers when a certain function would work flawlessly in their development
environment (Visual Café), but would have problems when accessed using Microsoft’s
Internet Explorer browser, and even different problems when used with Netscape’s
Navigator browser. This problem complicated the already challenging task of developing
in a totally new language (none of the developers had prior Java experience).

Testing was also a major challenge of the system building task for several reasons.
First, the expedited development schedule simply did not provide sufficient time for
testing. For example, major problems with the SPA prototype were identified three days
before the first Air Force meeting. These problems were not resolved until the evening
before the meeting, so only absolutely minimal testing was conducted on those fixes.
Second, the SPA architecture (Java applet running with an Internet browser on a client
workstation, connected via the Internet, to the separate server application) made testing
extremely difficult. It simply was not possible to test all possible combinations of
functions using different browsers on different workstations with different quality
communication lines with the server under different workload conditions. In addition,
when problems occurred, it was often difficult to allocate those problems to a specific
component. Finally, the collaborative and flexible nature of the tool made it impossible
to fully test all possible combinations of actions and action sequences which could be
used by a group. The combination of the Java development environment challenges and
the testing problems have caused major reliability problems with the SPA prototype
which persist to this day. However, all is not bleak. The prototype has been used very
successfully with Air Force groups as described in the next section. In addition, the
lessons learned during SPA development have already benefited the other Java
applications currently being developed by the Center for the Management of Information and will continue to benefit future distributed tool development projects.

6.5 Phase II Observations

Results of the observation of the SPA prototype during the Air Force technology demonstration meetings are described in section 5.6. Lessons learned from the observation of these meetings and limitations of these case studies are discussed in the next two sections.

6.5.1 Interpretation of the Case Study Results

The SPA prototype successfully met all Air Force requirements and was rated very highly by meeting participants. However, there were some very interesting lessons learned during those meetings which have direct implications on the SPA prototype and the methodology used to develop SPA models. These lessons learned include:

- A simple, easy-to-use interface is critical for SPA. Users were easily able to add comments to the free-form text panels (e.g., the description panel), but had more difficulty with the structured actor panel. The actor panel interface needs to be simplified.

- Import and export capabilities with GroupSystems would simplify moving between SPA and GroupSystems and would allow the most appropriate tool to be used for each task.

- As the size of the SPA model increased, synchronous review of all model information with the entire group became problematic. For example, during the second meeting, some participants lost interest during the almost two-hour
facilitated review and improvement of the model. Possible methods for avoiding this problem include:

- The facilitator could review the model at a high-level only, and then have participants review and improve the model in parallel. Alternatively, if participants are familiar with the SPA prototype, they should be asked to review the model before the meeting in an any time, any place mode.

- During the review, time was wasted reviewing items everyone agreed on. The review would have been much faster if the group could have focused discussion only on those processes or information panels about which the group had questions or with which participants disagreed. SPA needs a way for participants to "mark" processes/comments to indicate their agreement or disagreement. This capability would allow participants to review the model in detail individually, either synchronously or asynchronously. The entire group could then quickly complete the review together by simply reviewing those areas "marked" by participants.

- In general, as the size of the group increases, the meeting will be more effective if more time is spent with participants working in parallel with less time spent on chauffeured reviews and discussions.

- Back-up capabilities are essential for all technology critical to meeting success. For example, during the second meeting, dial-up communications were used when Mountain Home’s direct Internet connection went down for over an
hour. If this back-up had not been available, the distributed meeting would have been cancelled.

- Participation in the different time, different place meeting was disappointing. This may have been caused by the delay between the second and third meetings or by lack of incentives for participation. Motivation would have been higher if it had immediately followed the second meeting or if participants knew that their input was critical for a scheduled follow-on meeting.

These lessons learned will be helpful in informing enhancement of the SPA prototype process hierarchy and information panels.

6.5.2 Limitations of the Case Studies

The primary limitation of the case studies is that the graphical process modeling capability was not used by participants during either meeting and therefore could not be evaluated. The graphical capability was not used because: (1) its functionality was very limited and automatic diagram layouts less than ideal, (2) the AFTO 107 process was adequately represented in the process hierarchy and did not require a supplemental graphical view to understand the process, and (3) it is not yet clear exactly when and how the graphical modeling should be incorporated into the overall scenario elicitation process. These issues will require additional research as described in section 7.4. In contrast, the Air Force sessions did provide sufficient information about use of the process hierarchy and information panels to revise the collaborative scenario elicitation methodology. The revised methodology is discussed in the next section.
6.6 Phase II Theory Building

The methodology developed during Phase I was designed for use with GroupSystems. It provided an iterative collaborative scenario elicitation methodology with appropriate prompts at the scenario and action level to improve scenario completeness. However, GroupSystems' limited template support made the methodology somewhat awkward to implement. It also only provided the most basic support for the desired iterative approach for scenario elicitation and did not provide a graphical process modeling capability. SPA was designed to overcome these limitations. Evaluation of SPA showed that SPA successfully achieved these goals. Therefore, the collaborative scenario elicitation methodology was refined to take advantage of the increased functionality provided by SPA and the lessons learned during SPA evaluation. A description of the revised collaborative scenario elicitation methodology is included in Appendix H. The methodology was initially documented using the SPA prototype. CMI researchers and DESCIM PMO personnel reviewed the methodology using SPA, adding recommendations for improvement as required. A copy of the reported generated by SPA for this session served as documentation of the revised methodology (Appendix H).

The final issue for this discussion is to explore possible theoretical explanations for the Phase I and II results. The basic GSS theoretical model proved very effective for describing what happened and comparing results across research tasks and phases. However, it does not provide any theoretical explanations for why these results occurred, i.e., for why SPA provided better support for collaborative scenario elicitation than GroupSystems. Zigurs and Buckland (1998) recently proposed a model relating
task/technology fit to GSS effectiveness which may begin to provide such a theoretical explanation. They propose that GSS technologies differ in the degree of support provided in three dimensions: communication, process structuring, and information processing (including information structuring). Similarly, different categories of tasks have different ‘ideal’ levels of support in each of these dimensions. They combine this information into a proposed theory of task/technology fit for GSS effectiveness. Using their task categories, collaborative scenario elicitation would be considered a fuzzy task. The ideal profile for a fuzzy task is high communication support, medium process structuring, and high information processing (Table 15).

<table>
<thead>
<tr>
<th>Profile</th>
<th>Communication</th>
<th>Process Structuring</th>
<th>Information Processing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fuzzy Task Profile (Zigurs &amp; Buckland, 1998)</td>
<td>High</td>
<td>Medium</td>
<td>High</td>
</tr>
<tr>
<td>GS Group Outliner</td>
<td>High</td>
<td>Low</td>
<td>Medium</td>
</tr>
<tr>
<td>SPA Prototype</td>
<td>High</td>
<td>Medium</td>
<td>High</td>
</tr>
</tbody>
</table>

Table 15 – Task/technology fit and GSS effectiveness

Both GroupSystems Group Outliner and SPA provide high levels of communication support. As a general-purpose tool, Group Outliner generally provides only low process structuring. In contrast, SPA’s specific, but flexible panels which can be used to guide the process provide medium processing support. Finally, while Group Outliner provides extensive information processing capability, its information structuring capability is much more limited, resulting in an overall rating of medium for this dimension. The SPA prototype includes all the information processing capabilities of Group Outliner plus additional information structuring capabilities provided by the
specific information panels, resulting of a rating of high. As can be seen in Table 15, SPA more closely ‘fits’ the ideal profile, providing a possible theoretical explanation for the observed results. This possibility will be explored further during future research as described in section 7.4.
CHAPTER 7 – CONCLUSIONS AND FUTURE RESEARCH

7.1 Conclusions

Scenarios have great potential for helping to solve many of the requirements problems that have been plaguing the software industry since its inception. Their primary value centers on their simplicity – users can easily describe concrete examples of how they do their jobs. These examples provide a rich source of information for aiding the discovery of requirements and for evaluating alternative ways for meeting those requirements. However, when used in an undisciplined manner, scenarios can also be incomplete, unfocused, fail to provide critical information, and extremely inefficient to define and analyze. The purpose of this research was to advance understanding of the issues involved in user-focused, scenario-based information systems development and to create a disciplined collaborative scenario elicitation methodology that can accrue the benefits of scenarios while avoiding their pitfalls. Progress towards this goal is summarized by comparing the questions which drove this research with the responses to those questions developed as part of this dissertation. As stated in Chapter 1, the primary research question of this dissertation was:

What are the collaborative modeling processes, tools, and facilitation techniques needed to effectively elicit scenarios from users in a group environment?

This question was decomposed into seven more specific questions relating to the purpose, lifecycle, content, and format of the scenarios as well as the group process and collaborative tool support used to elicit them. Each of these is reviewed in turn.
**Purpose.** Scenarios can be used for a wide range of purposes, some of which are applicable to the Collaborative Software Engineering Methodology (CSEM). The need to increase understanding of this area resulted in the question:

How will scenarios be used during all CSEM development phases and how are the scenarios developed for these different purposes related?

The proposed framework for scenario usage in CSEM (section 4.1) describes how business, interaction, internal, and test scenarios can be used during each of the CSEM development phases. The relationships between scenarios in CSEM are graphically portrayed in Figure 10 – Figure 11. This framework was developed by integrating results of the scenario literature review with an in-depth understanding of the goals of CSEM. Although only scenario usage during the requirements phase has been evaluated, results to date indicate that, as a minimum, the framework can be effectively used to guide research in this area. The framework can also be used to guide system development projects for those practitioners desiring to implement a user-focused, scenario-based approach, at least until better guidance is provided in the literature.

**Lifecycle.** It is clear from the proposed framework and the literature that scenarios will have a lifecycle of their own as they move through the development process. It is much less clear how that lifecycle should be managed, raising the question:

How will scenarios change and evolve during development and how can that change be effectively supported?

A preliminary response to this question was provided as part of the description of the relationships between the different types of scenarios in Chapter 4. The collaborative scenario elicitation methodology designed for use with GroupSystems (Appendix G)
provides some additional detail of how business scenarios evolve during the initial scenario capture from users. However, since the detailed research has focused on this initial capture, not much is yet known on scenario evolution throughout development. In fact, more questions were raised by this research than answers provided. Therefore, this question requires additional research as part of future research in scenario usage beyond the initial scenario capture. This future research is described in section 7.4.

**Content.** Any specific guidelines for scenario elicitation require clear definition of the scenario information requirements. Several alternative scenario data models have been proposed and therefore must be reconciled to answer the question:

> What information should be included in the contextual scenarios elicited directly from user groups?

Very specific responses to this question were developed as part of this research. The contextual scenario data model described in section 4.3 defines scenario and action-level data requirements. These data requirements were used with positive feedback during the scenario meetings studied during Phase I. An integrated scenario and process modeling data model was developed from the contextual scenario data model and general process modeling data requirements (Appendix E). This data model has not yet been evaluated during scenario elicitation sessions, but it is a direct evolution from the Phase I data model which was positively evaluated by users. This evaluation showed that users could easily provide the information requested by these models. The only concern is that scenarios developed based on these data models have not been followed throughout the entire systems development process. As a result, the models have not been formally
evaluated to determine whether they provide the information required by analysts and developers. This issue should also be evaluated during future research.

**Format.** Many alternative formats have been proposed for scenarios, but with limited guidance on the impacts on those choices. This problem generated the question:

What format should be used for contextual scenarios and how does that format impact scenario quality and productivity of user definition of scenarios?

Results of the Phase I experiment and observations, clearly showed the need for prompts to ensure scenario quality. A template based on the contextual scenario data model was implemented in the GroupSystems collaborative scenario elicitation methodology (Appendix G) to provide these prompts. Later Phase I observations showed that the templates did improve scenario completeness, but at some cost to productivity. The Collaborative Distributed Scenario and Process Analyzer (CoID SPA) prototype was designed to provide a simpler mechanism for prompting for information without this negative productivity impact. Initial observations of SPA use have been positive, but focused experiments are required to specifically prove this advantage. As a minimum, SPA makes it much easier for facilitators to include prompts in the tool than is currently provided by Group Systems' awkward template support. SPA also provides an integrated graphical process modeling capability, but this capability has not yet been evaluated in a real-world setting.

**Group Process and Facilitation.** Prior GSS research has proven that effective meeting processes and facilitation techniques, tailored to the specific GSS tool, task, and context, are essential to GSS meeting success. These findings highlighted the question:
What group process and facilitation techniques are appropriate for each collaborative tool used to elicit contextual scenarios from user groups?

Specific collaborative scenario elicitation methodologies were defined for use with GroupSystems Group Outliner (Appendix G) and the SPA prototype (Appendix H). Variations of the GroupSystems methodology were evaluated during Phase I case studies with generally positive feedback (given the limitations of GroupSystems). The SPA methodology evolved from the GroupSystems methodology with changes incorporated to take advantage of SPA's increased functionality. Since the SPA methodology addresses many of the concerns identified with the GroupSystems methodology, it should be positively received, but it has not yet been formally evaluated.

**Collaborative Tool Support.** GSS have also proven highly effective in real-world settings for increasing the productivity of group meetings. The need to find as effective a solution for collaborative scenario elicitation led to the questions:

- Can general-purpose GSS tools be used for collaborative scenario elicitation or is a special-purpose tool needed?

- What are the functional requirements for a collaborative scenario modeling tool?

Phase I results showed that the general-purpose tool, Group Systems Group Outliner, could be used to support collaborative scenario elicitation. However, there are some significant limitations that negatively impacted both the process which could be used as well as the productivity of that process. These results drove the Phase II research, designed to explore the second question. As more fully described in Chapters 5 and 6, functional requirements were defined for the initial version of the SPA prototype.
(Appendix E). These requirements were successfully implemented in a proof-of-concept prototype, which was used with very positive feedback. Lessons learned during the systems building process and observations of system use will be used to further refine the SPA requirements and prototype.

In summary, specific solutions to the primary research question of how to collaboratively elicit scenarios from user groups have been proposed through this research’s responses to the detailed questions as described above. These responses, while sometimes preliminary, have significantly increased understanding of collaborative scenario elicitation. These contributions are discussed in more detail in section 7.3. Before looking at those contributions, it is important to understand some of the limitations associated with this research. These limitations are summarized in the following section.

7.2 Limitations

The limitations of the individual research tasks were discussed in Chapter 6. Additional limitations were identified in the preceding discussion of the responses to each specific research question. These discussions highlighted three recurring themes which summarize the primary limitations of this research.

First, although the SPA prototype was evaluated during two meetings, those meetings focused on process modeling. Process modeling is similar, but not identical, to scenario modeling. In addition, the graphical portion of the tool was not used by participants during either meeting. Therefore, conclusions on the effectiveness of the SPA prototype must be qualified by these caveats. Future research should address this
limitation by using the full capabilities of the SPA prototype during scenario elicitation meetings.

Secondly, only one experiment was conducted for this dissertation. It explored the narrow topic of the impact of scenario format for individual definition of scenarios using GroupSystems. No formal experimental comparisons of collaborative scenario elicitation methodologies or collaborative tools have been conducted. Conclusions regarding the benefits of specific methodologies and tools have been based on limited observations only. The strength of these conclusions would be enhanced significantly if additional experiments and observations were conducted to further explore these issues.

Finally, this research has primarily focused on the initial collaborative elicitation of business scenarios from users. However, the real benefits of using scenarios is not in the ability to simply capture descriptions of business processes, but in their ability to improve the requirements and systems development processes. So, any conclusions on the effectiveness of a specific methodology or tool must be qualified by the fact that the down-stream use of the scenarios elicited from users has not yet been studied. Therefore, longitudinal studies are required to study the use of scenarios throughout all phases of systems development.

Even given these limitations, this research has made several significant contributions. These contributions are discussed in the next section.
7.3 Contributions

This research has significantly increased understanding of the issues involved in elicitation of scenarios directly from user groups as part of an overall collaborative systems development process such as that defined by CSEM.

Existing guidelines for scenario usage in CSEM were extended to clearly define opportunities for scenario usage and evolution during all CSEM phases, providing a comprehensive life cycle perspective not addressed in the current literature.

Specific recommendations for scenario content, form, group process and facilitation techniques were defined for collaborative scenario elicitation using a general-purpose GSS, GroupSystems Group Outliner, in a face-to-face meeting environment. These recommendations are ready for use now by both researchers and practitioners. Researchers can use the proposed collaborative scenario elicitation framework and methodology to guide future research. Practitioners can use the proposed methodology to implement a scenario-based requirements elicitation process now. The use of GroupSystems with this methodology provides significant improvements over the limited scenario tools and process guidance currently available for practitioners.

A special-purpose GSS tool, the Collaborative Distributed Scenario and Process Analyzer (CoID SPA), was developed which provides improved support for collaborative scenario elicitation in both face-to-face and distributed settings. This tool was integrated into a comprehensive methodology for collaborative scenario elicitation which will allow user groups to rapidly define, analyze, and achieve consensus on the large number of scenarios necessary to define requirements for today's complex...
information systems. The tool also provides a platform for continued research in collaborative scenario elicitation that is not constrained by the limitations of a general-purpose GSS.

SPA research has also identified a wide range of potential uses for the tool well beyond the initial focus on scenario usage during requirements determination. These opportunities have been created by the flexibility designed into SPA, which has also supported use of SPA as a first-step towards a build-your-own GSS tool. These potential uses provide many new directions and opportunities for future research as described next.

7.4 Future Research

Numerous opportunities for future research were identified during completion of each of the research tasks included in this dissertation. Some of the more promising opportunities are summarized next.

Research to date has focused on the initial capture of business scenarios. Longitudinal studies are needed to study the use of scenarios through all phases of system development. The collaborative scenario elicitation framework proposed for usage of scenarios in CSEM could guide this research. Research is needed to evaluate each of the proposed scenario usages, to track the evolution of scenarios between development phases, to identify additional tool functionality required to support these proposed usages and scenario evolution, and to analyze the benefits of a scenario-based approach to systems development.
There are also many areas of additional research associated with the Collaborative Distributed Scenario and Process Analyzer (ColD SPA) prototype. Some of the issues that need to be addressed include:

- Improving tool reliability and communication robustness for use over poor quality or slow dial-up Internet connections or by client computers with limited processing capability. Current reliability and robustness problems limit opportunities for use of SPA to support real-world groups.

- Enhancing SPA functionality to incorporate deferred requirements and new requirements identified based on lessons learned during SPA evaluations. In particular, additional functionality is required to support more effective facilitation and streamlined collaborative review.

- Identifying new functionality and processes required to support distributed scenario elicitation meetings. Distributed meeting support is one of the most powerful features of the SPA prototype, but additional capabilities such as matching views, chat windows, participation monitoring, and consensus polling are required to more effectively support those meetings.

- Automating the links between the SPA prototype, GroupSystems and other tools. Easy-to-use import and export capabilities are needed so that the facilitator can move between SPA and GroupSystems as required to use the tool which is best suited for each individual meeting task. The same capability with single-user CASE tools will enhance process support by enabling use of
the best tool for each task and eliminating the need to duplicate some CASE-tool functionality in SPA.

- Upgrading the graphical process modeling capabilities in SPA. The current process modeling capability needs to be improved, evaluated, and new requirements and processes identified and implemented in a cyclic research process until an effective tool and process can be developed. For example, it is unclear how the process decomposition used for textual scenario elicitation must be modified to accurately reflect process sequence and decision logic in the graphical model. It is also not clear what functions should be performed by the system versus by the user (e.g., should links between processes be automatically generated based on the sequence in the process tree or only added explicitly by users).

- Identifying data required to support process simulation. SPA includes some basic process metrics now. However, if these metrics were enhanced, the SPA process data could be exported to a stand-alone process modeling and simulation tool such as KBSI's ProSim. Process simulation can help users better visualize process bottle-necks and opportunities for process improvement.

- Exploring other potential uses of SPA. The flexibility to define and change the process information panels on-the-fly has been built-in to the current version of the SPA prototype. This flexibility opens the door to use SPA for many applications beyond scenario elicitation, including use as a build-your-own
GSS tool. These additional opportunities need to be identified and explored so that the benefits of SPA can be extended to a wider variety of collaborative processes.

There also needs to be increased emphasis on theory building and more formal evaluation of proposed methodologies and SPA enhancements. This could be accomplished by conducting experiments comparing these new approaches to more traditional methods. Additional case studies of real-world uses of these approaches also need to be initiated. Finally, possible theoretical explanations such as the task/technology model proposed by (Zigurs & Buckland, 1998) need to be evaluated so that findings from the experiments and case studies can be generalized into an enhanced theory for collaborative scenario elicitation.

In conclusion, the research conducted as part of this dissertation has set the foundation for both enhanced scenario research as well as research in several new areas designed to improve the effectiveness of many types of collaborative work in both face-to-face and distributed environments.
APPENDIX A – COMPLETE SAMPLE SCENARIO

Scenario Name: Generate Hazardous Material Authorization Request

Description: Requestor fills out the requestor's portion of the electronic or paper request to authorize hazardous material (HM) for inclusion on the Authorized User List (AUL). The authorization request identifies the who, what, when, where, why, how much, and specific process for the HM that is being requested.

Objective: Complete a request for authorization (manual or electronic form) to use a hazardous material.

Preconditions: The requester must know what material is required and must have a legitimate requirement to use the requested hazardous material. A paper form or an electronic form must exist and be available for the requestor to complete.

Viewpoint: Person who requests hazardous material.

Trigger: Mission requirement to perform a process with a particular hazardous material not yet authorized.

Result: Form accepted by hazardous material validation team, control # assigned and provided to requestor.

Exceptions: In emergency, hazardous material is issued immediately and the authorization request is completed after the fact.

Alternatives: Requestors may complete manual or automated request form. Individual services or installations may have specific forms for this process.

Actions.

A1. Requestor completes hazardous material authorization request.

Description: Requestor enters requested information about the requestor, material requested, requiring documents, and the process which will use the material.

Data Required:
Requestor information
- Type of request - Initial or recurring
- Shop/Process code - a code that can be obtained from either the hazardous material center or team.
- Command/Organization Office Symbol
- Work Center Title
- Supply Account Code (cost account code)
- Building Number
- Location: Specific location in the building where hazmat is used.

Material Information:
- Material Name
- NSN/LSN
- Unit of Issue
- Material Specification: MilSpec or Commercial Spec
- Draw Amount: unit of measure to perform task
- Draw Frequency
- Sole Source Manufacturer Name/Cage
- Sole Source Part Number/Trade Name
- MSDS numbers if known
- Description

Requiring Documents (e.g., technical order/manual or owners' manual)
- Document Number
- Paragraph Number
- Page Number
- Revision/Change Number
- Revision Change Date

Process Information
- Is this request for a new workload or process?
- Task - A full description of the work activity and process in which the hazardous material in question is used (i.e., application method, type of industrial equipment.)
- Amount of Material used per task
- Frequency of task
- Duration of task
- Describe any engineering controls in use during the process.
- Indicate any PPE currently being used in conjunction with this process.
- Describe the method of disposal for all waste that is generated.

**Exception:** If requestor does not know the shop/process code, requestor must get data from Hazardous material team or center before completing request. If requestor does not know material information, requestor contacts installation logistics office.
A2. Requestor submits the form.

**Description:** The requestor submits the completed hazardous material authorization request to the organization responsible for validating the request.

**Alternative:** Requestor cannot complete all information, so keeps form until all information is available and submits at a later date.

A3. Validation team member accepts form.

**Description:** When form is successfully completed, a member of the hazardous material validation team accepts the form from the requestor, assigns a control number, and provides the control number to the requestor.

**Data Required:** Control # from validation team log book.

**Exception:** Team member rejects incomplete form and returns form to requestor without assigning a control number.
APPENDIX B – UNSTRUCTURED SCENARIO INSTRUCTIONS

B.1 Generating Scenarios

What is a Scenario?
A Scenario describes a situation or a process as a story told as an ordered sequence of events of activities. A scenario can be thought of as: (1) a particular recurring situation within an organization for which documentation is required, (2) a set of situations that describe a typical class of problems addressed by an organization or system, or (3) the setting within which a process occurs.

Scenario Development Guidelines.
1. Identify the parent business activity to which the scenario applies.
2. Identify from who's viewpoint the scenario is being described.
3. Provide a brief description of the scenario (a name or one sentence description).
4. List the steps that must be performed to accomplish the scenario.

B.2 Scenario Examples

B.2.1 Outline Scenario (Example 1)

Activity: Provide ATM Service
Viewpoint (Who): Customer
Scenario 1: Customer wants to withdraw $100.00 cash from checking account.
2. Machine asks customer to enter PIN.
3. Customer enters PIN.
4. ATM System checks if PIN is correct.
5. If PIN is incorrect, the machine asks the customer to reenter PIN. If customer enters the wrong PIN three times the machine keeps the card and notifies the customer who to contact at the bank.
6. If PIN is correct, machine asks customer to select the type of transaction.
7. Transaction types include: withdraw cash, deposit money, check account balance.
9. Machine asks customer what account the customer wants to withdraw the cash from.
10. Account types are checking or savings.
13. Customer enters $100
14. Machine gives customer $100, a receipt, and ejects the ATM card from the machine.
B.2.2 Outline Scenario (Example 2)

Activity: Provide ATM Service
Viewpoint (Who): Customer
Scenario 2: Customer wants to check savings account balance.
Steps 1 - 6 are the same as for Scenario 1.
10. Machine prints out receipt and asks customer if he wants to do another transaction.
11. If the customer says yes, the machine returns to step 6.
    If the customer says no, the machine ejects the ATM card from the machine.

B.2.3 Narrative Scenario (Example 3)

Scenario: I have been told to reduce my generation of hazardous waste.
Steps: I would go to the P2 Plan or the listing of hazardous chemicals or my P2 coordinator to determine what and how much hazardous waste I am generating (if I didn’t already know). I would then look in the lessons learned to see if anyone had the same problems that I have and then would use whatever they were using assuming they had the same operation. I might check to see if the change has been approved and at what level. I would then check to see if there was procurement guidance, someone who would help me install, startup and operate any changes. I would look for operating and maintenance manuals. I would also look for some training on the new material or process. Also, sometime in here I would place a request for procurement. I would need to know to get the funding necessary, i.e., how to get in the budget, if necessary.
APPENDIX C – EXPERIMENT DOCUMENTATION

C.1 Experiment Outline/Script

**Min Description**

10 Random Assignment of Students to Treatments & Letting them get settled
To ensure that I conduct each experiment in the same way, I will be reading from a
prepared script.

5 Introduction/Purpose of Experiment
The purpose of this experiment is to explore ways of gathering information systems
requirements directly from users through the use of scenarios. A scenario is a narrative
description of activities that a user engages in when performing a specific task. It
describes a user’s view of what happens, how it happens, and why in sufficient detail so
that requirements can be identified and analyzed. During the experiment, you will use a
GroupSystems tool, GroupOutliner, to define scenarios for the U of A’s Course
Registration System. The experiment will last approximately 2 hours and will include a
short GroupOutliner training session, a practice exercise, definition of the Course
Registration scenarios, and completion of a post-session questionnaire. Your
participation in this experiment is voluntary but you must complete the entire experiment
to earn the two points extra credit in MIS 341. All your responses through-out this
experiment are completely anonymous.

15 Training
Group Outliner: Add item (position, +, type, choose where, submit), Edit item (F5, type,
submit) Move item; Add description (double-click, type, submit, close window), Edit
description, Collapse/Expand Tree). Now highlight the outline item “Group Outliner
Practice” and try all the skills I just showed you – add item, edit item, move item, add
description, edit description, collapse/expand tree.

Scenarios: As I mentioned previously, a scenario is a narrative description of activities
that a user engages in when performing a specific task. To develop a scenario
description, you must specifically identify each action that must be accomplished. For
each action or step in the scenario, you must identify who performs that step (a person or
a system) and what specific action is done. You should also identify any data or
information that is needed to complete that step, the reason the step is done, any major
error checking or exceptions that may occur, as well as alternative ways of
accomplishing that step. These instructions are included in your scenario hand-out. Your
hand-out also includes a sample scenario “Student Checks Semester Grades using UA
Info”. Please review the sample scenario now. **PAUSE.**
Your hand-out also includes specific instructions for how you should enter scenarios in GroupSystems Group Outliner. Please read these instructions now and then browse the Checking Grades scenario which has already been added to your Group Outliner Session.

15 Practice (UA Info Student Schedule)
Now that you are familiar with Group Outliner and Scenarios, let's create a practice scenario. The scenario I want you to create is for a student who wants to check their current semester's class schedule using UA Info. To shorten the scenario, please assume that the student is already successfully logged on to Student Link and is at the Student Link main screen. All of you should highlight the scenario name, "Check Semester Schedule using UA Info" on your screen. Now, follow the instructions in your hand-out and the sample scenario provided to define the steps a student goes through to check their schedule on UA Info. We will spend approximately 10 minutes doing this exercise.

45 Scenario Generation for Spring Semester Courses
Now we are ready to begin the main part of this experiment. Please look at Section 3 of your outline. You will spend the next 45 minutes describing several scenarios related to the U of A's Course Registration System. Please start with the first scenario "Student Registers for Spring Semester Classes". If you complete that scenario description, you may move on to "Student Adds a Class after Initial Registration" and begin describing that scenario. You should continue describing scenarios for the entire 45 minutes. You do not have to describe all the scenarios. The goal is to provide the best possible scenario description for the first scenario before you move on to succeeding scenario descriptions. Use your Scenario Hand-Out and Instructions as a reminder of what should be included in a good scenario description and how you should use Group Outliner to enter your description. Please begin describing the first Course Registration scenario now.

3. UA Course Registration Scenarios
3.1 Student Registers for Spring Semester Classes
3.2 Student Adds a Class after Initial Registration
3.3 Student Drops a Class
3.4 Student Changes Sections for a Class

15 Post-Session Questionnaire
The final activity in this experiment is completion of the post-session questionnaire. In this questionnaire, when I ask you about "the method", I mean the way that I have asked you to describe your scenarios. While you are completing this questionnaire, please pay particular attention to any questions you have difficulty understanding. The last question of this survey asks you to tell me about these problems.
To complete this questionnaire, you will use another GroupSystems tool, Group Survey. … Provide brief instructions…
When you have finished the questionnaire and have submitted it by clicking on the Ballot Box at the top of the page, you may leave. Thank you very much for you participation.
C.2 Treatment 1 Instructions and Example

C.2.1 Scenario Description and Instructions

What is a scenario?
A scenario is a narrative description of activities that a user engages in when performing a specific task. It describes a user's view of what happens, how it happens, and why in sufficient detail so that requirements can be identified and analyzed.

What should a scenario description include?
To develop a scenario description, you must specifically identify each action that must be accomplished. For each action or step in the scenario, you must identify who performs that step (e.g., a person or a system) and what specific action is done. You should also identify any data/information that is needed to complete that step, the reason the step is done, any major error checking or exceptions that may occur, as well as alternative ways of accomplishing that step.

How to enter a scenario description in Group Outliner.
1. Use your mouse to highlight the name of the scenario you wish to create or review.
2. Double-click on the scenario name. A new window for text descriptions will open up on your screen.
3. To add a scenario description, simply type the description in the bottom portion of the text window and then click “Submit” when done.
4. To edit an existing scenario description, simply double-click on the description in the upper portion of the text window. The description is copied into the lower editing window. You can edit the description as required, then click “Submit” when done.
5. To review a long scenario description, you may need to click on the “More” button beside the description.
6. Close the text window by clicking on the “Close” button in the editing window.
C.2.2 Scenario Example 1: Student Checks Grades using UA Info

The student starts whatever Internet Browser (e.g., Netscape Navigator or Internet Explorer) is available on a computer they are authorized to use which has World Wide Web access. The system starts the browser and displays the browser's main screen. The student tells the browser to go to the UA's main web site (www.arizona.edu). The system then displays the UA's main web page. The student selects the "Student Information" hypertext link from that page as the most likely location for grades based on its description. The system displays the Student Information web page. The student then selects the "Student Link" link again based on its description. Temporarily, the system displays a web page which asks the student to select Student Link Version 1 or Version 2 which are each briefly described. Assuming the student selects Version 2 since the system says it has a nicer, simpler user interface, the system will then display the secure logon screen which asks the student to enter his/her Student ID Number and Personal Identification Number (PIN) to ensure that students only access their own information. The student enters the ID and PIN and then clicks the Login button. The system then validates the ID and PIN and requests the student re-enter them if they are not valid. If the student does not enter a valid ID and PIN or encounters any error in the previous steps that he/she can't fix, the student cannot look-up grades using UA Info's web access from that computer. The student can try another computer or use an alternative method for checking grades (e.g., RSVP, check with professor, wait for grades in mail, or check at Registrar's Office). If the ID and PIN were entered correctly, the system displays the main Student Link Screen. The student then clicks on the Icon for Grades. The system asks the student to select the term (semester and year) of the grades to display. The student selects the term from a pull-down menu and then clicks on the display grades button. The system then displays a form listing all the classes the student is registered for during that semester and the grades for each class if they have been posted to the system. The system also displays the semester and cumulative GPA. Students then have the choice of requesting grades for a different semester, displaying all grades, choosing another Student Link option, or logging off Student Link. When the student is done with Student Link, they should click on the Logoff icon. The system then displays a confirmation message and then asks the student if they want to return to Student Link or UA Info. The student normally would choose UA Info because they just exited Student Link. In this case, the system would then display the UA's main web page. From there the student could continue to browse the UA's web site, browse other sites, or exit the browser since the task is complete. The student would then logoff (if required) the system and turn off the computer power.
C.3 Treatment 2 Instructions and Example

C.3.1 Scenario Description and Instructions

What is a scenario?
A scenario is a narrative description of activities that a user engages in when performing a specific task. It describes a user's view of what happens, how it happens, and why in sufficient detail so that requirements can be identified and analyzed.

What should a scenario description include?
To develop a scenario description, you must specifically identify each action that must be accomplished. For each action or step in the scenario, you must identify who performs that step (e.g., a person or a system) and what specific action is done. You should also identify any data/information that is needed to complete that step, the reason the step is done, any major error checking or exceptions that may occur, as well as alternative ways of accomplishing that step.

How to enter a scenario description in Group Outliner.
1. Use your mouse to highlight the name of the scenario you wish to create or review.
2. Expand the Outline by clicking the “>>” button if required to see any existing steps for that scenario.
3. To add a step for that scenario, click the “+” button to open the “add an item to the outline window”. Type the complete description of the step, then select “Sub” for the location and “Submit” to add that step under the Scenario Name which is highlighted. Note: If an existing scenario step is highlighted (versus the scenario name), then select “Append” and “Submit” to add your new step after the existing step.
4. To edit an existing scenario step, press F5. The description is copied into an editing window. You can edit the description as required, then click “Submit” when done.
5. Close the add item window by clicking on the “Close” button in that window.
C.3.2 Scenario Example 2: Student Checks Grades using UA Info

1. The student starts whatever Internet Browser (e.g., Netscape Navigator or Internet Explorer) is available on a computer they are authorized to use which has World Wide Web access.
2. The system starts the browser and displays the browser’s main screen.
3. The student tells the browser to go to the UA’s main web site (www.arizona.edu).
4. The system then displays the UA’s main web page.
5. The student selects the “Student Information” hypertext link from that page as the most likely location for grades based on its description.
6. The system displays the Student Information web page.
7. The student then selects the “Student Link” link, again based on its description.
8. Temporarily, the system displays a web page which asks the student to select Student Link Version 1 or Version 2 which are each briefly described.
9. Assuming the student selects Version 2 since the system says it has a nicer, simpler user interface, the system will then display the secure login screen which asks the student to enter his/her Student ID Number and Personal Identification Number (PIN) to ensure that students only access their own information.
10. The student enters the ID and PIN and then clicks the Login button.
11. The system then validates the ID and PIN and requests the student re-enter them if they are not valid. If the student does not enter a valid ID and PIN or encounters any error in the previous steps that he/she can’t fix, the student cannot look-up grades using UA Info’s web access from that computer. The student can try another computer or use an alternative method for checking grades (e.g., RSVP, check with professor, wait for grades in mail, or check at Registrar’s Office).
12. If the ID and PIN were entered correctly, the system displays the main Student Link Screen.
13. The student then clicks on the Icon for Grades.
14. The system asks the student to select the term (semester and year) of the grades to display.
15. The student selects the term from a pull-down menu and then clicks on the display grades button.
16. The system then displays a form listing all the classes the student is registered for during the selected semester and the grades for each class if they have been posted to the system. The system also displays the semester and cumulative GPA.
17. Students then have the choice of requesting grades for a different semester, displaying all grades, choosing another Student Link option, or logging off Student Link. When the student is done with Student Link, they should click the Logoff icon.
18. The system then displays a logoff confirmation message and then asks the student if they want to return to Student Link or UA Info.
19. The student normally would choose UA Info since they just exited Student Link. In this case, the system would then display the UA’s main web page.
20. From there the student could continue to browse the UA’s web site, browse other sites, or exit the browser since the task is complete.
C.4 Treatment 3 Instructions and Example

C.4.1 Scenario Description and Instructions

What is a scenario?
A scenario is a narrative description of activities that a user engages in when performing a specific task. It describes a user’s view of what happens, how it happens, and why in sufficient detail so that requirements can be identified and analyzed.

What should a scenario description include?
To develop a scenario description, you must specifically identify each action that must be accomplished. For each action or step in the scenario, define a step name of the form “Who+Verb+Object”, e.g., Student Starts Internet Browser. For each step name, you must provide a complete description of that step to identify who performs that step (e.g., a person or a system) and what specific action is done. You should also identify any data/information that is needed to complete that step, the reason the step is done, any major error checking or exceptions that may occur, as well as alternative ways of accomplishing that step.

How to enter a scenario description in Group Outliner.
1. Use your mouse to highlight the name of the scenario you wish to create or review.
2. Expand the Outline by clicking the “>>” button if required to see any existing steps for that scenario.
3. To add a step name for that scenario, click the “+” button to open the “add an item to the outline window”. Type the name that step as “Who + Verb + Object”, then select “Sub” for the location and “Submit” to add that step under the Scenario Name which is highlighted. Note: If an existing scenario step is highlighted (versus the scenario name), then select “Append” and “Submit” to add your new step after the existing step.
4. To add or review the complete description for a scenario step, double-click on the scenario step name. A new window for text descriptions will open up on your screen.
5. To add a scenario step description, simply type the description in the bottom portion of the text window and then click “Submit” when done.
6. To edit an existing scenario step description, simply double-click on the description in the upper portion of the text window. The description is copied into the lower editing window. Edit the description as required, then click “Submit” when done.
7. To review a long scenario step description, you may need to click on the “More” button beside the description.
8. Close the text window by clicking on the “Close” button in the editing window.
9. To edit an existing scenario step name, press F5. The description is copied into an editing window. Edit the description as required, then click “Submit” when done.
10. Close the add outline item window by clicking on the “Close” button in that window.
C.4.2 Scenario Example 3: Student Checks Grades using UA Info

1. **Student Starts Internet Browser**
The student starts whatever Internet Browser (e.g., Netscape Navigator or Internet Explorer) is available on a computer they are authorized to use which has World Wide Web access.

2. **System Displays Browser's Main Screen**
The system starts the browser and displays the browser's main screen.

3. **Student Requests UA's Web Site**
The student tells the browser to go to the UA's main web site (www.arizona.edu).

4. **System Displays UA's Web Page**
The system then displays the UA's main web page.

5. **Student Selects "Student Information"**
The student selects the "Student Information" hypertext link from that page as the most likely location for grades based on its description.

6. **System Displays Student Information Web Page**

7. **Student Selects "Student Link"**
The student then selects the "Student Link" link, again based on its description.

8. **Student Selects Version 2 of Student Link**
Temporarily, the system displays a web page which asks the student to select Student Link Version 1 or Version 2 which are each briefly described. Assumption is that the student selects Version 2 since the system says it has a nicer, simpler user interface.

9. **System Requests Student ID and PIN**
The system will then display the secure login screen which asks the student to enter his/her Student ID Number and Personal Identification Number (PIN) to ensure that students only access their own information.

10. **Student Enters ID and PIN**
The student enters the ID and PIN and then clicks the Login button.

11. **System Validates ID and PIN**
The system then validates the ID and PIN and requests the student re-enter them if they are not valid. If the student does not enter a valid ID and PIN or encounters any error in the previous steps that he/she can't fix, the student cannot look-up grades using UA Info's web access from that computer. The student can try another computer or use an
alternative method for checking grades (e.g., RSVP, check with professor, wait for grades in mail, or check at Registrar's Office).

12. System Displays Student Link Main Screen
If the ID and PIN were entered correctly, the system displays the main Student Link Screen.

13. Student Clicks Grades Icon
The student then clicks on the Icon for Grades.

14. System Requests Term
The system asks the student to select the term (semester and year) of the grades to display.

15. Student Selects Term
The student selects the term from a pull-down menu and then clicks on the display grades button.

16. System Displays Grades
The system then displays a form listing all the classes the student is registered for during the selected semester and the grades for each class if they have been posted to the system. The system also displays the semester and cumulative GPA.

17. Student Selects Logoff
Students then have the choice of requesting grades for a different semester, displaying all grades, choosing another Student Link option, or logging off Student Link. When the student is done with Student Link, they should click on the Logoff icon.

18. Student Requests UA Info
The system then displays a logoff confirmation message and then asks the student if they want to return to Student Link or UA Info. The student normally would choose UA Info since they just exited Student Link.

19. System Displays UA’s Web Page
In this case, the system would then display the UA’s main web page.

20. Student Exits Browser
From there the student could continue to browse the UA’s web site, browse other sites, or exit the browser since the task is complete.
C.5 Post-Session Questionnaire

1. Terminal number.
2. The quality of the scenario descriptions I developed was: (Excellent, Very Good, Good, Fair, Poor)
3. This method allowed me to do everything I needed to do to define scenarios. (Strongly Agree to Strongly Disagree)
4. Another UA student could easily understand the meaning of the scenario descriptions I developed with a minimum of explanation. (Strongly Agree to Strongly Disagree)
5. Using this method to define scenarios enabled me to accomplish this task quickly. (Strongly Agree to Strongly Disagree)
6. The scenario descriptions I developed accurately describe the current UA Course Registration process. (Strongly Agree to Strongly Disagree)
7. I was very motivated to make this experiment a success. (Strongly Agree to Strongly Disagree)
8. The scenario descriptions I developed were unambiguous because the descriptions have only one possible interpretation. (Strongly Agree to Strongly Disagree)
9. Using this method increased my efficiency at scenario generation. (Strongly Agree to Strongly Disagree)
10. The scenario descriptions I developed were: (Very Complete, Mostly Complete, Average, Somewhat Incomplete, Very Incomplete)
11. An analyst could easily understand the meaning of the scenario descriptions I developed with a minimum of explanation. (Strongly Agree to Strongly Disagree)
12. The scenario descriptions I developed included the information requested in the instructions. (Strongly Agree to Strongly Disagree)
13. Describing scenarios using this method was: (Very Easy to Very Hard)
14. The scenario descriptions I developed were: (Completely Correct, Mostly Correct, Average, Somewhat Incorrect, Mostly Incorrect)
15. The scenario descriptions I developed identified the data used during each scenario step. (Strongly Agree to Strongly Disagree)
16. I spent my time efficiently describing the requested scenarios. (Strongly Agree to Strongly Disagree)
17. The scenario descriptions I developed were: (Very Clear, Mostly Clear, Average, Somewhat Vague, Very Vague,)
18. The scenario descriptions I developed identified the exceptions or problems that could occur during the UA Course Registration Process. (Strongly Agree to Strongly Disagree)
19. I put a lot of effort into developing good scenario descriptions. (Strongly Agree to Strongly Disagree)
20. The UA Registrar could easily understand the meaning of the scenario descriptions I developed with a minimum of explanation. (Strongly Agree to Strongly Disagree)
21. I was able to describe scenarios using this method: (Very Quickly, Quickly, Average, Slowly, Very Slowly)
22. The scenario descriptions I developed identified alternative ways of registering for classes. (Strongly Agree to Strongly Disagree)
23. Learning to operate GroupSystems was easy for me. (Strongly Agree to Strongly Disagree)
24. I found it easy to get GroupSystems to do what I wanted it to do. (Strongly Agree to Strongly Disagree)
25. My interactions with GroupSystems were clear and understandable. (Strongly Agree to Strongly Disagree)
26. I found GroupSystems flexible to interact with. (Strongly Agree to Strongly Disagree)
27. It was easy for me to become skillful at using GroupSystems. (Strongly Agree to Strongly Disagree)
28. Interacting with GroupSystems did not require a lot of mental effort. (Strongly Agree to Strongly Disagree)
29. I found GroupSystems easy to use. (Strongly Agree to Strongly Disagree)
30. What is your current class standing? (Freshman, Sophomore, Junior, Senior, Other)
31. How many years have you been taking classes at the University of Arizona. (Number)
32. What is your major? (MIS, Business (not MIS), Computer Science, Science (not CS), Other)
33. List all the MIS and CS classes (by course number) which you have taken as a college student (e.g., MIS 111). (Open)
34. How often do you use a computer? (Several times each Day, Once a Day, Several times each Week, Once a Week, Rarely.)
35. Select the answer which best describes your level of computer expertise. (Excellent, Very Good, Good, Fair, Poor)
36. How would you rate your word processing expertise? (Excellent, Very Good, Good, Fair, Poor)
37. How quickly do you type? (Very Quickly, Quickly, Average, Slowly, Very Slowly)
38. Please rate on a scale from 1 to 5 how well you type with 5 = Excellent Typist, 3 = Average, 1 = Hunt and Peck.
39. Before today, how many times have you used GroupSystems? (Many times, Several times, Couple of times, One time, Never)
40. Was there anything that made any part of this experiment difficult to accomplish? (Open)
41. Were there any questions in this survey which you did not understand or had difficulty answering? Please identify them by question number and explain your concerns. (Open)
# Question Description | Factor 1 Quality | Factor 2 Ease of Use | Factor 3 Productivity
--- | --- | --- | ---
Q2 Overall scenario quality | .74 | -.02 | .34
Q4 Scenarios understandable by students | .70 | -.01 | .14
Q6 Scenarios accurate | .78 | .15 | .02
Q8 Scenarios unambiguous | .45 | .11 | .15
Q10 Scenarios complete | .77 | .20 | .08
Q11 Scenarios understandable by an analyst. | .56 | .24 | .23
Q12 Scenarios included requested information | .40 | .17 | -.03
Q14 Scenarios correct | .77 | .03 | .06
Q15 Scenarios included data used | .35 | .16 | .13
Q17 Scenarios clear | .84 | .04 | .13
Q18 Scenarios identified exceptions/problems | * | * | *
Q20 Scenarios understandable by registrar | .68 | .14 | .15
Q22 Scenarios identified alternatives | * | * | *
Q3 Method supported scenario definition | .34 | .33 | .47
Q5 Defined scenarios quickly | .07 | .24 | .80
Q9 Defined scenarios efficiently | .06 | .39 | .66
Q13 Scenario definition method easy | .15 | .40 | .37
Q16 Used time efficiently | .33 | .11 | .43
Q21 Scenario definition speed | .24 | .25 | .61
Q23 GS easy to learn | .33 | .56 | .14
Q24 GS easy to do what wanted | .14 | .88 | .24
Q25 GS interactions clear/understandable | .19 | .89 | .13
Q26 GS flexible to interact with | .05 | .67 | .48
Q27 GS easy to become skillful | .26 | .75 | .21
Q28 GS did not require a lot of mental effort | .02 | .47 | .09
Q29 GS easy to use | .05 | .85 | .32

* Questions on alternatives and exceptions dropped due to low commonality with other variables.

Table 16 – Factor loadings
## E.1 SPA Functional Requirements List

<table>
<thead>
<tr>
<th>Functional Requirements</th>
<th>V1</th>
<th>V2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1. Process Tree Hierarchy</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.1. Create Process Tree</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.1.1. Root = Session Name</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.1.2. Orphan</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.1.3. Marked for Deletion</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td><strong>1.2. Add Process</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.2.1. After (default)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.2.2. Child</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.2.3. Before</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.2.4. To session/main process</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.2.5. As orphan</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.2.6. Long names</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.2.6.1. Limit to 60</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.2.6.2. Scroll to see</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.2.7. Focus on new process</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.2.8. New roots (Fac only)</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td><strong>1.3. Modify Name</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.3.1. Change Name (Except 3 roots)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.3.2. No change to blank</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.3.3. Change session name, changes root node</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.3.4. Change root names (Fac only)</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td><strong>1.4. Move Process in Tree</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.4.1. Cut &amp; Paste</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.4.1.1. Cut &amp; paste entire branch w/all info</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.4.1.1.1. Paste After (default)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.4.1.1.2. Paste Child</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.4.1.1.3. Paste Before</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.4.1.2. Cut &amp; paste = Change parent</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.4.1.3. Undo cut if no paste</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.4.2. Drag &amp; Drop</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.4.2.1. Functions like Cut &amp; Paste</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td><strong>1.5. Copy Process</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.5.1. Copy Branch w/all info (panels)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.5.2. Copy Process w/all info (panels)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.5.3. Copy Process w/ProcTable info only</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>1.5.4. Copy Process Name only</td>
<td>Y</td>
<td></td>
</tr>
</tbody>
</table>
## Functional Requirements

<table>
<thead>
<tr>
<th>1.6. Paste Process</th>
<th>V1</th>
<th>V2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.6.1. Paste Branch w/all info (panels)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.6.2. Paste Process w/all info (panels)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.6.3. Paste Process w/ProcTable info</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>1.6.4. Paste Process Name only</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1.6.5. Paste After (default)/Child/Before</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>

### Delete Process

| 1.7.1. Move Branch to “Marked for Deletion” | Y |
| 1.7.2. Move Process to “Marked for Deletion” | Y |

| 1.7.2.1. Move Children to new parent | Y |
| 1.7.3. Delete/Move “Orphan” (Fac only) | N | Y |

### Purge Mark for Deletion – (Fac only)

| 1.8.1. Purge All | Y |
| 1.8.2. Purge Branch | Y |
| 1.8.3. Purge Single Process (No Children) | Y |
| 1.8.4. Purge Process Only/Move Children | Y |

### Collapse/Expand Tree

| 1.9.1. Default = Fully Expanded | Y |
| 1.9.1.1. Start Fully Expanded | Y |
| 1.9.1.2. Facilitator Directed Start | Y |

| 1.9.2. Collapse Branch at a time (User) | Y |
| 1.9.3. Collapse All (User) | Y |
| 1.9.4. Retain User’s Selections | Y |

| 1.9.5. Facilitator Directed Collapse | Y |

### Mark Process Nodes

| 1.10. | Y |

### Indicate nodes with new information

| 1.11. | Y |

## Information Panels

### Create Default Panels

| 2.1.1. 9 Default Text Panels | Y |

| 2.1.2. Overview | Y |
| 2.1.3. Actor | Y |
| 2.1.4. Object | Y |
| 2.1.5. Metrics | Y |

### General Text Panels

| 2.2.1. Display comments for process | Y |

| 2.2.1.1. Display comments in entry order | N | Y |
| 2.2.1.2. Scroll if multiple comment screens | Y |
| 2.2.1.3. Managing long comments | Y |

<p>| 2.2.1.4. Display comment #/author/date/time | Y |
| 2.2.1.5. Display #comments on Panel Tab | Y |</p>
<table>
<thead>
<tr>
<th>Functional Requirements</th>
<th>V1</th>
<th>V2</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2.2. Add new comments</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.2.2.1. Add as last comment (default)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.2.2.2. Add before/after a comment</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.2.2.3. Store author/date/time added</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.2.3. Select comment (click)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.2.4. Edit existing comment</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.2.4.1. Select (click) then Edit button</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.2.4.2. Double-click to Select &amp; Edit</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.2.5. Cut comment</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.2.6. Copy comment</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.2.7. Paste comment – After/Child/Before</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.2.8. Delete comment</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.2.8.1. Request confirmation before delete</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.2.9. Author/Comment Tracking</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.2.10. Mark comment (agree/disagree)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.2.11. Mark panel (agree/disagree)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.3. Overview Panel</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.3.1. Display Parent &amp; # Children</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.3.2. Add/edit other data fields</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.3.3. Inherit parent’s viewpoint/allow change</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.3.4. Show Sys Auto Candidate in Process Tree</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.3.5. Allow field name changes (e.g., Sys Auto)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.4. Actor Panel</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.4.1. Add Actor to Process</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.4.1.1. Select actor from pull-down list</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.4.1.2. Assign actor role from pull-down list</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.4.1.3. Display actors in order added</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.4.1.4. Display in alpha or role order (Fac option)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.4.2. Display pull-down lists in Alpha order</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.4.3. Add Actor to Master Actor List</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.4.4. Edit Actor in Master Actor List</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.5. Object Panel</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.5.1. Add Object to Process</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.5.1.1. Select object from pull-down list</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.5.1.2. Display objects in order added</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.5.1.3. Display in alpha order (Fac option)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.5.2. Display pull-down lists in alpha order</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.5.3. Add Object to Master Object List</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.5.4. Edit Object in Master Object List</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.6. Metrics Panel</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.6.1. Add/edit current data fields</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td><strong>Functional Requirements</strong></td>
<td>V1</td>
<td>V2</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td>2.6.2. Add/edit new/other data fields</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.7. Other Fixed Panels</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>2.8. View Process Information</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3. Graphical View</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.1. Add Link</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.1.1. From/To Process/Junction</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.1.2. Only Legal To’s</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.1.2.1. Not to self</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.1.2.2. Not duplicate link</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.1.2.3. Warn on circularity</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.1.3. Create default links from Process tree</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.2. Move Link</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.2.1. Need to check legal to’s</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.3. Change Link Information</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.3.1. Optional Label</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.3.2. Add/edit multiple Object &amp; Opt State pairs</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.4. Delete Link</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.4.1. Manual Delete</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.4.2. Automatic Delete (if to/from deleted)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.5. Add Junction</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.5.1. Manual Add</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.5.2. Automatic Creation</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.6. Edit Junction Information</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.6.1. Optional Label</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.6.2. Junction Type</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.6.3. Junction Logic</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.7. Delete Junction</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.7.1. Manual Delete (w/associated links)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.7.2. Automatic Delete if no information/not required</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.8. Graphical Display</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.8.1. Automatic layout</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.8.1.1. Minimize link cross-overs</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.8.1.2. Maximize info on single screen</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.8.2. Portrait layout</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.8.3. Landscape layout</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.8.4. UML Activity Diagram standard</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.8.4.1. Include basic activity/junction/links</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.8.4.2. Add additional features from standard</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>3.8.5. Support other standards, e.g., IDEF3</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4. Facilitator</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.1. Defining/Changing Session Privileges</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Functional Requirements</td>
<td>V1</td>
<td>V2</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td>4.1.1. General Process Privileges</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.1.1. Process Tree – Add/Mod/Del</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.1.2. Master Actor List – Add/Mod/Del</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.1.3. Master Object List – Add/Mod/Del</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.1.4. Link – Add/Mod/Del</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.1.5. Link Info – Mod</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.1.6. Junctions – Add/Mod/Del</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2. Process Panel Privileges</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2.1. Privileges for Each Panel</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2.1.1. Visible</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2.1.2. Add/Mod/Delete</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2.2. Modify Panel Info</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2.2.1. Change panel name</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2.2.1.1. Do not allow dupe panel names</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2.2.2. Change panel public prompt</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2.2.3. Change panel mouse over text</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2.2.4. Live display on user text panels</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2.2.5. Live display on user fixed panels</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2.2.6. Change panel display sequence</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2.3. Add New Text Panels</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2.3.1. Add panel as last text panel</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2.3.2. Add panel in selected position</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2.3.3. Live update of user displays</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2.4. Delete Panels</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2.4.1. Request confirmation before delete</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2.4.2. No delete if process info for panel</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2.4.3. No delete of fixed panels</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.2.5. Live display update for all changes</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.3. General Session Privileges</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.3.1. Display Comment #/Author/Date/Time</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.1.3.2. Other privileges</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.2. View Process Information Counts</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.3. Lock Process Nodes</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.4. Clear all locks</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.5. Change facilitator view</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.5.1. User View (Just user panels)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.5.2. Facilitator View (User panels &amp; Marked for Deletion)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.5.3. Complete View (All Panels &amp; Marked for Deletion)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.6. Change participant view</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.6.1. Match facilitator view</td>
<td>Y</td>
<td></td>
</tr>
</tbody>
</table>
### Functional Requirements

<table>
<thead>
<tr>
<th>Requirement</th>
<th>V1</th>
<th>V2</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.6.1.1. Move to current process</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.6.1.2. Move to current process &amp; panel</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.6.1.3. Request user permission to match view</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.6.2. Follow facilitator view</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.6.2.1. Move to process when facilitator does</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.6.2.2. Move to process &amp; panel with facilitator</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.6.2.3. Request user permission to follow</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>4.6.2.4. Allow user to break-off follow</td>
<td>Y</td>
<td></td>
</tr>
</tbody>
</table>

#### 5. Report

<table>
<thead>
<tr>
<th>Requirement</th>
<th>V1</th>
<th>V2</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1. Generate HTML Report w/all text info</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>5.1.1. Sequence of report same as on screen</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.2. Allow user-selection of info to include</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>5.2.1. Include processes/sections w/info (default)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>5.2.2. Include process tree</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>5.2.3. Include which process panels</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>5.2.4. Include all/selected processes</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>5.2.5. Include graphical view, which levels</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>5.3. Provide other report formats</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>5.3.1. Provide MS Word/RTF report format</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>5.3.2. Modify HTML format for easier Word import</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>5.4. Allow report/print-out of graphical diagrams</td>
<td>Y</td>
<td></td>
</tr>
</tbody>
</table>

#### 6. Menu

<table>
<thead>
<tr>
<th>Requirement</th>
<th>V1</th>
<th>V2</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1. File</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>6.1.1. Change session</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.1.2. Report</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>6.1.3. Export as File</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.1.4. Exit</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>6.2. Edit</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>6.2.1. Cut</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>6.2.2. Copy</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>6.2.3. Paste</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>6.2.4. Delete</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>6.2.5. Select All</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>6.2.6. Find</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>6.2.7. Replace</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>6.3. View</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>6.3.1. Toggles</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>6.3.1.1. Process Tree</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>6.3.1.2. Process Panels</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>6.3.1.3. Graphical View</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>6.3.1.4. Status Bar</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Functional Requirements</td>
<td>V1</td>
<td>V2</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td>6.3.2. Chat Window</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.3.3. Group Members</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.3.3.1. Show all members of group</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>6.3.3.2. Indicate which currently logged in</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>6.3.3.3. Show when users last logged in</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.3.4. Process Information Counts</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>6.3.5. Refresh</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>6.3.5.1. Refresh process tree</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>6.3.5.2. Refresh process information panels</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>6.3.5.3. Refresh graphical view</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>6.4. Facilitator</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.4.1. Privileges</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>6.4.2. Process Information Counts</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>6.4.3. Toggle</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.4.3.1. User View (Just user panels)</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>6.4.3.2. Facilitator View (User panels &amp; Marked for Deletion)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>6.4.3.3. Complete View (All Panels &amp; Marked for Deletion)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>6.5. Help</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>6.6. General capabilities</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.6.1. Gray/Invisible menu items if no rights</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>6.6.2. Gray/Invisible menu items if not implemented</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>6.6.3. Alt/Hot Keys</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>6.6.4. Control/Short-cut Keys</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>7. Other</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.1. Edit Locking</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.1.1. For DB integrity</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>7.1.1.1. Lock record (&amp; parent record)</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>7.1.2. Desired locking</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.1.2.1. Process tree – lock name only/parent</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>7.1.2.2. Text panel – lock individual comments</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>7.1.2.3. Fixed panels – panel data fields</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>7.1.3. Display lock indicator</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>7.1.4. Identify who has lock</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>7.1.5. Clear all locks (Fac only)</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>7.2. Dynamic IPs</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>7.3. Multiple Browsers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.3.1. Internet Explorer 4.01 (&amp; later)</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>7.3.2. Netscape Navigator 4.5 (&amp; later)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>8. System Administrator Module</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Functional Requirements</td>
<td>V1</td>
<td>V2</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td>8.1. Create users &amp; passwords</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>8.1.1. Administrators – all rights</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>8.1.2. Session facilitators/owners</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>8.1.2.1. Create new sessions</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>8.1.2.2. Maintain own sessions</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>8.1.2.3. Create new users</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>8.1.3. Users – maintain own record only</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>8.2. Maintain user list/passwords</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>8.2.1. Change user info &amp; password</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>8.2.2. Display users in alpha order by last name</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>8.2.3. Allow blank passwords</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>8.2.4. Create user groups</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>8.3. Create new session</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>8.3.1. Creator automatically User &amp; Facilitator</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>8.3.2. Create session tables</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>8.3.3. Defaults start/end date = today</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>8.4. Maintain session info</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>8.4.1. Display sessions in alpha order</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>8.4.2. Allow optional display of session in date order</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>8.5. Maintain session participants</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.5.1. Add users to session</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>8.5.2. Delete users from a session</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>8.5.3. Add/remove “Read-Only” users to session</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>8.5.4. Add owners/facilitators of a session</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>8.5.5. Remove session owners/facilitators</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>8.6. Copy session</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>8.7. Save session</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>8.8. Load session</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>8.9. Delete session</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>8.9.1. Request confirmation before delete</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>8.9.2. Delete session from roster</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>8.9.3. Delete session tables from DB</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>9. Import</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.1. Import from GroupSystems</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>9.1.1. Process Tree Hierarchy &amp; Descriptions</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>9.1.2. Processes &amp; Comments to Selected Panel</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>9.1.3. Actors &amp; Descriptions to Master Actor List</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>9.1.4. Objects &amp; Descriptions to Master Object List</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>9.1.5. Import to new SPA Session</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>9.1.6. Import to existing SPA Session</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>9.2. Import from Group Date Modeler</td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Functional Requirements</td>
<td>V1</td>
<td>V2</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td>9.2.1. Import Entities/Attributes/Descriptions to Object List</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>9.3. Import from Other Systems (MS Word, others)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>10. Export</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.1. Export to GroupSystems</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>10.1.1. Process Tree to outline</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>10.1.2. Process Tree with one text panel as comments</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>10.1.3. Process Tree with all text panels as comments</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>10.1.4. Process Tree with all panels as comments</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>10.1.5. Text panel to outline</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>10.2. Export to common format (e.g., tab delimited)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>10.2.1. For import to MS Project/Excel/others</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>10.3. Export to specific tool format</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>10.3.1. Graphical process modeling tool (ProSim/others)</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>10.3.2. Simulation tool</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>11. Distributed Session Support</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11.1. Chat window</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>11.2. Match/follow views</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>11.3. Group member status</td>
<td>Y</td>
<td></td>
</tr>
</tbody>
</table>

Key:


V2 – Version 2. Represents all future versions of SPA prototype. More detailed allocations will be made as part of planning for version 2.

Y – Yes. Functionality included in or added to specified version and all future versions (unless specifically excluded (indicated by N))

N – No. Function specifically blocked in version
E.2 SPA Logical Data Model
E.3 SPA Data Entity and Attribute Definitions

Entity Name: Abstract Process
Entity Definition: Either a Generic Process or a Junction. This abstract entity created because a link may have a process or junction as its source and/or destination.

Attribute Name: ID
Attribute Definition: ID # for Process, Internal/Assigned by Software
Column Datatype: INTEGER
Null Option: NOT NULL
PK: (PK)

Attribute Name: Pseudo_Type
Attribute Definition: Y if Pseudo-Process (Junction) N if Generic Process
Column Datatype: CHAR(1)
Null Option: NULL

Entity Name: Actor
Entity Definition: List of who or what does or participates in the processes. Can be an individual, group of individuals, or a system. Individuals usually identified by role name vs. person name.

Attribute Name: Role_Name
Column Datatype: CHAR(40)
Null Option: NOT NULL
PK: (PK)

Attribute Name: Actor_Desc
Column Datatype: LONG VARCHAR
Null Option: NULL

Entity Name: All_Text_Types
Entity Definition: Entity to store the number and name of the types of text/memo information that may be stored for a process

Attribute Name: Text_Type
Column Datatype: INTEGER
Null Option: NOT NULL
PK: (PK)

Attribute Name: Type_Name
Column Datatype: CHAR(20)
Null Option: NOT NULL
Type_Name: Alternative
Definition: Used to capture different way(s) of doing the process

Type_Name: Description
Definition: Used to capture the description of the process only. Objective is captured separately.

Type_Name: Exception
Definition: Used to capture error conditions, constraints on continuing.

Type_Name: Info_Resource
Definition: Used to capture objects, data attributes or entities needed to accomplish the process. These could be either data attributes (fields) from specific entities (tables) or just the entity (table) names; depending on the level of detail necessary.

Type_Name: Objective
Definition: Used to capture Objective, Purpose, Reason for doing the process, i.e., why are you doing the process, what do you expect to get.

Type_Name: Other_Resource
Definition: Used to capture any resources, other than information (captured as Info_Resources), needed to accomplish the generic process. Could be other people, systems, materials, products, production technology (equipment or machines), communication media, etc., that are consumed or needed during a process.

Type_Name: Post-Condition
Definition: Used to capture the Output, Post Condition, and/or Ending Constraints, Results for a process. This is not something that will usually be completed by SMEs, will be completed by the analysts, probably during use case development.

Type_Name: Prerequisite
Definition: Used to capture the Assumptions, Preconditions, and/or Starting Constraints for a process.

Type_Name: Related Issues
Definition: Used to capture other factors, constraints, states, pending changes, improvement ideas, disagreements, or any other information that is not captured elsewhere that influences the process.

Type_Name: Trigger
Definition: Used to capture what event/action jump starts this particular process/scenario.

Attribute Name: Type_Prompt
Attribute Definition: Prompt that will be shown on user's screen to ask for this type of information.
Column Datatype: CHAR(60)
Null Option: NULL

Attribute Name: Mouse_Text
Attribute Definition: Information that will display on status line when user moves mouse over text panel.
Column Datatype: CHAR(60)
Null Option: NULL

Entity Name: Any_Text
Entity Definition: Entity to store any individual memo entry for a process

Attribute Name: ID
Attribute Definition: ID # for Process, Internal/Assigned by Software
Column Datatype: INTEGER
Null Option: NOT NULL
PK: (PK)
FK: (FK)

Attribute Name: Text_Type
Column Datatype: INTEGER
Null Option: NOT NULL
PK: (PK)
FK: (FK)

Attribute Name: Text_No
Attribute Definition: Counter for Text which when combined with ID and Text_Type makes unique primary key. Internal/assigned by software.
Column Datatype: INTEGER
Null Option: NOT NULL
PK: (PK)

Attribute Name: Text
Attribute Definition: Free-format memo field
Column Datatype: LONG VARCHAR
Null Option: NOT NULL

Attribute Name: Text_Author
Attribute Definition: Identification of the individual who created or last modified the text.
Column Datatype: CHAR (20)
Null Option: NOT NULL

Attribute Name: Text_Date
Attribute Definition: Date text was created or last modified.
Column Datatype: DATE
Null Option: NULL

Attribute Name: Text_Time
Attribute Definition: Time text was created of last modified.
Column Datatype: TIME
Null Option: NULL

Entity Name: Children
Entity Definition: Tracks all the child processes of a process, so that you can move down the tree from parent to child.

Attribute Name: Child.ID
Attribute Definition: ID # for Process, Internal/Assigned by Software
Column Datatype: INTEGER
Null Option: NOT NULL
PK: (PK)
FK: (FK)

Attribute Name: Parent.ID
Attribute Definition: ID # for Process, Internal/Assigned by Software
Column Datatype: INTEGER
Null Option: NOT NULL
PK: (PK)
FK: (FK)

Entity Name: Generic Process
Entity Definition: Any type of process including business process, sub-process, process task, scenario, scenario action/step. Same as UOB in IDEF3/ProSim

Attribute Name: ID
Attribute Definition: ID # for Process, Internal/Assigned by Software
Column Datatype: INTEGER
Null Option: NOT NULL
PK: (PK)
FK: (FK)
Attribute Name: Process_Name
Attribute Definition: Verb + Object; Used as node name in outline view and box label in graphical view.
Column Datatype: CHAR(40)
Null Option: NOT NULL

Attribute Name: Viewpoint
Column Datatype: CHAR(40)
Null Option: NULL

Attribute Name: Parent.ID
Attribute Definition: ID # for Parent Process. Internal/Assigned by Software
Column Datatype: INTEGER
Null Option: NULL
FK: (FK)

Attribute Name: Cost
Attribute Definition: Average estimated cost of executing process one time
Column Datatype: DECIMAL()
Null Option: NULL

Attribute Name: Frequency
Attribute Definition: How often process is done
Column Datatype: INTEGER
Null Option: NULL

Attribute Name: Frequency_UOM
Attribute Definition: Unit of Measure for Frequency, e.g., seconds, minutes, hours, days, weeks, months, years, etc.
Column Datatype: CHAR(6)
Null Option: NULL

Attribute Name: User_Reference
Column Datatype: LONG VARCHAR
Null Option: NULL

Attribute Name: Volume
Attribute Definition: How many times process is done per UOM
Column Datatype: INTEGER
Null Option: NULL

Attribute Name: Volume_UOM
Attribute Definition: Unit of Measure for Volume, e.g., # per second, minute, hour, day, week, month, year, etc.
Column Datatype: CHAR(6)
Null Option: NULL

Entity Name: Link
Entity Definition: Links are used primarily to denote the significant constraining relationships among processes. For example, precedence links show temporal precedence, i.e., that one process must be done before the next.

Attribute Name: Source.ID
Attribute Definition: ID # for Source Process for Link, Inherited from Source
Column Datatype: INTEGER
Null Option: NOT NULL
PK: (PK)
FK: (FK)

Attribute Name: Destination.ID
Attribute Definition: ID # for Destination Process for Link, Inherited from Destination
Column Datatype: INTEGER
Null Option: NOT NULL
PK: (PK)
FK: (FK)

Attribute Name: Link_Name
Attribute Definition: Used to label link in graphical view, optional
Column Datatype: CHAR(40)
Null Option: NULL

Attribute Name: Link_Type
Attribute Definition: IDEF3 Link Types are: Precedence, Relational, Object Flow
Column Datatype: CHAR(5)
Null Option: NULL

Attribute Name: Link_Description
Column Datatype: LONG VARCHAR
Null Option: NULL

Entity Name: Object
Entity Definition: Objects that flow between, used in, or produced by processes.

Attribute Name: Object_Name
Column Datatype: CHAR(40)
Null Option: NOT NULL
PK: (PK)

Attribute Name: Object_Desc
Column Datatype: LONG VARCHAR
Null Option: NULL

Entity Name: Object_State
Entity Definition: Defines the state of an object as it flows on a link between one process and the next.

Attribute Name: Source.ID
Attribute Definition: ID # for Source Process for Link, Inherited from Source
Column Datatype: INTEGER
Null Option: NOT NULL
PK: (PK)
FK: (FK)

Attribute Name: Destination.ID
Attribute Definition: ID # for Destination Process for Link, Inherited from Destination
Column Datatype: INTEGER
Null Option: NOT NULL
PK: (PK)
FK: (FK)

Attribute Name: Object_Name
Attribute Definition: Name of object flowing on link
Column Datatype: CHAR(40)
Null Option: NOT NULL
PK: (PK)
FK: (FK)

Attribute Name: State
Attribute Definition: State of the object when it flows on the link
Column Datatype: CHAR(18)
Null Option: NULL

Entity Name: Process_Actor
Entity Definition: Documents who or what does or participates in the process. Can be an individual, group of individuals, or a system. Individuals usually identified by role name vs. person name.

Attribute Name: Process.ID
Attribute Definition: ID # for Process, Internal/Assigned by Software
Column Datatype: INTEGER
Null Option: NOT NULL
PK: (PK)
FK: (FK)

Attribute Name: Role_Name
Column Datatype: CHAR(40)
Null Option: NOT NULL
PK: (PK)
FK: (FK)

Entity Name: Process_Sim_Info
Entity Definition: Stores the data needed to run a simulation of the process. Several attributes still must be determined. This node is not shown in the data model (Appendix E.2) because of the need for more detailed analysis.

Attribute Name: ID
Attribute Definition: ID # for Process, Internal/Assigned by Software
Column Datatype: INTEGER
Null Option: NOT NULL
PK: (PK)
FK: (FK)

Attribute Name: Exec_Time
Attribute Definition: Time from process start to process end
Column Datatype: DECIMAL()
Null Option: NULL

Attribute Name: Exec_Time_UOM
Attribute Definition: Unit of Measure for Execution Time, e.g., seconds, minutes, hours, days, weeks, months, years, etc.
Column Datatype: CHAR(6)
Null Option: NULL

Attribute Name: Queue_Time
Attribute Definition: Time from when all objects necessary to do process arrive to when the process actually starts. Average Queue Time
Column Datatype: DECIMAL()
Null Option: NULL

Attribute Name: Queue_Time_UOM
Attribute Definition: Unit of Measure for Queue Time, e.g., seconds, minutes, hours, days, weeks, months, years, etc.
Entity Name: Pseudo-Process
Entity Definition: Junctions can be considered a type of pseudo-process. They provide a mechanism for specifying the logic of process branching.

Attribute Name: ID
Attribute Definition: ID # for Junction, Internal/Assigned by Software
Column Datatype: INTEGER
Null Option: NOT NULL
PK: (PK)
FK: (FK)

Attribute Name: Junction_Label
Attribute Definition: Optional label for Junction in graphical view
Column Datatype: CHAR(40)
Null Option: NULL

Attribute Name: Junction_Type
Attribute Definition: UML Activity Diagram has: Fork/Branch or Join. Decision is special type of Branch. IDEF3 has: Fan-out and Fan-in
Column Datatype: CHAR(18)
Null Option: NULL

Attribute Name: Junction_Logic
Attribute Definition: IDEF3 has: AND, OR, XOR
Column Datatype: CHAR(3)
Null Option: NULL
APPENDIX F - SPA EVALUATION RESULTS

F.1 Air Force Meeting 1 Evaluation Results

Tool Functionality

1. The system allowed me to do everything I needed to develop the process model.
   SA-Strongly Agree  A-Agree  N-Neutral  D-Disagree  SD-Strongly Disagree

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>1</td>
</tr>
<tr>
<td>A(4)</td>
<td>5</td>
</tr>
<tr>
<td>N(3)</td>
<td>0</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

   Statistics
   Mean          A(4.17)
   STD           0.41

2. Developing the process model using this tool was:
   Very Easy (VE), Easy (E), Average (A), Hard (H), Very Hard (VH)

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>VE(5)</td>
<td>2</td>
</tr>
<tr>
<td>E(4)</td>
<td>4</td>
</tr>
<tr>
<td>A(3)</td>
<td>0</td>
</tr>
<tr>
<td>H(2)</td>
<td>0</td>
</tr>
<tr>
<td>VH(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

   Statistics
   Mean          E(4.33)
   STD           0.52

3. What additional features would make it easier to develop a process model using this tool? (Open-Ended)
   1. using a format similar to MICROFAX Form Flow
   2. The availability to Cut and Paste would be useful.
   If the screen would not reset every time someone made an input it would be easier to work.
3. Higher levels of authority...i.e. Air Staff representation to make decisions for further development/improvements of/to the AFTO 107 process.

4. Longer advance briefing

**Tool Usability**

4. The screen layout/design was clear and easy to use.

*SA-Strongly Agree  A-Agree  N-Neutral  D-Disagree  SD-Strongly Disagree*

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>3</td>
</tr>
<tr>
<td>A(4)</td>
<td>3</td>
</tr>
<tr>
<td>N(3)</td>
<td>0</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

*Statistics*

- **Mean:** SA(4.50)
- **STD:** 0.55

5. What changes to the screen layout would make the tool easier to use? (Open-Ended)

1. Eliminate the blanking out screen and having to scroll down after each submission
2. Cut and paste.
3. Group selections/assembly
4. The process tree could be centered better

6. Understanding the instructions provided by the tool was:

*Very Easy (VE), Easy (E), Average (A), Hard (H), Very Hard (VH)*

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>VE(5)</td>
<td>2</td>
</tr>
<tr>
<td>E(4)</td>
<td>3</td>
</tr>
<tr>
<td>A(3)</td>
<td>1</td>
</tr>
<tr>
<td>H(2)</td>
<td>0</td>
</tr>
<tr>
<td>VH(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

*Statistics*

- **Mean:** E(4.17)
- **STD:** 0.75
7. If you encountered any errors, understanding the error messages was:
Very Easy (VE), Easy (E), Average (A), Hard (H), Very Hard (VH)

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>VE(5)</td>
<td>1</td>
</tr>
<tr>
<td>E(4)</td>
<td>4</td>
</tr>
<tr>
<td>A(3)</td>
<td>1</td>
</tr>
<tr>
<td>H(2)</td>
<td>0</td>
</tr>
<tr>
<td>VH(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean E(4.00)
STD 0.63

8. If you encountered any errors, recovering from the errors was:
Very Easy (VE), Easy (E), Average (A), Hard (H), Very Hard (VH)

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>VE(5)</td>
<td>0</td>
</tr>
<tr>
<td>E(4)</td>
<td>3</td>
</tr>
<tr>
<td>A(3)</td>
<td>3</td>
</tr>
<tr>
<td>H(2)</td>
<td>0</td>
</tr>
<tr>
<td>VH(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean E(3.50)
STD 0.55

Tool Ease of Use

9. Learning to operate the Process Modeler tool was easy for me.
SA-Strongly Agree A-Agree N-Neutral D-Disagree SD-Strongly Disagree

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>2</td>
</tr>
<tr>
<td>A(4)</td>
<td>4</td>
</tr>
<tr>
<td>N(3)</td>
<td>0</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean A(4.33)
STD 0.52
10. I found it easy to get the Process Modeler tool to do what I wanted it to do.  
SA-Strongly Agree  A-Agree  N-Neutral  D-Disagree  SD-Strongly Disagree  

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>0</td>
</tr>
<tr>
<td>A(4)</td>
<td>6</td>
</tr>
<tr>
<td>N(3)</td>
<td>0</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics  
Mean A(4.00)  
STD 0.00

11. My interactions with the Process Modeler tool were clear and understandable.  
SA-Strongly Agree  A-Agree  N-Neutral  D-Disagree  SD-Strongly Disagree  

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>0</td>
</tr>
<tr>
<td>A(4)</td>
<td>6</td>
</tr>
<tr>
<td>N(3)</td>
<td>0</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics  
Mean A(4.00)  
STD 0.00

12. I found the Process Modeler tool flexible to interact with.  
SA-Strongly Agree  A-Agree  N-Neutral  D-Disagree  SD-Strongly Disagree  

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>1</td>
</tr>
<tr>
<td>A(4)</td>
<td>4</td>
</tr>
<tr>
<td>N(3)</td>
<td>1</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics  
Mean A(4.00)  
STD 0.63
13. It was easy for me to become skillful at using the Process Modeler tool.

SA-Strongly Agree  A-Agree  N-Neutral  D-Disagree  SD-Strongly Disagree

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>1</td>
</tr>
<tr>
<td>A(4)</td>
<td>5</td>
</tr>
<tr>
<td>N(3)</td>
<td>0</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean: A(4.17)
STD: 0.41

14. Interacting with the Process Modeler tool did not require a lot of mental effort.

SA-Strongly Agree  A-Agree  N-Neutral  D-Disagree  SD-Strongly Disagree

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>1</td>
</tr>
<tr>
<td>A(4)</td>
<td>4</td>
</tr>
<tr>
<td>N(3)</td>
<td>1</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean: A(4.00)
STD: 0.63

15. I found the Process Modeler tool easy to use.

SA-Strongly Agree  A-Agree  N-Neutral  D-Disagree  SD-Strongly Disagree

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>1</td>
</tr>
<tr>
<td>A(4)</td>
<td>5</td>
</tr>
<tr>
<td>N(3)</td>
<td>0</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean: A(4.17)
STD: 0.41

16. What changes would you recommend to improve the ease of use of the Process Modeler tool? (Open-Ended)
1. Improve the disappearance and reappearance of the Process Modeler every time a change is made so that recovery brings you back to where you had highlighted and not at the top of the modeler every time.
2. none at this point
3. In the process model we used, the blinking screen whenever a change was made was disturbing and disruptive. Also whenever a change came through, the drop down chart returned to the top line even though I was working on a separate line entry.

Overall Assessment of the Tool

17. What did you like least about the Process Modeler tool? (Open-Ended)

1. See number 16
2. Stated in earlier comment
3. The screen resetting after other user inputs
4. Lack of edit capability (copy/cut paste).
   Constant jumping of fields during group activity.
5. See previous comments

18. What did you like best about the Process Modeler tool? (Open-Ended)

1. It's ease of use
2. Anonymity
3. Provides clear display of entire process for objective analysis.
4. Fun and easy to work with. Just needs a few bugs worked out.

19. If you could change only one thing in the Process Modeler tool, what would you tell the designers to change? (Open-Ended)

1. Give the capability to make multiple selections
2. See 17
3. Reconsider the limit of 40 characters.
4. Stop interference with users screen during operation and inputs by other users.

AFTO 107 Process Model Quality

20. The overall quality of the AFTO 107 process model developed was:
   Excellent (E), Very Good (VG), Good (G), Fair (F), Poor (P)

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>E(5)</td>
<td>2</td>
</tr>
<tr>
<td>VG(4)</td>
<td>2</td>
</tr>
</tbody>
</table>
21. Personnel involved in the 107 Process could easily understand the meaning of the process model developed with a minimum amount of explanation.

SA-Strongly Agree  A-Agree  N-Neutral  D-Disagree  SD-Strongly Disagree

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>1</td>
</tr>
<tr>
<td>A(4)</td>
<td>5</td>
</tr>
<tr>
<td>N(3)</td>
<td>0</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean        A(4.17)
STD         0.41

22. The process model developed was:
Completely Correct (CC), Mostly Correct (MC), Average (A), Somewhat Incorrect (SI), Mostly Incorrect (MI)

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>CC(5)</td>
<td>0</td>
</tr>
<tr>
<td>MC(4)</td>
<td>6</td>
</tr>
<tr>
<td>A(3)</td>
<td>0</td>
</tr>
<tr>
<td>SI(2)</td>
<td>0</td>
</tr>
<tr>
<td>MI(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean        MC(4.00)
STD         0.00

23. The process model developed was:
Very Complete (VC), Mostly Complete (MC), Average (A), Somewhat Incomplete (SI), Very Incomplete (VI)

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>VC(5)</td>
<td>2</td>
</tr>
</tbody>
</table>
24. The process descriptions developed were:
Very Clear (VC), Mostly Clear (MC), Average (A), Somewhat Vague (SV), Very Vague (VV)

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>VC(5)</td>
<td>2</td>
</tr>
<tr>
<td>MC(4)</td>
<td>2</td>
</tr>
<tr>
<td>A(3)</td>
<td>2</td>
</tr>
<tr>
<td>SV(2)</td>
<td>0</td>
</tr>
<tr>
<td>VV(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean       MC(4.00)
STD        0.89

25. How satisfied were you with the final AFTO 107 process model?
Very Satisfied (VS), Mostly Satisfied (MS), Average (A), Somewhat Unsatisfied (SU), Very Unsatisfied (VU)

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>VS(5)</td>
<td>1</td>
</tr>
<tr>
<td>MS(4)</td>
<td>5</td>
</tr>
<tr>
<td>A(3)</td>
<td>0</td>
</tr>
<tr>
<td>SU(2)</td>
<td>0</td>
</tr>
<tr>
<td>VU(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean       MS(4.17)
STD        0.41

Meeting Approach

26. Using this approach to define the AFTO 107 process enabled me to accomplish this task quickly.
SA-Strongly Agree  A-Agree  N-Neutral  D-Disagree  SD-Strongly Disagree
27. I spent my time efficiently describing the AFTO 107 process.
SA-Strongly Agree  A-Agree  N-Neutral  D-Disagree  SD-Strongly Disagree

28. The meeting approach allowed me to everything I needed to do to define the AFTO 107 process.
SA-Strongly Agree  A-Agree  N-Neutral  D-Disagree  SD-Strongly Disagree

29. How could the approach followed in the meeting to create/validate the AFTO 107 process be improved? (Open-Ended)
1. Disappointed in that the suggestion to review other bases -107 process only provided a list of apparent bias personal comments rather than the expected intended model process flow chart as first formulated and reviewed at/by Mountain Home AFB, ID.
2. Need to get all the players involved in this process.
   All the MAJCOMS need to represented as well as Wing Schedulers.
3. Author of TO 00-15-107 should have been in attendance to explain/defend the current TO.
4. Again, have the proper level of authority/decision makers present....i.e. Air Staff policy makers.

30. How satisfied were you with the overall meeting approach?
Very Satisfied (VS), Mostly Satisfied (MS), Average (A), Somewhat Unsatisfied (SU),
Very Unsatisfied (VU)

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>VS(5)</td>
<td>4</td>
</tr>
<tr>
<td>MS(4)</td>
<td>2</td>
</tr>
<tr>
<td>A(3)</td>
<td>0</td>
</tr>
<tr>
<td>SU(2)</td>
<td>0</td>
</tr>
<tr>
<td>VU(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean VS(4.67)
STD 0.52

Demographics

31. How many years have you been working for the military (combined active duty and civilian time)? (Assign a number)

<table>
<thead>
<tr>
<th>Range</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>20 - 32</td>
<td>6</td>
</tr>
</tbody>
</table>

Statistics
Mean 23.83
STD 4.17

32. How many years have you been working with the AFTO 107 Process? (Assign a number)

<table>
<thead>
<tr>
<th>Range</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - 23</td>
<td>6</td>
</tr>
</tbody>
</table>
### Statistics

<table>
<thead>
<tr>
<th>Mean</th>
<th>12.67</th>
</tr>
</thead>
<tbody>
<tr>
<td>STD</td>
<td>9.09</td>
</tr>
</tbody>
</table>

33. **How often do you use a computer?**

*Select the appropriate response*

- [6] Several times each day
- [0] Once a day
- [0] Several times each week
- [0] Once a week
- [0] Rarely

34. **How would you rate your expertise with use of Internet Browsers?**

*Excellent (E), Very Good (VG), Good (G), Fair (F), Poor (P)*

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>E(5)</td>
<td>0</td>
</tr>
<tr>
<td>VG(4)</td>
<td>2</td>
</tr>
<tr>
<td>G(3)</td>
<td>3</td>
</tr>
<tr>
<td>F(2)</td>
<td>1</td>
</tr>
<tr>
<td>P(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
</tr>
<tr>
<td>STD</td>
</tr>
</tbody>
</table>

35. **Before this meeting, how many times had you used GroupSystems?**

*Select the appropriate response*

- [0] Many times
- [2] 2-3 times
- [1] 1 time
- [3] Never

36. **Before this meeting, how many times had you developed a process, activity, or similar type model?**

- [1] Many times
- [2] 2-3 times
- [2] 1 time
- [1] Never
F.2 Air Force Meeting 2 Evaluation Results

Process Modeler Tool Ease of Use

1. Learning to operate the Process Modeler tool was easy for me.
   SA-Strongly Agree  A-Agree  N-Neutral  D-Disagree  SD-Strongly Disagree

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>3</td>
</tr>
<tr>
<td>A(4)</td>
<td>8</td>
</tr>
<tr>
<td>N(3)</td>
<td>1</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean  A(4.17)
STD   0.58

2. I found it easy to get the Process Modeler tool to do what I wanted it to do.
   SA-Strongly Agree  A-Agree  N-Neutral  D-Disagree  SD-Strongly Disagree

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>5</td>
</tr>
<tr>
<td>A(4)</td>
<td>6</td>
</tr>
<tr>
<td>N(3)</td>
<td>1</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean  A(4.33)
STD   0.65

3. My interactions with the Process Modeler tool were clear and understandable.
   SA-Strongly Agree  A-Agree  N-Neutral  D-Disagree  SD-Strongly Disagree

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>6</td>
</tr>
<tr>
<td>A(4)</td>
<td>5</td>
</tr>
<tr>
<td>N(3)</td>
<td>1</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
270

4. I found the Process Modeler tool flexible to interact with.

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>5</td>
</tr>
<tr>
<td>A(4)</td>
<td>5</td>
</tr>
<tr>
<td>N(3)</td>
<td>2</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics

| Mean   | A(4.25) |
| STD    | 0.75    |

5. It was easy for me to become skillful at using the Process Modeler tool.

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>4</td>
</tr>
<tr>
<td>A(4)</td>
<td>7</td>
</tr>
<tr>
<td>N(3)</td>
<td>1</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics

| Mean   | A(4.25) |
| STD    | 0.62    |

6. Interacting with the Process Modeler tool did not require a lot of mental effort.

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>4</td>
</tr>
<tr>
<td>A(4)</td>
<td>8</td>
</tr>
<tr>
<td>N(3)</td>
<td>0</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics

| Mean   | A(4.33) |
| STD    | 0.49    |
7. I found the Process Modeler tool easy to use.

SA-Strongly Agree  A-Agree  N-Neutral  D-Disagree  SD-Strongly Disagree

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>3</td>
</tr>
<tr>
<td>A(4)</td>
<td>8</td>
</tr>
<tr>
<td>N(3)</td>
<td>1</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean: A(4.17)
STD: 0.58

8. If you participated in the earlier meeting, did the changes in the Process Modeler tool improve its ease of use?

SA-Strongly Agree  A-Agree  N-Neutral  D-Disagree  SD-Strongly Disagree

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>3</td>
</tr>
<tr>
<td>A(4)</td>
<td>2</td>
</tr>
<tr>
<td>N(3)</td>
<td>6</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean: A(3.73)
STD: 0.90

9. What changes would you recommend to improve the ease of use of the Process Modeler tool? (Open-Ended)

1. none
2. My computer did not update all of the information, I had to use another members monitor to participate in the discussion.
3. NO real changes, I found it easy to use.
4. Add mini pop-up help notes that appear when you place the cursor over a tool bar icon
   - Add same for showing the entire description of a node when the cursor is placed over it, vice having to scroll across
   - Add the ability to drag nodes in the process tree
   - Add a graphical depiction of the process/decision tree and the ability to click on it to jump to that node of the process
Overall Assessment of the Process Modeler Tool

10. What did you like least about the Process Modeler tool? (Open-Ended)

1. The loss of capability when network went down


1. It’s ease of use
2. It is easy to use and great to be able to see others comments at the same time,
3. ease of use

12. If you could change only one thing in the Process Modeler tool, what would you tell the designers to change? (Open-Ended)

1. Nothing
2. Nothing

AFTO 107 Process Model Quality

13. The overall quality of the AFTO 107 process model developed was:
Excellent (E), Very Good (VG), Good (G), Fair (F), Poor (P)

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>E(5)</td>
<td>3</td>
</tr>
<tr>
<td>VG(4)</td>
<td>8</td>
</tr>
<tr>
<td>G(3)</td>
<td>1</td>
</tr>
<tr>
<td>F(2)</td>
<td>0</td>
</tr>
<tr>
<td>P(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean   VG(4.17)
STD    0.58

14. The overall quality of the recommended improvements to the AFTO 107 process identified were:
Excellent (E), Very Good (VG), Good (G), Fair (F), Poor (P)

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>E(5)</td>
<td>4</td>
</tr>
<tr>
<td>VG(4)</td>
<td>5</td>
</tr>
<tr>
<td>G(3)</td>
<td>3</td>
</tr>
<tr>
<td>F(2)</td>
<td>0</td>
</tr>
</tbody>
</table>
15. Personnel involved in the 107 Process could easily understand the meaning of the process model developed with a minimum amount of explanation.

SA-Strongly Agree  A-Agree  N-Neutral  D-Disagree  SD-Strongly Disagree

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>6</td>
</tr>
<tr>
<td>A(4)</td>
<td>6</td>
</tr>
<tr>
<td>N(3)</td>
<td>0</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Total   54
Mean    SA(4.50)
STD     0.52

16. The process model developed was:

Completely Correct (CC), Mostly Correct (MC), Average (A), Somewhat Incorrect (SI), Mostly Incorrect (MI)

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>CC(5)</td>
<td>2</td>
</tr>
<tr>
<td>MC(4)</td>
<td>10</td>
</tr>
<tr>
<td>A(3)</td>
<td>0</td>
</tr>
<tr>
<td>SI(2)</td>
<td>0</td>
</tr>
<tr>
<td>MI(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean    MC(4.17)
STD     0.39

17. The process model developed was:

Very Complete (VC), Mostly Complete (MC), Average (A), Somewhat Incomplete (SI), Very Incomplete (VI)

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>VC(5)</td>
<td>3</td>
</tr>
<tr>
<td>MC(4)</td>
<td>8</td>
</tr>
</tbody>
</table>
18. The process descriptions developed were:
Very Clear (VC), Mostly Clear (MC), Average (A), Somewhat Vague (SV), Very Vague (VV)

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>VC(5)</td>
<td>3</td>
</tr>
<tr>
<td>MC(4)</td>
<td>9</td>
</tr>
<tr>
<td>A(3)</td>
<td>0</td>
</tr>
<tr>
<td>SV(2)</td>
<td>0</td>
</tr>
<tr>
<td>VV(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean: MC(4.25)
STD: 0.45

19. How satisfied were you with the final AFTO 107 process model?
Very Satisfied (VS), Mostly Satisfied (MS), Average (A), Somewhat Unsatisfied (SU), Very Unsatisfied (VU)

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>VS(5)</td>
<td>4</td>
</tr>
<tr>
<td>MS(4)</td>
<td>8</td>
</tr>
<tr>
<td>A(3)</td>
<td>0</td>
</tr>
<tr>
<td>SU(2)</td>
<td>0</td>
</tr>
<tr>
<td>VU(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean: MS(4.33)
STD: 0.49

20. How satisfied were you with the recommended AFTO 107 process improvements?
Very Satisfied (VS), Mostly Satisfied (MS), Average (A), Somewhat Unsatisfied (SU), Very Unsatisfied (VU)

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
</table>

Distributed Meeting Assessment

21. Having representatives participate from both locations improved the quality of the AFTO 107 process model.
SA-Strongly Agree A-Agree N-Neutral D-Disagree SD-Strongly Disagree

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>12</td>
</tr>
<tr>
<td>A(4)</td>
<td>0</td>
</tr>
<tr>
<td>N(3)</td>
<td>0</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean  SA(5.00)
STD    0.00

22. Having representatives participate from both locations improved the quality of the recommended AFTO 107 process improvements.
SA-Strongly Agree A-Agree N-Neutral D-Disagree SD-Strongly Disagree

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>11</td>
</tr>
<tr>
<td>A(4)</td>
<td>1</td>
</tr>
<tr>
<td>N(3)</td>
<td>0</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean  SA(4.92)
STD    0.29

23. Representatives from both meeting locations participated equally.
SA-Strongly Agree A-Agree N-Neutral D-Disagree SD-Strongly Disagree
24. Working with other meeting participants at my location was:
Very Easy (VE), Easy (E), Average (A), Difficult (D), Very Difficult (VD)

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>VE(5)</td>
<td>7</td>
</tr>
<tr>
<td>E(4)</td>
<td>4</td>
</tr>
<tr>
<td>A(3)</td>
<td>1</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>VD(1)</td>
<td>1</td>
</tr>
</tbody>
</table>

Statistics
Mean VE(4.50)
STD 0.67

25. Working with meeting participants at the other location was:
Very Easy (VE), Easy (E), Average (A), Difficult (D), Very Difficult (VD)

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>VE(5)</td>
<td>4</td>
</tr>
<tr>
<td>E(4)</td>
<td>2</td>
</tr>
<tr>
<td>A(3)</td>
<td>4</td>
</tr>
<tr>
<td>D(2)</td>
<td>1</td>
</tr>
<tr>
<td>VD(1)</td>
<td>1</td>
</tr>
</tbody>
</table>

Statistics
Mean E(3.58)
STD 1.31

26. My meeting location was:

[ 9 ] Mountain Home AFB
Overall Meeting Approach

27. Using this approach to define/improve the AFTO 107 process enabled me to accomplish this task quickly.

SA-Strongly Agree A-Agree N-Neutral D-Disagree SD-Strongly Disagree

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>5</td>
</tr>
<tr>
<td>A(4)</td>
<td>5</td>
</tr>
<tr>
<td>N(3)</td>
<td>2</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean: A(4.25)
STD: 0.75

28. I spent my time efficiently describing the AFTO 107 process.

SA-Strongly Agree A-Agree N-Neutral D-Disagree SD-Strongly Disagree

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>5</td>
</tr>
<tr>
<td>A(4)</td>
<td>5</td>
</tr>
<tr>
<td>N(3)</td>
<td>2</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
Mean: A(4.25)
STD: 0.75

29. The meeting approach allowed me to everything I needed to do to define/improve the AFTO 107 process.

SA-Strongly Agree A-Agree N-Neutral D-Disagree SD-Strongly Disagree

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA(5)</td>
<td>5</td>
</tr>
<tr>
<td>A(4)</td>
<td>6</td>
</tr>
<tr>
<td>N(3)</td>
<td>1</td>
</tr>
<tr>
<td>D(2)</td>
<td>0</td>
</tr>
<tr>
<td>SD(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
30. How could the approach followed in the meeting to define/improve the AFTO 107 process be improved? (Open-Ended)

1. Better LAN connectivity
2. Improve connectivity between bases, i.e. loss of video and lack of audio quality
3. The only problem was losing the internet but other than that it is a great approach
4. Get all the MAJCOMS involved.
5. The video conferencing link we had with Robins was not all that it should have been to facilitate a good conference. The Internet link at Robins did not allow us constant access to them. The Internet at Robins kept dumping them - forcing telephone speakerphone discussions that did not allow us good communications. The software was excellent, but the connection to Robins was weak.

31. How satisfied were you with the overall meeting approach?

**Very Satisfied (VS), Mostly Satisfied (MS), Average (A), Somewhat Unsatisfied (SU), Very Unsatisfied (VU)**

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>VS(5)</td>
<td>4</td>
</tr>
<tr>
<td>MS(4)</td>
<td>7</td>
</tr>
<tr>
<td>A(3)</td>
<td>1</td>
</tr>
<tr>
<td>SU(2)</td>
<td>0</td>
</tr>
<tr>
<td>VU(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

**Statistics**

<table>
<thead>
<tr>
<th>Mean</th>
<th>MS(4.25)</th>
</tr>
</thead>
<tbody>
<tr>
<td>STD</td>
<td>0.62</td>
</tr>
</tbody>
</table>

32. How satisfied were you with the overall meeting outcome?

**Very Satisfied (VS), Mostly Satisfied (MS), Average (A), Somewhat Unsatisfied (SU), Very Unsatisfied (VU)**

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>VS(5)</td>
<td>2</td>
</tr>
<tr>
<td>MS(4)</td>
<td>9</td>
</tr>
<tr>
<td>A(3)</td>
<td>1</td>
</tr>
<tr>
<td>SU(2)</td>
<td>0</td>
</tr>
<tr>
<td>VU(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

**Statistics**
Mean $MS(4.08)$
STD 0.51

Demographics

33. How many years have you been working for the military (combined active duty and civilian time)? (Assign a number.)

<table>
<thead>
<tr>
<th>Range</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>12 - 32</td>
<td>12</td>
</tr>
</tbody>
</table>

Statistics

| Mean   | 19.92 |
| STD    | 5.68  |

34. How many years have you been working with the AFTO 107 Process? (Assign a number.)

<table>
<thead>
<tr>
<th>Range</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 - 23</td>
<td>12</td>
</tr>
</tbody>
</table>

Statistics

| Mean   | 8.50  |
| STD    | 6.60  |

35. How often do you use a computer?

[ 11 ] Several times each day
[  0 ] Once a day
[  1 ] Several times each week
[  0 ] Once a week
[  0 ] Rarely

36. How would you rate your expertise with use of Internet Browsers?
Excellent (E), Very Good (VG), Good (G), Fair (F), Poor (P)

<table>
<thead>
<tr>
<th>Choices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>E(5)</td>
<td>3</td>
</tr>
<tr>
<td>VG(4)</td>
<td>5</td>
</tr>
<tr>
<td>G(3)</td>
<td>3</td>
</tr>
<tr>
<td>F(2)</td>
<td>1</td>
</tr>
<tr>
<td>P(1)</td>
<td>0</td>
</tr>
</tbody>
</table>

Statistics
<table>
<thead>
<tr>
<th></th>
<th>Mean</th>
<th>VG(3.83)</th>
<th>STD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.94</td>
</tr>
</tbody>
</table>

37. Before this meeting, how many times had you used GroupSystems?

[ 1 ] Many times  
[ 3 ] 2 -3 times  
[ 2 ] 1 time  
[ 6 ] Never

38. Before this meeting, how many times had you used the Process Modeler tool?

[ 0 ] Many times  
[ 0 ] 2 -3 times  
[ 4 ] 1 time  
[ 8 ] Never

39. Before this meeting, how many times had you developed a process, activity, or similar type model?

[ 0 ] Many times  
[ 2 ] 2 -3 times  
[ 4 ] 1 time  
[ 6 ] Never
APPENDIX G – METHODOLOGY FOR COLLABORATIVE SCENARIO ELICITATION USING GROUPSYSTEMS

The methodology for collaborative scenario elicitation using GroupSystems was significantly refined and revised based on the Phase I research results. The key components of the final methodology are: (1) scenario templates to improve the completeness of the scenario descriptions, (2) iterative process blending parallel user input with group review and consensus building to improve the efficiency of the process, and (3) collaborative tool support using GroupSystems Group Outliner. The recommended methodology incorporates these components into a nine step scenario/use case definition process.

2. Facilitator adds scenario template to leaf nodes.
3. Users add scenario information as comments.
4. Group reviews comments. Facilitator copies result to template item in outline.
5. Users identify scenario actions.
6. Group reviews and agrees on scenario actions.
7. Users identify exceptions and related issues.
8. Analysts review for completeness, questions.

Each of these steps is discussed using sample Group Outliner scenario screen captures to explain what is occurring during that step.
**Step 1 — Group Decomposes Business Activities**

The first step in the collaborative scenario elicitation process is to identify business activities and decompose those activities into an activity hierarchy identifying sub-activities and processes. This step is usually completed activity in the CSEM Requirements Phase (see Step 1 description in Section 4.2). Although a complete activity model could be developed using Activity Modeler, usually only an activity hierarchy as shown in Figure 19 is created. Definitions are developed for each activity to ensure group members have a common understanding of each business activity. An activity model could be developed if the group is having a great deal of difficulty developing the activity hierarchy or significant process improvement is required at the macro/organizational level. The user group also uses the activity hierarchy to focus in on those activities requiring information system support and/or activities included in the current system increment scope.

**Step 2 — Facilitator Adds Scenario Template**

Because GroupSystems Group Outliner is a general-purpose outlining tool, it has no provisions for capturing different categories of information about an outline item. Therefore, to use a scenario template, the facilitator must copy the template items as children of each of the leaf nodes in the activity hierarchy (see Figure 20). Leaf nodes are used because scenarios usually occur at the lowest nodes of the activity tree. However, occasionally users may want to capture scenario information for other activity tree nodes. This is avoided if at all possible because the activity tree could get extremely messy if template information was added throughout the tree versus just at the leaf
Figure 19 - Group decomposes business activities (Step 1)

Figure 20 - Facilitator adds scenario template outline (Step 2)
node level. Even when limiting the template to the leaf nodes, users begin to lose a sense of the activity hierarchy because so much non-activity information is included in the tree.

**Step 3 — Users Add Scenario Information as Comments**

The users then work in parallel to define the main scenario-level information (e.g., objective, trigger, preconditions) for each of the scenarios. The scenario information is entered as comments by double-clicking on the appropriate template item in the activity hierarchy (see Figure 21). Multiple comments may be added for a single template item as users may want to add to, comment on, or disagree with each other’s entries.

**Step 4 — Facilitator Copies Group Review Results to Template**

Next the group reviews the scenario-level information input by the users. When the group reaches consensus for each template item, the facilitator copies the agreed upon information and adds it to the outline following the template item (see Figure 22).

**Step 5 — Users Identify Scenario Actions**

Once the group reaches agreement on the main scenario information and clearly understands what the scenario is, users then begin to work in parallel to identify the specific actions or steps they take to accomplish that scenario. The action descriptions are added as comments to the Scenario template item (see Figure 23) and should include all information identified in the action-level template. The actual action-level templates are not added to the outline because of the limitations of a simple outline with a single comment window per outline item as implemented in Group Outliner.
### Preconditions
- Cost Center/Shop Zone exists as an authorized area on the installation
- Process is part of authorized mission
- Technical document (TA, TO, LO, etc.) briefly describes process and HAZMAT use

### 1.1.1 Generate HAZMAT Authorization Request

**Definition:** Generate, review, approve, and create an authorization for a shop/area to use a hazardous material in a process.

- **Objective:** Complete a request for authorization (manual or electronic form) to use a hazardous material.
- **Preconditions:** A paper form or an electronic form must exist and be available for the requester to complete. The requester must know what material is requested and must have a legitimate requirement to use the requested hazardous material.
- **Who does this?** Any person who requests HAZMAT授权 to perform a process with a particular hazardous material not yet authorized.
- **Trigger:** Mission requirement to perform a process with a particular hazardous material not yet authorized.
- **Exceptions:**
- **Related Issues:**

### 1.2 Validate/Edit HAZMAT Authorization Request

- **Definition:** Ensure that the request for hazardous material authorization is complete in terms of all required data elements to continue with the approval of the authorization.
- **Objective:** Ensure that HAZMAT Authorization Request is complete and accurate to include necessary attachments.
- **Preconditions:** Requester has submitted a request.
- **Trigger:** Requesting shop must be in the shop table.
- **Process:** The following must exist:
  - NSN (UPC, description of the product, part number, item name) tables must exist.
  - Vendor/CAGE/MSDS record(s) must exist (alphanumeric fields).

---

**Figure 21** — Users add scenario information as comments (Step 3)

**Figure 22** — Facilitator copies group review results to template (Step 4)
Figure 23 – Users identify scenario actions (Step 5)

Figure 24 – Group reviews and agrees on scenario actions (Step 6)
Step 6 – Group Reviews andAgrees on Scenario Actions

The group then follows a process similar to step 4, but this time focusing in on the scenario actions. Once the group agrees on the actions for a specific scenario, the facilitator copies the agreed upon action descriptions to the Scenario template item in the outline (see Figure 24). Problems occur when individual action descriptions are long or complex or there are a lot of individual actions. As can be seen in Figure 24, only a limited amount of information is displayed for each outline item. In this case, only part of action 1 is seen with no indication that there is an action 2. Users must edit or display the individual outline item to see its entire contents when they exceed the allotted space.

Step 7 – Users Identify Exceptions and Related Issues

Users then work in parallel to complete the rest of the scenario template, identifying exceptions that may occur during execution of the scenario and any other related issues they feel may be important in understanding the scenario. Users enter their input as comments behind the appropriate template item. The group reviews these comments, but oftentimes will just refer to the comments in the outline rather than recopying all comments to the outline item (see Figure 25).

Step 8 – Analysts Reviews and Completes Scenarios

After the scenario session is over, the analyst reviews the scenarios for completeness. The analyst will complete scenarios when they have the information, e.g., developing the desired system result based on the user’s objective. The analyst may also coordinate with the developer to see if they have any questions regarding the scenarios. New scenarios may be identified to address special cases alluded to by users in the
normal-case scenarios. The next step in the requirements process will require the analysts and developers to design alternative system use cases, so any unanswered questions or issues that the analyst or developer has should be surfaced now. Some of these questions may be able to be resolved by representatives of the user group, but others may need to be addressed by the entire group.

Figure 25 – Users identify exceptions and related issues (Step 7)

**Step 9 – Group Validates Analyst-Completed Scenarios**

The final step in the scenario elicitation process occurs when the user group validates the scenarios completed by the analyst and responds to any questions the analyst or developer may have surfaced during their review of the scenarios. Depending on the number of scenarios and questions, this validation may occur in distributed mode, as part of another session, or in a session of its own.
APPENDIX H – METHODOLOGY FOR COLLABORATIVE SCENARIO ELICITATION USING SPA

Scenario and Process Analyzer Report

Session Name: Scenario Methodology
Report Generated: 3/2/99, 2:42:52 PM

Process Tree Hierarchy

Scenario Methodology
- Identify Activities
- Identify Scenarios
  - Define scenarios
  - Review scenarios
- Identify Scenario Actions
  - Define normal course of actions
  - Review actions
  - Describe exceptions/alternatives
- Complete Scenario Templates
  - Review scenarios for completeness
  - Complete scenarios as necessary
- Validate Scenarios
  - Validate completed scenarios
  - Identify improvement opportunities
  - Brainstorm IS opportunities
  - Assess integration proposals
- Identify System Use Cases

Orphan
Process: Scenario Methodology

Description
1. Proposed scenario elicitation methodology, revised to take advantage of the flexibility of the Scenario and Process Analyzer tool.
2. Basic scenario methodology was designed for use with GroupSystems Group Outliner. The basic methodology is documented in the Appendix G.

Overview
1. Number of Children: 6

Process: Identify Activities

Description
1. Identify and describe business activities, select which ones for which will define scenarios (i.e., identify scope of current increment/scenario effort).
2. This activity "Identify Business Activities" precedes "Generate and Integrate Business Scenarios" in CSEM Requirements Phase. So, normally would import this info from Activity Modeler/Group Outliner output of that step. If not done, must do this as first step in Scenario Methodology.

Overview
1. Parent Process: Scenario Methodology
2. Number of Children: 0

Actors
1. User/SME Group – Group of subject matter experts (SMEs) representing the user community.

Process: Identify Scenarios

Description
1. For each of the leaf node activities, will normally identify at least one scenario which describes the normal course of action for performing that activity. If there are other critical scenarios (e.g., major exceptions or alternatives) or meeting participants have extremely different ways of accomplishing that activity, may
want to create additional scenarios. May also have scenarios at non-leaf node activities, but majority will probably be at leaf node.

Overview

1. **Parent Process:** Scenario Methodology
2. **Number of Children:** 2

Actors

1. **User/SME Group** – Group of subject matter experts (SMEs) representing the user community.

---

**Process: Define scenarios**

---

**Description**

1. For each scenario, add scenario name to process hierarchy under appropriate activity. Add Description and Objective as a minimum. If users are having problems identifying scenarios, may want to add Trigger as well. Also identify preconditions, the viewpoint from which the scenario will be described, and any issues related to the scenario. This information may be added before or after group review of the basic scenario information.

Overview

1. **Parent Process:** Identify Scenarios
2. **Number of Children:** 0

Actors

1. **Individual Users/SMEs** – Members of user group working individually in parallel.

---

**Process: Review scenarios**

---

**Description**

1. Review scenarios to ensure understanding and agreement on scenario scope, i.e., its basic description, starting point (trigger, prerequisites), and ending point (objective).

Overview

1. **Parent Process:** Identify Scenarios
2. **Number of Children:** 0
Actors

1. User/SME Group – Group of subject matter experts (SMEs) representing the user community.

Process: Identify Scenario Actions

Description

1. After each scenario is clearly defined, then identify the steps required to accomplish that scenario. Initial focus is on defining the normal course of actions.

Overview

1. Parent Process: Scenario Methodology
2. Number of Children: 3

Actors

1. User/SME Group – Group of subject matter experts (SMEs) representing the user community.

Process: Define normal course of actions

Description

1. Identify and describe each step that will normally be done to ensure successful completion of the scenario. For each step, identify who/what performs that step (i.e., the actor) and resources (e.g., data, other resources) required to accomplish the step.

Overview

1. Parent Process: Identify Scenario Actions
2. Number of Children: 0

Actors

1. Individual Users/SMEs – Members of user group working individually in parallel.

Process: Review actions
Description
1. Review/revise actions until reach agreement on basic flow of actions required to successfully accomplish each scenario.

Overview
1. Parent Process: Identify Scenario Actions
2. Number of Children: 0

Actors
1. User/SME Group — Group of subject matter experts (SMEs) representing the user community.

Process: Describe exceptions/alternatives

Description
1. Identify the major exceptions/errors/problems which could occur for each scenario/action and/or alternative ways of accomplishing the scenario/action. Also identify any critical issues related to the scenario/action which have not yet been identified.

Overview
1. Parent Process: Identify Scenario Actions
2. Number of Children: 0

Actors
1. Individual Users/SMEs — Members of user group working individually in parallel.

Process: Complete Scenario Templates

Description
1. Analyst works with selected User Reps to review scenarios for completeness and identify information required to complete the entire scenario template.

Overview
1. Parent Process: Scenario Methodology
2. Number of Children: 2
Actors
1. Analyst – Systems analyst (or developer working as an analyst) assigned to the project.

Process: Review scenarios for completeness

Description
1. Analyst reviews scenario definitions, possibly with the help of a User Rep, to identify problem areas/questions.

Overview
1. Parent Process: Complete Scenario Templates
2. Number of Children: 0

Actors
1. Analyst – Systems analyst (or developer working as an analyst) assigned to the project.

Process: Complete scenarios as necessary

Description
1. Analyst defines "Result" of completion of scenario in IS/technical terms based on user's stated objective. Also adds (with assistance from User Rep) other information needed to complete the scenario templates or required to respond to analyst/developer questions about the scenario.

Overview
1. Parent Process: Complete Scenario Templates
2. Number of Children: 0

Actors
1. Analyst – Systems analyst (or developer working as an analyst) assigned to the project.
2. User Rep – Selected members of the user group who assist the analyst by providing subject matter advise between meetings.

Process: Validate Scenarios
Description
1. User/SME Group reviews scenarios to ensure that they accurately reflect current/desired business processes by first reviewing the scenarios describing current processes and then identifying ways to improve those scenarios through process changes, IS support, and/or data/systems integration.

Overview
1. **Parent Process**: Scenario Methodology
2. **Number of Children**: 4

Actors
1. User/SME Group – Group of subject matter experts (SMEs) representing the user community.

---

**Process: Validate completed scenarios**

Description
1. Group reviews, revises, and validates analyst-completed scenarios.

Overview
1. **Parent Process**: Validate Scenarios
2. **Number of Children**: 0

Actors
1. User/SME Group – Group of subject matter experts (SMEs) representing the user community.

---

**Process: Identify improvement opportunities**

Description
1. As part of review process, may also identify opportunities for improving the basic process described in the scenario.

Overview
1. **Parent Process**: Validate Scenarios
2. **Number of Children**: 0
Actors

1. Individual Users/SMEs – Members of user group working individually in parallel.
2. User/SME Group – Group of subject matter experts (SMEs) representing the user community.

Process: Brainstorm IS opportunities

Description

1. This is also an ideal opportunity for users/SMEs to identify how they think that an IS could assist/improve the basic process. This is a first step in identify user requirements.

Overview

1. Parent Process: Validate Scenarios
2. Number of Children: 0

Actors

1. Individual Users/SMEs – Members of user group working individually in parallel.
2. User/SME Group – Group of subject matter experts (SMEs) representing the user community.

Process: Assess integration proposals

Description

1. Identify potential IS integration opportunities and review how proposed IS/data integration modifies the business scenarios. Determine if changes are acceptable from a business perspective.

Overview

1. Parent Process: Validate Scenarios
2. Number of Children: 0

Actors

1. User/SME Group – Group of subject matter experts (SMEs) representing the user community.
Process: Identify System Use Cases

Description

1. Analysts/developers identify alternative system use cases to satisfy the business/IS requirements identified by users. Initially, may just use the Scenario Modeler tool to expand on scenario definitions by adding details of the user-system interaction required to accomplish system-supported steps in the scenario. May also use other tools to create screen mock-ups, process specs/models (e.g., DFDs, OO interaction diagrams), and/or prototypes as required to clearly define the system use cases.

Overview

1. Parent Process: Scenario Methodology
2. Number of Children: 0

Actors

1. Analyst – Systems analyst (or developer working as an analyst) assigned to the project.
2. Developers – Representatives of the development team.

Process: Orphan

Overview

1. Number of Children: 0
REFERENCES


