UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

DEVELOPING AND EVALUATING A WEB-BASED EMERGENCY MANAGEMENT SPATIAL DECISION SUPPORT SYSTEM

A DISSERTATION

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

DOCTOR OF PHILOSOPHY

By

NACI DILEKLI
Norman, Oklahoma
2010

DEVELOPING AND EVALUATING A WEB-BASED EMERGENCY MANAGEMENT SPATIAL DECISION SUPPORT SYSTEM

A DISSERTATION APPROVED FOR THE DEPARTMENT OF GEOGRAPHY

BY

Dr. Darren Purcell, Chair

Dr. Tarek Rashed

Dr. Renee McPherson

Dr. Aondover Tarhule

Dr. Bruce Hoagland

Dr. Kevin Kloesel

TABLE OF CONTENTS

TABLE OF CONTENTS iv
LIST OF TABLES........ vii
LIST OF ILLUSTRATIONS........ viii
Chapter 1 Introduction and Background........ 1
1.1. Statement of the Problem 1
1.2. Research Challenges, Objectives and Questions........ 3
1.3. Relevance and Contribution 5
Chapter 2 Literature Review 8
2.1. Summary 8
2.2. Single Hazard to Multi-Hazard Paradigm Shift in Emergency Management...... 8
2.3. Decision Support and Information Systems........ 12
2.3.1. Introduction........ 12
2.3.2. Decision Support Systems (Information Systems for Decision Support)... 12
2.3.3. User Centered Design for Decision Making........ 15
2.3.4. Geographic Information Systems (GIS) and Spatial Decision Support Systems (SDSS)........ 26
2.4. Information Systems and decision Support for Disaster Organizations 29
2.4.1. Emergency Decision Making Characteristics and Properties........ 29
2.4.2. Examples of EMDSSs........ 29
2.4.3. Requirements for EDSS 32
2.5. Development of Information Systems........ 36
2.5.1. Software Engineering........ 36
2.5.2. Information Systems Development Methodologies........ 36
2.5.3. Agile Development 48
2.5.4. Socio-technic Methodologies........ 52
2.5.5. Software Engineering Techniques and Tools 56
2.5.6. Capability Maturity Model Integration (CMMI) 62
2.5.7. CMMI and Agile Development Integration........ 67

2.5.8. Use of Methodologies across Applications of Interest 81
2.5.9. Comparison of Methodologies 85
2.5.10. A Final Look at the Development Methodologies 93
2.6. Summary and Conclusions 98
Chapter 3 Methodology 100
3.1. Overview of Development and Research Methodology 100
3.2. XP and CMMI Integration for this Study 103
3.2.1. Overview of Products: Selected CMMI Process Areas 103
3.2.2. Extreme Programming Activities in the Integrated Development 105
3.2.3. Overview of Proposed Integrated Methodology 109
3.2.4. Achieving Generic Goals and Practices 112
3.2.5. Achieving Specific Goals and Practices 114
3.2.6. Practices completely outside CMMI 119
3.3. Preliminary Design Considerations 119
3.4. Brief Overview of the Methodology 120
3.4.1. Project Initiation 121
3.4.2. Implementation 123
3.4.3. Project Evaluation 123
3.5. Anticipated Timeline 123
3.6. Summary 124
Chapter 4 Implementation and Analysis 125
4.1. Introduction 125
4.2. Organization of User Input 125
4.2.1. Acquisition and Evaluation of User Stories 126
4.2.2. Selection of User Stories for Development Cycles 127
4.3. Implementation 130
4.3.1. First Development Cycle 132
4.3.2. Second Development Cycle 155
4.3.3. Third Development Cycle 167
4.4. Changes throughout the Project 181
4.5. Evaluation of the Entire Implementation 184

Chapter 5 Conclusions and Recommendations 192
5.1. Summary 192
5.2. Contributions 193
5.3. Limitations of the research and Future Work 197
6. Appendix 202
6.1. Operator Manual for the First Development Cycle 202
6.2. Operator Manual for the Second Development Cycle 204
6.3. Operator Manual for the Third Development Cycle 205
6.4. Interface Description and User Manual for the First Development Cycle of the Project 207
6.5. Interface Description and User Manual for the Second Development Cycle of the Project 214
6.6. Interface Description and User Manual for the Third Development Cycle of the Project 220
7. References 231

LIST OF TABLES

Table 2-1 Types of requirements. 57
Table 2-2 Process areas and their associated categories and maturity levels. 66
Table 2-3 Agile method vs. CMMI process area conflict findings................................. 73
Table 2-4 Agile method vs. CMMI generic practice conflict findings…………………. 74
Table 2-5. Comparison of development methodologies 90
Table 2-6. Average group effort by activity in hours 92
Table 2-7. Average group effort by activity as percentages. 92
Table 2-8. Product quality assessment on a scale 0 - 10.. 92
Table 4-1 User stories evaluation table before implementation 128
Table 4-2 Issues and risks that were raised before development cycles......................... 131
Table 4-3 User stories evaluation form for first development cycle 133
Table 4-4 Project planning for the tasks developed in first development cycle 134
Table 4-5 Solution alternatives in first development cycle with selected solutions highlighted 138
Table 4-6 First development cycle schedule in detail.. 153
Table 4-7 Risk and issue registers for the first development cycle 154
Table 4-8 User stories selected for second sevelopment sycle 156
Table 4-9 Project planning for the tasks developed in second development cycle 158
Table 4-10 Solution alternatives in third development cycle with selected solutions highlighted 159
Table 4-11 Second development cycle in detail 165
Table 4-12 Risk and issue registers for the second development cycle.......................... 166
Table 4-13 User stories selected for third development cycle 168
Table 4-14 Project planning for the tasks developed in third development cycle 169
Table 4-15 Solution alternatives in third development cycle with selected solutions highlighted 170
Table 4-16 Third development cycle schedule in detail 179
Table 4-17 Risk and issue registers for the third development cycle 180
Table 4-18 Requirements change form.. 183
Table 4-19 Project execution estimated and actual timelines summarized 188

LIST OF ILLUSTRATIONS

Figure 2-1 A Typical DSS. 14
Figure 2-2. From techno-centric to socio-technical computing. 16
Figure 2-3 The information gap. 16
Figure 2-4 Model of situation awareness in dynamic decision making. 20
Figure 2-5 Types of uncertainty in the decision process. 20
Figure 2-6 examples of reliability when human and machine components are operating in parallel vs serial modes. 22
Figure 2-7 The triangular structure of organizations. 28
Figure 2-8 Structure of a knowledge-based distributed EDSS 33
Figure 2-9 Waterfall development model. 39
Figure 2-10 Incremental model. 41
Figure 2-11 RAD model. 43
Figure 2-12. Evolutionary Prototyping. 45
Figure 2-13 Spiral Model of software process. 47
Figure 2-14. XP Process. 50
Figure 2-15 Stages of multiview methodology. 55
Figure 2-16 XP Satisfaction of key process areas, given the appropriate environment. .. 69
Figure 2-17 Degree of satisfaction for the 52 CMM goals by implementing an XP process. 71
Figure 2-18 The planning spectrum 77
Figure 2-19 Risk exposure profile. 78
Figure 2-20 Comparative risk exposure profile for an agile home-ground company with a small installed base and less need for high assurance. 79
Figure 2-21 Comparative risk exposure profile for a plan-driven home-ground company that produces large, safety-critical systems. 80
Figure 2-22. A generic GIS development methodology. 82
Figure 2-23. User interface design process. 84
Figure 2-24. Users don't always get what they wanted. 95
Figure 2-25. What the user wanted. 96
Figure 3-1 CMMI, XP and Situational Awareness 102
Figure 3-2 A visual process model for XP. 107
Figure 3-3 XP and CMMI oriented development methodology.. 111
Figure 4-1 User stories and tasks that were developed in the first development cycle .. 137
Figure 4-2 Interface design for first development cycle 140
Figure 4-3 Manage layers panel 143
Figure 4-4 Application interface with some layers turned off 143
Figure 4-5 Transparency for the background layer adjusted 144
Figure 4-6 Activity diagram for sketching and selection. 147
Figure 4-7 Sketching a facility selection 148
Figure 4-8 Some hazard action list items checked. 150

Figure 4-9 Viewing phonebook items.. 150
Figure 4-10 General view of the application after first cycle was finished................... 151
Figure 4-12 Use case diagram for user stories and tasks that were developed in the second development cycle 157
Figure 4-13 Interface design for second development cycle 157
Figure 4-14 Facility zoomed and its attributes shown 161
Figure 4-15 An arbitrary emergency plan near oklahoma memorial stadium attached to the application.. 163
Figure 4-16 General view of the application after second cycle was finished 163
Figure 4-17 User stories and tasks that were developed in the third development cycle 168
Figure 4-18 Interface design for third development cycle... 171
Figure 4-19 Layers menu with added wms layers highlighted 173
Figure 4-20 Layers menu with traffic cameras layer highlighted.................................. 173
Figure 4-21 Hazmat icon highlighted and clicked 175
Figure 4-22 User can close a hazmat guide by the “X” button...................................... 175
Figure 4-23 Activity diagram for hazmat management 176
Figure 4-24 General view of the application after third development cycle 178
Figure 4-25 Project execution estimated and actual timelines....................................... 188
Figure 4-26 Number of days it took to close issues in the second development cycle... 190
Figure 4-27 Number of days it took to close issues in the first development cycle 190
Figure 4-28 Number of days it took to close issues in the third development cycle 191

ABSTRACT

This dissertation proposes and evaluates a consolidated design methodology for web-based emergency management decision support systems (WEM-DSSs). The development of the proposed methodology draws upon a literature review which crosslinks substantive topics related to evolving theoretical paradigms in disaster research and the role of information systems within organizations, and competing approaches to the development of GIS and participatory decision support systems. As a conclusion of the literature review, it was suggested that a good software development methodology should be balanced between agility and discipline. Due to the nature of this research, a mixture of Extreme Programming and Capability Maturity Integration approaches with an emphasis on agility is proposed. Then the design of the proposed methodology is refined and tested through a case study that seeks to develop a WEM-DSS for the emergency managers working in Oklahoma. The methodology's effectiveness is mainly evaluated by investigator's ability to follow proposed methodological tasks, ability to involve sufficient user input and ability to follow proposed timeline.

The findings of this research enhance our understanding of delivering geographic information to users, and drawing user input from emergency management communities. From a systems development point of view, this study shows that XP and CMMI are in fact compatible with each other. From an empirical viewpoint, the study shows a complete process of following a methodology that is implemented for developing a WEMDSS. Finally, this research delivers a technical product that is built upon user input. This product employs ArcGIS Silverlight API, Microsoft Silverlight and service oriented architectures.

Chapter 1 Introduction and Background

1.1. Statement of the Problem

This research develops and tests a consolidated design methodology for web-based emergency management decision support systems (WEM-DSSs). A WEM-DSS is a decision support system utilizing recent developments in communications, especially Internet technology, for holistic and effective emergency management. An emergency management decision support system (EM-DSS) is a tool to assist emergency managers in all elements related to the holistic planning and management of emergencies, from the earlier efforts aiming at preventing emergencies, to the preparation for the occurrence of an emergency and the management of the actual response should an emergency occur.

The field of emergency management and planning is undergoing a switch in paradigm that entails fundamental shifts in concepts and perspectives. One of such shifts implies a growing realization that emergencies, whether natural or technological, are not simply isolated incidents or events (Erickson, 2006). Rather, emergencies are social phenomena which are influenced by broader economic, social, political variables. Since many such variables are subject to "human control," emergency response is increasingly viewed as a *proactive and participatory* endeavor (contrary to the classical *reactive* view which lacks hazard mitigation and planning) that cannot be realized effectively without a combination of various governmental and private sector partnerships. Another distinct shift in emergency planning theory is related to "multi-hazards" thinking (Department of Homeland Security, 2008). Following the attacks of September 11, 2001, the US Department of Homeland Security (DHS) developed a National Response Plan (Department of Homeland Security, 2008) which has recently changed into the National Response Framework (NRF). The NRF adopts a "comprehensive, national, all-hazards approach to domestic incident response," which describes how various parties (communities, tribes, states, the federal government, private-sectors, etc.) can work together to coordinate national response and best practices in this regard. The multi-hazard view presents the main premise of the NRF since it emphasizes the complex and compound roots of emergencies in society that cannot be simply attributed to a singular, triggering event. This view explains the need for a holistic emergency management approach taking into account the chain of events leading to emergencies.

Emergency management is considered holistic and more effective when it is based on a thorough understanding of the communication channels amongst a large diversity of involved parties (e.g., government agencies, response services, community services, etc.), as well as other factors related to affected communities, the nature of emergencies, and needed actions. Although many of the activities carried out, and the information required to support such activities, are often specific to the nature and scale of emergencies (e.g., wildfire vs. chemical release, local community vs. regional), as well as to the location of the emergency (e.g., rural vs. urban), there are certain universal aspects and information processing requirements applicable to all emergency management activities. A well designed, maintained and operated WEM-DSS requires a comprehensive examination of the diverse types of data and information related to the broad range of such universal activities, procedures, operations, equipment and materials commonly falling under the umbrella of emergency planning.

Equally important to the development of a WEM-DSS is the understanding of the nature of emergency decision making; the limitation, potentials, capabilities of information technology when applied to emergency situations. A thorough assessment of candidate software engineering approaches to the development of a WEM-DSS is a rather important issue in this regard since one of the greatest challenges in the application of DSS is to determine where and how the technology fits into the process of making decisions (emergency decisions in the case of this research). Innovations in communications technology and GIScience enable system designers to build more sophisticated DSSs by integrating spatial analysis and remote access. It is now possible to design a system which involves user participation from different organizations around the world, working on the same data and maps while interacting with each other. Despite all these advances, major methodological problems (see next section) need to be resolved before a WEM-DSS can fully be integrated in emergency management.

1.2. Research Challenges, Objectives and Questions

The literature review conducted in support of this research revealed three major challenges for effective WEM-DSS development:

- The lack of a systematic approach to developing, evaluating and identifying which technology is best suited to a particular type of decision situation during an emergency;
- A need to resolve methodological issues that confound the widespread application of WEM-DSS across different kinds of emergencies; and
- There is a need to demonstrate how WEM-DSS can be integrated into the process of emergency management

Addressing some of the challenges outlined above, this research examines the theory and application of software engineering, decision making, and problem solving approaches, and consolidates and integrates these with the broadly defined field of emergency management. To guide this investigation, and in order to reach the overarching research goal that has been stated previously, three research objectives have been developed. The first research objective is:

- Identify the characteristics of a development methodology for Emergency Management Decision Support System.

This research objective required understanding and exploration of several topics, including state of the art thinking in disasters research, the nature of emergency decision making, a review of alternative methodologies for information systems development as well as human and information systems interactions. These topics were investigated in the literature review chapter of this dissertation. Two research questions were formulated to help address this research objective:

- What are the key elements of an Emergency Management Decision Support System?
- What additional benefits does a Web Based Emergency Management Spatial Decision Support System offer over an EM-DSS? Does a WEM-DSS intrinsically

have different requirements and challenges than an ordinary EM-DSS? If so, what are the differences?

The second research objective is:

- Design and implement a suitable methodology for developing a WEM-DSS for emergency managers in Oklahoma

The core of this research relies on achievement of this research objective. This objective required designing a methodology that is based on surveying topics in the literature review, with a philosophical and practical discussion of them. This design has been realized in the methodology chapter of this dissertation. This objective then required implementing the particular methodology that was designed. The implementation was realized in the implementation chapter of this research. A research question was formulated to address this literature survey.

- What is the optimal strategy for the design and implementation of a WEM-DSS to support holistic planning and management of emergencies?

The third and last research objective is:

- Document and evaluate the development process

It is important to define the term “document” in this dissertation. Documentation is not only a mere collection of “paperwork” necessitated by processes. Rather, it refers to the form of collecting personal experiences, as well as lessons learned during and at the end of the process essentially serving as field or laboratory notes and observations of the processes I worked to employ. In addition to observing the entire process and recording it, these documents recorded problems faced, methods developed to cope with problems, and changes in plans and schedules. This objective was formulated to help direct efforts as the lessons learned in this study would constitute a base for future efforts. The documentation part of this objective was addressed in the implementation part of this dissertation. This objective also included evaluating the development process. It should be emphasized that the focus of evaluation is the development process, and not end product (although the proposed methodology included mechanisms to evaluate the end product as well). The evaluation portion of this objective is addressed in the discussion section of the implementation chapter and in the conclusions chapter. To address this objective, the following research question was formulated.

1.3. Relevance and Contribution

An optimal WEM-SDSS system architecture is simply one that best serves the goals of an emergency organization. To develop such a system, the design methodology needs to ensure the presence of a solid information strategy in harmony with the organization's goals. This, in turn, requires a balanced consideration of technological requirements, the organizational factors, and personal factors. Some visible challenges in achieving this balance stem from the complexity of spatio-temporal data entities making up the bulk of information used in emergency planning and management, and the inter-enterprise structure involving federal departments, state agencies, military, non-governmental organizations, international organizations and potentially many others.

The major contribution of this study is the examination of the relevance of existing architectures such as distributed components and service oriented architectures, and development methodologies including prescriptive models (e.g. waterfall, incremental process, evolutionary and spiral development), specialized process models (e.g. formal methods model) and unified process models to the development of a WEM-DSS. These examinations included evaluation of the applicability of such methods based on objective criteria. The results of this evaluation provided a basis for developing the proposed consolidated methodology for WEM-DSS development.

A second major contribution of this research concerns the relationship between technology adaptation and the adoption of a multi-hazards view within emergency organizations. A number of issues impeding progress in adopting multi-hazards views within emergency organizations have been identified in the literature (Mileti 1999; Mitchell 1999; Alexander 2000; Cutter 2001; Turner *et al.* 2003). Examples include divergent views with regard to the nature of emergency management, lack of comparative indicators, and a broad range of challenges brought about by examining the collective impact of multiple hazards. The role of technology adaptation and more specifically the role of geospatial technologies, in facilitating (or impeding) the adoption of multi-hazards thinking has been overlooked in ongoing discussions so far. The study addresses existing gaps by closely investigating the extent to which successful adoption of spatial support systems in emergency management organizations facilitates their shift to multi-hazard strategies.

A third contribution of this research lies in its empirical contribution. The proposed methodology has been tested on a real-world case study that involves development of a WEM-DSS. The implementation was followed by a post-evaluation, to assess the effectiveness of the new system. To this end, the dissertation provides an empirical contribution through demonstrating a start-to-finish exercise for the application of the proposed methodology that can be replicated in other projects.

This dissertation continues with a literature review in Chapter 2. The literature review starts with observations of a paradigm shift from single hazard to multi hazard oriented emergency management. This shift is then discussed regarding its implications in operational emergency management and subsequent information needs and systems. Discussion of decision support systems included examination of user-centered design for decision making and geographic information systems. Then, information systems for emergency management were discussed in particular, with their characteristics, requirements and examples. Much of the literature review is devoted to examining information systems development methodologies. This portion of the review ranged from traditional sequential models to agile and flexible development methodologies. The chapter ends with a comparison of development methodologies and a summary and conclusions section regarding information systems development methodologies.

Chapter 3 is built on the theoretical discussions drawn from the literature review. It includes a methodology reflecting the integration of the Extreme Programming approach and the Capability Maturity Model Integration (CMMI), and how they were to be implemented in this project. The details of this integration discussed specifically how the concepts of agility and discipline would be balanced in this new methodology. This chapter concludes with a step-by-step explanation of the methodology and an anticipated timeline.

Chapter 4 includes discussion of the application of the proposed methodology as a case study. This methodology was carried out with project initiation, and then three development cycles in an iterative manner. For each development cycle, a number of user stories were implemented. Usually, a number of user stories are organized under a “task”. A task in this study refers to generalized forms encompassing specific user stories. Use case diagrams and activity diagrams accompany the features. Since the methodology is an agile one, changes occurred throughout the project. These changes, along with the justifications for those changes were explained in this chapter. In addition to the methodological modifications, there were some variations from the proposed time schedule. These variations were then analyzed using documentation that contained quantitative data. In addition to the modifications, issues and risks throughout the project were documented during the project and they were discussed in this chapter.

This dissertation ends with Chapter 5 that represents conclusions of this dissertation, in which an evaluation of the proposed methodology for WEM-DSS is undertaken in the light of qualitative and quantitative analyses conducted in Chapter 5. In addition, theoretical and practical contributions and future research directions are discussed in this chapter.

Chapter 2 Literature Review

2.1. Summary

The purpose of this review is two-fold. First, I make a case for the working hypothesis of this dissertation with regard to the promising paradigm of multi-hazard emergency planning and the extent to which successful adoption of spatial support systems in emergency management organizations facilitates their shift to multi-hazard strategies. Second, the chapter provides the basis of the proposed methodology through evaluating and consolidating the major ideas underpinning various approaches to the development of GIS and spatial decision support systems, based on their relevance to the success or failure of information systems projects in emergency organizations. The review concludes with a discussion of the various technological and organizational factors influencing the adoption of decision support systems in emergency organizations, which need to be addressed by the proposed methodology.

2.2. Single Hazard to Multi-Hazard Paradigm Shift in Emergency Management

As Erickson (2006) emphasizes, the all-hazard approach constitutes a paradigm shift in the field of emergency management. This shift is a result of a realization by many emergency planners and practitioners that the old ways of responding to emergencies may not be totally right, or not as effective as many would like them to be. A call for a change towards new ways in emergency planning is increasingly voiced and this will require development of new methodologies, technologies and approaches. The multi- or all-hazards approach has been proposed as one of the potential paths for the needed change. Compared to a single hazard approach, a multi-hazards approach requires more collaboration between federal, state and private organizations and as thus, it comes with associated costs both in term of finances, information sharing, technology adaptation, and organizational factors (Erickson, 2006, p. 232).

The call for adopting a multi-hazards approach in emergency planning is not a new idea. As early as 1985, McLoughlin (1985) emphasized the needs for a shift from a narrow purpose, single hazard view to a broader, multi-hazard view of emergency management, which he identifies as an Integrated Emergency Management System. McLoughlin's idea of multi-hazards thinking focused on commonalities of emergency functions across different hazards, while addressing specific requirements unique to particular hazards and emergencies. According to him, realizing this goal was one of the reasons for establishing Federal Emergency Management Agency in the USA. .

The call for a shift towards a multi-hazards thinking in emergency planning is not limited to the U.S. context. New Zealand is one of the countries that values a multi-hazards approach. As Jensen (1998) indicated, the Emergency Management Office of Wellington City adopted an all-hazards, all-agencies program for emergency management in 1993. Jensen draws attention to the danger of communities focusing on one single hazard too much but neglecting other kinds of hazards. Like others, he expresses the importance of extracting commonalities among hazards to avoid duplication in efforts.

Britton and Clark (2000) define the elements of a new emergency management framework as comprehensive, integrated emergency management systems, and an all-hazards approach. Dennis S. Mileti and Lori Peek-Gottschlich (2001) conclude that local emergency management will require a multi hazard approach utilizing risk assessment maps and tools that should be reinforced by federal investment. Tralli *et al.* (2005) discuss the potential benefits of risk modeling integrated into multi-hazard analysis and decision support in order to provide more accurate results for the international disaster community. Tolentino (2007) specifically addresses tsunamis, and discusses how tsunami early warning can be cost effective when it is integrated into a multi-hazard system, which in addition to tsunami also considers less frequent hazards. Carreño *et al*. (2006) proposes a method for urban risk evaluation that is multi-hazard and holistic for decision making support. Kershaw and Mason (2006) narrate discuss about the implications of the Indian Ocean tsunami disaster for multi-hazards mitigation and preparedness at the national and international levels.

Among all the authors, Quarantelli (1999) is the one who makes a well-articulated and comprehensive case for why a multi-hazard or generic hazard approach is a better choice for emergency management. His initial argument is that there is no important distinction between technological and natural disasters, since their impacts are very similar while sharing many common elements. He lists three major reasons, theoretical, empirical and practical, for why such switch to this approach is deemed important. From a theoretical point of view a hazard does not automatically result in a disaster but in negative social consequences that have common properties irrespective of the type of the hazard. From an empirical point of view, in many human related problems in emergency tasks, such as warning, evacuation, sheltering, feeding, search and rescue etc., the type of hazard causing the social disruption does not matter that much. Practical reasons involve cost efficiency, politically informed strategy, avoiding duplication of efforts and increase of efficiency.

The impacts of this paradigm shift can be seen in the developing policy statements and planning guides such as the National Response Framework (NRF). This guide for conducting all-hazards response prepared by the Department of Homeland Security (2008) reflects the incorporation of a multi-hazard approach. It also describes organizational structures and defines key roles and responsibilities to link governmental and non-governmental institutions engaged in national emergency response at all scales. It provides best management practices for potential and actual incidents that range from local events to larger incidents, and from terrorist attacks to natural disasters.

The National Response Framework is mainly constructed upon National Incident Management System (NIMS), which provides a template for managing emergencies. The NIMS provides a range of standardized command and management structures allowing the responders from different jurisdictions and disciplines to work together for emergency management.

The NRF’s target audience is comprised of government executives, private-sector and non-governmental organization (NGO) leaders, and all emergency management practitioners. It assigns the governments the responsibility to develop comprehensive all hazards response plans. These plans should have some generic attributes, such as defining leaderships and roles. The contents should cover generic plans that cross hazards as well as hazard-specific strategies.

But, what are the implications of this shift on emergency management? What are the changes that are required in the emergency management mechanisms in order to accommodate new frameworks for understanding hazards responses?

As it has been made clear, the multi-hazard or all-hazard approach focuses on first defining the commonalities and then defining the differences among considerations for and effects of hazards. While this is argued to significantly reduce the total information required for emergency management, it increases the need for effective ways to manage such information. Considering that single hazard emergency models already demand efficient information management, a multi-hazard view will increase this demand while necessitating improvements of existing information systems. Necessities stemming from the user base, data, information products and communication channels may even extend to a point where the existing information systems are no longer sufficient and cannot be improved further. Improvement of information systems may bring about several challenges. Underlying hardware structure, data throughput, organizational resistance and etc. may impede the implementation of required improvements. However, usually the most fundamental challenge for change is the existing system architecture. Many times, the redevelopment of a system from scratch will be easier than attempting to improve upon existing systems, since continuous modifications and maintenance would be the more costly option.

In the context of disasters, critical information is valuable and can be used to save lives or critical infrastructure. Information Systems can store, maintain and transmit large volumes of data that are crucial for emergency management. Disaster information may span from preparedness and early warning information, e.g. weather and population data, to response information, e.g. critical facilities locations and to response information e.g. damage and cost information. Especially shortly before and during emergencies, the volume of the information necessary to the emergency management officials increases dramatically. Information management without information and communication technologies is very difficult, if not impossible during these times. A multi-hazard approach compared to a single-hazard approach will take into account more hazards, resulting in more information requirements both in size and complexity.

2.3. Decision Support and Information Systems

2.3.1. Introduction

Holsapple and Whinston (2001) discuss that classical decision-making focuses on examination of alternative courses of action. This examination may involve issues such as the extent an alternative should be studied; reliability of expected impacts of each alternative; the framework to compare alternatives; and identification of a strategy to choose an alternative as the final decision. Aside from the classic view, there is the knowledge-based view, according to which any decision is a piece of knowledge that is descriptive in nature. A step by step description of actions suggested by a decision is an example to this. Holsapple and Whinston (2001) also argue that both views are compatible with each other. The basic assumption for their reasoning is that the process for producing decisions (including the process in classic view) always results in knowledge.

A decision context is an important parameter in decision support systems, and it refers to the characteristics of the setting where the decisions are made. From an organizational point of view, top management makes strategic decisions and middle management makes control and policy assurance related decisions. Another contextual attribute is related to whether the situation is *established,* which relies heavily on past decisions or *emergent,* which relies on qualitative judgment. Organizational structure also constitutes an important part of the decision context. Centralized organizations have fewer power centers that are authorized to make decisions (Holsapple and Whinston, 2001).

2.3.2. Decision Support Systems (Information Systems for Decision Support)

According to Holsapple and Whinston (2001), a DSS serves the following purposes: (1) providing decision makers with necessary information that can be used in decision problems; (2) providing design and choice alternatives; (3) facilitating problem solving; (4) providing aid for unstructured decisions; (5) managing knowledge by storing and organizing user experience. A typical DSS as depicted in Fig 2-1will have several attributes including a body of knowledge that involve aspects of decision making process. The system should allow knowledge acquisition by various sources, and should be able to serve knowledge in customized ways such as presenting synthesized or subsets of information.

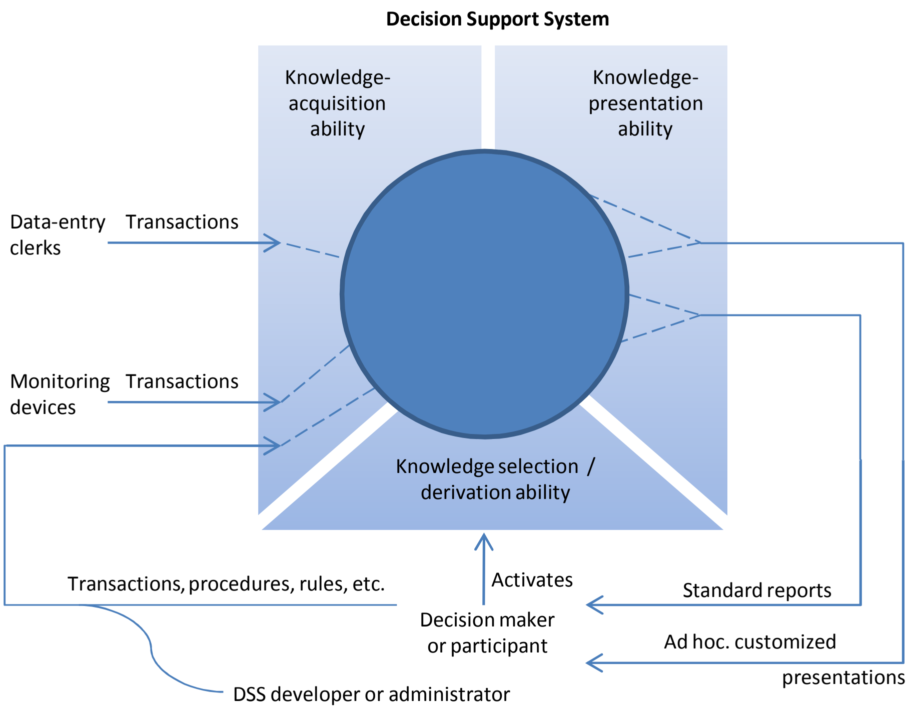

FIGURE 2-1 A TYPICAL DSS. MODIFIED FROM: (HOLSAPPLE AND WHINSTON, 2001)

Types of DSS

Power (2005) identifies five generic types of DSS. These are

- Communications Driven DSS: They are used to facilitate meetings, so that users can share their ideas.
- Data Driven DSS: They help decision makers to analyze, display and manipulate large and structured internal and external data sets.
- Document Driven DSS: They allow decision makers access and manipulate text based documents containing qualitative data.
- Knowledge Driven DSS: They are essentially used to provide decision makers with advice.
- Model Driven DSS: They allow decision makers to utilize statistical, algebraic, financial and simulation models embedded within the system.

These DSS are not mutually exclusive. Power (2005) discusses the difficulty of classifying a DSS into a single category. He for example uses the term Group DSS as a hybrid between Communications Driven DSS and Model Driven DSS. A Group DSS is used to facilitate decision making for semi structured and unstructured problems for decision makers working as a group.

2.3.3. User Centered Design for Decision Making

2.3.3.1. Techno-centered vs. User-centered Design

The main identifying aspect of techno-centered development is the focus on the technology instead of the human factors. There are important differences between techno-centered and user-centered information systems designs as they are contrasted in Figure 2-2. In techno-centered type of development, information systems are both specified and developed by technologists, and these technological products are “pushed” to end users, without waiting the end users to “demand” such technologies.

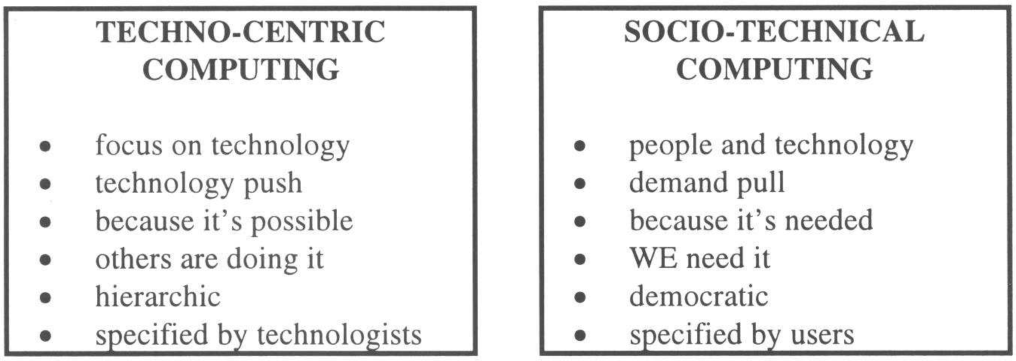

FIGURE 2-2 FROM TECHNO-CENTRIC TO SOCIO-TECHNICAL COMPUTING. SOURCE: REEVE AND PETCH (1999)

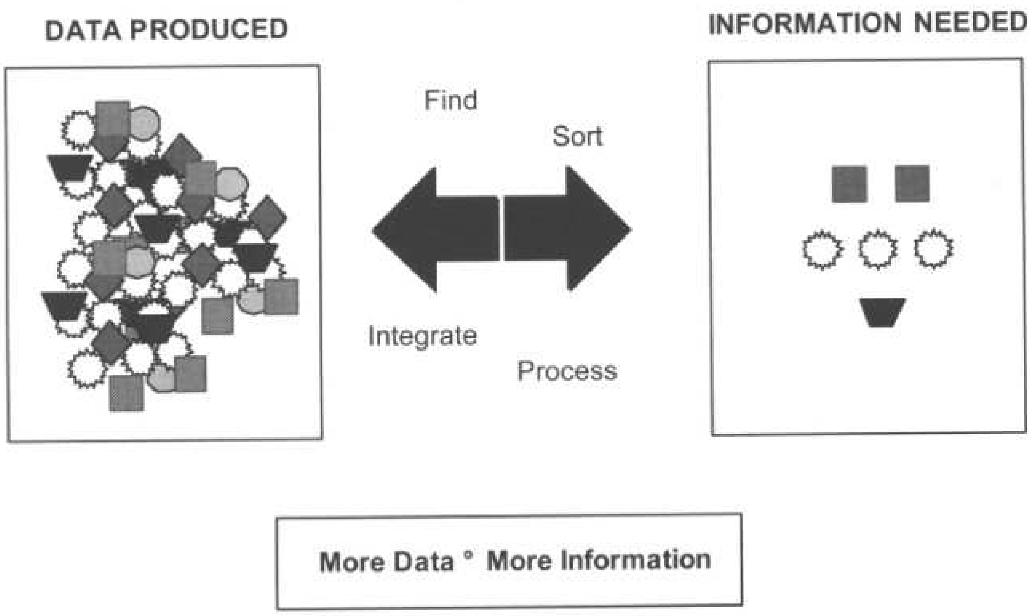

FIGURE 2-3 THE INFORMATION GAP (FROM ENDSLEY, 2000B). SOURCE: ENDSLEY ET AL. (2003)

An issue that arises with the techno-centric development is management of data that is produced in large quantities. Endsley *et al.* (2003) draw attention to increasing amount of information available to end users to emphasize the need for user centered information systems design and development. They argue that such an information-rich environment challenges people in making good and timely decisions, as there can be too much information to handle. They also claim that many people are less informed than before, as it is more difficult to find what people actually need to know; and they point out to the widely accepted fact: “more data do not mean more information”. This gap between massive amounts of data and inability to process it into useful information takes us from the conventional techno-centered designs to user-centered designs. A *techno-centered* design lies on the assumption that a good system should serve desired technologies and have all necessary functionalities which are to be developed by engineering methods. This type of design overlooks whether users of the systems would be able to utilize the system as intended. As the functionalities and corresponding display mechanisms increased, that starts to be a crucial problem as humans have a certain processing capability. In other words, humans have information processing bottlenecks, as they can only focus on and process certain types and amounts of data at once, or for a certain, limited duration. Such bottlenecks often drive people to make mistakes, make wrong decisions, many of which may have fatal results. *User centered* designs on the other hand, aim to organize information based on the capacities and most important needs of decision makers. “This philosophy is not borne primarily from humanistic or altruistic desire, but rather from a desire to obtain optimal functioning of the human-machine system”. The main purpose is to decrease human error and increase user satisfaction. Its main principles are:

- Organize technology around user’s goals, tasks, and abilities
- Technology should be organized around the way users process information and make decisions
- Technology must keep the user in control and aware of the state of the system

These principles are the main factors leading to the realization of situational awareness as the key to the user centered design.

2.3.3.2. Situation Awareness

Situation awareness (SA) is "the perception of the elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future" (Endsley, 1988). It is a concept originated in aviation discipline; however its principles can be applied to other disciplines as well. According to Endsley (1988), it is a critical input to decision making. There are three levels of situational awareness.

The first level of SA is perception of elements in the environment. More specifically, this refers to perceiving the attributes and dynamics of relevant elements in the environment of interest using senses. For each type of task these elements and the particular attributes of interest may be different. Perceived information can come from a variety of senses, including visual, auditory, tactile, taste and olfactory, or a combination of these. It is important to note that each piece of perceived information has a certain reliability level. This reliability becomes the basis for confidence in information, which is an integral part of Level 1 SA (Endsley *et al.*, 2003).

The second level of SA is comprehension of the current situation. This level constitutes understanding and interpreting the cues that are collected at level 1. Comprehension involves synthesizing disjointed level 1 elements; and integrating them to create useful information (information that is associated with the goals and has importance to potential decision making processes) and making necessary prioritizations among the information pieces (Endsley *et al.*, 2003).

The third level of SA is projection of future status. If a person knows what the elements of interest are in an environment, and how they relate to the goals, then the person can predict the actions of those elements in the future. This is only possible with a good comprehension of the situation. This can be very demanding mentally, and usually requires much time and effort to generate predictions constantly to form strategies whenever necessary (Endsley *et al.*, 2003). This entire mechanism as placed in a greater context is illustrated in Figure 2-4.

Situation awareness is tightly related to space and time. According to Endsley (1995), "SA is not necessarily acquired instantaneously but is built up over time. Thus it takes into account the dynamics of the situation that are acquirable only over time and that are used to project the state of the environment in the near future"; and; "Pilots and traffic controllers, for instance, are concerned with the spatial relationships among multiple aircraft, and this information also yields important temporal cues"; and; "[…] spatial information is highly useful for determining exactly which aspects of the environment are important for SA".

When there are several individuals working together to make decisions, each team member may have a specific set of SA elements. Some overlap among the members should also be present, to make possible any team coordination and overall team SA.

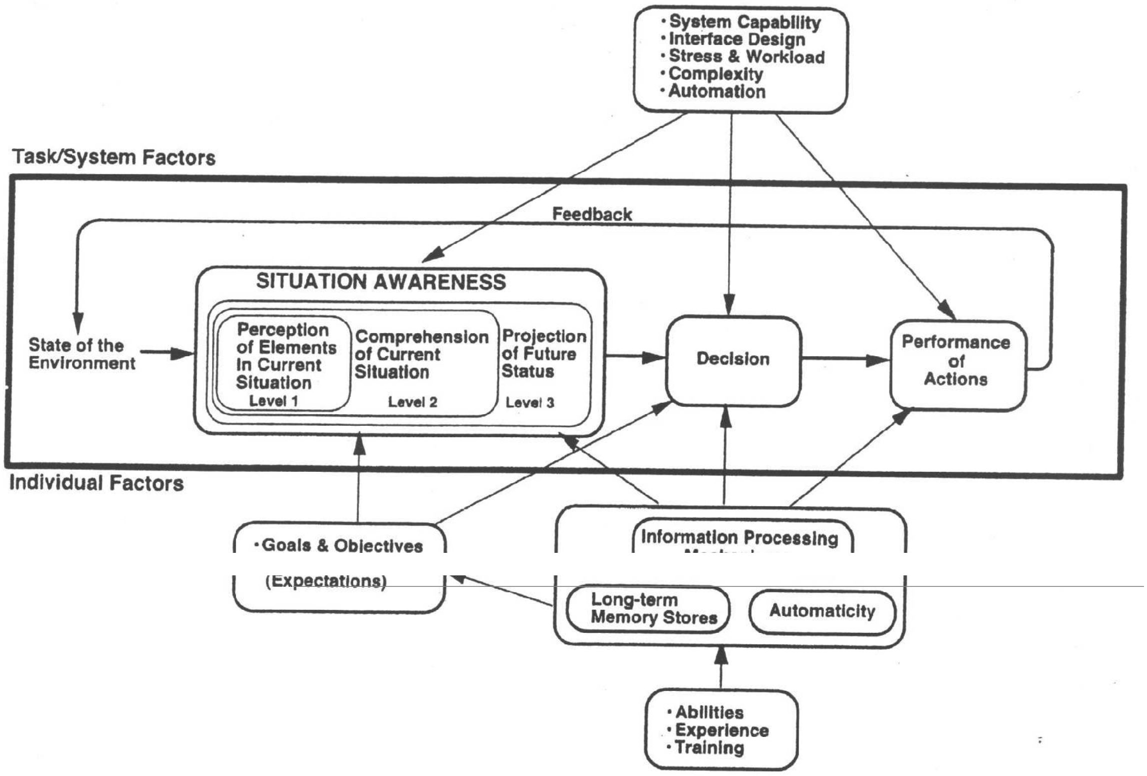

FIGURE 2-4 MODEL OF SITUATION AWARENESS IN DYNAMIC DECISION MAKING. SOURCE: ENDSLEY (1995)

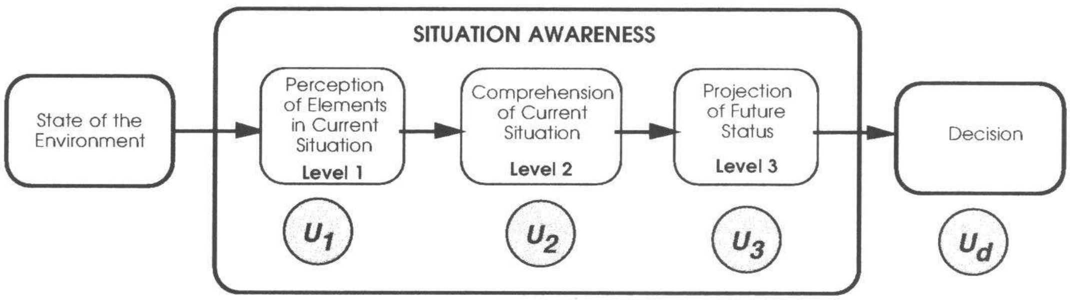

FIGURE 2-5 TYPES OF UNCERTAINTY IN THE DECISION PROCESS. SOURCE: ENDSLEY ET AL. (2003)

Situation awareness and therefore decision making performance is dramatically affected by uncertainty and its negative correlate, confidence. There can be varying degrees of uncertainty associated with user perception (u_1), comprehension (u_2), projection (u_3) and decision making (u_d) as shown in Figure 2-5. Accordingly, the quality of the decisions will be higher with good situational awareness and high confidence levels (Endsley et al, 2003).

Besides being an input to decision making, situational awareness can directly impact the decision making process. Manktelow and Jones (1987) concluded that situational characteristics may influence a person's ability to make a decision through appropriate mental models. Endsley (1995) reviewed the literature and showed that the way a problem or situation is presented can significantly determine how the problem is solved, or how a decision is made after.

Situation awareness can be affected by automation processes, which may be a part of or supplement to decision making processes and are intended to improve such processes. Endsley *et al.* (2003) argue that automation can adversely affect situation awareness in three aspects. Firstly, it decreases users' ability to detect system related errors when working with automated systems. Secondly, with automation users may not be able to acquire a clear understanding of why and how the system operates which are required for comprehension and projection phases of situation awareness. Thirdly, Endsley *et al.* (2003) concluded that automation output may become highly crucial and influential for the user as in serial systems instead of using the automated output as recommendations as in parallel systems. In serial systems system recommendations are required to take actions. On the other hand parallel systems refer to users operating independently from the decision support system which provides optional recommendations to the user. The problem with serial systems is that they usually decrease the performance of the decisions as seen in Figure 2-6.

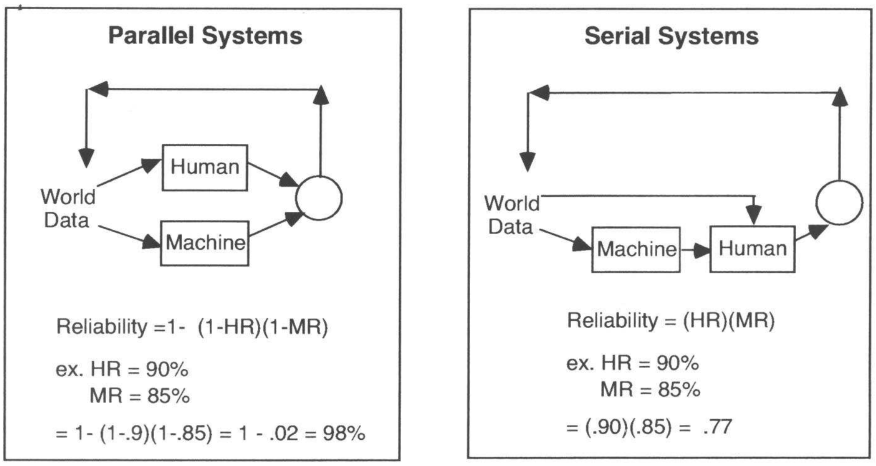

FIGURE 2-6 EXAMPLES OF RELIABILITY WHEN HUMAN AND MACHINE COMPONENTS ARE OPERATING IN PARALLEL VS SERIAL MODES. SOURCE: ENDSLEY ET AL. (2003)

Endsley (1995) provides three levels of elements for situational awareness for air-to-air aircraft that are derived from the methodology presented by Endsley (1993).

> *Level 1:* location, altitude and heading of ownship and other aircraft; current target; detections; system status; location ground threats and obstacles
>
> *Level 2:* mission timing and status; impact of system degrades; time and distance available on fuel; tactical status of threat aircraft (offensive/defensive/neutral)
>
> *Level 3:* projected aircraft tactics and maneuvers, firing position and timing

This is the only model describing the elements for situational elements within a hierarchy and can be useful model for disaster management as well. As it can be deduced from above mentioned descriptions, level 1 refers to current spatial properties, level 2 refers to current temporal properties and level 3 refers to future spatio-temporal properties of elements. These can be adopted into disaster management by translating same properties of elements in a disaster situation:

> *Level 1:* distribution of landuse characteristics (e.g. vulnerable features), location of resources (e.g. fire dispatch units, emergency centers etc.), spatial characteristics of weather conditions, location of potential hazards
>
> *Level 2:* approximate timing of hazard arrival, availability of manpower and equipment to respond, estimation of time necessary to take measures such as sheltering or evacuation
>
> Level 3: projected response actions, projected allocation of resources and comparison of alternative scenarios.

Enemies of Situation Awareness

Endsley *et al.* (2003) explains SA demons, the factors that deter situational awareness as follows:

Attentional Tunneling

Decision makers need to be aware of potentially important and relevant information. In case of emergency management, emergency managers need to continuously pay attention to various factors and various information channels that may have critical significance. However, sometimes a decision maker may lock into certain aspects of the environment, or certain information flows resulting in a situation called attentional tunneling or attentional narrowing. In that case, decision maker will be outdated on other potentially important aspects of the environment (Endsley *et al.*, 2003).

Requisite Memory Trap

Requisite memory trap originates from the fact that human short term memory (which is also known as working memory) is limited. This can pose a problem as the important elements about a situation may be eliminated as time passes (Endsley *et al.*, 2003).

Workload, Anxiety, Fatigue, and Other Stressors

Stressors like workload, anxiety and fatigue can reduce the working memory. People also may collect less information if exposed to stress. Therefore, these conditions might result in undermined situational awareness and more errors (Endsley *et al.*, 2003).

Data Overload

Data overload can happen if the amount of the data and the rate the data changes overwhelms the person’s sensory and cognitive system (Endsley *et al.*, 2003).

Misplaced Salience

Salience in situational awareness refers to relevance or prominence of information, and is characterized by physical characteristics of the representation due to the fact that people’s perceptual system is more sensitive to certain forms. People will usually pay attention to information relevant to their goals. However, there may be irrelevant information that attracts more attention than they are supposed to due to misplaced salience (Endsley *et al.*, 2003).

Complexity Creep

Complexity creep originates due to systems with too many features in quantity and complexity. This is a result of insufficient mental models or internal representations of how these systems operate. This may cause reduced ability to receive information, besides deterring the ability to correctly interpret information presented by the system (Endsley *et al.*, 2003).

Errant Mental Models

Mental models are formed through learning systems, and they tell how to interpret given information provided by those systems. There may be occasions when users employ similar systems that require different interpretation of similar looking information or representations. This can cause the use of a wrong or incomplete mental model, therefore resulting in poor interpretation (Endsley *et al.*, 2003).

Out of the Loop Syndrome

Out of the loop syndrome is caused by automation. If much of the system is on automated mode, users may not have good situation awareness. This is because users do not know neither how exactly automation works, nor the state of the elements that are in the automation. Problems can also exacerbate further if automation fails and user does not notice it (Endsley *et al.*, 2003).

Principles of Situation Awareness Oriented Design

A number of principles for SA oriented design are adopted from Endsley (1988): Accordingly, to achieve better SA, divided attention requirements need to be minimized. This can be done by presenting and organizing information based on spatial proximity, optimizing short term memory, storing multiple attribute information in spatial objects while minimizing number of objects and minimizing required attentional shifts by minimizing number of separate displays

Long term storage should be able to be accessed as quickly as possible with information organization and object categorization. Cues that are more important to long term memory should stand out in the design to provide rapid pattern matching. For both short and long term memory, the most important and relevant information should stand out the most perceptually. As a principle, verbal information should be minimized, especially regarding spatial data. As access to spatio-temporal information is important, the system should provide information regarding the trends and rates of changes in conditions.

User's memory can be optimized by adjusting the amount of information they are exposed to. This can be done by functionalities that allow users to increase or decrease the level of detail. Similarly, as attentional constraints may be present, the system should be able to filter the abundant information based on the relevance and importance. As a part of interface design principle, peripheral vision can be utilized to input some of the non-crucial information. Additional modes of input can be utilized such as auditory and tactile simultaneously with the visual input. More important information should have input redundancy, especially visually. Attentional narrowing should be avoided. This can be improved by providing simultaneous access to secondary information which will not interfere with primary tasks. System should provide means to relate the user himself/herself to the information spatially.

2.3.4. Geographic Information Systems (GIS) and Spatial Decision Support Systems (SDSS)

Power (2005) defines SDSS to be utilizing GIS technologies to support managers for analyzing data with some spatial component. According to Lianfa *et al.* (2005), traditional and commercial GIS are powerful for capturing, storing, visualizing and manipulating geospatial data, but their analytical, modeling and inference capabilities are rather limited. An SDSS offers additional analytical capabilities for integration of spatial and mathematical models. SDSSs can deal with semi structured and unstructured problems as well by integrating fuzzy and uncertainty functionalities.

According to Reeve and Petch (1999), GIS can be used at different levels in organizations (See Figure 2-7). At an operational level, GIS is primarily used for larger volumes of information processing with limited spatial analysis capabilities. At management level GIS is used to utilize information coming from the operational level, as well as external resources. The IS systems to help managers making decisions are often called DSS. At the executive level decisions are more strategic, requiring more unstructured data. Unstructured data refers to data without data models that allow querying, or with structures that are not useful for particular intent. Images (which are very large in size and often needed for spatial decision making), videos, audios and text documents are examples to unstructured data. Penetration of GIS is rather limited at this level. While Reeve and Petch point out that IS for each level are usually separate software packages, they argue that ideally IS across the whole organization should be seamlessly integrated.

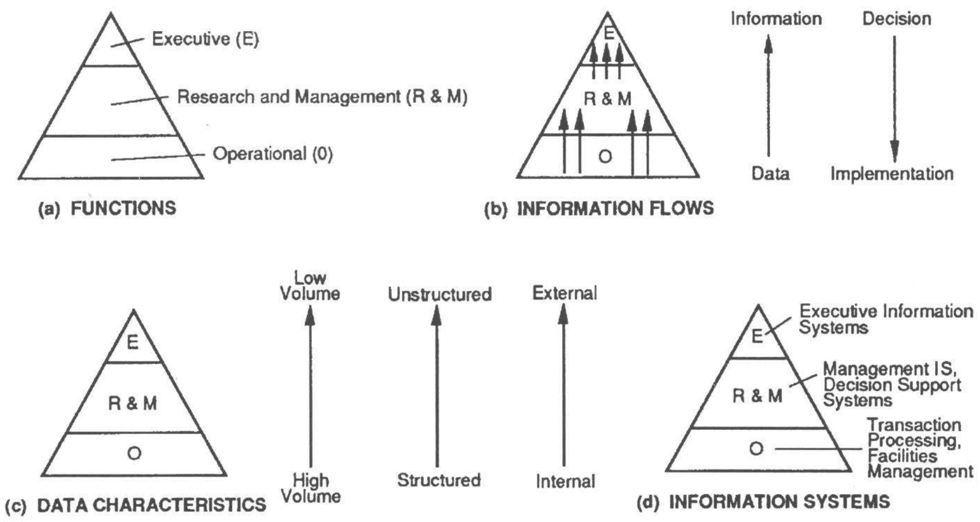

FIGURE 2-7. THE TRIANGULAR STRUCTURE OF ORGANIZATIONS. SOURCE: REEVE AND PETCH (1999)

2.4. Information Systems and decision Support for Disaster Organizations

2.4.1. Emergency Decision Making Characteristics and Properties

Jianshe *et al.* (1994) argues that emergency decision making has some subtle differences compared to common decision making; (1) Usually attributes of the problem are uncertain (form, nature, where and when); (2) Decision environment is more prone to change uncontrollably; (3) Need for decision making in very limited time with lack of information; (4) Only one or two most important goals should be pursued, and a satisfaction criterion should be adopted

Based on these factors, they further argue that a practical emergency decision making methods should have certain properties including: (1) Conciseness: Decisions are easy to understand by common users; (2) Limited Interaction: The prior setting of preferences and parameters that will apply to most conditions, so that users will not spend time adjusting said parameters during an emergency; (3) Robustness: Methods should be able to incorporate imperfect data; (4) Dynamic adaptability, which allows modification of both internal and external parameters any time.

2.4.2. Examples of EMDSSs

Dai Jianshe *et al.* (1994) point out that pre-disaster and post-disaster functionalities of EMDSS are not the same. Prior to disasters, EMDSS should serve the purpose of preparation, such as data collection, scenario development and prediction. During and after the disasters EMDSS should aid in incoming data analysis, developing decision alternatives and helping to decide among alternatives.

HAZUS (**HAZ**ards **U**nited **S**tates) was conceived in the early 1990s as a free, general purpose natural hazards loss estimation software tool for use by a broad range of persons and agencies concerned with natural hazards mitigation and decision making. HAZUS has undergone continuous improvements both in terms of programming environment, database platform, and the GIS platform it is based on, resulting in faster operating speeds. Its functionality has been expanded by addition of new models and tools (Schneider and Schauer, 2006).

RODOS (Real-time On-line DecisiOn Support system) is designed as a generic tool for providing support for off-site emergency management of nuclear accidents with the support of European Commission and German Ministry of Environment. It can serve at different levels: (1) for acquisition and presentation of radiological data, (2) analysis and prediction of radiological situation, (3) simulation of countermeasures such as evacuation and their feasibility and finally (4) evaluation and ranking of alternative scenarios (Ehrhardt, 1997).

CEMPS (Configurable Evacuation Management and Planning Simulator) is a prototype SDSS utilizing simulation modeling and GIS to serve evacuation planning. The system is made configurable using terrain, road network, population, hazard source and shelter information (Silva, 2001).

Levy *et al.* (2007) discuss a framework for Multi-Criteria Decision Support Systems which involve a database component (including environmental, social and chemical data), flood modeling functions and multi criteria decision analysis (MCDA) techniques and a graphical user interface (GUI) to display analytical results and model outputs. They discuss potential benefits of temporal GIS data and remotely sensed imagery as well as Analytical Hiearchical Process and Analytical Network process as MCDA models.

FIMS (the Fire Management Information System) is an application combining commercial products with specially designed software. Its architecture consists of DBMS, an information manager and a GIS. Its functionality includes weather monitoring, fire risk rating, fire fighting advisory, fire detection and fire modeling (Wybo, 1998). NADSS (National Agricultural Decision Support Systems) project is developing a geospatial decision support system for drought risk management. The project researchers have been developing data mining and retrieval techniques, constraint databases, spatial analysis and visualization tools. Its architecture is composed of three low level layers (data, information and knowledge) and presentation layer (Goddard *et al.* 2003).

Sahana, a free and open source disaster management system that is built by Sri Lankan IT volunteers. Its development was initiated by Indian Ocean Earthquake and the following tsunami. It was developed within three weeks, and was authorized as part of the official portal for the Center of National Operations, the official body for coordinating relief efforts in Sri Lanka. Sahana system is composed of components, interacting with each other via shared databases. Core components are (1) Organization registry, (2) Missing People / Disaster Victim Registry, (3) Camp Registry, (4) Request Management System. Layer additional components were added including (5) Inventory Management System, (6) Messaging Module, (7) Situation Mapping Module, (8) Synchronization Module. Sahana has officially been deployed for response efforts for 2005 Pakistan earthquake, 2006 Philippines mudslides and the 2006 Yogyakarta earthquake in Indonesia (Currion *et al.* 2007)

Schenker-Wicki (1997) presents the development of Swiss DSS for evaluating countermeasures reducing ingestion dose after an accidental release of radioactivity. The system has four modules for threat assessment, generation of countermeasures, specification and decision making at the political and technical level.

The Office of Emergency Preparedness (OEP) had the responsibility to develop a system called Emergency Management Information System for the Wage Price Free (EMISARI) in 1971 as a communications system. This system has been used for strikes, energy shortages and some natural disasters, while allowing 200 to 300 users scattered around the US (Turoff, 2002).

Lianfa *et al.* (2005) designed a prototype spatial DSS to provide decision support regarding hazard simulation, fuzzy comprehensive evaluation of risk and query for insurance pricing in China. This SDSS takes into account four factors for insurance pricing (1) The spatio-temporal patterns of natural hazards ; (2) The spatio-temporal variation of exposure; (3) Past claims and their correlation with different policies; (4) Uncertainty and other factors: This refers to the quality and availability of spatial data and modeling methods.

Insurance modeling library includes five modules made up by Component Object Modeling objects (Lianfa *et al.*, 2005): (1) Hazard occurrence module for statistically and mechanically estimating spatio-temporal patterns of hazards based on past data as well as meteorological hypotheses; (2) Comprehensive risk analysis module to estimate the comprehensive risk level based on probabilistic and deterministic simulations; (3) Zonal correlation module to calculate spatial correlations among vulnerable areas; (4) Vulnerability and loss analysis of exposure module to find the loss in money value ; (5) Rate-making and pricing module to calculate insurance rates and premiums based on potential loss.

2.4.3. Requirements for EDSS

Jianshe *et al.* (1994) lays out general requirements for EDSS. These include large storage, high speed information processing, analysis tools, expert level inference and a reliable communication network. Functionalities are organized based on whether the function is in operation before or during the disaster. Before the emergency, the EDSS should (1) collect and store information relevant to potential emergencies, (2) acquire and store expertise on disasters, (3) help design emergency plans, (4) evaluate emergency plans and scenarios with simulations, (5) monitor situation and predict potential disasters. During the emergency the EDSS should (1) collect and transform information about emergency (2) analyze the emergency situation, (3) provide alternatives for decisions, (4) to help decision maker select an alternative. The authors also stress the importance of flexibility in system structure, having high degree of software automation and a knowledgebase of common emergency measures.

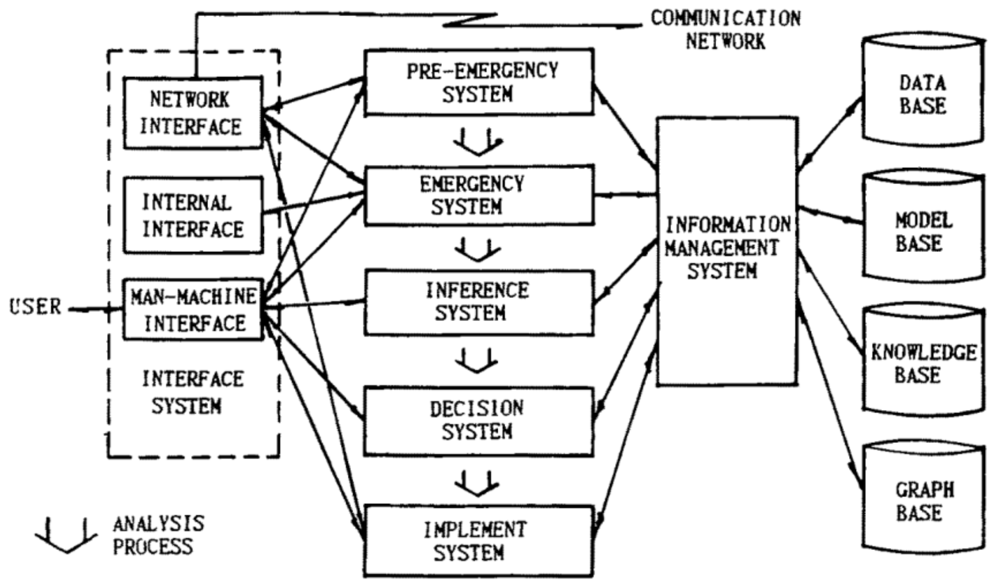

FIGURE 2-8 STRUCTURE OF A KNOWLEDGE-BASED DISTRIBUTED EDSS. SOURCE: JIANSHE ET AL. (1994)

The structure in Figure 2-8 is composed of core parts of (1) pre-emergency system for data collection, storage, data analysis and prediction; (2) emergency system to deliver information during emergency; (3) inference system to give response measures to decision maker; (4) decision making system for evaluating alternative actions; (5) information management system to access data, models and other sources for other modules.

Oklahoma's First-Response Information System using Telecommunications (OK-FIRST) is a support system developed by Oklahoma Climatologic Survey in 1996, based on "perception of a near-complete lack of real-time weather information". The purpose was then to provide real-time weather information available to local public agencies via comprehensive Web services (Morris et al, 2001). It was first developed as a tool for delivering the formal education program of Oklahoma Climatological Survey. The original idea and the goal were to "develop a transportable, agency-driven information support system to help public safety agencies harness the information age". By 2001, more than 100 public safety agencies, most of which were from rural areas, received training on utilizing OK-FIRST for their operations (Morris et al, 2002).

Its development involved design decisions for integration to Web Browsers (Morris 1998). Using web and plug-in integration strategy instead of providing imagery in a common displayable format served several benefits including: (1) reducing server's information processing load; (2) scalability: ability to serve multiple users; (3) interactivity: users can query, animate and zoom; and (4) flexibility: users can customize view, generate overlays and etc. Software development was also highly improved based on the user feedback by addition of new features (Morris et al, 2001). In addition, OK-FIRST staff provided assistance with the software and data products regularly throughout the development processes to increase familiarity (Morris et al, 2002).

Most of OK-FIRST's resources are directed into the delivery of NEXRAD information dissemination service (NIDS) and Mesonet data (Morris et al, 2002). The Mesonet has 114 sites measuring 20 variables and NIDS is provided by 20 radars having 20 data products each (Morris, 1998). An Oklahoma Mesonet station includes a datalogger, solar panel, radio receiver, lightning rod and environmental sensors that are installed on a 10 meters high tower (McPherson *et al.*, 2007).

Provision of the decision support system, along with the training sessions and workshops decreased the amount of interaction between public safety agencies and National Weather Service (NWS), partially due to the fact that agencies did not need detailed descriptions of digital information as much as before (Morris *et al.*, 2002). Before acquiring access to the OK-FIRST system, each completes a weeklong workshop, to gain necessary skills in computer literacy and interpretation of weather data for decision making. Following three sets of workshops a program evaluation was conducted. Evaluation revealed that over 95% of the participants were satisfied with the OK-FIRST web site, training and ongoing support (Morris *et al.*, 2001).

Morris *et al.* (2001) collected specific comments from users highlighting the use and benefit of the system. They include examples of use of the system so that officials made right decisions for evacuation decisions, convincing other officials, and shelter warnings for severe weather; early response for fire; and early warning for floods.

2.5. Development of Information Systems

2.5.1. Software Engineering

The term software engineering was first introduced by Fritz Bauer (Naur and Randell, 1969) in 1968 in a conference as

> "The establishment and use of sound engineering principles to obtain economical software that is reliable and works efficiently on real machines"

Many other definitions of software engineering were later introduced. One of the more comprehensive definitions was introduced by Institute of Electrical and Electronics Engineers (IEEE, 1990) stating:

> "Software Engineering: (1) The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software. (2) The study of approaches as in (1)."

According to Pressman (2005), software engineering methods provide the technical how-to's for developing software. Such methods include a variety of tasks such as communication, requirements analysis, design modeling, testing and support.

2.5.2. Information Systems Development Methodologies

Information systems development methodologies (ISDM) which are also known as software process models are an essential part of software engineering activities. Kurbel (2008) defines these as:

> "A software process model is an ordered set of activities with associated results that are conducted in the production and evolution of software. It is an abstract representation of a type of software process."

Avison and Fitsgerald (2008) define ISDM as

> "A collection of procedures, techniques, tools, and documentation aids which will help the systems developers in their efforts to implement a new information system. A methodology will consist of phases, themselves consisting of subphases, which will guide systems developers in their choice of the techniques that might be appropriate at each stage of the project and also help them plan, manage, control, and evaluate information systems projects"

Avison and Fitsgerald (2008) identified four eras throughout the emergence of methodologies:

- Pre-methodology era
- Early-methodology era
- Methodology era
- Methodology reassessment era

There was little reliance on any formal development methodologies in the early years of software development. Such efforts were usually characterized as “code and fix” approach. According to Avison and Fitsgerald (2008), until the 1960’s, development efforts were focused on programming and the skills of programmers. Programmers displayed very good technical skills, but their communication skills were lacking. They would then use some rule of thumbs, and depend on their own experiences rather than relying on end user input. This would result in expenditure of time mostly in fixing the codes they created. A “vicious circle”, as defined by Avison and Fitsgerald (2008), compromised of programmers being asked for modifications, which resulted in undesirable effects, which in turn needed to be changed as well.

Avison and Fitsgerald (2008) point to two main factors helping the emergence of formal methodologies for software development (p. 24). First of all, it was realized that design and development of information systems required skills that programmers may not possess. Secondly, as information systems were growing in size and complexity, software standards and disciplined methods for development were appreciated more.

Numerous ISDMs have been proposed throughout the history of software engineering. It is possible to classify them in a variety of ways. According to Kurbel (2008), they can be compared according to their attributes which can be: (1) linear or iterative; (2) sequential or incremental; (3) plan-driven vs. agile development; or (4) model-driven or evolutionary. However, many authors organize these methodologies in different ways, since it is possible to look at them from different points of view. In fact, three studies encompassing a large scope of ISDM’s were examined in this study for reviewing classification schemes. All three studies which are done by Deek *et al.* (2005), Pressman (2005) and Kurbel (2008) have very different classification schemes.

Although there are some similarities among all the categorization approaches, it is not easy and straightforward to place all the methodologies into certain categories. In this research Pressman’s (2005) scheme was selected, because his classification involved all the aspects (attributes) Kurbel (2008) specifies. In addition to Pressman’s (2005) classification, socio-technic development methodologies are also added as another category.

Prescriptive Software Engineering Models

Prescriptive models prescribe a set of process elements, workflows, software engineering actions, work products and quality assurance (Pressman, 2005). They are the plan-driven models, and they require the developers to follow a strict development plan, which are high contrast to agile methods.

Waterfall Model

Royce (1970) proposed a linear model, which is also known as Classic Life Cycle Model (or) Linear Sequential Model (or) Waterfall Method (See Figure 2-9), a model which he pointed out that was flawed. He explored then how this linear model could be improved into an iterative model.

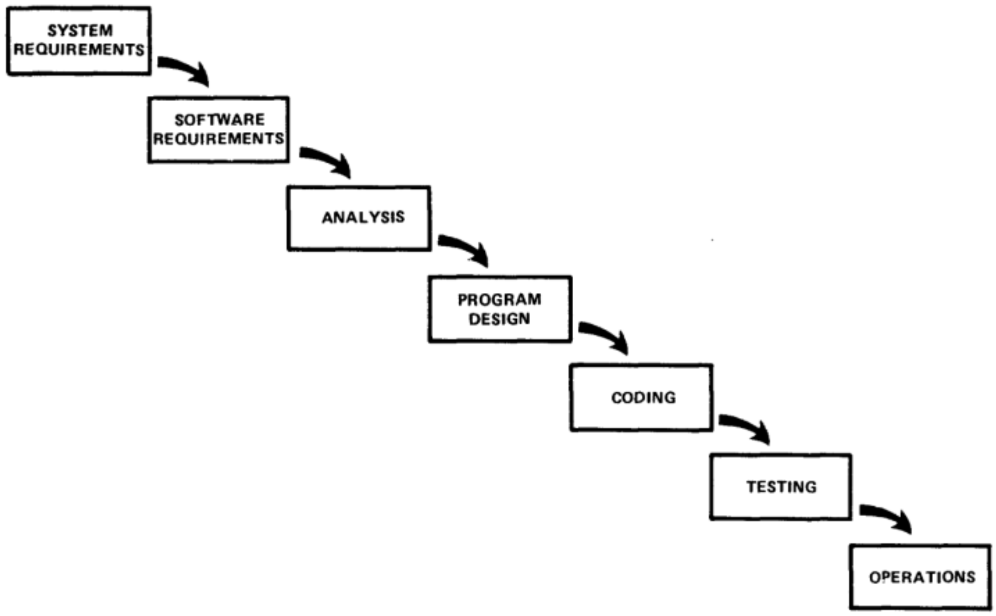

FIGURE 2-9 WATERFALL DEVELOPMENT MODEL. SOURCE: ROYCE (1970)

The Waterfall model is an old model with apparent problems causing its supporters to abandon it (Hanna, 1995). Pressman (2005) lists three important problems associated with linear development:

- Real projects do not follow a sequential process. Due to its nature, changes can confuse team members.
- Getting all the exact and necessary requirements from costumers at once is very difficult. Waterfall processes cannot accommodate uncertainty to development either.
- Working version of the project will not be available until the late stages of development. This may cause problems with impatient customers.

Incremental Process Models

Incremental process models consist of deliveries that are completed in increments.

Incremental Model

The incremental model uses the waterfall process for each increment, the first increment being the core product (See Figure 2-10). Similar to evolutionary approaches it is an iterative approach. The difference is the incremental model focuses on delivering working versions with each increment, with less complete versions in early increments which is similar to agile development methods (Pressman, 2005).

Incremental development is particularly useful if the staffing is limited, and staffing support for the complete development seems unlikely. This development allows for

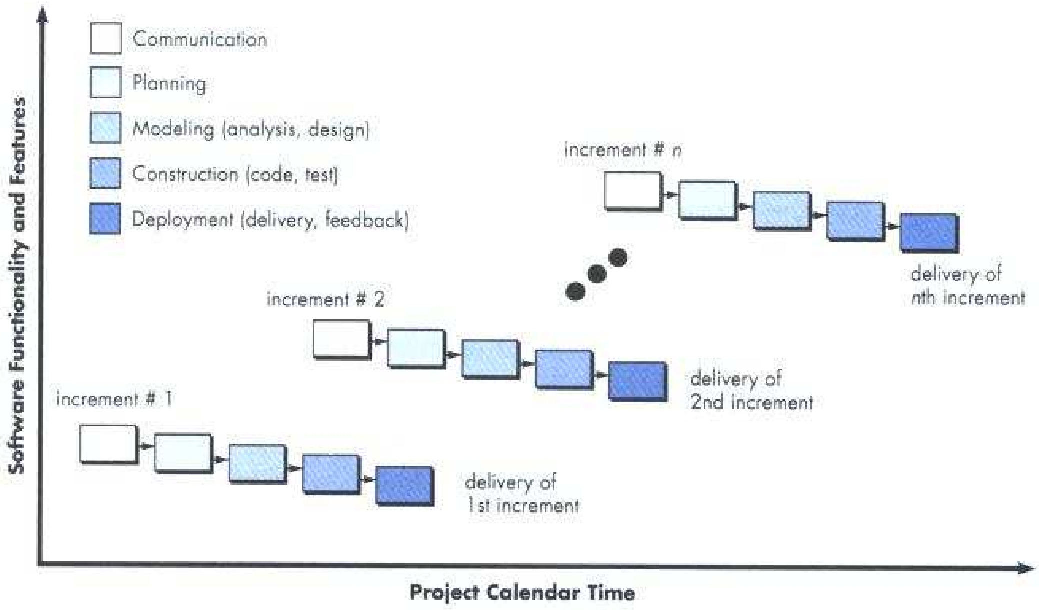

FIGURE 2-10 INCREMENTAL MODEL. SOURCE: PRESSMAN (2005)

Rapid Application development (RAD)

RAD focuses on short development cycles, adapting high-speed waterfall models as increments. If a project can be modularized so that each major function can be finished in less than three months, then RAD is a good fit (Pressman, 2005). Its process model has been illustrated in Figure 2-11.

A common problem for RAD development is that developers use the RAD tools in a selective way, which is usually the speedy one, rather than the ones ensuring quality and maintainability of the systems. This makes sense with the philosophy of RAD, as the developers are under the managerial directives to deploy the project rapidly (Deek *et al.*, 2005).

Pressman (2005) cites from Butler (1994) on the problems associated with RAD. For large projects RAD required sufficient staffing so that the project can be assigned to sufficient number of RAD teams. One problem relates to required level of discipline of RAD methodology. Accordingly, if either developers or customers are not committed to the Rapid activities, then the project is likely to fail. Another problem relates to design of the system to be developed. If the system cannot be modularized into manageable pieces, RAD is not a suitable choice. Additionally, if performance relies on tuning the interfaces to system components, then RAD may be a good choice. Lastly, if development depends much on the new technologies, RAD may not be a good alternative either.

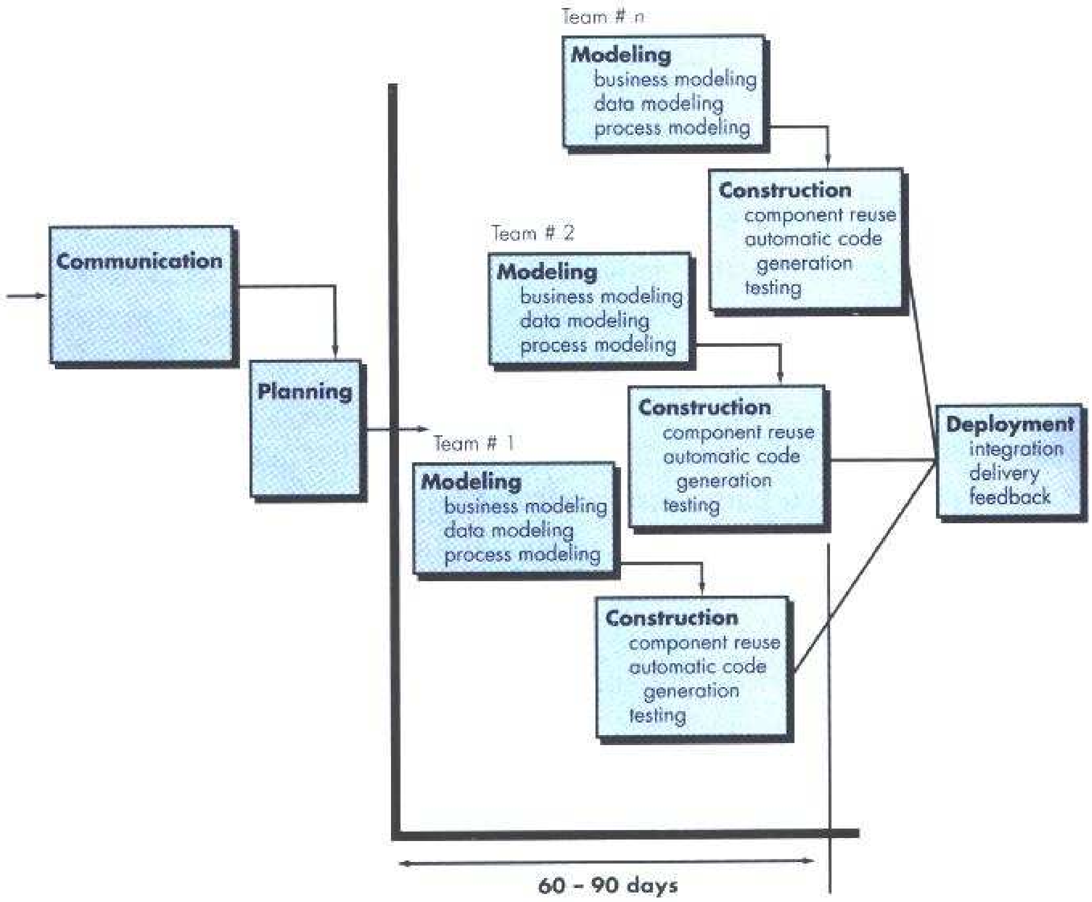

FIGURE 2-11 RAD MODEL. SOURCE: PRESSMAN (2005)

Evolutionary Process Models

Business and product requirements change often during the software development process, weakening the validity of the initial design and plan. However while delivery of comprehensive products in short time is not possible, businesses require at least limited versions in the competitive market conditions. Under such circumstances, evolutionary development strategies which are iterative can accommodate products that are likely to evolve over time (Pressman, 2005).

Evolutionary Prototyping

Prototyping is a software process that is composed of a series of prototypes. In throwaway prototyping, all the earlier prototypes are discarded and are used for showcase purposes. In evolutionary prototyping, the earlier prototypes become the core of the implementation. Pressman (2005) states that iterations in prototyping (iteration) first start by communicating with customers to collect the requirements. This is followed by quick planning and modeling that is focused on representation of aspects that are visible to the customer, such as the graphical user interface. After the construction, the prototype is deployed and then evaluated by the user. This evaluation serves as requirements analysis to let better systems specification in later iterations. Evolutionary prototyping has been illustrated in Figure 2-12.

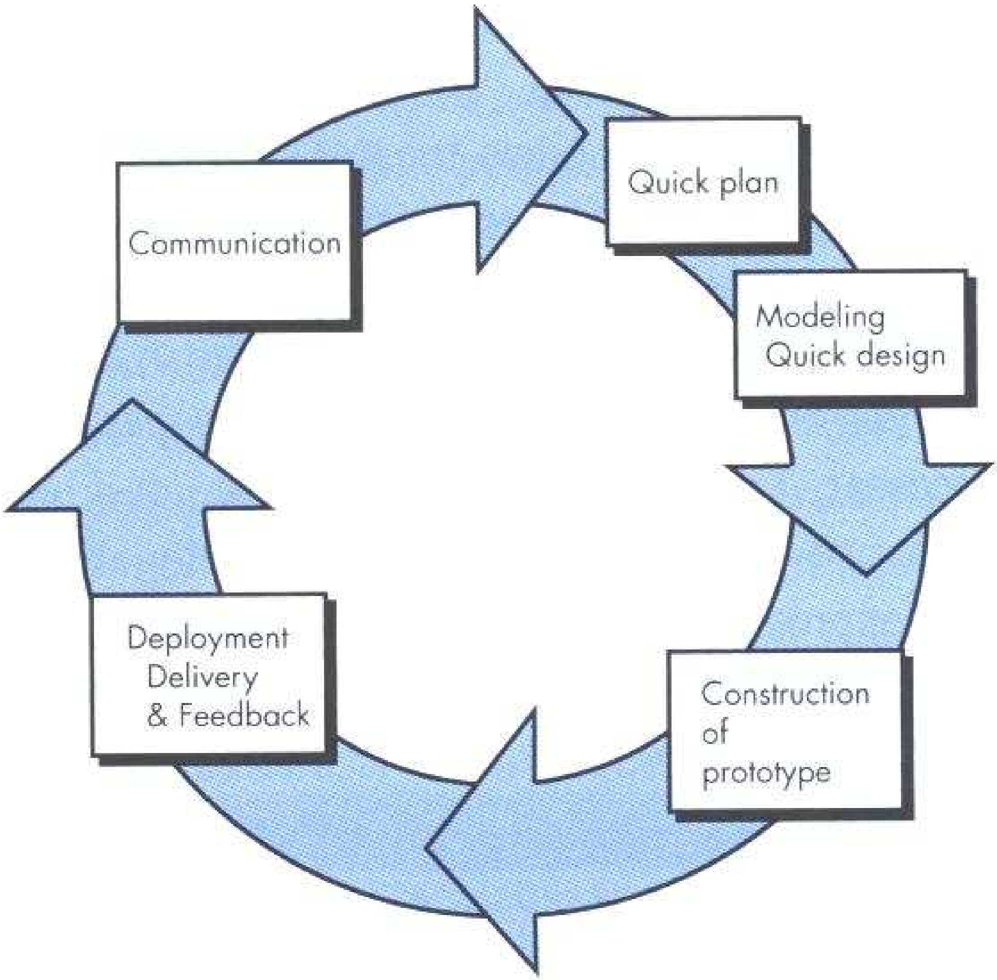

FIGURE 2-12 EVOLUTIONARY PROTOTYPING. SOURCE: PRESSMAN (2005)

Pressman (2005) indicates two potential problems with evolutionary prototyping. First problem is that customers may see a working version of the system, while being unaware of potential problems with overall software quality. This may result in customers rushing and thinking that the prototype can be finalized with a few fixes while this is not possible. Secondly, developers may make implementation compromises to have a working version as soon as possible. Such compromises may include inefficient algorithms to demonstrate functionality, operating systems and programming languages that are known by the developer, but inappropriate for the actual implementation. Such compromises may be forgotten by the developer in time, and ultimately become the integral part of the final system.

The Spiral Model

The spiral model is originally formulated by Boehm (1988) as shown in Figure 2-13, combining the iterative evolutionary prototyping with the systematic features of linear waterfall processes (Pressman, 2005). Boehm (2001) describes the model as follows:

> "The spiral development model is a *risk*-driven *process model* generator that is used to guide multi-stakeholder concurrent engineering of software-intensive systems. It has two main distinguishing features. One is a *cyclic* approach for incrementally growing a system's degree of definition and implementation while decreasing its degree of risk. The other is a set of *anchor point milestones* for ensuring stakeholder commitment to feasible and mutually satisfactory system solutions."

According to Pressman (2005), spiral models can be broken into frameworks that are defined by the development team. In one scenario, the first spiral may result in product specifications, while the others may involve execution in evolutionary style. With the spiral approach, it is easier to react to the changing requirements for developer and customer, while keeping a systematic stepwise approach.

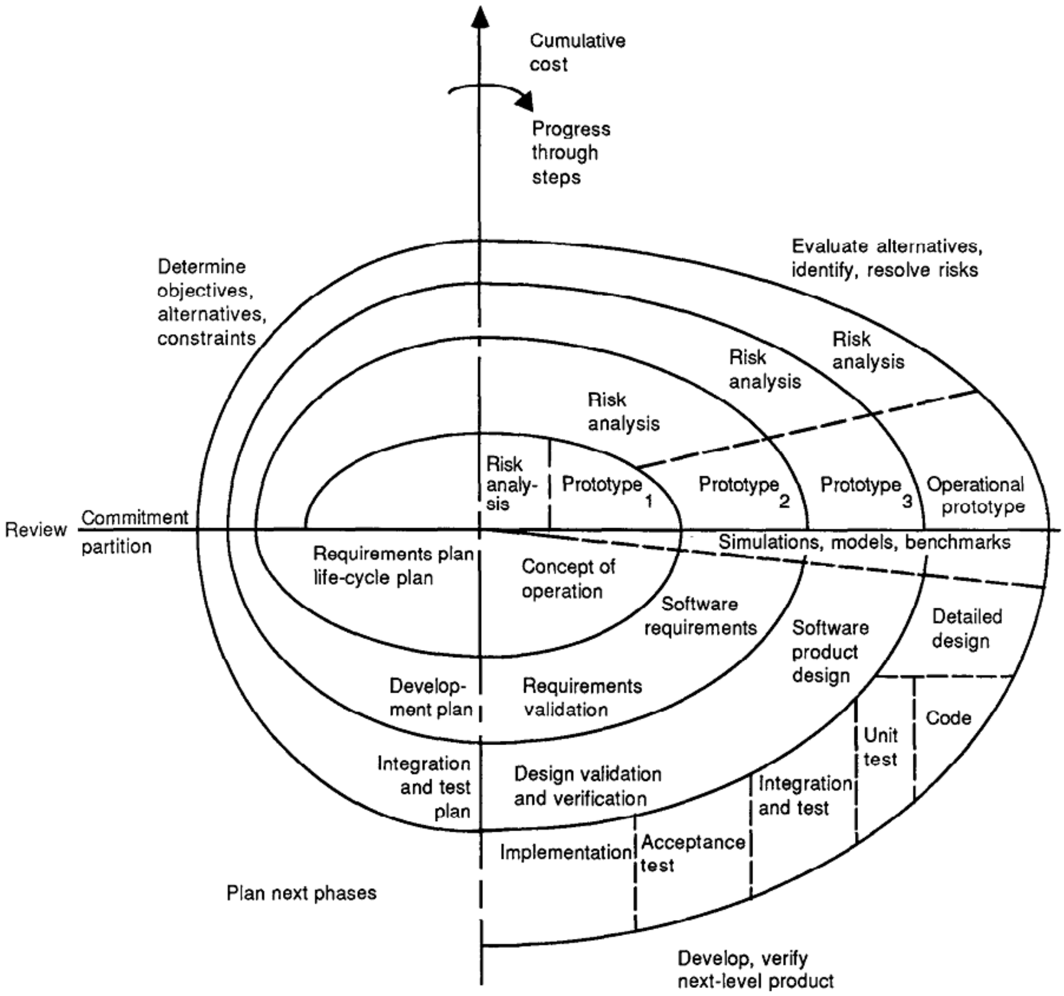

FIGURE 2-13 SPIRAL MODEL OF SOFTWARE PROCESS. SOURCE BOEHM (1988)

Nogueira *et al.* (2000) draw attention to three main problems associated with evolutionary processes. First problem emerges due to uncertain number of cycles required while usually in project management activities are estimated in a linear manner. Secondly, they claim that evolutionary development methodologies do not set the optimum evolution speed. Too fast evolution could cause chaos, while too slow evolution would obviously impact efficiency. Thirdly, Nogueira *et al.* criticize evolutionary methodologies' focus on flexibility. This property can be perceived as if software quality standards may be compromised to meet deadlines.

2.5.3. Agile Development

Agile development refers to software development methodologies that are based on continuous user input, responsiveness to change in an iterative manner. It originated from statement of software development principles by Kent Beck and 16 other authors. The "Manifesto for Agile Software Development" (Beck et. al., 2001) is as follows:

> We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value: Individuals and interactions over processes and tools; Working software over comprehensive documentation; Customer collaboration over contract negotiation; Responding to change over following a plan. That is, while there is value in the items on the right, we value the items on the left more.

Pressman (2005) state that agile principles have been known for a long time, however prioritization of these ideas happened more recently. These principles bring about important benefits over conventional software engineering methods; however they may not be applicable in all situations and projects.

There are a number of process models that are built on agile principles including Extreme Programming. In the following section, two well-known models of agile methodology are described briefly.

Extreme Programming (XP)

Particular methods associated with XP were first published by Beck (1999). Pressman (2005) states that XP development is composed of four frameworks; planning, design, coding and testing (See Figure 2-14).

1. Planning starts with a set of stories describing functionality of the software written by the customer on index cards. The customer also assigns values depending on the priority based on its business value. Developers then assess the stories (functionalities and features) and assign them costs based on the time required. If a story requires more than three weeks then the customer is asked to split the story into smaller pieces. Customers and developers together decide which stories will be published in the next releases.
2. Design is strongly focused on the simplicity principle. XP advocates use of class-responsibility collaborator (CRC) cards to design software in object oriented framework. In case the design problem is difficult, then XP suggests something called “spike solution”, the development of an operational prototype immediately.
3. XP suggests unit testing before the coding so that testing can be done immediately as coding is finished. A key concept in XP programming is pair programming, which is a recommended process and involves two programmers working on the same machine to create a code of story.
4. Availability of unit tests allows a regression testing strategy whenever code is modified. With XP, customers specify acceptance tests.

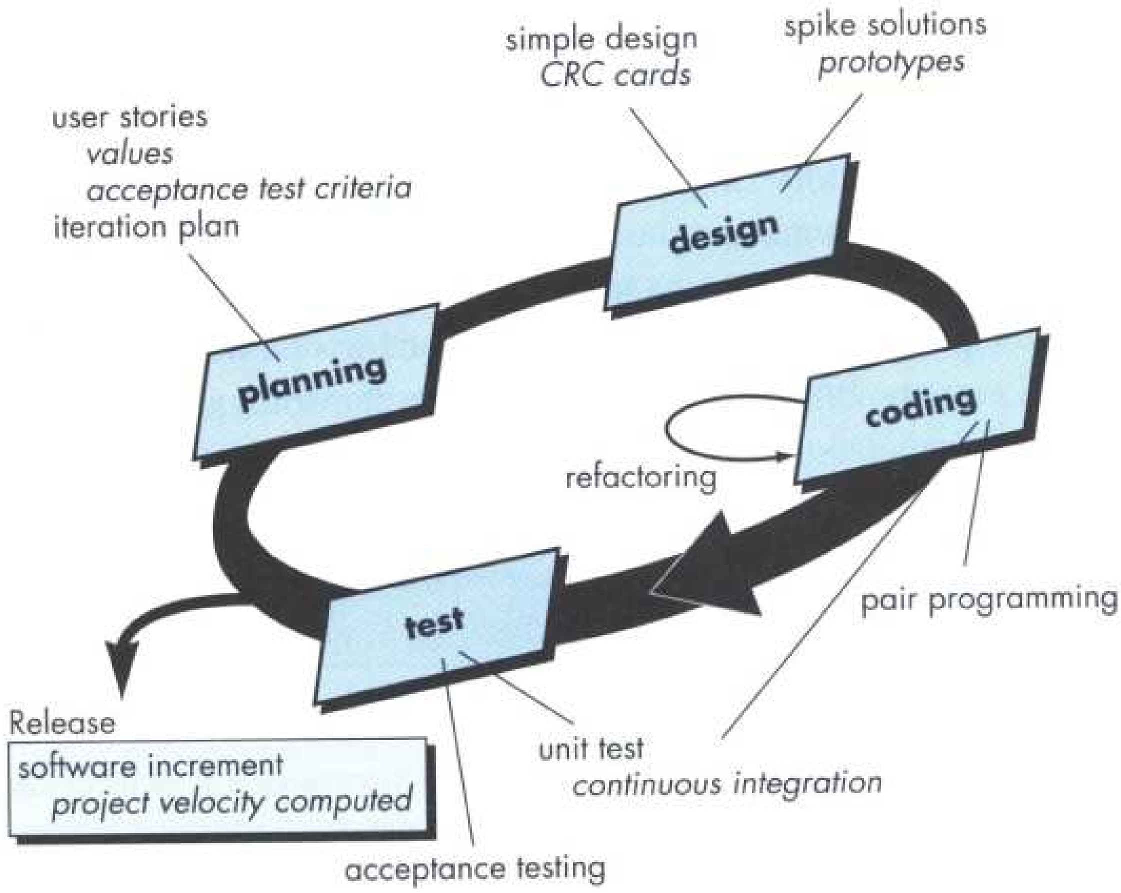

FIGURE 2-14 XP PROCESS. SOURCE: PRESSMAN (2005)

Scrum

Pressman (2005) highlights the software process patterns that should work for tight project schedules, changing requirements and priorities. Each process pattern has four activities. The first activity is backlogging, which involves getting a prioritized list of features from the customer that can be changed at any time. Secondly, there are sprints which are composed of work units that should implement the backlogs in a given time frame which is usually a month. During a sprint, requirements cannot be changed by the customer. This results in a work structure that is flexible yet maintaining certain stability. Third activity is the meetings. Scrum masters lead scrum meetings that are typically 15 minutes long assessing the performance and feedback from each individual. Fourth activity is demos which are delivered as increments to customers to be evaluated at the end of the iteration.

Evaluation of Agile Methods

While agile development received many positive responses in the industry, criticisms do exist. One prominent criticism came from Stephens (2003), pointing out some issues including the lack of structure and necessary documentation. It relies very much on senior-level developers which can be inconvenient in some organizations. While introduced as a strong point, Stephens criticizes that XP will work with insufficient design since it may lead to low quality in the final product. Additionally he argues that XP requires a lot of cultural changes within the organization. Lastly, he points out that it is difficult to negotiate contracts, because in the beginning no one knows the scope of the project and the list of requirements.

Wailgum (2007) argues that some enterprise architects could object to agile methods because there is not much room for a complete architectural design, and the projects often result in spaghetti code (meaning that the source code of the project has a excessive complex and tangled control structure). Just as the coding process, the development teams are very flexible too. A project leader’s role can change too much during the project. At a certain point a leader may be giving orders, at another point he/she may be limited to a facilitator position. Additionally, executive level management may feel out of touch with the development process as development teams can be self managing for the sake of flexibility.

2.5.4. Socio-technic Methodologies

Socio-technic methodologies including Effective Technical and Human Implementation of Computer-based Systems (ETHICS) and Multiview emerged in 1983 and 1985 respectively, long before emergence of agile methodologies. These are the first software development methodologies that put considerable emphasis on user involvement. Agile methodologies are similar to socio-technic methodologies in that respect. The main difference of agile methodologies however is the focus on flexibility and adaptability, which were not explicit in ETHICS or Multiview.

Effective Technical and Human Implementation of Computer-based Systems (ETHICS)

Reeve and Petch (1999) state that ETHICS arose in 1983 from the idea that participation of the users can help avoiding human issues that arise in complex software development projects. The whole idea is to involve the users as much as possible, so that the technologies will be as much fit as possible to the organizations, which results in more effectiveness. The methods focus on the changes that are introduced by new technologies. ETHICS also enable users to define job satisfaction objectives in addition to technical objectives. Job satisfaction is very important in this methodology, so that employees will think (a) their skills are appropriately used, and further improved; (b) their aspirations are recognized and met; (c) they achieve rewards and control; and (d) new technologies help them achieve their goals.

Multiview

Multiview is a methodology that was first proposed in 1985, which tries to combine techno-centric and socio-centric views for software development processes. It is literally a “multiview” approach, to look at the Information Systems from multiple points of view. This is especially a good strategy when dealing with multiple disciplines, and when there are different types of system usage by the technical and non-technical users (Reeve and Petch, 1999).

Bell (1996) describes the five stages involved in the Multiview method. The first phase involves analysis of human activity system. This analysis requires three steps to identify (a) rich picture: structures and processes in the organization; (b) root definition: actors and their links to structures; (c) conceptual model: the simplification and abstraction of the tasks in the new information system. Second phase is called informational modeling to visually model entities, functions and events within the information system. Third stage is the identification of social and technical requirements. The human-computer interface is designed at the fourth stage. The focus of this design is on user interaction via screen dialogues. The last stage is the design of the technical aspects. These aspects are application (what functionalities will it have), information retrieval, design of database, maintenance of database, recovery methods for application and monitoring of application. In Figure 2-15, this methodology is presented visually.

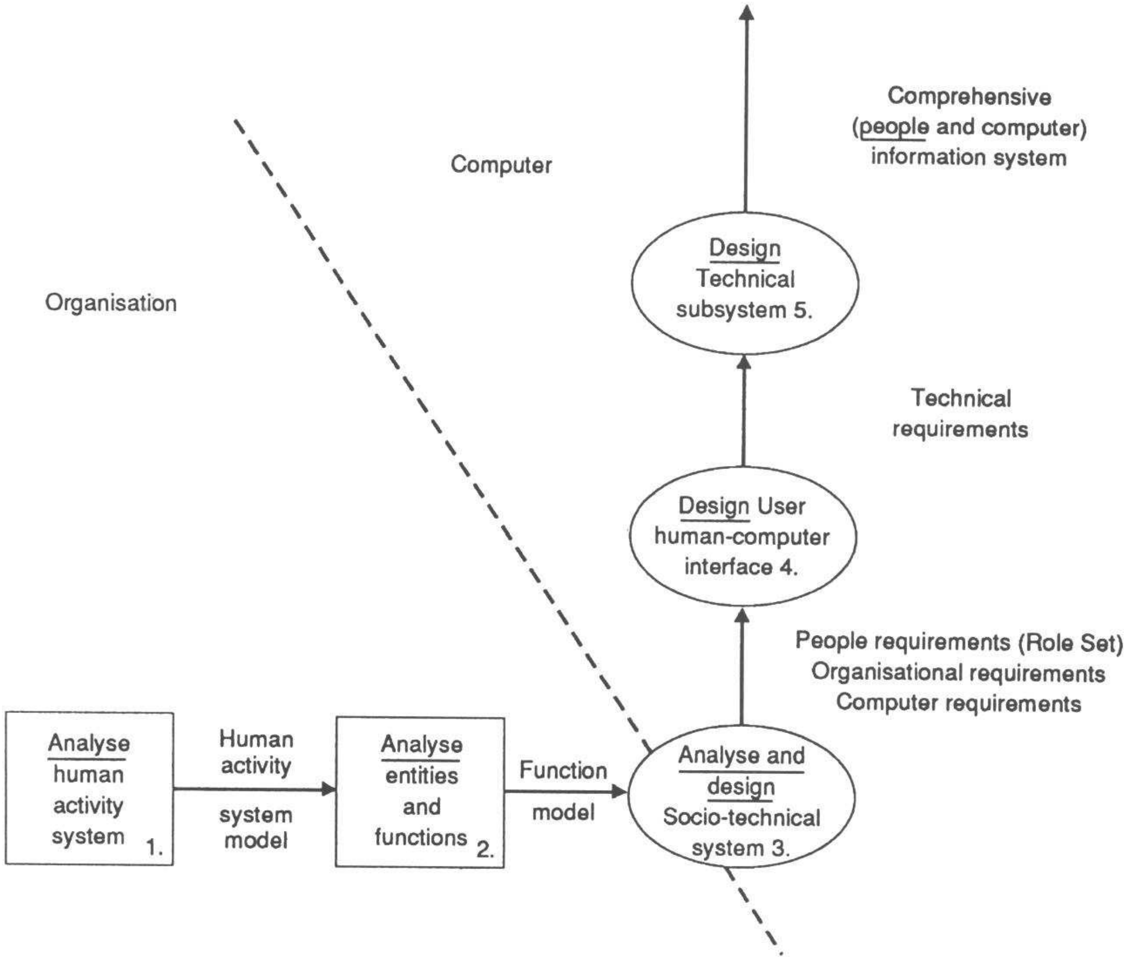

FIGURE 2-15 STAGES OF MULTIVIEW METHODOLOGY. SOURCE: REEVE AND PETCH (1999)

2.5.5. Software Engineering Techniques and Tools

Requirements Engineering

Requirements engineering is integral to all modern information systems development methodologies. This step is necessary to understand what the user wants, see if it is feasible and to specify them in a clear manner so that the implementation can take place. Various methodologies such as waterfall or Extreme Programming may suggest differences in collection and organization of requirements while there is a substantial amount of commonalities in requirements engineering. Purpose of this section is to discuss important aspects of requirements engineering.

Aurum and Wohlin (2005) describe 'Requirements Engineering' as a collection of life-cycle activities that relate to requirements; and its main tasks include gathering, documenting and managing requirements.

Requirements engineering, and its sub tasks are crucial to successful software and system development. This is a necessary process to involve stakeholders in the development process, as well as to establish a good communication between the customers and developers.

IEEE (Institute of Electrical and Electronics Engineers) 610.12-1990 standard 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 or system component to satisfy a contract, standard, specification, or other formally imposed documents; (3) A documented representation of a condition or capability as in (1) or (2).

There are several ways to classify requirements. As shown in Table 2-1, a requirement can be classified as either functional or non-functional, classified according to its level (goal, domain, product or design levels), classified as either primary or derived or classified according to another scheme.

TABLE 2-1 TYPES OF REQUIREMENTS. SOURCE: AURUM AND WOHLIN (2005)

Requirements Classification
• *Functional requirements* - what the system will do • *Non-functional requirements* - constraints on the types of solutions that will meet the functional requirements e.g. accuracy, performance, security and modifiability
• *Goal level requirements* - related to business goals • *Domain level requirements* - related to problem area • *Product level requirements* - related to product • *Design level requirements* - what to build
• *Primary requirements* - elicited from stakeholders • *Derived requirements* - derived from primary requirements
Other classifications, e.g. • *Business requirements* versus *technical requirements* • *Product requirements* versus *process requirements* - i.e. business needs versus how people will interact with the system • *Role based requirements*, e.g. customer requirements, user requirements, IT requirements, system requirements, and security requirements

Pressman (2005) presents a 7 step requirements engineering approach. The first step is inception: Usually a new project begins when a business need is identified or a potential service is discovered. At this stage, software engineers ask some context free questions to understand the nature of the problem and solution, as well as to establish a communication with clients. The second step is elicitation. This stage involves asking users what the objectives and functionality of the system will be. The third step is elaboration. This stage involves modeling and refinement of the information acquired in the inception and elicitation phases. Usually techniques like Use-Case modeling and UML diagramming are used to define the scenarios, actors, processes and the relationships. The fourth step is negotiation. This step is necessary because usually customers may ask for more than what can feasibly be achieved, or such demands can be conflicting with other's demands. Under these circumstances the software engineer should negotiate with the users on what requirements are really important and feasible. The fifth step is specification. After requirements are finalized, requirements should be specified on written documents in an unambiguous and consistent way, sometimes accompanied with graphical models, mathematical models and scenarios. The sixth step is validation. This stage is necessary to make sure all the requirements are unambiguous, free of error, consistent and conformant to the standards. The last stage is requirements management. Purpose of requirements management is identifying and keeping track of changes to requirements as the project progresses.

Aurum and Wohlin (2005) on the other hand identify five main requirements engineering activities. The first activity is requirements elicitation, specification and modeling. This activity is necessary to understand the needs of stakeholders, eliciting their requirements, collecting, modeling and storing them. The second activity is requirements prioritization. This activity is necessary to decide which requirements are more important, solving conflicts between customers and developers with regards to priorities, and planning for deliveries. Third activity is requirements dependencies and impact analysis. According to Dahlstedt and Persson (2005), a study has shown that only about one fifth of all requirements are not related to any other requirements. Requirements dependencies is very important, as a dependency can: (1) Constrain how another requirement can be designed and implemented; (2) Affect the cost of implementation of another requirement;

or (3) Increase or decrease customer satisfaction regarding another requirement. Similarly, a change in one requirement may result in changes in other requirements, necessitating further changes in the original requirement. Requirements traceability is a basis for understanding requirements interdependencies and avoiding associated and potential problems. Dahlstedt and Persson (2005) define it as the "ability to describe and follow the life of a requirement, in both forward and backward direction, ideally through the whole system life cycle". In addition to identification of requirement dependencies, impact analysis is an important part of requirements engineering, since any change to a requirement results in significant changes in the final software product (Jönsson and Lindvall, 2005). Arnold and Bohner (1996) define it as "the activity of identifying the potential consequences, including side effects and ripple effects, of a change, or estimating what needs to be modified to accomplish a change before it has been made". The fourth activity in Aurum and Wohlin's requirements engineering process is requirements negotiation. This process is necessary to resolve conflicts inherent in requirements between customers and developers. The last activity is quality assurance. Quality assurance is required to ensure specified requirements meet the necessary quality criteria, to help the success of the final product. Denger and Olsson (2005), state that traditionally, quality assurance (QA) applied to post software development. A typical example is testing outputs from the implemented software. However, QA is also very important in the requirements engineering process. Without QA at the requirements phase, it is likely that incorrect requirements may be embedded into the system design, resulting in an implementation that is not the desired outcome. It is logical to think that as the development proceeds, the cost of a defect will increase as it does not get noticed. Therefore it is important to assure quality standards at the beginning, in the requirements phase. Denger and Olsson (2005) give a list of quality attributes for requirements, including correctness, unambiguity, completeness, consistency, rank of importance, verifiability, modifiability, traceability, comprehensibility, feasibility and right level of detail.

Requirements elicitation is present in both Pressman (2005) and Aurum and Wohlin's (2005) requirements engineering activities and can be discussed further. Zowghi and Coulin (2005) further divide the requirements elicitation activities into five. The first one is understanding the application domain. This sub activity is necessary as it is important to understand and examine the real world situation in which the system will be operating. This situation can be examined with respect to its political, organizational and social aspects. Second sub activity is identification of the sources of requirements. Requirements may exist in a variety of sources. While stakeholders are one obvious source, existing systems, processes and documentation can be useful in extracting important requirements. Third one is analyzing stakeholders. Stakeholders, who are people that have an interest in the system, or are somehow affected by the development, should be analyzed, identified and involved/consulted for requirements elicitation. Fourth one is selecting the techniques, approaches and tools to use for requirements elicitation. A range of such techniques are explained in the next paragraph. There is no one fits for all type of requirements elicitation technique, as the choice of technique depends on the context and nature of the project. Last requirements elicitation activity is the actual elicitation of the requirements from stakeholders and other sources. After the requirements sources and stakeholders are identified, the requirements elicitation at a detailed level can begin using selected techniques described below.

Zowghi and Coulin (2005) list a comprehensive list of techniques that can be used for eliciting the requirements from stakeholders and other sources. These are (1) Interviews: Interviews are one of the most commonly used methods for collecting data on requirements. The quality and the amount of the data largely depend on the skill level of the interviewer and the communication with the interviewees; (2) Questionnaires: Questionnaires are usually used for the early stages of requirements elicitation, and may involve open or closed type of questions; (3) Task Analysis: Task analysis is a top-down approach where high level tasks are broken down into lower level tasks, eventually resulting description of detailed processes of all actions; (4) Domain Analysis: This analysis is carried out by examination of existing documentation regarding the scope of the project; (5) Introspection: This technique requires the requirements analyst to determine the requirements based on his / her understanding of what the stakeholders want and need from the system. Obviously, this technique should be used with care, and preferably only employed at the initial elicitation stage, and if the analyst has a good understanding and expertise in similar domains; (6) Repertory Grids: This method involves building a matrix by categorizing the elements, detailing their instances, and assigning variables to each of the instances; (7) Card Sorting: This technique requires users to organize and sort cards which include domain related concepts. They are also asked to explain the rationale for their particular sort. This helps the analyst to understand stakeholders' understanding of the domain and requirements; (8) Laddering: This activity involves asking stakeholders asking a series of questions and requiring them to organize their answers into categories, such as hierarchical trees; (9) Group Work: Group work is a very common; often default method for requirements elicitation; (10) Brainstorming: This activity involves extracting as many ideas as possible from stakeholders in a relatively shorter amount of time; (11) Joint Application Development (JAD): This activity is similar to brainstorming in that, all the stakeholders are included to investigate through requirements and potential solutions. Main difference is that, with JAD the main goals should already have been defined, and all the sessions are well structured with predefined steps and roles of participants; (12) Ethnography: Being the study of people in their natural settings, this method involves the analyst actively or passively participating in activities that are carried out by stakeholders; (13) Observation: This is a particular technique of ethnography, and involves analyst passively observing the users of a system; (14) Protocol Analysis: This is a method involves stakeholders speaking about the processes aloud, and describing specific actions and explaining the rationale behind them; (15) Apprenticing: This is an activity of the analyst learning about the domain of the system by under instruction and supervision of an experienced stakeholder; (16) Prototyping: Prototyping involves providing stakeholders prototypes of the system to be planned to be developed in order to understand the potential requirements and solutions; (17) Goal Based Approaches: This method involves breaking down the high level goals objectives into lower level objectives, and elaborating on them in such a way that individual requirements are extracted (18) Scenarios: This technique is widely used for requirements elicitation, and similar to use cases, it describes current and future processing actions of the system including the users and other stakeholders; (19)

Viewpoints: This is a method for attempting to model the domain of the system from multiple point of views in order to have a clear and comprehensive understanding of the system. For example, a system can be modeled from an operational, implementation or interface point of views.

Prioritization of requirements is also an important activity in requirements engineering. Berander and Andrews (2005) explain the aspects of prioritization that can be used as criteria to evaluate requirements. One of the aspects is importance. Stakeholders should classify the requirements by their importance, and the most important requirements should be given the most attention and priority. Another aspect is penalty. It can be useful to think of a potential penalty if a requirement is not realized. Cost is also a useful aspect in prioritization of a requirement. Cost of a requirement can be measured in money and effort, which can further be analyzed by its complexity or ability to reuse existing code. One other aspect of prioritization of requirement is risk. For each requirement, performance risks, process risks and schedule risks can be included, which can affect the overall risk of the project.

Requirements negotiation is present in both Pressman (2005) and Aurum and Wohlin's (2005) requirements engineering activities. Grünbacher and Seyff (2005) identify three steps in negotiation for requirements. First step is pre-negotiation that involves definition of the negotiation problem, identification of stakeholders, and analysis of goals of stakeholders to find conflicts. Second step is the negotiation. Actual negotiation occurs in this phase, as stakeholders look for solutions that are acceptable to all parties. This usually involves exchanging offers and proposing alternatives for mutual gain. Third and last step of requirements negotiation is post-negotiation. After finding a solution, stakeholders may analyze the negotiation and its outcomes, and may suggest re-negotiation if necessary.

2.5.6. Capability Maturity Model Integration (CMMI)

CMMI is an approach developed by Software Engineering Institute (SEI) to provide organizations elements for effective processes, which include development, acquisition and maintenance of products or services. It is used to integrate systems engineering, software engineering and process development developments in a single framework. CMMI can be used for (1) Product and service development, (2) Service establishment, management, and delivery or (3) Product and service acquisition (CMMI Product Team, 2006). Literature survey in this study focuses on CMMI for product and service development. There are several important concepts in CMMI, including representations, maturity and capability levels and process categories and process areas.

Representation Types

There are two representations of CMMI, continuous and staged. Either of the representations type needs to be chosen for a project. Continuous representation was chosen for the implementation part of this study (See Chapter 3 for details). With the continuous representation which is more flexible, it is possible to select certain process areas of CMMI. It is also possible to improve different process areas at different intensity. The staged representation is more systematic and structured alternative. With this approach, every process needs to be addressed at the same rate. Instead of specifying the process areas and their particular capability levels to be developed as in continuous representation, a maturity level is defined for the entire project.

Capability and Maturity Levels

The other important concept is levels, that is, capability and maturity levels. The term “capability level” is used in the context of continuous representation of improvement, which allows the organization to choose specific improvement areas and improve them incrementally. Levels are used to show how ideal a certain process is, or the organization as a whole. There are six capability levels starting with number 0: (0) Incomplete: This level indicates that a process is not performed, or only partially performed; (1) Performed: This level indicates that a process is performed, meaning that it satisfies necessary goals of the particular process area; (2) Managed: This level indicates that a process is managed, meaning that it was planned and implemented according to organizational policies that involve resource allocation, stakeholder involvement, monitoring, controlling, testing, and evaluating; (3) Defined: For a managed process the standards, process descriptions and procedures can be very different for each specific instance of a process across different projects. For a defined process, standards, process descriptions and procedures must conform to the organizations standards, and be more consistent; (4) Quantitatively Managed: A quantitatively managed process is managed and controlled using quantitative techniques, such as statistical ones; (5) Optimizing: An optimizing process conforms to all previous maturity requirements, and focuses on continually improving process performance (CMMI Product Team, 2006).

The term “maturity level” is used in the context of staged representation of improvement, which is concerned with the overall maturity of the organization and it allows organizations to improve processes in a set of processes areas. Maturity levels are very similar to capability levels, in that they reflect levels of planning and understanding of the processes. There are five maturity levels and are denoted by numbers ranging from 1 to 5: (1) Initial: At the initial level, processes are usually not planned and chaotic. Success in these processes depends on the individual skills or people working in the organization; (2) Managed, which is as the capability level 2; (3) Defined, which is same as the capability level 3; (4) Quantitatively managed, which is same as the capability level 4; (5) Optimizing, which is same as the capability level 5 (CMMI Product Team, 2006).

Process Categories and Process Areas

There are four process area categories, and 22 process areas at CMMI for product and service development (CMMI Product Team, 2006). If a continuous representation is selected, an organization has the freedom to select a desired number of process areas, and develop each at different capability levels. If a staged representation is selected, first a maturity level is chosen. Some process areas are only addressed at certain maturity levels. For example, organizational innovation and deployment process area can only be addressed at fifth level. Accordingly, when a maturity level is chosen, there is no selection of individual process areas. Rather, maturity level determines what process areas need to be developed. The process categories are as follows:

Process Management: This category involves five process areas that are oriented towards “defining, planning, deploying, implementing, monitoring, controlling, appraising, measuring, and improving processes” (CMMI Product Team, 2006, p. 52). This process category involves process areas of organizational process focus, organizational process definition, organizational training, organizational process performance and organizational innovation and deployment.

Project Management: This category involves process areas activities related to "planning, monitoring and controlling the project" (CMMI Product Team, 2006, p. 55). This process category involves process areas of project planning, project monitoring and control, supplier agreement management, integrated project management, risk management and quantitative project management.

Engineering: This category involves process areas that are related to development and maintenance activities across engineering disciplines. This process category involves process areas of requirements development, requirements management, technical solution, product integration, verification and validation.

Support: This category involves process areas that are used to support product development and maintenance. This process category involves process areas of configuration management, process and product quality assurance, measurement and analysis, decision analysis and resolution and causal analysis and resolution.

Each of these process categories contain several process areas that correspond to different levels as indicated in Table 2-2.

TABLE 2-2. PROCESS AREAS AND THEIR ASSOCIATED CATEGORIES AND MATURITY LEVELS. (CMMI PRODUCT TEAM, 2006, P. 44) (IPPD STANDS FOR INTEGRATED PRODUCT AND PROCESS DEVELOPMENT).

Process Area	**Category**	**Maturity Level**
Causal Analysis and Resolution	Support	5
Configuration Management	Support	2
Decision Analysis and Resolution	Support	3
Integrated Project Management +IPPD	Project Management	3
Measurement and Analysis	Support	2
Organizational Innovation and Deployment	Process Management	5
Organizational Process Definition +IPPD	Process Management	3
Organizational Process Focus	Process Management	3
Organizational Process Performance	Process Management	4
Organizational Training	Process Management	3
Product Integration	Engineering	3
Project Monitoring and Control	Project Management	2
Project Planning	Project Management	2
Process and Product Quality Assurance	Support	2
Quantitative Project Management	Project Management	4
Requirements Development	Engineering	3
Requirements Management	Engineering	2
Risk Management	Project Management	3
Supplier Agreement Management	Project Management	2
Technical Solution	Engineering	3
Validation	Engineering	3
Verification	Engineering	3

Goals and Practices

In CMMI terminology, a goal may involve several practices that need to be implemented. There are generic goals and practices, and specific goals and practices. Same generic goals and generic practices apply to all process areas. Application of generic goals and specific goals into process areas is mandatory in CMMI implementation. Generic goals and practices exist at corresponding capability or maturity levels. For example, "Institutionalize a Managed Process" is a generic goal at the maturity or capability level 2. "Plan the Process" is a generic practice among ten generic practices within that generic goal. If a capability or maturity level of 2 is targeted for example, all the generic goals and generic practices at level 1 and level two need to be implemented.

In addition, there are specific goals and specific practices that are particular to each process area. Specific goals and practices exist at different levels corresponding to capability and maturity levels. Specific goals and specific practices are required to be implemented. For example, "Develop the Design" is a specific goal for the process area "Technical Solution" at the capability or maturity level 2. "Design the Product or Product Component" is a specific practice among four specific practices within that specific goal. If a capability or maturity level of 2 is targeted, all the specific goals and specific practices at level 1 and level two that are particular to selected process areas need to be implemented.

2.5.7. CMMI and Agile Development Integration

Agile methods have gained in popularity due to their apparent advantages and success stories that have been told since their introduction. These advantages are flexibility, adaptability and user satisfaction. Old CMM (Capability Maturity Model) and its replacement CMMI have always been given much importance, especially due to the fact that it is prestigious to achieve CMMI compliance (evaluated by Standard CMMI Appraisal Method for Process Improvement lead appraisers that are authorized by Software Engineering Institute. Software Engineering Institute is a development and research center based at Carnegie-Melon University). From a development standpoint; CMMI and agile methods are two seemingly competing schools of thought. The CMMI approach privileges consistency across projects by standardizing processes, thus benefitting customers. The agile school of thought focuses upon retaining agility within and across projects by letting creative user problem solving take place within a flexible methodology. While these strategies appear contradictory in terms of approach to organizations and users, there are many points where they complement and supplement each other and can be combined to take advantage of the best elements of both approaches. As Turner (2002) says, "agile methods are very much how to do rather than the CMM's what to do" (p. 137). Thus, there are fundamental points of intersection that can be combined. Many studies were conducted and research articles were published that view such integration is possible, as two methods can be seen complementary to each other in many aspects.

Paulk (2001) asserts that when rationally implemented, agile methods (such as XP) can address many of the requirements in CMMI levels 2 and, and agile methods can be adopted into specific areas depending on the business environment. XP satisfaction for CMMI process areas has been shown in Figure 2-16. This integration can be considered especially in small to medium projects. He also advocates modifications on agile methods when necessary, and points out developers and managers should decide on setting the balance between documentation, planning and flexibility. He further points out that while agile methods are preferable in many contexts, in life critical systems it may be inappropriate due to extremely low tolerance to errors.

XP satisfaction of key process areas, given the appropriate environment

Level	Key process area	Satisfaction
2	Requirements management	++
2	Software project planning	++
2	Software project tracking and oversight	++
2	Software subcontract management	—
2	Software quality assurance	+
2	Software configuration management	+
3	Organization process focus	+
3	Organization process definition	+
3	Training program	—
3	Integrated software management	—
3	Software product engineering	++
3	Intergroup coordination	++
3	Peer reviews	++
4	Quantitative process management	—
4	Software quality management	—
5	Defect prevention	+
5	Technology change management	—
5	Process change management	—

+ *Partially addressed in XP*
++ *Largely addressed in XP (perhaps by inference)*
— *Not addressed in XP*

FIGURE 2-16 XP SATISFACTION OF KEY PROCESS AREAS, GIVEN THE APPROPRIATE ENVIRONMENT. SOURCE: PAULK (2001)

However, Martinsson (2003) maintains that XP is a foundation to build matured software and to improve processes as defined by CMM. Although XP is found unsatisfactory for CMM level 2 in Martinsson's study, it satisfies many key process areas at different levels, and therefore XP and CMM are complementary in general. As shown in Figure 2-17, he also maps XP practices to CMM goals, and mentions how well the satisfaction is based on his independent findings and his literature survey.

Others have advocated XP and SMM integration. Reifer (2003) is a supporter of the idea, but acknowledges potential problems facing integration efforts. While there can be specific solutions to these problems, he draws attention to map XP practices to software CMM.

Kähkönen and Abrahamsson (2004) give an example from a case study called "The eXpert Project", which achieved CMMI maturity level 2, through using enhanced XP, that is XP with some additional practices and documentation. Some of these additions included: (a) A team other than the programming team worked on planning the project at the beginning; (b) CM (Configuration Management) plan and CM audit procedures were written at the end of each iteration; (c) The planning team prepared an implementation plan. The project manager also maintained a document called Task Book that included the release plan. This plan contained planned and actual effort spent for each task; (d) Documents containing system architecture, database diagrams and information, and user interface descriptions were written during the last iteration. Additionally based on the system testing a system test report was created.

▲: XP core practices and roles fully satisfy the KPA goal.
►: XP core practices and roles partially or implicitly satisfy the KPA goal.
▼: XP core practices and roles do not satisfy the KPA goal.

Level	Key Process Area (KPA)	Goals 1 – 4			
2	Requirements management (RM)	▲	▲		
	Software project planning (SPP)	▲	▲	▲	
	Software project tracking and oversight (SPTO)	▲	►	▲	
	Software subcontract management (SSM)	Omitted			
	Software quality assurance (SQA)	▲	▼	▲	▼
	Software configuration management (SCM)	▲	►	►	►
3	Organization process focus (OPF)	►	▼	▼	
	Organization process definition (OPD)	▲	▲		
	Training program (TP)	►	►	►	
	Integrated software management (ISM)	▼	▼		
	Software product engineering (SPE)	▲	▲		
	Intergroup coordination (IC)	▲	▲	▲	
	Peer reviews (PR)	▲	▲		
4	Quantitative process management (QPM)	▲	►	▼	
	Software quality management (SQM)	▼	▼	▼	
5	Defect prevention (DP)	▲	▲	►	
	Technology change management (TCM)	▼	▼	▼	
	Process change management (PCM)	▼	▼	▼	

FIGURE 2-17 DEGREE OF SATISFACTION FOR THE 52 CMM GOALS BY IMPLEMENTING AN XP PROCESS. SOURCE: MARTINSSON (2003)

Turner and Jain (2002) carried out a research on over 40 participants at Center of Software Engineering, University of Southern California. Participants were asked to characterize the relationship of agile methods to CMMI process areas and generic practices (Characterizing the relationships as one of three categories; C for conflicts; N for neutral and S for supports). The results are seen in Table 2-3.

TABLE 2-3 AGILE METHOD VS. CMMI PROCESS AREA CONFLICT FINDINGS. MODIFIED FROM: TURNER AND JAIN (2002)

Process Area	**Survey Finding**
Organizational Process Focus	C
Organizational Process Definition	C-N
Organizational Training	N-S
Organizational Process Performance	C
Organizational Innovation and Deployment	C-S
Project Planning	S
Project Monitoring and Control	S
Supplier Agreement Management	N
Integrated Project Management	S
Risk Management	N
Integrated Teaming	S
Quantitative Project Management	C
Requirements Management	N
Requirements Development	S
Technical Solution	S
Product Integration	S
Verification	S
Validation	S
Configuration Management	None
Process and Product Quality Assurance	C-N
Measurement and Analysis	C-N
Decision Analysis and Resolution	C
Organizational Environment for Integration	S
Causal Analysis and Resolution	N

TABLE 2-4 AGILE METHOD VS. CMMI GENERIC PRACTICE CONFLICT FINDINGS. MODIFIED FROM: TURNER AND JAIN (2002)

CMMI Generic Practices	**Survey Finding**
2.1 Establish an Organizational Policy	N-S
2.2 Plan the Process	N-S
2.3 Provide Resources	N-S
2.4 Assign Responsibility	S
2.5 Train People	N
2.6 Manage Configurations	C-S
2.7 Identify and Involve Relevant Stakeholders	S
2.8 Monitor and Control the Process	N
2.9 Objectively Evaluate Adherence	C
2.10 Review Status with Higher Level Management	N-S
3.1 Establish a Defined Process	C
3.2 Collect Improvement Information	C
4.1 Establish Quantitative Objectives for the Process	N
4.2 Stabilize Subprocess Performance	C-N
5.1 Ensure Continuous Process Improvement	C-N
5.2 Correct Root Causes of Problems	N

The results from Table 2-3 and Table 2-4 can be summarized as follows: (a) 7 components are seen as clearly in conflict; (b) 10 components are seen as possibly in conflict; (c) 11 components are seen as clearly supportive; (d) 11 components are seen as no worse than neutral; (e) 1 component had no consensus finding.

Turner (2002) claims that agile processes can fit into process improvement practices very well. However, in order to achieve this, CMMI should be interpreted in a more essential and less literal manner, so that the freedom to exercise adaptability and flexibility of the working environment will not be threatened. Marçal *et al.* (2008) focuses on application of Scrum practices for the staged CMMI development model, particularly process areas of project management. They map specific practices and Scrum practices with each other, and determine what percentage of the practices was satisfied, partially satisfied or unsatisfied. Considering each process area according to its maturity level, process areas at maturity level 2 have 43.8% of its specific practices satisfied by Scrum, and 21.9% are partially satisfied. If the supplier agreement management process is not applicable, satisfaction coverage increases to 58.3%, and partial satisfaction increases to 29.2%. Similarly, Santos (2007) mentions Scrum practices helped his company meet the requirements of CMMI maturity level 2.

Bozheva and Gallo (2005) argue that adoption of agile processes for CMMI framework needs to be done gradually. This should be done in three steps. First, processes for which agility need to be increased need to be identified. Secondly, processes need to be built from scratch by applying necessary agile patterns to achieve process areas. The third step is introducing patterns and the processes that are based on them to developers. This should be done gradually while keeping customers involved in processes. Additionally, automation should be employed as much as possible.

Pikkarainen and Mäntyniemi (2006) map agile practices to CMMI goals and conclude that agile methods can be used for process improvement. They find problems are likely to arise when documentation is a priority. This is due to the agile value of having “working software over comprehensive documentation”.

Anderson (2005) from Microsoft Corporation explains their efforts on stretching Microsoft Solutions Framework (MSF) Agile Software Development to fit CMMI level 3, under the name MSF for CMMI. Since CMMI requires artifacts that are not directly produced by agile methods, it was necessary to enhance MSF for Agile Software Development with additional activities to produce these artifacts. As a result of enhancement, the footprint of the guidance material for MSF for CMMI Process Improvement became 150% larger than that of MSF for Agile Software Development. Similarly, number of product artifacts increased to 59 for MSF for CMMI method, contrasting with 25 product artifacts produced by MSF for Agile Software Development.

Boehm (2002) maintains that a combined approach of agile methods and extensive planning, codified processes and predictable techniques is feasible and preferable. He draws attention to the fact that compared to undisciplined hacking, agile methods require considerable amount of documentation. This view is visualized in Figure 2-18 that shows two ends of software development methodologies, hacking which is a completely unorganized process, and ironbound contracts which contain no room for flexibility.

Boehm (2002) also suggests that there should be a certain balance between agility and traditional approaches. He also recommends this balance should be adjusted depending on the situation, as shown in Figure 2-19, Figure 2-20 and Figure 2-21.

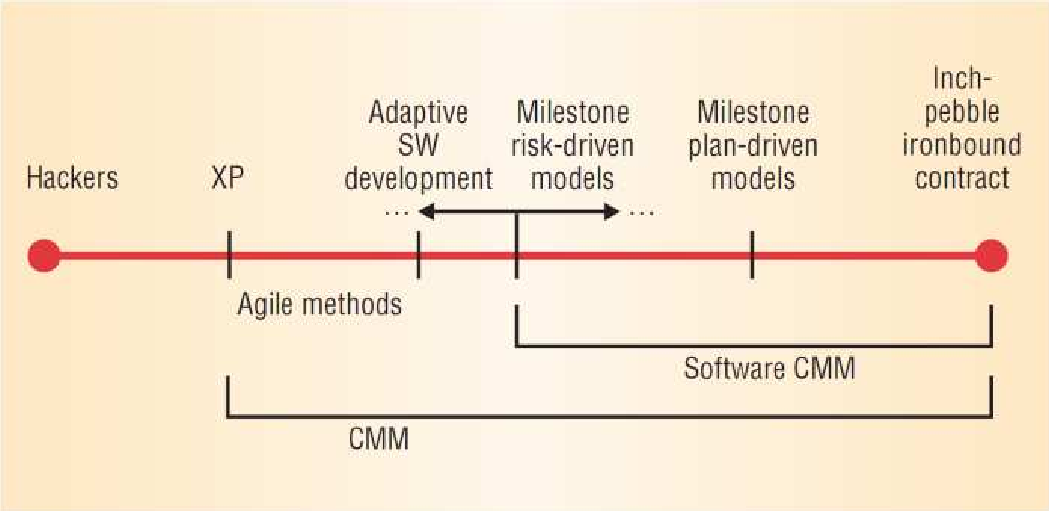

FIGURE 2-18 THE PLANNING SPECTRUM: UNPLANNED AND UNDISCIPLINED HACKING OCCUPIES THE EXTREME LEFT, WHILE MICROMANAGED MILESTONE PLANNING, ALSO KNOWN AS INCH PEBBLE PLANNING, OCCUPIES THE EXTREME RIGHT. SOURCE: BOEHM (2002)

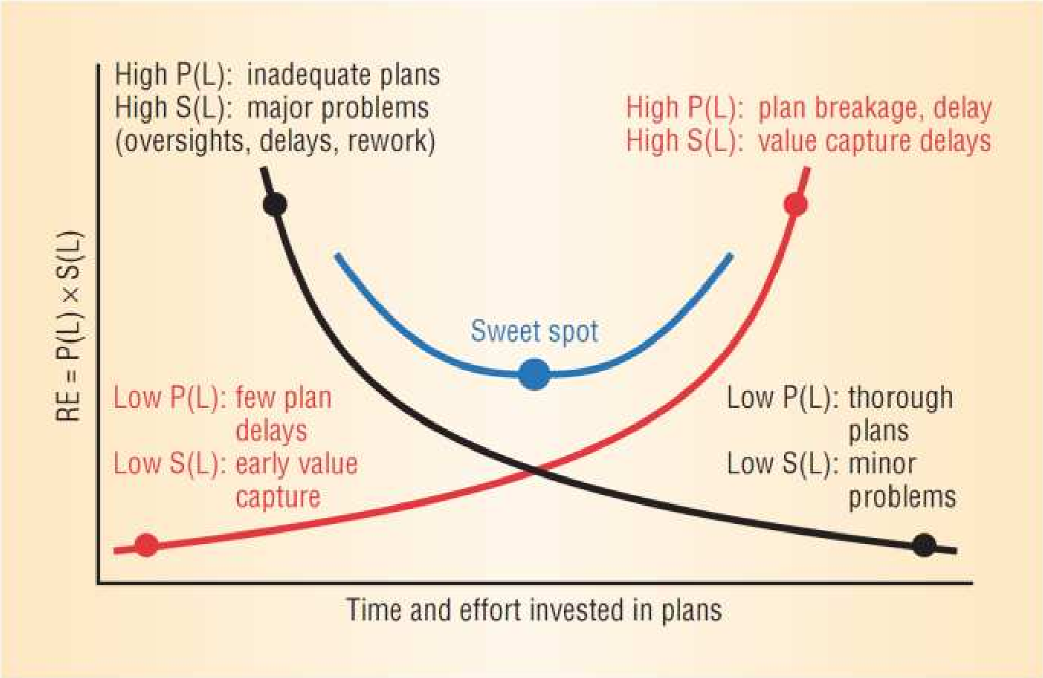

FIGURE 2-19 RISK EXPOSURE (RE) PROFILE. THIS PLANNING DETAIL FOR A SAMPLE E-SERVICES COMPANY SHOWS THE PROBABILITY OF LOSS P(L) AND SIZE OF LOSS S(L) FOR SEVERAL SIGNIFICANT FACTORS. SOURCE: BOEHM (2002)

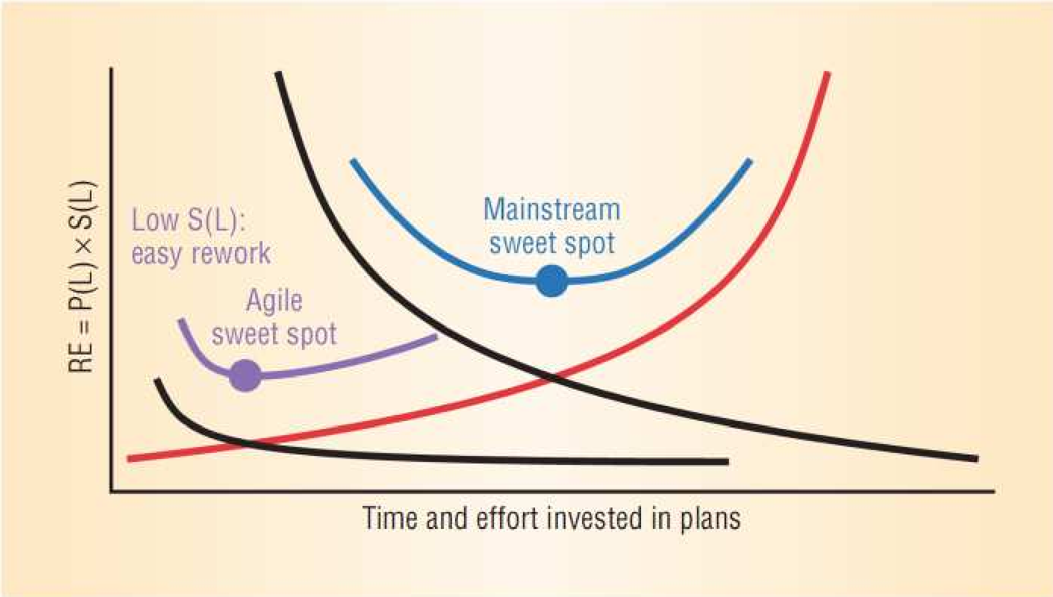

FIGURE 2-20 COMPARATIVE RISK EXPOSURE PROFILE FOR AN AGILE HOME-GROUND COMPANY WITH A SMALL INSTALLED BASE AND LESS NEED FOR HIGH ASSURANCE. SOURCE: BOEHM (2002)

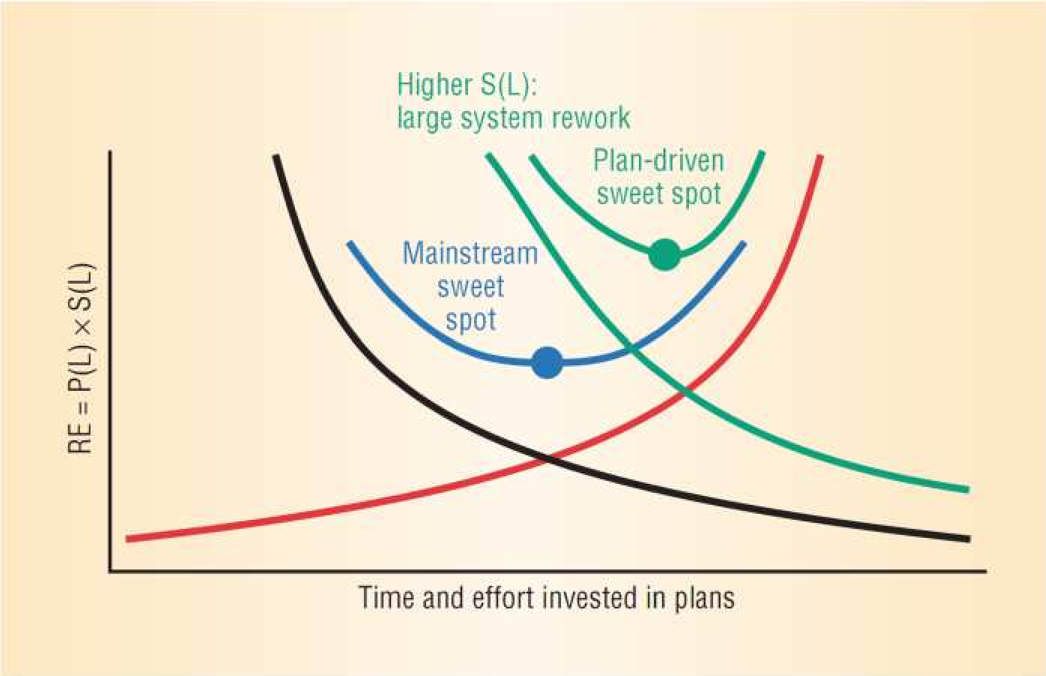

FIGURE 2-21 COMPARATIVE RISK EXPOSURE PROFILE FOR A PLAN-DRIVEN HOME-GROUND COMPANY THAT PRODUCES LARGE, SAFETY-CRITICAL SYSTEMS. SOURCE: BOEHM (2002)

2.5.8. Use of Methodologies across Applications of Interest

GIS Development

Reeve and Petch (1999) point out that there have not been much done for the adaptation of socio technical methodologies in large GIS development projects. They observed from journals such as *Mapping Awareness* and some guides prepared by Local Government Training Board (1992) and Royal Town Planning Institute (1992) on how various authorities investigated methodologies and how they implemented their GIS projects. Reeve and Petch (1999) concluded that such implementations usually follow a classical waterfall process as they abstract it in Figure 2-22.

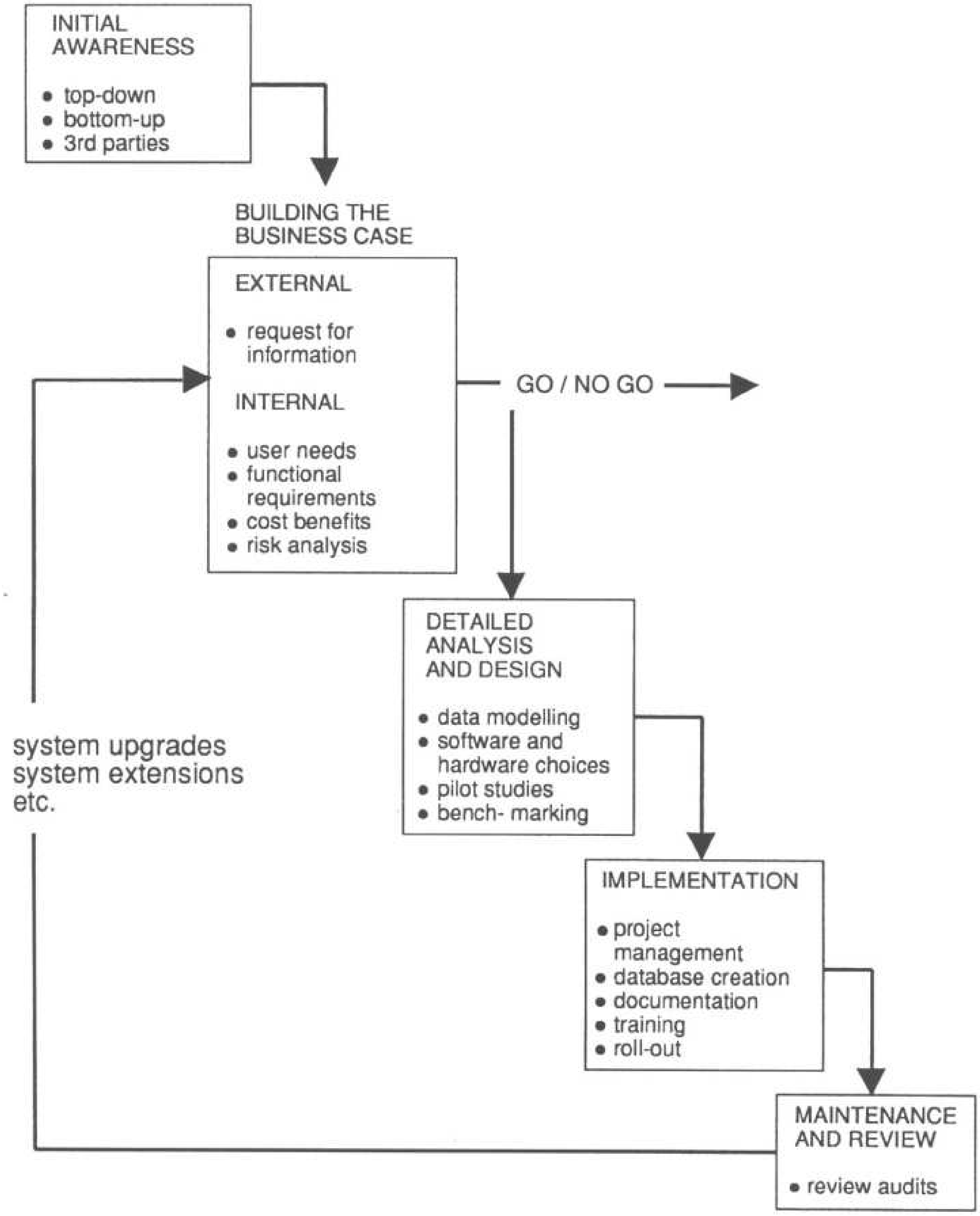

FIGURE 2-22 A GENERIC GIS DEVELOPMENT METHODOLOGY. SOURCE: REEVE AND PETCH (1999)

Pick (2008) states that, the use of software development methodologies and tools have been limited in the GIS industry. He gives three possible reasons for this: (a) GIS has been developed mostly in public sector, which is not rapid in terms of methodology adaptation. One potential reason for this is the stricter contract requirements in public sector, therefore classic development methodologies such as waterfall models may be more suited for public sector; (b) GIS teams are usually composed of non-technical people in the organization; (c) fewer formal methodologies that are designed for large projects have been implemented for GIS, since GIS has usually not been viewed as a profitable service.

Pick (2008) introduce the classical waterfall process as a typical GIS development methodology. "Phases in Systems Development for GIS" include: (1) planning that involves identification of the problem, solution, and also feasibility, budget, staffing and scheduling; (2) analysis that involves information collection and requirements elicitation; (3) Design that involves system architecture design, data, functionality and process modeling; (4) Implementation that involves actual building of the system by programming necessary components and putting the data into databases; (5) Maintenance that involves keeping the system running, enhancing functionality, eliminating the problems and providing training and support for users.

Situation Awareness Oriented Development

While it would not be unusual to argue that many methodologies, especially user centric methodologies are suitable to develop situation awareness oriented information systems, Endsley (1988) argues that situation awareness needs separate attention in information systems development:

> "Situation awareness forms the critical input to, but is separate from, pilot decision making, which is the basis for all the subsequent pilot actions. Even the best trained and most experienced pilots can make the wrong decisions if they have incomplete or inaccurate SA. Conversely, a pilot may accurately understand what is occurring in the environment, yet not know the correct action to take or be unable to carry out that action. For this reason, it is important that SA be considered in the design process separately from decision making and performance." (p. 97)

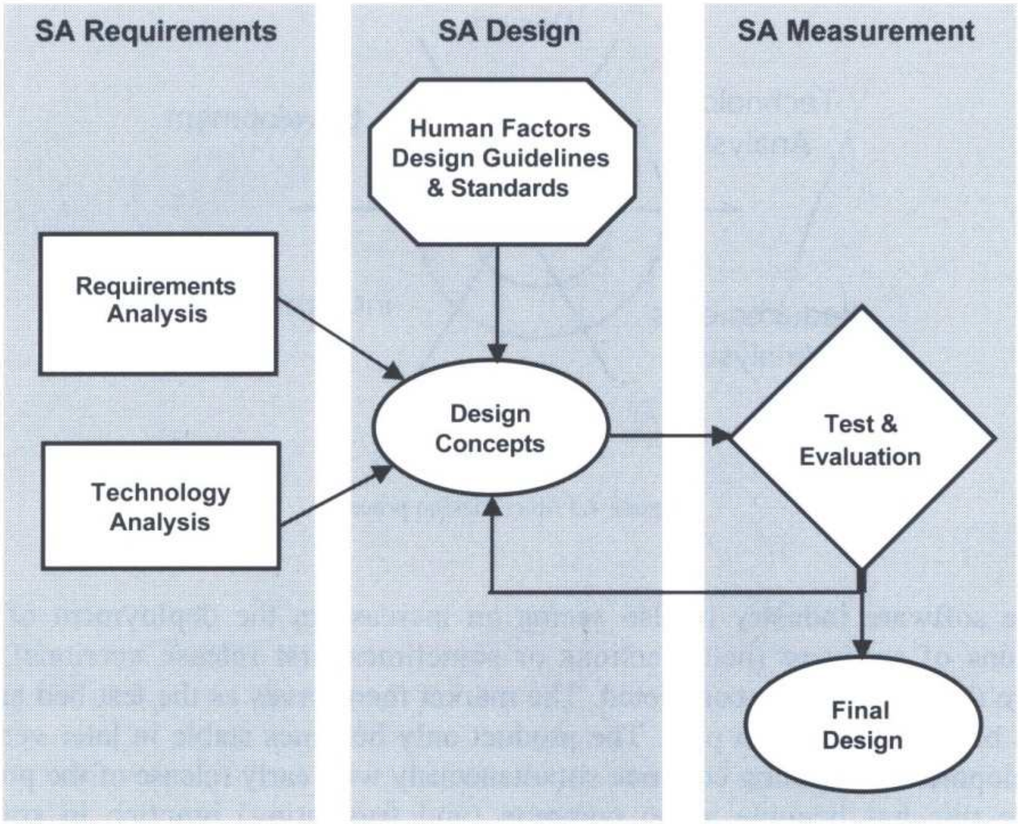

FIGURE 2-23 USER INTERFACE DESIGN PROCESS. SOURCE: ENDSLEY ET AL. (2003)

Endsley *et al.* (2003) mention waterfall, concurrent engineering (in which developers can work in multiple phases e.g. requirements analysis, design and implementation at the same time) and spiral development. They however, focus on user interface design process rather than the complete systems development life cycle. This process as visualized in Figure 2-23 includes (1) requirements analysis that may take into account environmental conditions, user characteristics and operational requirements; (2) technology analysis that is carried out simultaneously with requirements analysis that involves surveying a range of technological products (both hardware and software) available in the market; (3) Design conceptualization that involves analyzing the functions of the system, interface design

2.5.9. Comparison of Methodologies

This chapter reviewed some of the most prominent software development methodologies that were developed throughout the history of software engineering. While techno-centric methodologies were much common in the past, the socio-centric ideas were also introduced into development of methodologies. Reeve and Petch (1999) claim that addressing one problem leads to another in these methodological issues. They point out that while ETHICS and Multiview bring forward some good ideas on how to include the users to design, there have not been formal methodologies presented to developers so that they can follow an “A to Z” pattern. Similarly, adaptive and evolutionary methodologies cannot be executed strictly based on a rulebook, since they heavily depend on key players’ decisions.

Criteria for Methodologies Evaluation

Bjorn-Andersen (1984) as cited by Avison and Fitsgerald (2008), provides a set of questions to evaluate information systems development methodologies. Answering them may help decide selecting an appropriate methodology for an intended development project. The questions include: (1) What are the research paradigms forming the foundation of methodology? (2) What are the underlying value systems? (3) What is the context where a methodology is useful? (5) To what extent is modification enhanced or even possible? (6) Does communication and documentation operate in the users’ dialect, either expert or not? (7) Does transferability exist? (8) Is the societal environment dealt with, including the possible conflicts? (9) Is user participation ‘really’ encouraged or supported?

Avison and Fitsgerald (2008) also give a comprehensive list of criteria that can be used for evaluating methodologies: (1) Rules: An ideal methodology should provide rules and formal guidelines to proceed with phases and techniques, using tools, producing documentation, and estimating requirements; (2) Coverage: Different methodologies may cover varying spans of development, however ideally a methodology should cover from strategic planning to maintenance; (3) Understanding the information resource: Methodology should be helpful to capture and utilize the information resource, such as the available data; (4) Documentation standards: An ideal methodology should provide standards for documentation that is understandable for customers and developers, and should help facilitate communication between them; (5) Separation of designs: Logical design (what an application does, what data and processes it uses etc.) should be separate from the physical design; (6) Validity of design: Methodology should provide techniques for checking the completeness, consistency and accuracy of the design; (7) Early change: Changes that emerge as necessary should be identifiable during the development; (8) Inter-stage communication: Entire contents of a whole stage should ne communicable to all other stages; (9) Effective problem analysis: Methodology should provide techniques for capturing the problems and objectives; (10) Planning and control: Development process should controlled and planned while being fir into a time frame; (11) Performance evaluation: Methodology should have techniques for evaluating the performance of the developed products; (12) Increased productivity: Proper methodology should help increase the productivity; (13) Improved quality: An ideal methodology should improve the quality of the whole development process (e.g. analysis, design, implementation and evaluation) as well as the end product; (14) Visibility of product: The visibility of the product should be maintained during the whole development process; (15) Teachable: Not only developers, but also the customers or users should be able to able to learn the methodology; (16) Information systems boundary: Methodology should help define the extent of the information systems as well as the organizational boundaries; (17) Designing for change: It should be relatively easy to modify logical and physical designs; (18) Effective communication: Methodology should provide means for developers and users to communicate; (19) Simplicity: Methodology should be easy to learn and use; (20) Ongoing relevance: Methodology should be adoptable as new techniques and tools are developed, while maintaining an overall consistency and philosophy; (21) Automated development aids: Methodology should benefit from productivity tools whenever possible; (22) Consideration of user goals and objectives: Methodology should assist meeting user goals and objectives by integrating them into the system development; (23) Participation: Methodology should encourage participation of different parties and provide means for effective communication; (24) Relevance to practitioner: Methodology should be appropriate for its users (practitioners) in terms of technical knowledge and social skills; (25) Relevance to application: Methodology should be appropriate for the type of system developed (e.g. web based, decision support, distributed, service oriented etc.); (26) The integration of the technical and non-technical systems: methodology should provide means to integrate technical and non-technical aspects of the system developed; (27) Scan for opportunity: Methodology should encourage looking for better and new problem solving strategies; (28) Separation of analysis and design: Methodology should encourage this separation so that user requirements are not influenced by design considerations.

In addition to above mentioned criteria, several other criteria can be useful. One of them is methodology's adoptability for improving existing systems that are developed with other methodologies or with no apparent methodology. Many organizations may suffer from poor standardization of activities and documentation. In that case, methodology should assist in inter and intra organizational standardization processes

Comparison Frameworks

In addition to evaluation criteria, comparison frameworks can be useful in comparing various methodologies side by side. Such a comparison structure would reveal a big picture that is helpful in deciding on what methodology is useful under a particular context. Jayaratna (1994) proposes a comparison framework called NIMSAD (Normative Information Model-based Systems Analysis and Design) for methodologies. This framework consists of three elements: (1) Problem situation (methodology context);

(2) Intended problem solver (methodology user); (3) Problem solving process (methodology).

Avison and Taylor (1996) identify five distinct problem situations and assign them appropriate methodologies: (1) Well structured problem situations with a well defined problem and clear requirements. Methodologies based on the traditional systems development life cycle are the most appropriate in these situations; (2) Well structured problem situations with clear objectives but uncertain user requirements. Methodologies based on data modeling, process modeling or prototyping are the most appropriate in these situations; (3) Unstructured problem situations with unclear objectives. Soft systems approaches are the most appropriate in these situations; (4) Situations with high user interaction. Socio-technical approaches are the most appropriate in these situations; (5) Situations with high levels of uncertainty. Contingency approaches, such as Multiview, are the most appropriate in these situations.

Avison and Fitsgerald (2008) also lay a framework for comparing methodologies. This framework is not used for normative purposes (e.g. which methodology is the most appropriate under certain conditions), but is only used to classify methodologies according to various aspects. The first element in this framework is philosophy. This is the element that defines the set of principles that underlie the methodology. It has four sub-elements including (a) Paradigm: It refers to specific way of thinking. Avison and Fitsgerald use objectivist and subjectivist paradigms for a simplified classification; (b) Objectives: Different methodologies may have different objectives. For example, while most of the methodologies' objective is to develop information systems, some of them are used to see if there is a need to develop information systems; (c) Domain: Methodologies may address different domains, such as narrow and isolated problems or interrelated and complex problems; (d) Target: Some methodologies may target specific environments and/or organizations, while others may be designed for general purposes.

Second element in Avison and Fitsgerald's methodology is the model. "The model is the basis of the methodology's view of the world, it is an abstraction and a representation of the important factors of the information system or organization". Models are means of communication among stakeholders, they can be translated into different forms and they can be used to provide insights into problem domain. All the information systems development methodologies are of iconic/schematic type in nature. Other types of models include verbal, analytic/mathematical and simulation. Techniques and tools constitute the third element. This element refers to the productivity, analysis and modeling tools. Fourth element is the scope referring to the extent of the life cycle that is covered by the methodology. Outputs constitute the fifth element. Methodologies produce certain deliverables at the end of each phase/cycle. This can involve requirements specification, conceptual design diagrams, working system etc. Practice is the sixth element: This element is measured according to methodology background (academic or commercial), user base (number and types of users) and participants of methodology (if professional analysts must be involved) and required skill sets. Last and seventh element in Avison and Fitsgerald's methodology is the product: This element refers to the final deliverable in the contract. It can be training, consultancy, documentation, software etc.

McConnell (1996) and Reeve and Petch (1999) provide frameworks with examples to compare software development methodologies. Table 2-5 was developed combining the two frameworks. As seen in this table, some methodologies are superior in some areas, while having poor performance in some other areas. For example evolutionary prototyping is very suitable with working with poorly understood requirements and allowing mid-course corrections. However, its performance is not as good as waterfall methodology when it comes to producing a reliable system. This result reinforces McConnell (1996)'s statement that there is no development methodology that can be used for all kinds of projects because the effectiveness of the model depends on the context it is used.

TABLE 2-5. COMPARISON OF DEVELOPMENT METHODOLOGIES BASED ON MCCONNELL (1996) AND REEVE AND PETCH (1999)

Question	Code and Fix	Waterfall	ETHICS	Multiview	Spiral Development	Evolutionary Prototyping	Commercial Off-the-Shelf
Works with poorly understood requirements	poor	poor	excellent	fair	excellent	excellent	excellent
Works with poorly understood architecture	poor	poor	excellent	fair	excellent	poor to fair	poor to excellent
Produces reliable system	poor	excellent	excellent	good	excellent	fair	poor to excellent
Manages risks	poor	poor	good	fair	good	fair	n/a
Allows mid-course corrections	poor to excellent	poor	fair	fair	excellent	excellent	poor
Provides visible progress	poor	fair	excellent	excellent	excellent	excellent	n/a
Requires little manager skill	excellent	fair	poor	poor	poor	poor	fair

Benediktsson *et al.* (2006) use metrics to compare software development methodologies in an actual experiment involving 15 software teams. They used VM Model (An extended version of Waterfall model), Evolutionary Model (EM), Incremental Model (IM) and Extreme Programming (XP). Several metrics including time effort, quality and intermediate design products (length of code, number of diagrams etc.) were collected. Quality of the products was assessed by focusing on five attributes of functionality, reliability, usability, efficiency and maintainability. Accordingly Table 2-6 and Table 2-7 show how much effort was spent on each activity for each development methodology. The modified waterfall model took the most time, a total of 748 hours, and exceeding all other methodologies in requirements specification, designing and coding. Extreme programming required the least amount of time for requirements specification and designing. Integration and testing took the longest for Extreme Programming, which coincides with the integration issues faced in this study as examined in Chapter 4, the implementation chapter. Repair activities also took the longest time for Extreme Programming, which was natural given less effort in requirements specification and design. In Table 2-8 quality parameters of functionality, usability and maintainability were examined. While there is not much difference in functionality, extended waterfall model had significantly higher maintainability compared to Extreme Programming. This is also understandable given limited planning for Extreme Programming.

TABLE 2-6. AVERAGE GROUP EFFORT BY ACTIVITY IN HOURS. (PM= PROJECT MONTH) SOURCE: BENEDIKTSSON ET AL. (2006)

Group	ReqsSpec	Design	Code	Integration and testing	Review	Repair	Other	Total hours	Total PM
VM	73.9	68.6	206.1	35.6	56.9	60.4	246.5	748.0	4.92
EM	67.8	58.0	169.1	57.8	23.0	48.0	125.6	549.3	3.61
IM	43.8	51.2	185.7	40.7	37.7	53.8	121.6	534.5	3.52
XP	16.2	26.2	205.6	82.7	46.9	92.7	122.4	592.5	3.90
OAve	53.3	52.8	191.1	53.1	41.0	62.4	158.6	612.1	4.03

TABLE 2-7. AVERAGE GROUP EFFORT BY ACTIVITY AS PERCENTAGES. SOURCE: BENEDIKTSSON ET AL. (2006)

Group	ReqsSpec, %	Design, %	Code, %	Integration and testing, %	Review, %	Repair, %	Other, %	Total, %
VM	10	10	27	5	7	8	33	100
EM	12	10	29	11	5	9	23	99
IM	8	9	35	7	7	10	23	99
XP	3	4	34	14	8	17	21	101
OAve	8.9	8.9	30.8	8.9	6.5	10.0	25.7	100

TABLE 2-8. PRODUCT QUALITY ASSESSMENT ON A SCALE 0 - 10. SOURCE: BENEDIKTSSON ET AL. (2006)

Group	Functionality	Usability	Maintainability
VM	7.9	7.4	8.9
EM	7.9	7.3	7.9
IM	8.3	8.3	7.3
XP	8.0	7.7	7.7
OAve	8.0	7.7	8.0

Benediktsson *et al.* (2006) reached the conclusion that the assumptions regarding XP (therefore agile) methods were satisfied, since small XP teams were able to deliver the biggest and most comprehensive products with more additional functionalities. XP offered developers flexibility that allowed adaptation to changing contexts. Also parallel with expectations, VM team required the most time, and produced products that were behind the most recent requirements.

Shine Technologies (2003) published a survey on the use of agile processes. Accordingly, 93% said team productivity improved; 88% found the quality of applications was better and 83% experienced better business satisfaction with the software. Furthermore, according to this survey, 95% of the respondents indicated their costs were unchanged or lower after adopting agile processes.

Another study by Cohn (2004) contrasts companies that produce heavy documentation with agile methods, and concludes that agile methods provide a strong competitive advantage in the professional world.

2.5.10. A Final Look at the Development Methodologies

Goldfinch (2007) states that most of the information systems development projects result in failures. Larger development projects have higher rates of failure. While the numbers are uncertain and there are not universally accepted criteria to decide failure, 20% to 30% of all development projects result in complete failure (abandonment); 30% to 60% of all projects are partial failures due to cost and time overruns. The rest, which is a small minority, can be regarded as examples of success.

There can be many reasons for development project failures. One of them is that, usually there are large gaps between that the actors have in mind about the final product to be delivered. Such problems will largely originate due to the lack of communication and understanding between these actors. This is humorously approached in Figure 2-24, which shows that while the end user asked for a simple solution, the various members of the information systems team had different ideas about the user requirements due to inaccurate communications in between actors, resulting in an end product that is completely different than the desired simple solution. Kurbel (2008) state that especially the waterfall process has the drawback that the original requirements and the requirements at the delivery are different.

Reeve and Petch (1999)'s example differ from Kurbel (2008)'s in that it also shows the potential problems associated with communication issues, again humorously portrayed in Figure 2-25. In addition to similar problems, final users of the system may use the system in a way other than the way it was designed by the developers, and the way users themselves asked for in the first place. Currently no development methodologies including agile development, which is relatively the most recent methodology, provide any methods for somehow including user observation to see how the users actually use the software. Available methodologies usually assume that sufficient active user input will be sufficient for a successful implementation.

Another explanation comes from Reeve and Petch (1999), as they claim that information systems used to be solely seen as technological products. When an information system fails, the immediate and almost instinctive reaction is to look for technical explanations. Technical people usually think that software could not cope, network infrastructures and protocols were inadequate, or the system response times were poor. However some computer specialists conclude that failure lies behind neglecting the human and organizational aspects of computing. Systems were ill-fitted to the organizations they were delivered. Organizations have been expected to accommodate technologies, rather than the other way around.

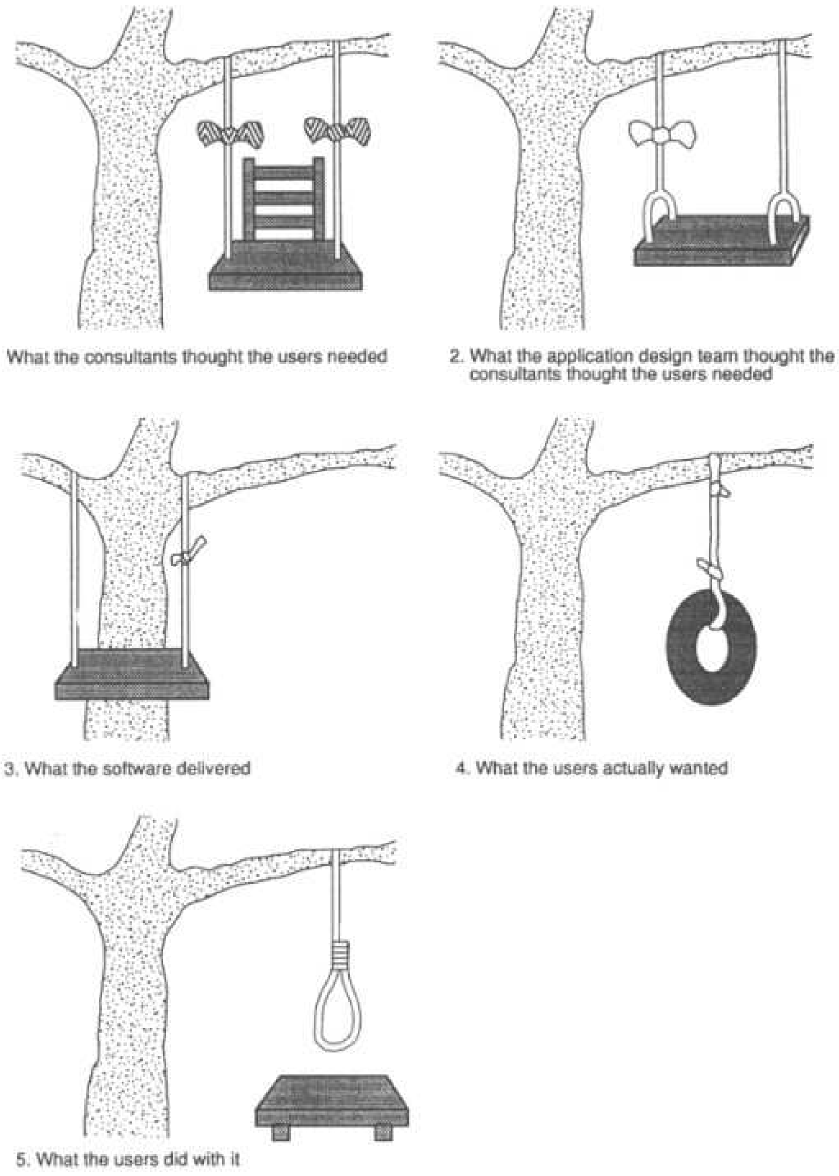

FIGURE 2-24 USERS DON'T ALWAYS GET WHAT THEY WANTED. SOURCE: REEVE AND PETCH (1999)

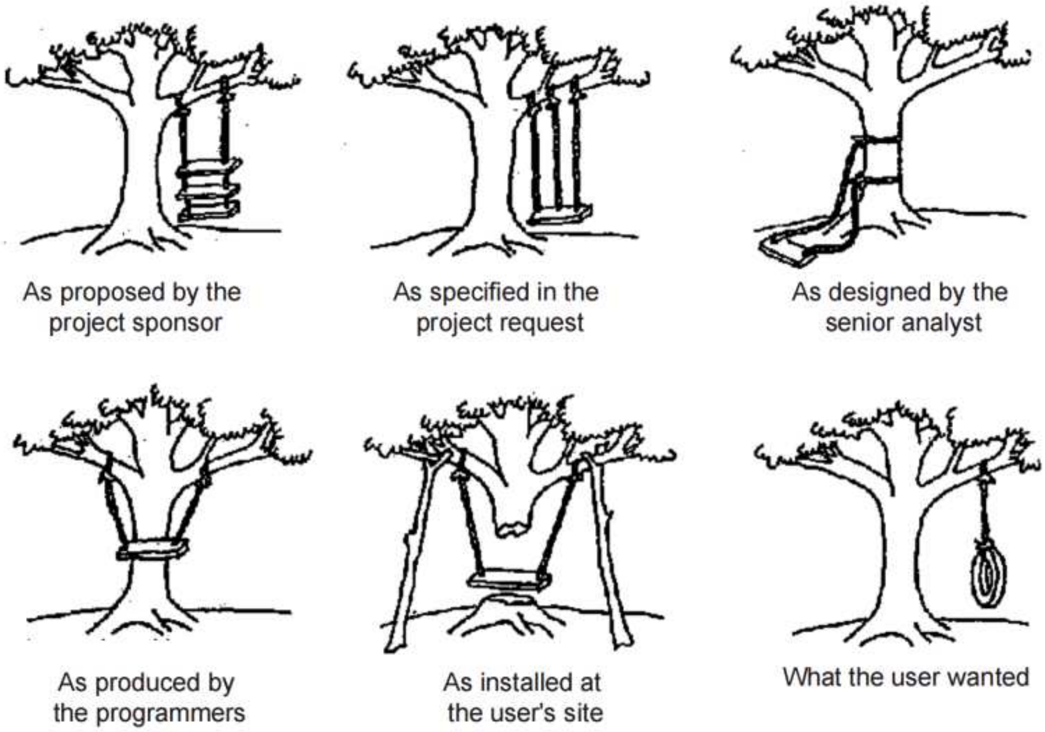

FIGURE 2-25 WHAT THE USER WANTED. SOURCE: KURBEL (2008)

At this point it is clear that methodologies that integrate human and organizational aspects of computing are needed to addresses issues relating end user's not getting what they wanted. Benediktsson *et al.*'s (2006) experiment using different methodologies revealed that each methodology has certain strong points. However specific methodologies (XP in their case) offer better overall results, especially with regards to being able to adapt to changing user requirements. Another conclusion they also pointed out was that the most suitable methodology depended on the contextual characteristics and participants.

For example, if there is a highly technical project on a security theme, with end users capable of producing detailed technical requirements, waterfall type of development which ensures efficient reliability and maintainability may be among the best options. Similarly, more than 20 years ago, Brooks (1987) claimed that there is no silver bullet, and this is a result of the essential and accidental complexities. Berry (2004) agrees with Brooks, and supports the argument by basing the problem on requirements, as they always change, and they are always misunderstood. Despite of his criticisms, he also agrees that methodologies actually do offer certain benefits.

Methodologies certainly offer benefits; otherwise developers would stick to code and fix method, which is merely an attempt to start programming right after facing a problem. Efforts to introduce systematic development have been clearly helpful throughout the history of software engineering. These efforts later been enhanced with more flexible (such as the evolutionary), rapid and agile methods. For example, Goldfinch (2007) cites examples which showed an increase in the success of software development projects from 26% to 29% in USA, from years 2001 to 2004.

The software methodologies evaluation framework proposed by McConnell (1996) offers a wide range of viewpoints a system development manager would be interested in. While it is obvious that there is no magical development methodology (or silver bullet), some methodologies have certain aspects making them better when compared to the rest, as it can be observed in McConnell's framework. The question then becomes, which aspects of software development are more important in a given set of general requirements, context and staffing environment. For example, Turban and Aronson (2000) argued prototyping is the ideal development methodology for DSS development, because of semi-structured and unstructured nature of the problems DSS users face. Their decision resided in the idea that DSS designers cannot have a complete and accurate understanding of the scope of the problem, appropriate models that need to be implemented and the information needs.

2.6. Summary and Conclusions

This literature review has a wide scope. The topics ranged from the changes in emergency management thinking, the application of information systems for emergency management and user-oriented system design for emergency management highlighted by principles of situational awareness. All these topics were contributions forming a foundation to design and apply an information systems development methodology to construct a WEM-DSS, as it is one of the challenges of this research.

This methodology stems from the major strengths of current methodologies and additional social theories that are mentioned in this literature review and are critical for enhanced development processes.

Studies show that volatile work environments and conditions require agility. Agility is a collection of principles, and the selection of particular agile methodology should depend on particular needs and conditions of the project, organization type and work environment. Extreme programming methodology is one of the apparent choices for this research study. It fits well into many criteria identified, and it is an appropriate option particularly for this case since time and resources are limited, and the immediate purpose is to develop a prototype that demonstrates functionality that can be implemented in the future. However, pure and unenhanced use of agile methods may not be recommended for life critical systems with extremely low tolerance to errors. A CMMI framework is recommended can be especially useful implementation of life critical systems. Enhanced XP integrated into CMMI framework is a viable solution in this case, and the benefits of combining the methodologies, such as:

- The combination provides a balance between agility and discipline
- It also provides a foundation for transitioning into higher levels of CMMI
    - Quality and reliability considerations are addressed
    - Process improvement is envisioned

Such a combination will guide the development of the software for this case study. Testing and evaluating it in the situation I have chosen will present a solid opportunity to design an appropriate development protocol combining XP and CMMI principles, and to evaluate the efficacy of the design used to produce the software for the client.

The next chapter will outline the translation of the XP and CMMI principles to a working methodology and how evaluation of the efforts was planned.

Chapter 3 Methodology

3.1. Overview of Development and Research Methodology

There were many uncertainties regarding the project before implementation. Additionally, it was not very clear what features needed to be developed. I anticipated that there could be multiple changes to systems requirements during the development. Additionally it was not clear how much end users could contribute or how much technical knowledge they had. One assumption was that if we showed them some prototypes early in the development, they could see how it could be further improved and we could incorporate their input midway in the development process. This would also require short development cycles. Due to these factors Extreme Programming (XP) seemed as a viable candidate development methodology. This study also required documentation to monitor the process and evaluate the performance of the implementation to guide the development. Due to such needs Capability Maturity Model Integration (CMMI) was seen as a methodology that required appropriate documentation, which would serve as lessons learned.

For above mentioned concerns and needs, the development methodology used in this study relied on an integration of XP and continuous representation of CMMI. This choice of a continuous representation was made since this representation offers maximum flexibility. Using continuous representation allows for a focus upon specific process areas, rather than attempting to improve the whole organizational process. Concentration on specific process areas, especially the ones related to software production fits the purpose of developing for priority areas in the WEM-DSS better.

Situational awareness principles were integral to the design step of the development. These principles guided the interface design as well as determining how to represent spatial and non- spatial information. Melding the two development approaches and situational awareness principles required careful selection of the correct elements to base a design methodology upon. This melding is illustrated in Figure 3.1. Capability Maturity Model Integration (CMMI) constituted the framework for process improvement part of the methodology. Accordingly, when there were problems in a particular practice during the development, that practice would require modification. The target goal was capability level 2 for selected process areas (continuous representation of CMMI). Extreme programming primarily guided how the product would be developed by setting four stages of planning, design, programming and testing to develop product releases. While situational awareness only contributed to design stage, CMMI process areas corresponded to all XP stages. Each of CMMI process areas required documentation during development, and this documentation corresponded to the stages set by XP as well. The details of XP and CMMI are explained in following sections in detail.

The proposed methodology covered two important aspects of systems engineering in order to satisfy the main research objective: What engineering products to produce and how such products are produced. With regard to the first aspect, the intermediate products of this research will be documentation required by selected CMMI areas. CMMI’s focus is producing certain products to improve processes, and it does not explicitly specify how to produce them. For the scope of this research, my corresponding purpose is to develop a WEM-DSS for the emergency managers in Oklahoma. This will be done by implementing selected process areas of CMMI to reach a certain capability level for the software development process as described by the Software Engineering Institute. Process improvement will be achieved by producing a set of documentation that conforms to CMMI standards, which is a key to maintain discipline in software development. Selected CMMI areas for this research have been explained as well as justifying why some certain process areas are omitted in Section 3.3.

FIGURE 3-1 CMMI, XP AND SITUATIONAL AWARENESS

With regard to the second aspect (that is how the engineering of the systems will be implemented) the proposed methodology utilizes the XP methodology as a means to analyze, design, and implement a WEM-DSS. In the literature review agile methodologies were discussed, and they were found the most adaptive and popular methodologies in the literature reviewed. Extreme Programming (XP) is a particular agile methodology, and will be used in this study for software development methodology. Certain parts of XP methodology will be omitted however, due to constraints in research environment. These modifications and constraints have been explained in Section 3.3.

3.2. XP and CMMI Integration for this Study

3.2.1. Overview of Products: Selected CMMI Process Areas

Among the twenty-two process areas in CMMI for development, six were initially chosen to try to achieve due to time limitations of this research. These are the process areas that correspond well with XP practices and therefore mostly related to software development.

Project Planning

Project management is a process area under project management process category. The purpose of this process area is to develop and maintain plans that define what to produce during the course of the development project. Project planning is very important as any established software methodology requires certain amount of planning. For the development of WEM-DSS, planning was an important activity during development and will involve meeting with project advisors to determine which functionalities to develop within one to three weeks (which is a time span deemed optimal according to XP practices).

Requirements Development

Requirements development is a process area under engineering process category. The purpose of this process area is to identify user requirements and establish corresponding product requirements. This process is necessary as user involvement is central to this research. For the development of OK-FIRST, the requirements development process area is integrated with the XP practice of user stories collection. As CMMI requirements development requires more formal documentation than user stories, user stories will be written into formal and technical documentation.

Requirements Management

Requirements management is a process area under engineering process category. The purpose of this process is to gauge the requirements of the products to identify inconsistencies between project plans and products requirements, and identify any necessary changes in requirements. This process area is the only CMMI item that does not have a direct correspondence in XP. The reason requirements management is selected is that user involvement (communicating requirements with users and getting their commitment) is central to this research. Additionally, requirements management is closely tied to requirements development and project planning.

Risk Management

Risk management is a process area under project management process category. The purpose of risk management is tracking and monitoring risks before and after they occur. This process area also involves activities to determine actions to handle risks such as mitigation planning.

Technical Solution

Technical solution is a process area under engineering process category. The purpose of this process area is to design and implement technical solutions to user requirements. This process area involves evaluation and selection of solution alternatives, preparing designs for selected solutions and implementing designs.

Validation

Validation is a process area under engineering process category. The purpose of validation is to show that a product fulfills its intended. This process area is important as this is the way to check is the user requirements have been satisfied. Additionally, validation corresponds to user acceptance tests of XP.

Verification

Verification is a process area under engineering process category. The purpose of verification is to make sure that a product meets the specified requirements. This process area is included as verification is a means for quality assurance. In the context of OK-FIRST, verification is important in order to deliver a prototype meeting certain standards.

Often, verification and validation can be confused with each other. According to CMMI Product Team (2006), "verification ensures that you built it right; whereas, validation ensures that you built the right thing" (p. 483). In order to make sure the right products have been developed, users are expected to be involved in validation process. The reason for this is developers may not always be able to foresee what the actual user expectations are regarding the software.

Different aspects can be incorporated into verification and validation. Cohn (2004) mentions the following four: (1) User interface testing: to make sure the interface functions and can be used as expected; (2) Usability testing: to make sure software is easy to use; (3) Performance testing: to measure how fast and efficient the software behaves with varying workloads; (4) Stress testing: to observe how the software will respond when there is extreme number of users, parameters etc.

Other types of testing such as reliability testing (e.g. if the system can go without any crashes for two months, security testing (e.g. blockage of unauthorized access to the system) can be added to this list as well.

3.2.2. Extreme Programming Activities in the Integrated Development

There is no consensus on whether XP and CMMI are fully compatible, however it has been shown that XP can be modified to achieve CMMI level 2 (Kähkönen and Abrahamsson 2004). For a broad review on how XP and CMMI are compatible, see the literature review in section 2.5.7 at page 67.

There will be some modifications to XP due to integration with CMMI and due to limitations of this research. For example, since the development team will consist of one person, “pair programming” cannot be implemented. All these modifications and their justifications are explained in the remainder of this section.

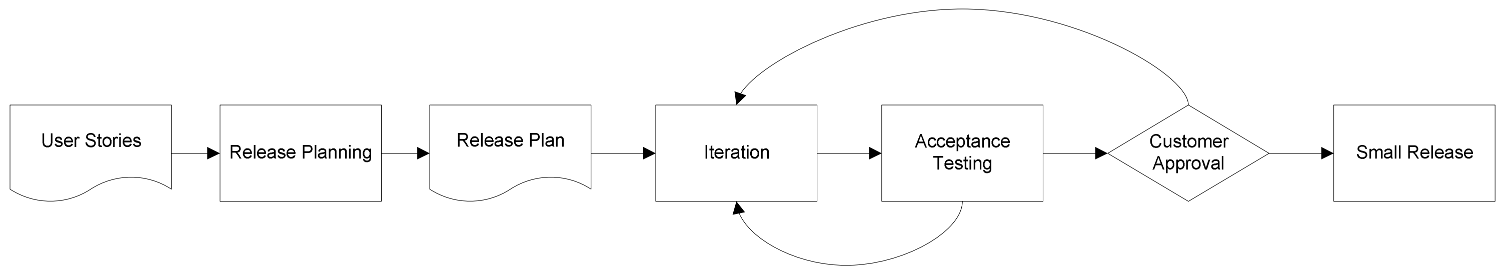

FIGURE 3-2 A VISUAL PROCESS MODEL FOR XP. MODIFIED FROM WELLS (2006)

The activities that comprise the development process to be implemented include:

Writing Story Cards

It is the responsibility of the customer to write stories, prioritizing them and testing that stories were developed as expected. The XP customers should be highly interested persons and can be the future users of the system, or product manager, project manager and analyst. This activity is related to requirements development of CMMI. A typical user story in the context of WEM-DSS for this study could be:

User story: Emergency manager checks the dew point for Oklahoma City area.
Priority: 9/10

Small Releases

XP is executed though a series of iterations, each of which usually takes one to three weeks. At the beginning of each iteration the development team decides on each iteration length. The duration of iteration should be as short as possible. Iterations cannot be extended, meaning that the amount of the work should be accommodated within the predetermined time frame without compromising the quality. The complete small release corresponds to the technical solution process area of CMMI.

The Planning Game

The planning game refers to the iteration planning during which customers and developers predict the future, depending on the story cards, and the cost estimates based on these cards. Cost estimates are made by the developers, they are simply man hours and required money expenditure. Customers prioritize the stories and then place the stories with highest priorities into the first iteration, which is limited by the amount of work developers think they can do. After the assignment for the first iteration, the remaining iterations are assigned to stories with decreasing level of priority. The customer then decides on which iterations will constitute releases, whenever he/she thinks there are sufficient stories. The planning game corresponds to the project planning process area of CMMI.

Refactoring

Refactoring refers to restructuring or rewriting of the code without modifying its behavior and functionality. This prevents the decay of the code that may cause serious problems in the near future.

Acceptance Testing

Acceptance testing is an important part of Extreme Programming. During an iteration, acceptance tests are created based on the user stories selected during the development. The customer specifies scenarios and expected results on how the particular function will be realized so that the implemented user story can be tested. It is the responsibility of customers to confirm the correctness of the acceptance tests and evaluate test scores to provide feedback to developers (Wells 2006).

Continuous Integration

The practice of integrating early and often is suggested in the XP environment, to avoid situations where there are separate applications that need to be integrated. The purpose of continuous integration is to save time eventually.

Modifications and Reduced Activities within the Extreme Programming Process

Since the programming team will consist of one person, "pair programming" cannot be attempted and "team code ownership" cannot be realized. Additionally, "unit testing" is also omitted due to limited time. XP also advocates sustainable pace, which means that development should move at a consistent and fast rate. This is certainly a desirable activity, however, due to the conditions in this research environment (such as uncertainties in commitment of customers and technical assistance from OK-FIRST people), it could not be foreseen whether it was possible to keep the development pace consistent.

3.2.3. Overview of Proposed Integrated Methodology

This methodology was the main component of the overall project. The model in Figure 3-3 can be viewed as the extended version of a typical XP process, which is enhanced with additional documentation described by CMMI standards. All the documentation to be produced is grouped under 6 CMMI process areas that are selected for development for OK-FIRST. Note that Project Planning of CMMI refers to Release Planning of XP, and Technical Solution of CMMI refers to Iteration of XP. Red colored activities are XP based, and they determine the flow of the method. CMMI activities (blue colored) are anchored to XP processes, and each CMMI activity contains relevant documentation.

As a principle, while there are many instances of documentation to be produced, documents were kept as compact and simple as possible considering each iteration should be done in at most a month (as a principle of XP), and there is only one principle developer in this project.

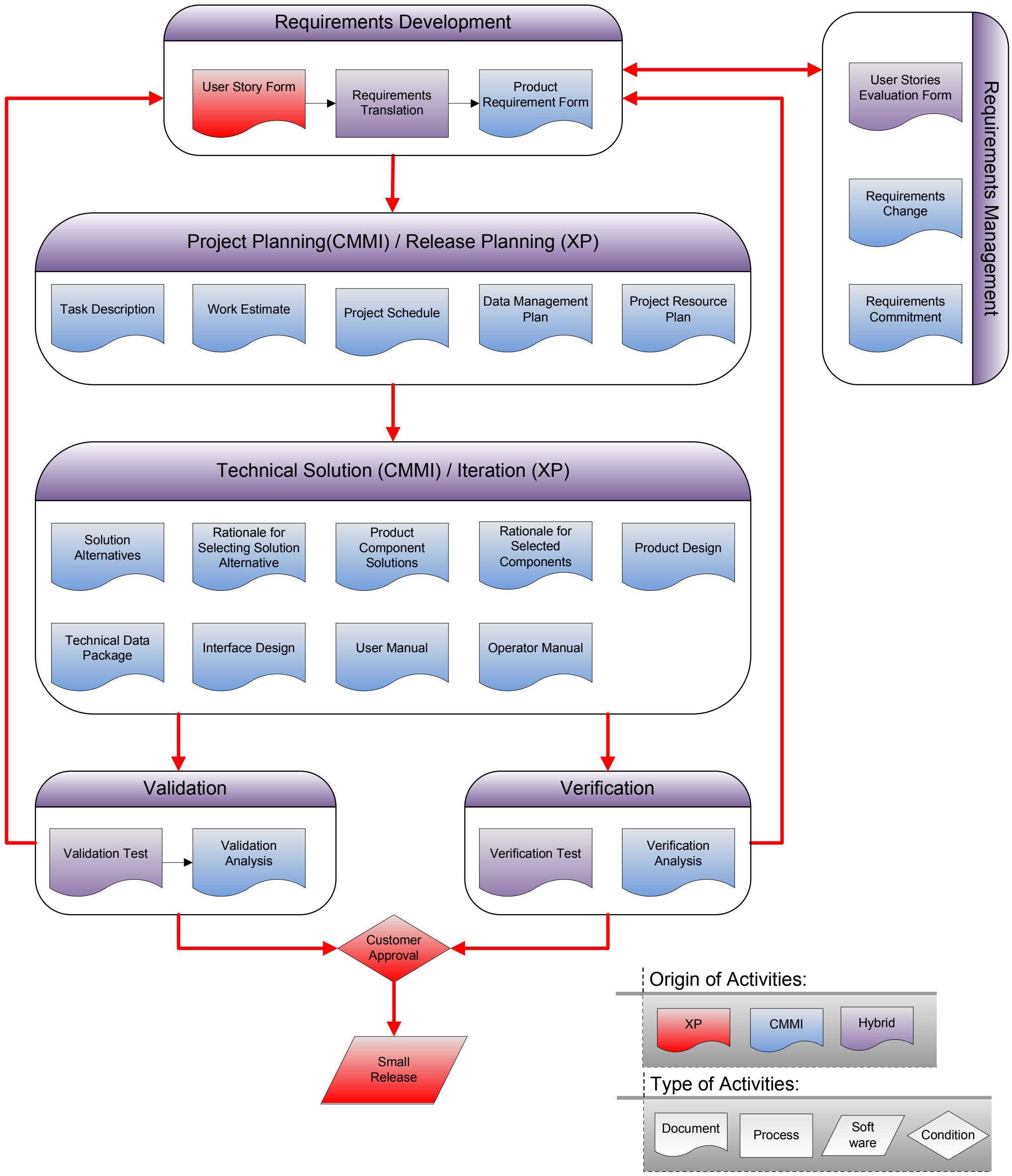

FIGURE 3-3 XP AND CMMI ORIENTED DEVELOPMENT METHODOLOGY. NOTE THAT PROJECT PLANNING OF CMMI REFERS TO RELEASE PLANNING OF XP, AND TECHNICAL SOLUTION OF CMMI REFERS TO ITERATION OF XP.

3.2.4. Achieving Generic Goals and Practices

In order to achieve a capability level of 2, all the generic goals for each of the process areas need to be satisfied. Under each generic goal there are a number of generic practices, which are recommended practices, but they are not obligatory. Below I explain which generic practices will be implemented, and how. Some of the practices may be implemented partially or not at all, and the reasons are given for each generic practice.

Generic Goal 1 Achieve Specific Goals

Generic Practice 1.1 Perform Specific Practices

Perform the specific practices of the process area to develop work products and provide services to achieve the specific goals of the process area.

For each of the process area, it is explained how specific practices will be implemented in Section 2.3.2 “Achieving Specific Goals and Practices”.

Generic Goal 2 Institutionalize a Managed Process

Generic Practice 2.1 Establish an Organizational Policy

This practice is addressed in project initiation.

Generic Practice 2.2 Plan the Process

The purpose of planning process practice is determining requirements to perform the process area. This involves preparation of a process description, and to get agreement on the plan from relevant stakeholders (CMMI Product Team, 2006). This goal is addressed in sections XP and CMMI integration (explained specifically for each process area) and project initiation. For the purposes of the dissertation research, the entire dissertation proposal was a plan of the research process, and the proposal defense was the procedure to get the agreement on the plan from primary stakeholders (end users and committee members) who are the equivalent of managers in this case.

Generic Practice 2.3 Provide Resources

The purpose of providing resources practice is making sure that the resources required to perform the process area are available. These resources may include funding, time, physical facilities, skills, and tools (CMMI Product Team, 2006). This practice was addressed at in project initiation, resource planning section. For the purposes of the dissertation research, resource planning involves meeting with interested parties (may involve people in dissertation committee, OK-FIRST users, and emergency managers) and figuring out required amount of resources in terms of man hours, access rights to any facilities, necessary hardware and software.

Generic Practice 2.4 Assign Responsibility

The purpose of assigning responsibility for processes and products is to ensure that there is accountability for performing these activities to achieve aimed results through the project (CMMI Product Team, 2006). This practice was mainly reduced to assigning the project development responsibility to the principal developer. Dissertation committee members naturally had the responsibility to monitor the processes and evaluate the work.

Generic Practice 2.5 Train People

This practice was addressed in project initiation, methodology communication section. For the purposes of the dissertation research, the only training users (emergency managers and other potential users who will determine requirements) needed to have is to learn how to write story cards and validation tests.

Generic Practice 2.6 Manage Configurations

This goal primarily applies for configuration management process area of CMMI, and it can be used in other process areas as well. However, this goal will not be addressed in this project for developing for WEM-DSS; since this is a smaller sized project, change and version control is not expected to be a frequent activity. Therefore managing configurations is not a priority, nor a necessity in this project.

Generic Practice 2.7 Identify and Involve Relevant Stakeholders

This goal is addressed in project initiation section, particularly stakeholder identification and stakeholder commitment acquisition subsections. For the purposes of the dissertation research, additional potential stakeholders that are other than emergency managers, committee members and OCS people will be identified if necessary.

Generic Practice 2.8 Monitor and Control the Process

The purpose of monitoring and controlling the process is to be able to take corrective actions whenever necessary (CMMI Product Team, 2006). The development process was structured as it is broken down into pieces that follow XP methodology, so that it is possible to monitor and control the processes within the development cycles for the WEM-DSS to be developed.

Generic Practice 2.9 Objectively Evaluate Adherence

The purpose of objectively evaluating adherence is providing assurance that the processes have been implemented as planned that the processes matches descriptions, standards and procedures (CMMI Product Team, 2006). Objective evaluation for adherence to the methodology and requirements can only be done by the people outside the development team. These people may involve committee members, and scope of this evaluation depends on how much time they can put into evaluation of the process and products.

Generic Practice 2.10 Review Status with Higher Level Management

The purpose of reviewing status with higher level management practice is providing higher level management with the process as the project proceeds (CMMI Product Team, 2006). There is no "higher level management" in this study, except that the dissertation committee members and directors of OK-FIRST can be seen as management people. Compliance with this goal depends on how much time they can put into review of the research, just like Generic Practice 2.9.

3.2.5. Achieving Specific Goals and Practices

This section also contains description of steps and CMMI process areas shown in Figure 3-3.

Requirements Development

Product Requirement Form

This is a pure CMMI practice and needed for Requirements Development Specific Goal 2. Based on the user story, the requirements engineer (that is the developer in this project) will develop a list for descriptions of architecture requirements, functional requirements, component requirements, related processes and necessary resources regarding WEM-DSS development.

Requirements Management

User Stories Evaluation Form

This is a hybrid product built on user stories (of XP) that satisfies CMMI Requirements Management (Specific Practice 1.1) requirements. For each user story filled out by OK-FIRST users, priority, clarity, completeness, appropriateness, testability and traceability are evaluated by requirements engineer (the developer in this project).

Requirements Change

Requirements change is necessary in application of CMMI Requirements Management Specific Practice 1.3. If a requirement changes in the WEM-DSS development due to any reason, it is recorded into Requirements Change form.

Requirements Commitment

Requirements commitment is necessary in application of CMMI Requirements Management Specific Practice 1.2. That is, for each selected user story, commitments from people who are needed to contribute (they can be committee members, OCS people or emergency managers) are collected and recorded.

Project Planning

Task Description

This document is necessary in application of CMMI Project Planning Specific Practice 1.1. Each task in the WEM-DSS development needs to be described and relevant requirements need to be specified.

Work Estimate

This document is necessary in application of CMMI Project Planning Specific Practice 1.2. For each of the task that is selected to be implemented, estimates of criteria, size and complexity of tasks and work products, and work estimate are evaluated and recorded.

Project Schedule

This document is necessary in application of CMMI Project Planning Specific Practice 2.1. For each iteration in the development of the WEM-DSS, covered tasks, schedule assumptions, task dependencies and amount of needed assistance are recorded.

Data Management Plan

This document is necessary in application of CMMI Project Planning Specific Practice 2.3. For each iteration in the development of the WEM-DSS, data content and format description, privacy requirements, security requirements, mechanism for data retrieval, reproduction and distribution and schedule of collection of project data are recorded.

Project Resource Plan

This document is necessary in application of CMMI Project Planning Specific Practice 2.4, 2.5, 2.6 and 3.1. For each iteration in the development of the WEM-DSS, critical facilities and equipment lists, skill needs, identification of necessary stakeholder involvement and stakeholder commitment are recorded.

Risk Management

Risk and Issue Registers

Risk and issue registers are kept for tracking problems and foreseen risks throughout the project. In this study, a risk is identified as a threat to project objectives that has not occurred yet. An issue is identified as a risk that actually occurred. This is particularly useful for calibrating next phase project planning. For example, if there were too many issues in one phase / cycle, for the next cycle, fewer features or user stories may be chosen for implementation.

Technical Solution

Solution Alternatives

This document is necessary in the application of CMMI Technical Solution Specific Practice 1.1. For each task group (iteration) in the WEM-DSS development, a set of solutions should be identified and documented. All the alternative solutions should be evaluated according to cost, technical limitations, risks, scalability, performance and complexity criteria.

Rationale for Selecting Solution Alternative

This document is necessary in the application of CMMI Technical Solution Specific Practice 1.1. Among the alternatives, the best solution should be selected, and the reasons for selecting that alternative should be stated in this document.

Product Component Solutions

This document is necessary in the application of CMMI Technical Solution Specific Practice 1.2. For each of the solution alternatives, necessary and/or alternative components need to be identified in this document.

Rationale for Selected Components

This document is necessary in the application of CMMI Technical Solution Specific Practice 1.2. Among the alternatives, the best component arrangement solution should be selected, and the reasons for selecting that alternative should be stated in this document.

Product Design

This document is necessary in the application of CMMI Technical Solution Specific Practice 2.1. For the selected product in OK-FIRST development, use cases, static UML diagram and activity diagrams need to be created and documented.

Technical Data Package

This document is necessary in the application of CMMI Technical Solution Specific Practice 2.2. For all the products to be developed for OK-FIRST, product architecture description, product component descriptions, and allocated resources will be described in this document.

Interface Design

This document is necessary in the application of CMMI Technical Solution Specific Practice 2.3. For all the products to be developed for OK-FIRST, interface design will be described in this section. It will be evaluated from a Situation Awareness point of view. Interfaces with internal and external components will be identified.

User Manual

This document is necessary in the application of CMMI Technical Solution Specific Practice 3.2. This document will contain the instructions on how to use the developed product(s) for the users of the WEM-DSS.

Operator Manual

This document is necessary in the application of CMMI Technical Solution Specific Practice 3.2. This document will contain the descriptions regarding the product(s), so that developers will be able to modify it/them in the future for future developments of the WEM-DSS.

Validation

Validation of the product is carried out by customer. If the results of validation indicate any problems with regards to the implementation or the design, these issues will be fixed before moving to the next development step in the WEM-DSS development.

Validation Test

This document is necessary in the application of CMMI Validation Specific Practice 1.1, 1.3 and 2.1. For each product/service to be tested, relevant user stories and requirements are identified. The validation process is described, with instructions and expected results. After the tests are executed regarding the WEM-DSS development, errors (if any) are recorded, and results are described and evaluated.

Validation Analysis

This document is necessary in the application of CMMI Validation Specific Practice 2.2. For each of the validation test regarding the WEM-DSS development, expected results, actual results and products/services with issues are listed. This is performed by the end users and the results relayed to the developer for use in the next process, verification.

Verification

Verification process is the same as validation, except that it is carried out by developer. If the results of verification indicate any problems with regards to the implementation or the design, these issues will be fixed before moving to the next development step in the WEM-DSS development.

Verification Test

This document is necessary in the application of CMMI Verification Specific Practice 1.1, 1.3 and 2.1. For each product/service to be tested regarding the WEM-DSS development, relevant user stories and requirements are identified. The validation process is described, with instructions and expected results. After the tests are executed, errors (if any) are recorded, and results are described and evaluated.

Verification Analysis

This document is necessary in the application of CMMI Verification Specific Practice 2.2. For each of the verification test, expected results, actual results and products/services with issues are listed.

3.2.6. Practices completely outside CMMI

Use of user story form is an XP practice that is employed in this study. Users will write a description of tasks they would like to carry out in user story forms, and they will constitute the user requirements. In order to do this, there needs to be a transition from the informal user story forms (required by XP) to product requirement form (required by CMMI). This process will involve interpretation of user story forms and writing them in a more technical format, so that it conforms to CMMI standards. Lastly, once the iteration is completed, customers will evaluate the product by validation tests (acceptance tests), and approve the product is their requirements are met. This step is called customer approval.

3.3. Preliminary Design Considerations

While an agile method will be adopted (that is project requirements are open, and always subject to change), the researcher nevertheless has some preliminary design considerations. These considerations stem from state of the art practices in decision support, emergency management, working environment and similar WEM-DSS's that are being used by various organizations.

Such practices were discussed throughout the literature review presented in Chapter 2. This discussion demonstrated that Geographic Information Systems support and situational awareness plays a great role in emergency management decision support.

Additionally, the broad range of users requires consideration of user customization as a viable feature.

The multi-hazard approach has been raised throughout the introduction (especially in research questions and objectives) and literature review sections of this dissertation. This has been addressed by including emergency managers that are involved in various hazards for inputting to study. These hazards mainly include hazmat, weather related and fire hazards. Also situational awareness has been an important part of the literature review. Situational awareness oriented principles has been adopted during the design of the interface of the system.

Due to the familiarity of the developer with .NET and ESRI ArcGIS Server, ArcGIS Silverlight API was decided to be used. All these applications were targeted to be integrated in a Service Oriented Architecture (SOA) fashion. An SOA infrastructure allows different applications to exchange data with one another as they participate in various processes. Service-orientation aims at a loose coupling of services with operating systems, programming languages and other technologies which underlie applications. Additional data sources, such as Google Maps and KML files can be introduced under this particular architecture.

The iterative structure of Extreme Programming can be ideal to observe appropriation of developed technologies and how social structures influence these processes. This could simply be done by having emergency managers evaluate and/or use the product. However, as it is discussed in Chapter 4 (Implementation chapter), user input and feedback was so limited that such points could not be visited within the scope of the case study.

3.4. Brief Overview of the Methodology

This methodology starts with project initiation that requires documenting organizational policy, resource planning, methodology communication, stakeholder identification and stakeholder commitment acquisition. After the initiation, project continues with the implementation. When the implementation is finished, the project is evaluated. These activities are described in detail below:

3.4.1. Project Initiation

The research project will be initiated by completing five items as described below.

a. Document Organizational Policy

This activity satisfies CMMI Generic Practice 2.1: Establish an Organizational Policy. It involves the following:

> "Establish and maintain an organizational policy for planning and performing the process. The purpose of this generic practice is to define the organizational expectations for the process and make these expectations visible to those in the organization who are affected" (CMMI Product Team, 2006).

Such a policy can be generated for the corresponding six process areas with the help from committee members and some input from OCS.

b. Resource Planning

This activity satisfies CMMI Generic Practice 2.3: Provide Resources. It involves the following:

> "The purpose of this generic practice is to ensure that the resources necessary to perform the process as defined by the plan are available when they are needed. Resources include adequate funding, appropriate physical facilities, skilled people, and appropriate tools" (CMMI Product Team, 2006).

For each of the process areas, resource planning and allocation need to be done.

c. Methodology Communication

This activity satisfies CMMI Generic Practice 2.5: Train People. It involves the following:

> "The purpose of this generic practice is to ensure that the people have the necessary skills and expertise to perform or support the process" (CMMI Product Team, 2006).

The purpose of this document is to inform the committee members (aka project managers) about the process areas, methods and documentation to be produced. The customers will also be trained on how to produce user stories and validation tests.

d. Stakeholder identification:

This activity satisfies CMMI Generic Practice 2.7: Identify and Involve Relevant Stakeholders. Due to the work environment, the same person can be identified as both as a customer and a project advisor. Stakeholder identification involves:

- Definition of customers: These people will direct the development on "what's" of the project. E.g. "what needs to be on the flood monitoring screen?"
    - Definition of development team: The actual development team principally consists of the researcher (one person). However, a few more people could *and* should be involved to assist the developer regarding the technical matters and limitations especially during the planning game.
    - Explaining to customers about their duties (story cards, planning and testing): This is necessary considering probably no customer will have a prior knowledge about XP.
- Definition of project advisors: These people will direct the development on "how's" of the project. E.g. "how to arrange four information panels on the main screen?". These people can also act as customers, in other words, they can direct the development on "what's" of the project as well.
    - All the committee members are project advisors and their input will be seek as long as they can commit assistance
    - Other people, such as people with technical expertise on OK-FIRST
    - People with expertise on software interface design
    - People with expertise on emergency management

e. Stakeholder commitment acquisition:

This activity satisfies CMMI Generic Practice 2.4: Assign Responsibility and Generic Practice 2.7: Identify and Involve Relevant Stakeholders. Stakeholder commitment will be necessary during the software development process in terms of dedicating specified time and efforts. Parallel to this, during the project initiation, a general commitment needs to be acknowledged by identified stakeholders that they will be able to contribute to the study throughout the development process.

3.4.2. Implementation

Implementation corresponds to technical solution process area of CMMI and iteration activity of Extreme Programming. Development methodology will be implemented as defined in detail in previous section. The basic principles of the iteration are:

- Each iteration should be finished in three to four weeks. During an iteration, several requirements could be accommodated.
- Several iterations will result in a release. Depending on the project status, one or few releases can be delivered.
- Project planning will be carried out under supervision of "project advisers".

3.4.3. Project Evaluation

Project evaluation is different from validation and verification, which are parts of development process. Project evaluation will include an evaluation of

- Overall research project
- Success and Issues with XP and CMMI integration
- Emergency Management Decision support improvement

There will be ideally three groups of people who evaluate:

- Committee members at the management level
- OK-First managers and developers
- Current and prospectus OK-First Users

3.5. Anticipated Timeline

Project initiation phase was planned to be completed one month after proposal is presented to the dissertation committee, pending agreement of the committee to proceed.

The implementation part is to be conducted using facilities at University of Oklahoma Center for Spatial Analysis, University of Oklahoma Department of Geography and Oklahoma Department of Wildlife Fishery Research Laboratory. U input was to be collected from emergency managers in the State of Oklahoma. At the beginning of the study however, there was an input session meeting with three emergency managers in Ozark, Arkansas since there was not sufficient response to conduct an input session meeting with Oklahoma emergency managers.

Note that proposed software development methodology and the general methodology of this study are not the same. Rather, proposed software development methodology can be seen as a sub-methodology, whereas the general methodology of this study encompasses software development methodology as well as project initiation and project evaluation.

Another important point concerns the activities within the proposed software development methodology. Since the intent is to stick to agile principles, it would be inconsistent to predetermine exactly when the specific activities will take place. Rather, a certain amount of time will be devoted to application of proposed software development methodology. Accordingly, the number of iterations, releases, and the exact amount of hours devoted to coding, testing, planning and documentation will be decided depending the course of the development process, and the user input.

This phase was planned to be completed three to four months after project initiation was completed. However, since this is an agile approach in essence, the timeline needs to be modified during the project. A rough time span is proposed instead of a list of functionality as it cannot be foreseen whether it is possible to keep the development pace consistent.

3.6. Summary

This chapter describes the methodology that was employed, providing details of how the integration of Extreme Programming and CMMI were accomplished. Specifically, the selected CMMI process areas and the rationale as well as how to achieve the generic and specific goals were explained.

This chapter was finished with description of preliminary design considerations and the anticipated timeline.

Chapter 4 Implementation and Analysis

4.1. Introduction

This study was conducted based on the input from emergency managers that use OK-First, or had training for it. The reason for this selection was that they were knowledgeable about using computerized systems for emergency management, and it was possible to contact them through OCS for input. The scope of the product has been broadly identified as decision support system emergency management. The identification of the particular functionalities for emergency management and the prioritization of depended on the user input.

The implementation consisted of initial project planning and project execution that included three phases. Initial project planning consisted of IRB documentation submission, Draft of Mission Planning, Draft of Toolkit Description, Mission Planning Revision Statement, IRB Approval and Input Session with Emergency Managers.

After Project Planning, project continued with Project Execution. Project execution involved three development cycles that are typical to Extreme Programming practices. For each cycle, coding was the priority; therefore most of the time (10 to 15 days) was allocated for coding practice.

All the coding was done in MS Silverlight, Visual Studio .NET and using the ArcGIS Silverlight API. A Silverlight application is written in two parts, the XAML code (which is an extension of XML) and either C# or Visual Basic.

4.2. Organization of User Input

User input was mainly in the form of user stories. Two input session meetings were conducted before the implementation and another input session was conducted during the development to identify user stories, and to prioritize them. See Figure 4.1 for the evaluation of list of user stories at the beginning of the project (after the first two input sessions). The first input session was conducted on October 15th 2009 in Ozark, Arkansas with three emergency managers, the second was conducted on October 23th

2009 in Norman, Oklahoma with three emergency managers and the third was conducted on January 12th 2010 with four emergency managers in Norman, Oklahoma.

4.2.1. Acquisition and Evaluation of User Stories

In the first two input sessions prior to implementation, a list of potential user stories was presented. This list was prepared based on data which was put together as a result of the initial interviews with three emergency managers, and the recommendations of Dr. Rashed, a committee member, due to his expertise on the WEM-DSS. Another resource for user stories was user evaluations that were collected for evaluation of OK-FIRST. These evaluations were assessed and selected portions were converted into user story format.

In the user input sessions, every item in this list was read to them, and the users were asked if the particular user story was something useful for them. They were also asked how a particular functionality (that corresponds to a user story) would be used, and if there would be any modifications regarding this functionality.

They were also asked to identify which functions were more important than the others. While they said all of the user stories were important, the responses regarding the importance of each user story was not structured as the users were not able to provide precise answers. Since the unstructured responses were not useful, the prioritization was made by Dr. McPherson, another committee member. For each story, scores between zero and ten assigned for four parameters, importance, ease of implementation, clarity and completeness. A score 10 for importance meant the user story was extremely important, a score 10 for ease to implement meant the user story was extremely easy to implement, a score 10 for clarity to implement meant the user story was extremely clear to understand and a score 10 for completeness meant the user story was entirely complete. Another parameter, priory was then defined as a simple multiplication of parameters importance and ease of implementation, that ranged from zero to a hundred, with hundred indicating highest priority. Another parameter, called testable indicates whether a user story is testable or not. Values could either be “Yes” or “No”. This evaluation is shown in Table 4-1, along with the source for each user story and the task identification number if the user story has been selected. Additionally, selected user stories have been highlighted according to the development cycle they were developed in.

4.2.2. Selection of User Stories for Development Cycles

After the parameters of all user stories were prepared, I was able to see which user stories had a higher priority for development. While sorting them according to their priorities and selecting those with highest priorities would be an obvious choice, from an organizational and practical point of view it would not have been ideal. Therefore these specifications were not absolutely necessary to follow; rather they were treated as guides. Many of the user stories actually complemented each other or developing one was a requirement for developing another. Therefore developing a complementary user story was easier and more practical than implementing an unrelated one. Therefore, often for each development cycle a number of user stories were grouped into tasks. Since the development methodology focuses more on agility while avoiding to spend too much time on planning the evaluations were limited to these criteria mentioned before.

Additionally, selection of the user stories to be developed was done at the beginning of each cycle, not at once. This allowed the development process to be flexible, which is an agile development principle, providing the ability to incorporate any changes or new additions to user stories. The initial user stories and their evaluation are shown in Table 4-1, which was slightly modified with further user input during the implementation.

TABLE 4-1 USER STORIES EVALUATION TABLE BEFORE IMPLEMENTATION

Story ID	Story	Importance	Ease to Implement	Priority	Clarity	Completeness	Testable	Source
1	User zooms in to map	8	10	80	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
2	User zooms out of map	8	10	80	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
3	User pans across map	6	10	60	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
4	User identifies features (point selection)	9	10	90	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
5	User zooms to the map extent	3	6	18	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
6	User zooms to the bookmarked features using a dropdown menu	6	8	48	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
7	User draws a hazard response plan	10	1	10	8	4	Yes	Dr. Rashed and I, and approved at first two input session meetings
8	User edits a hazard response plan	10	1	10	8	4	Yes	Dr. Rashed and I, and approved at first two input session meetings
9	User shares hazard response plans	10	1	10	8	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
10	User erases hazard response plans	10	1	10	8	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
11	User attaches building specific response plan(s) to the hazard response plan	10	1	10	8	2	Yes	Dr. Rashed and I, and approved at first two input session meetings
12	User views locations of emergency vehicles on the map real time (prototype)	8	5	40	10	6	Yes	Dr. Rashed and I, and approved at first two input session meetings
13	User toggles between different types of emergency vehicles	5	5	25	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
14	User views locations of emergency managers on the map real time (prototype)	2	5	10	10	6	Yes	First Input Session Meeting
15	User views labels on top of emergency vehicles on the map	7	5	35	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
16	User views topographic maps	8	10	80	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
17	User views land cover satellite imagery	8	10	80	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
18	User views land use maps	8	10	80	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
19	User views real time radar data	10	5	50	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
20	User views building floor plans	3	9	27	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
21	User views building floor plans by clicking on the building on the map	7	1	7	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
22	User views building floor plans by clicking on the building name from a drop down menu	7	8	56	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
23	User draws polygons on the fly during response	8	6	48	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
24	User views what critical facilities are in a drawn polygon automatically	9	5	45	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
25	User ranks the selected facilities based on an attribute	7	6	42	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
26	User views a checklist for actions to do for certain events	9	8	72	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
27	User identifies a hot emergency area on the map	0	7	0	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
28	User defines hot/warm/cold emergency areas as buffer rings	0	6	0	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
29	User sees the interface change colors when the threat level changes	10	2	20	10	6	Yes	Dr. Rashed and I, and approved at first two input session meetings
30	User views near real time cameras on the map	9	5	45	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings

Story ID	Story	Importance	Ease to Implement	Priority	Clarity	Completeness	Testable	Source
31	User sends text based messages to other users	8	4	32	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
32	An administrative level user specifies a message that will be displayed on all the user's interfaces	2	2	4	10	7	Yes	Dr. Rashed and I, and approved at first two input session meetings
33	User toggles between different data source	10	8	80	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
34	User views weather radar data that is refreshed every 1 minute	10	6	60	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
35	User accesses archived incident map reports and statistics	9	7	63	10	6	Yes	Dr. Rashed and I, and approved at first two input session meetings
36	User customizes mapping application	10	3	30	10	4	Yes	Dr. Rashed and I, and approved at first two input session meetings
37	User saves customized settings	10	3	30	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
38	User splits the interface into two maps	8	4	32	10	8	Yes	OK-FIRST User evalulations
39	User selects the county name from a drop down list to zoom in	5	8	40	10	10	Yes	OK-FIRST User evalulations
40	User can view county names on the map	10	9	90	10	10	Yes	OK-FIRST User evalulations
41	User views names of towns	10	9	90	10	10	Yes	OK-FIRST User evalulations
42	User orients the map by rotating it with a compass	3	6	18	10	10	Yes	First Input Session Meeting
43	User views critical facilities' building square footage	5	7	35	10	6	Yes	First Input Session Meeting
44	User views critical facilities by selecting them from a drop down menu	8	8	64	10	10	Yes	First Input Session Meeting
45	User views the address book to contact people in critical facilities and other agencies	10	7	70	10	6	Yes	First Input Session Meeting
46	User views website of a critical building by clicking on it	9	7	63	10	10	Yes	First Input Session Meeting
47	User views phone number for a critical building by clicking on it	10	7	70	10	9	Yes	First Input Session Meeting
48	User views what shelters are in a drawn polygon automatically	10	5	50	10	8	Yes	First Input Session Meeting
49	User chooses whether shelters and/or critical facilities will be displayed by clicking on checkboxes when a polygon is drawn	8	5	40	10	10	Yes	First Input Session Meeting
50	User views a shelters capacity, proximity to drawn polygon and contact information	10	6	60	10	9	Yes	First Input Session Meeting
51	User views shelters as soon as a hot zone is identified	10	5	50	10	8	Yes	First Input Session Meeting
52	User can view fire department according to the level of training they have, type of fire trucks and other resources	8	7	56	8	7	Yes	First Input Session Meeting
53	User views flood plain maps	10	9	90	10	10	Yes	Second Input Session Meeting
54	User views hazmat information regarding critical facilities	10	7	70	10	8	Yes	Second Input Session Meeting
55	User can access chat logs later	7	7	49	10	8	Yes	Second Input Session Meeting
56	User observes the color of the buildings which are in the hot zone change color	8	5	40	10	6	Yes	Second Input Session Meeting
57	User observes flood gauge measurements (prototype)	10	7	70	10	8	Yes	Second Input Session Meeting
58	User views base reflectivity	NA					Yes	Second Input Session Meeting
59	User accesses to hazmat information based on ERG number with a hyperlink on the critical facilities	10	7	70	10	8	Yes	Second Input Session Meeting
60	User views downrange isolation and protective distance of hazmat by specifying type of hazmat, the amount of the leak and the wind direction	10	6	60	10	8	Yes	Second Input Session Meeting
61	User observes the wind direction from the interface	10	1	10	10	9	Yes	Second Input Session Meeting

4.3. Implementation

After the user stories were collected, evaluated, prioritized a number of them were selected for each development cycle, the implementation phase began.

As a requirement of CMMI project management process area, issues and risks regarding the project were collected before the coding process. As shown at Table 4-2, nine issues or risks were identified before the start of first development cycle. Two items (issue #4 and #5) that first were classified as risks, were later converted to issues when they occurred.

TABLE 4-2 ISSUES AND RISKS THAT WERE RAISED BEFORE DEVELOPMENT CYCLES

ID		Description	Impact				Response			Follow Up				
	Classification			Severity	Probability (P)	Exposure (SxP)	Action Person	Mitigation Plan	Contingency Plan	Date Raised	Status	Details	Actions Taken	Actual Closure Date
1	Issue	**Insufficient response for Input Session:** Sufficient emergency managers may not respond for an input session	Cannot start the project	Moderate (3)	Likely (4)	12	Naci Dilekli	NONE	(1) Start the project without an input session (2) Wait until conducting an input session	9/25/2009	Occurred	(1) There was insufficient response from OK. 8 emergency managers across OK responded. Responders were scattered and it was not possible to get together easily	(1) An alternative meeting was arranged in AR on 10-15-09 (2) Another meeting with OK Emergency Managers was arranged on 10-20-09	10/20/2009
2	Issue	**Developer may not contact emergency managers (as mass contact) for user stories on time:** All the emergency managers are subscribed to an email list. Developer is not subscribed; he can only contact them through OCS.	Project Delay	Moderate (3)	Moderate (3)	9	Naci Dilekli, OCS	Communicate with OCS in advance to contact emergency managers	(1) Communicate with the emergency managers that developer contacted before	10/19/2009	Occurred	(1) User story request email was sent on 10-29-09 instead of 10-19-09 (2) Communications person said they won't be emailing anymore since they don't want to use the list frequently	Use the contingency plan	10/29/2009
3	Issue	**Delay in User Communications:** Since OCS contacts all the Emergency Managers, developer has no control over whether an email is sent or not	Project Delay	Minor (2)	Likely (4)	8	Naci Dilekli	NONE	(1) Contact emergency managers whose contact information has been acquired through input sessions, individual meetings or from the ones replying to previous emails	10/29/2009	Occurred	(1) User story request email was sent on 10-29-09 instead of 10-19-09 (2) Communications person said they won't be emailing anymore since they don't want to use the list frequently	Use the contingency plan	Has not been closed
4	Issue	**Insufficient response for User Stories:** Emergency Managers may not respond to the request for the user stories	More limited user input	Minor (2)	Likely (4)	8	Naci Dilekli	NONE	(1) Use requirements from input session only	10/29/2009	Occurred	Since the beginning of the project, only one emergency personnel sent a user story	Use the contingency plan	Has not been closed
5	Issue	**Insufficient response for Acceptance Testing:** Emergency Managers may not rchoose to do acceptance testing	More limited user input	Minor (2)	Very Likely (5)	10	Naci Dilekli	NONE	(1) Ask committee members for acceptance testings (2) Have a third party do acceptance testing (3) Skip acceptance testing	10/29/2009	Occurred	Since the beginning of the project, only no emergency personnel sent an acceptance testing	Use the contingency plan	Has not been closed
6	Risk	**Hardware Problem at CSA:** Malfunction in the computer or the internet connection at CSA	Cannot work at CSA	Serious (4)	Very Unlikely (1)	4	Naci Dilekli and Brian Hart	NONE	(1) Work at ODWC Fish Lab (2) Have the computer fixed	11/12/2009	Open			
7	Risk	**Software Problem at CSA:** Malfunction in any of these software: The Windows XP, Visual Studio.NET, ArcGIS Server	Cannot work at CSA	Serious (4)	Very Unlikely (1)	4	Naci Dilekli and Brian Hart	NONE	(1) Work at ODWC Fish Lab (2) Have the relevant software fixed	11/12/2009	Open			
8	Risk	**Hardware Problem at ODWC:** Malfunction in the computer or the internet connection at CSA	Cannot work at anywhere	Critical (5)	Very Unlikely (1)	5	Naci Dilekli and Greg Summers	NONE	(2) Have the computer fixed	11/12/2009	Open			
9	Risk	**Software Problem at ODWC:** Malfunction in any of these software: The Windows XP, Visual Studio.NET, ArcGIS Server	Cannot work at anywhere	Critical (5)	Very Unlikely (1)	5	Naci Dilekli and Greg Summers	NONE	(1) Have the relevant software fixed (2) Accept the risk	11/12/2009	Open			

4.3.1. First Development Cycle

The first development cycle of implementation took 70 hours. 397 lines of XAML code and 518 lines of C# code were written. Based on a meeting with Dr. McPherson, 26 user stories were selected as shown in Table 4-3 along with their corresponding parameters. This table contains the user stories evaluation form that is a requirement of CMMI Requirements Management process area. It is specifically specific requirement 1.1 within the process area.

Selected user stories were organized according to six tasks including managing layers, navigation, tracking management, sketching / selection management, action checklist and address book. For these tasks, then, a project planning document was created shown in Table 4-4. This form is needed for CMMI project planning process area, and is composed of a combination of smaller forms that were integrated into a single document for practicality. The project planning form includes task description, work estimate, project schedule and project resource plan information, which are required by CMMI Project Planning specific practices 1.1, 1.2, 2.1, 2.4, 2.5, 2.6 and 3.1. While it was not among the original project planning requirements, the developer decided to include actual costs and actual schedules in project planning document, to help see project delays for individual tasks. Also the solution alternatives and the rationale for the selected solution have been discussed in Table 4-5.

TABLE 4-3 USER STORIES EVALUATION FORM FOR FIRST DEVELOPMENT CYCLE

Task ID	Story ID	Story	Importance	Ease to Implement	Priority	Clarity	Completeness	Testable	Source
1	1	User zooms in to map	8	10	80	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
1	2	User zooms out of map	8	10	80	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
1	3	User pans across map	6	10	60	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
1	5	User zooms to the map extent	3	6	18	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
1	42	User orients the map by rotating it with a compass	3	6	18	10	10	Yes	First Input Session Meeting
2	16	User views topographic maps	8	10	80	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
2	17	User views land cover satellite imagery	8	10	80	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
2	18	User views land use maps	8	10	80	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
2	20	User views building floor plans	3	9	27	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
2	33	User toggles between different data source	10	8	80	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
2	40	User can view county names on the map	10	9	90	10	10	Yes	OK-FIRST User evalulations
2	41	User views names of towns	10	9	90	10	10	Yes	OK-FIRST User evalulations
3	12	User views locations of emergency vehicles on the map real time (prototype)	8	5	40	10	6	Yes	Dr. Rashed and I, and approved at first two input session meetings
3	13	User toggles between different types of emergency vehicles	5	5	25	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
3	14	User views locations of emergency managers on the map real time (prototype)	2	5	10	10	6	Yes	First Input Session Meeting
3	15	User views labels on top of emergency vehicles on the map	7	5	35	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
4	4	User identifies features (point selection)	9	10	90	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
4	23	User draws polygons on the fly during response	8	6	48	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
4	24	User views what critical facilities are in a drawn polygon automatically	9	5	45	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
4	25	User ranks the selected facilities based on an attribute	7	6	42	10	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
4	47	User views phone number for a critical building by clicking on it	10	7	70	10	9	Yes	First Input Session Meeting
4	48	User views what shelters are in a drawn polygon automatically	10	5	50	10	8	Yes	First Input Session Meeting
4	49	User chooses whether shelters and/or critical facilities will be displayed by clicking on checkboxes when a polygon is drawn	8	5	40	10	10	Yes	First Input Session Meeting
4	50	User views a shelters capacity, proximity to drawn polygon and contact information	10	6	60	10	9	Yes	First Input Session Meeting
5	26	User views a checklist for actions to do for certain events	9	8	72	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
6	45	User views the address book to contact people in critical facilities and other agencies	10	7	70	10	6	Yes	First Input Session Meeting

TABLE 4-4 PROJECT PLANNING FOR THE TASKS DEVELOPED IN FIRST DEVELOPMENT CYCLE

Task ID	Task Description	Relevant Requirements	Estimate Criteria	Size and Complexity of tasks and work products	Work Estimate	Project start and end dates	Actual Time It Took	Actual Date Started and Finished
1	This task involves development of navigation controls	1, 2, 3, 5, 42	Size of similar Silverlight projects	A similar application at resources.esri.com (http://resources.esri.com/help/9.3/arcgisserver/apis/silverlight/samples/start.htm#ToolBarWidget) involves a XAML file of 82 lines, and a C# file of 200 lines containing 6 classes. Zoom in, zoom out and pan can be automatically done by mouse input (with the use of roller). The zoom extents event is the only one that will require programming. The estimation then is it will take half the size of the similar application at resources.esri.com	5 hours	This task will be started on 11/16/09 and finished the same day	6 Hours	This task was started on 11/16/09 and finished the same day
2	Mapping data collection, display and management	16, 17, 18, 20, 33, 40, 41	Size of similar Silverlight projects	This task is data collection intense. Toggling between data sources requires a radiobutton control. A similar application (http://resources.esri.com/help/9.3/arcgisserver/apis/silverlight/samples/start.htm#SwitchMap). It is estimated the developed application will have a slightly larger size due to the more data services that are required application.	8 hours	This task will be started on 11/18/09 and finished on 11/19/09.	30 hours	Generated arbitrary floor plans on 16-11-09, generated arbitrary Critical Facilities or Shelters information 17-11-09, Added local services on 11/30/2009
3	Emergency vehicle and person tracking and management	12, 13, 14, 15	Size of similar Silverlight projects	An application at resources.esri.com to display graphical objects (http://resources.esri.com/help/9.3/arcgisserver/apis/silverlight/samples/start.htm#AddGraphics) involves a XAML file of 38 lines, and a C# file of 222 lines containing 6 functions. The application that will be developed will not have as many different objects as in this reference application. However, there needs to be an algorithm to randomly generate emergency vehicle and people locations. Besides, since user will toggle the types of emergency vehicles, it will add some more complexity. Overall, it is estimated the developed application will have somewhat larger size compared to the reference application.	8 Hours	This task will be started on 11/17/09 and finished on 11/19/09.	6 hours	This task was started on 11-19-09 and finished on 11-18-09
4	Mapping and managing critical facilities and shelters information according to user specified polygon	4, 23, 24, 25, 48, 49, 50	Size of similar Silverlight projects	A similar application (http://resources.esri.com/help/9.3/arcgisserver/apis/silverlight/samples/start.htm#SpatialQuery) involves a XAML of with 129 lines, and a C# file of 163 lines with 5 functions. The example does not include the ability to sort the records according to a polygon. In addition, there needs to be a checkbox to limit what type of buildings can be selected. Lastly, distances from the drawn polygon and the shelter or facility location needs to be calculated. It is expected that these requirements will make the size of the task 2-3 times the size of the referenced application	12 hours	This task will be started on 11/19/09 and finished on 11/20/09.	18 hours	This task was started on 12-1-09 and finished on 12-3-09
5	Checkbox for necessary actions	26	Personal experience based on .NET	This is a fairly simple task, requiring no GIS component. This requires setting up a data table for the necessary actions, and classifying them according to the hazard type and magnitude. It will require setting up two drop down menus and several functions to read from the data tables based on the selection.	6 hours	This task will be started on 11/23/09 and finished the same day	4 hours	This task was started on 12-3-09 and finished on 12-4-09
6	Address book	45	Personal experience based on .NET	This is a fairly simple task, requiring no GIS component. This requires setting up a data table for the contact information. There will be a listbox that the user can scroll to view the contact information.	4 hours	This task will be started on 11/24/09 and finished the same day	4 hours	This task was started on 12-3-09 and finished on the same day

Task ID	Task Description	Schedule Assumptions	Task Dependencies	Potential Risks	List of Managed Data	Schedule of collection of project data	Critical facilities and equipment list	Problems Encountered
1	This task involves development of navigation controls	There will not be any major interferences to this work	This task does not depend on any other tasks	None	None	NA	Computer with Visual Studio.NET, Silverlight for Visual Studio.NET, Silverlight API for ArcGIS Server and ArcGIS Server.	None
2	Mapping data collection, display and management	There will not be any major interferences to this work	This task does not depend on any other tasks	11	Topographic maps, landcover satallite imagery, landuse maps, building floor plans, counties and their names, towns and their names	Task implementation will start with the data generation	Computer with Visual Studio.NET, Silverlight for Visual Studio.NET, Silverlight API for ArcGIS Server and ArcGIS Server.	See issues 15, 16, 17, 18, 19, 20, 21, 22 in the Risks and Issues register
3	Emergency vehicle and person tracking and management	There will not be any major interferences to this work	This task does not depend on any other tasks	10	None	NA	Computer with Visual Studio.NET, Silverlight for Visual Studio.NET, Silverlight API for ArcGIS Server and ArcGIS Server.	Tracking Data Inavailability: Developer does not have access to a data feed for emergency vehicle locations or emergency people. Therefore Developer generated a random algorithm
4	Mapping and managing critical facilities and shelters information according to user specified polygon	There will not be any major interferences to this work	This task does not depend on any other tasks	12, 13	Sample or arbitrarily generated data	Task implementation will start with the data generation	Computer with Visual Studio.NET, Silverlight for Visual Studio.NET, Silverlight API for ArcGIS Server and ArcGIS Server.	Developer was not able to calculate distances from the drawn polygon and the shelter or facility location.
5	Checkbox for necessary actions	There will not be any major interferences to this work	This task does not depend on any other tasks	None	Sample or arbitrarily generated data	Task implementation will start with the data generation	Computer with Visual Studio.NET, Silverlight for Visual Studio.NET, Silverlight API for ArcGIS Server and ArcGIS Server.	None
6	Address book	There will not be any major interferences to this work	This task does not depend on any other tasks	None	Sample or arbitrarily generated data	Task implementation will start with the data generation	Computer with Visual Studio.NET, Silverlight for Visual Studio.NET, Silverlight API for ArcGIS Server and ArcGIS Server.	None

Use case diagrams were created as shown in Figure 4-1. Use case diagrams along with other product design documents such as activity diagrams are necessitated by CMMI, specifically in Technical Solution process area, specific practice 2.1.

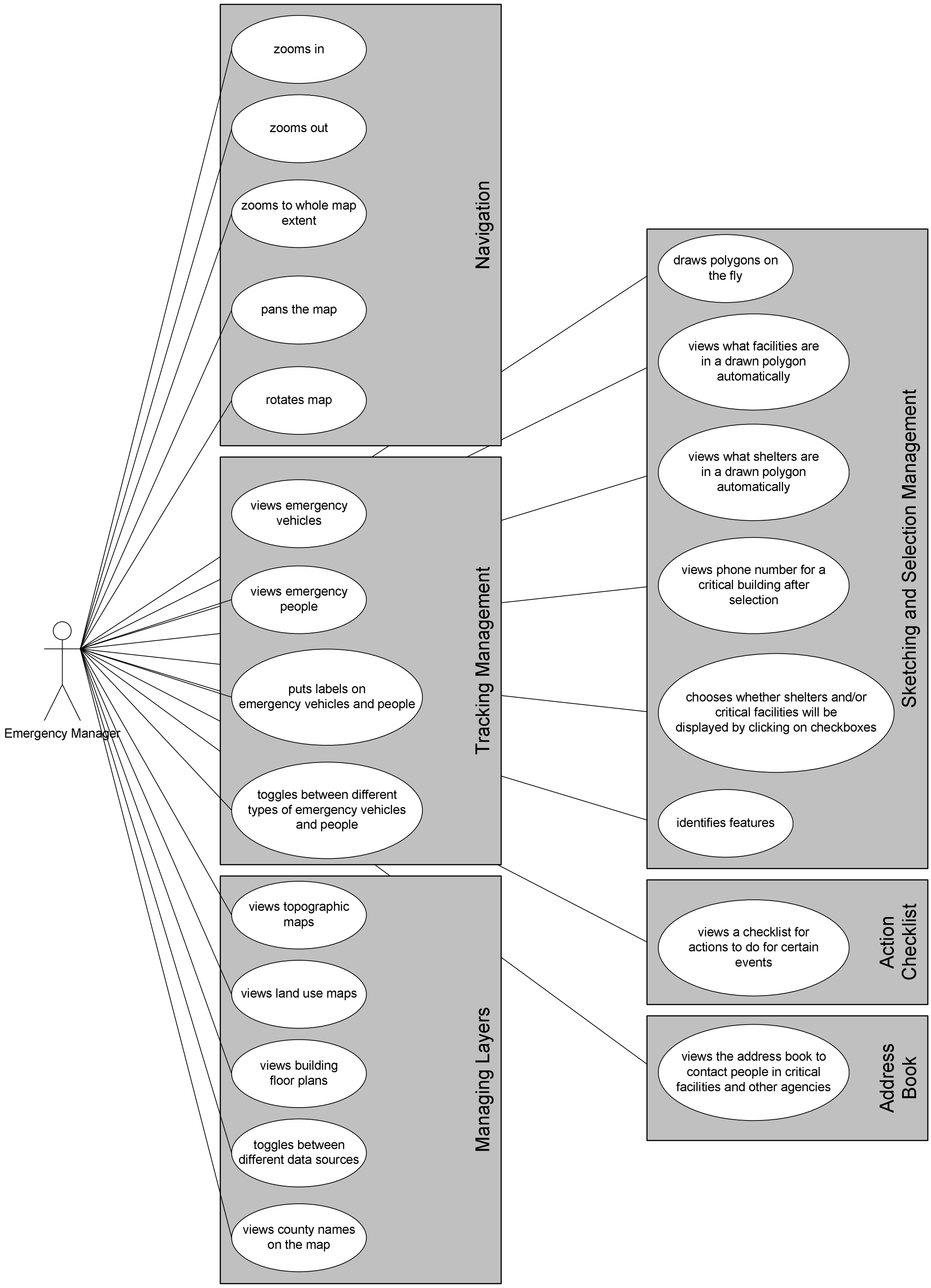

FIGURE 4-1 USER STORIES AND TASKS THAT WERE DEVELOPED IN THE FIRST DEVELOPMENT CYCLE

TABLE 4-5 SOLUTION ALTERNATIVES IN FIRST DEVELOPMENT CYCLE WITH SELECTED SOLUTIONS HIGHLIGHTED

User Story ID	Task ID	Task Title and Desciption	Solution Alternative	Cost estimate	Technical Limitations	Risks	Advantages and Disadvantages	Complexity
All	All	Determining the Platform for Development	Flex	Learning Flex syntax	NA	Learning may delay the actual development	Flex has been around for longer, there are more examples on ESRI website	In general, complexity of Flex and Silverlight are similar in the length of code and complexity
			Silverlight	Relatively shorter learning curve	NA	Silverlight Framework and Silverlight API have been released a lot more recently. Therefore	Programmer has familiarity with .NET, and Silverlight uses .NET framework for programming end	In general, complexity of Flex and Silverlight are similar in the length of code and complexity
Rationale for Selecting the Solution Alternative: Since the methodology of this study relies on extreme programming's short development cycle premise, developer has to adopt the fastest working solution. The programmer has familiarity with the .NET framework, therefore Silverlight solution alternative was chosen.								
1, 2, 3, 5, 42	1	This task involves development of navigation controls	Using the ESRI.ArcGIS.Client.Toolkit:Navigation	It already is in the ArcGIS for Silverlight API. 1 hour to integrate	NA	Customizing controls is not straightforward if needed	No development efforts are needed	If no mofifications are necessary, then there is no complexity regarding the use. If developer wants to make changes to the functionality, it will take time.
			Developing a navigation control from scratch	40 working hours	NA	It may not be accomadated within the available time frame	Developer will have a lot more control over the component	Very complex
Rationale for Selecting the Solution Alternative: It is available. No other components are available								
16, 17, 18, 20	2	Mapping data collection, display and management	Serving the data locally	Similar services exist locally, so it is a little more costly then the other alternative since it requires publishing the services on the server. 8 working hours	NA	It will increase the requests to the server. A lot of the required data is large raster data sets	In case ArcGIS data services are no longer available, the system as it is now will still work.	Complexity of the alternatives for this solution are similar
			Using outside data services whenever possible	Easy to implement. 6 working hours	NA	ArcGIS Services may not be online in the future	It reduces the load on the server	Complexity of the alternatives for this solution are similar
Rationale for Selecting the Solution Alternative: Both solutions are valid and selected depending on the situation. For server performance and ease of development outside services are preffered. If ArcGIS data services are discontinued, it is always possible to put the local services. Data regarding of critical facilities and shelters need to be local.								
12, 13, 14, 15	3	Emergency vehicle and person tracking and management	Installing GPS signal receivers and emitter devices, integrating them with the server and application	Vert costly, money-wise and time-wise. Not possible to make an estimate without a comprehensive analysis	No hardware available, bureaucracy, no platform to integrate into the silverlight system	It is likely that this would not be managed within the available time frame	Accurate and completely working system	Very complex
			Putting random points for showing the potential benefits of this functionality	Easier to implement. 10 working hours	None	It is not the real application	It gives an idea of the feature's usefulness without taking all the risks and taking on the costs	Far easier to implement than the other alternative
Rationale for Selecting the Solution Alternative: Due to the time frame and highly probable technical issues and risks, a prototype needed to be developed								
23, 24, 25, 48, 49, 50	4	Mapping and managing critical facilities and shelters information according to user specified polygon	Using the spatial query tool example, and modifying it	10 hours of development time for modification and integration	None	None	It seems like implemantation can be done with relatively little effort	Fairly complex
			Writing the code from scratch	30 working hours	None	None	Developer will have a lot more control over the implementation	Fairly complex
Rationale for Selecting the Solution Alternative: Due to the time frame and the risks, a prototype will be developed								
26	5	Checkbox for necessary actions	Writing the code from scratch	6 working hours	None	None	Developer will have a lot more control over the implementation	Fairly simple
			Using an existing application	Unknown hours	None	A short survey did not reveal any similar applications	Shorter development and integration time	Fairly simple
Rationale for Selecting the Solution Alternative: A short survey did not reveal any similar applications, so developer decided to implement the code from scratch instead								
45	6	Address book	Writing the code from scratch	5 working hours	None	None	Developer will have a lot more control over the implementation	Fairly simple
			Using an existing application	Unknown hours	None	A short survey did not reveal any similar applications	Shorter development and integration time	Fairly simple
Rationale for Selecting the Solution Alternative: A short survey did not reveal any similar applications, so developer decided to implement the code from scratch instead								

For visual organization of tasks in the, a simple interface design schema was designed as shown in Figure 4-2. This was required by CMMI Technical Solution process area, specific practice 2.3. While originally, navigation controls were planned to be placed on the lower left corner, however due to insufficient space in the interface during development, they were placed on the lower right corner.

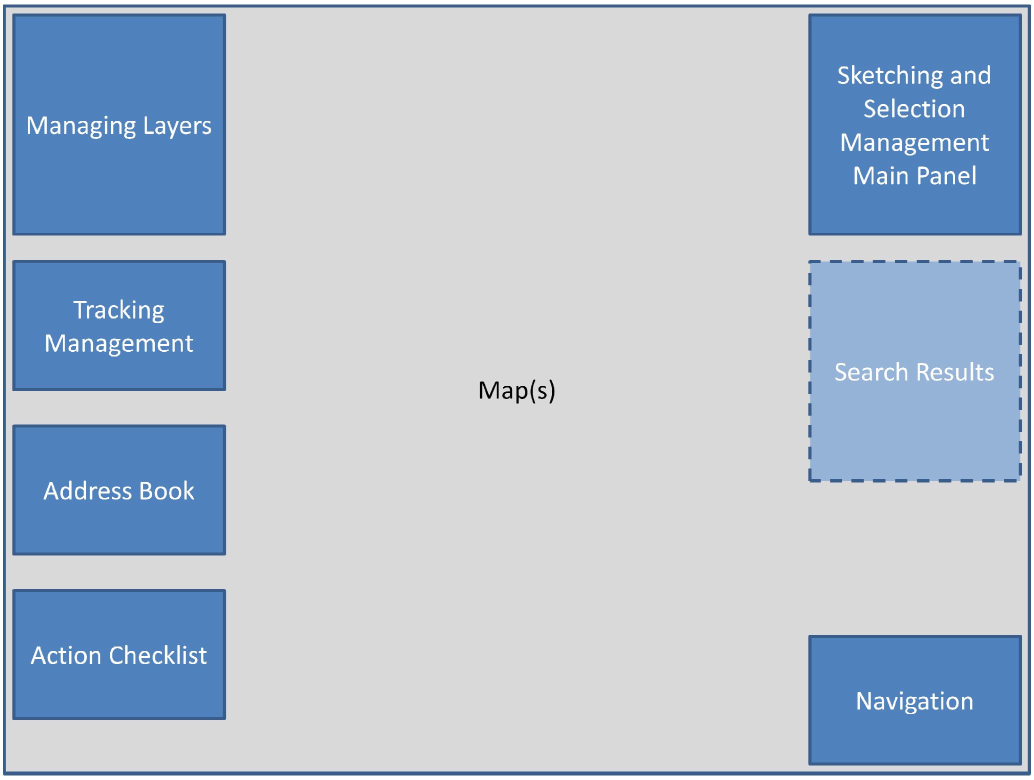

FIGURE 4-2 INTERFACE DESIGN FOR FIRST DEVELOPMENT CYCLE

Navigation (Task #1)

This task refers to the following user stories that were identified in the input sessions in the project.

- User zooms in to map (User story #1)
- User zooms out of map (User story #2)
- User pans across map (User story #3)
- User zooms to the map extent (User story #5)
- User rotates map (User story #42)

A navigation tool was placed on the application using XAML code, simply using the Navigation component in ESRI ArcGIS Silverlight Toolkit.

Managing Layers (Task #2)

This task refers to the following user stories that were identified in the input sessions in the project.

- User views topographic maps (User story #16)
- User views land cover satellite imagery (User story #17)
- User views building floor plans (User story #20)
- User toggles between background maps (User story #33)
- User can view county names on the map (User story #40)
- User views names of towns (User story #41)

Layer management panel is important from situational awareness point of view. Using this panel, that user can opt to decrease the amount of detail and complexity of data to avoid data overload and complexity creep, which are deterrents of situational awareness.

A layer can be turned on and off by the checkbox on the left. Transparency of a layer can be adjusted by the slide bar in the middle. At the end of first development cycle, there were 7 layers available for managing, including:

- Background Layer: A layer showing either the street map, topographical map or the satellite imagery. All the background layers are retrieved from ArcGIS online maps, as rest services.
- Counties: A layer showing the boundaries and names of counties in Oklahoma.
- Buildings: A layer showing the buildings in Norman, OK.
- Critical Facilities and Shelters: A prototype layer showing the critical facilities (Police stations, fire stations, schools, hospitals and shelters) in Norman, OK. This layer was generated by arbitrarily assigning some buildings in buildings layer as critical facilities and shelters.
- Building Plans: A prototype layer showing the building plans of the critical facilities in Norman, OK. A single building plan was drawn first and it was resized, rotated and placed over each of the critical facilities and shelters.
- Tracking Layer: A prototype graphics layer showing the locations for emergency vehicle and people locations. It was designed to display fire vehicles, police vehicles and field responders. In ArcGIS Server, a graphics layer only displays dynamic graphics that are generated in runtime.
- Sketching Layer: A graphics layer that controls the visibility of the sketching graphics. Sketching is done through the top right panel.

FIGURE 4-3 MANAGE LAYERS PANEL

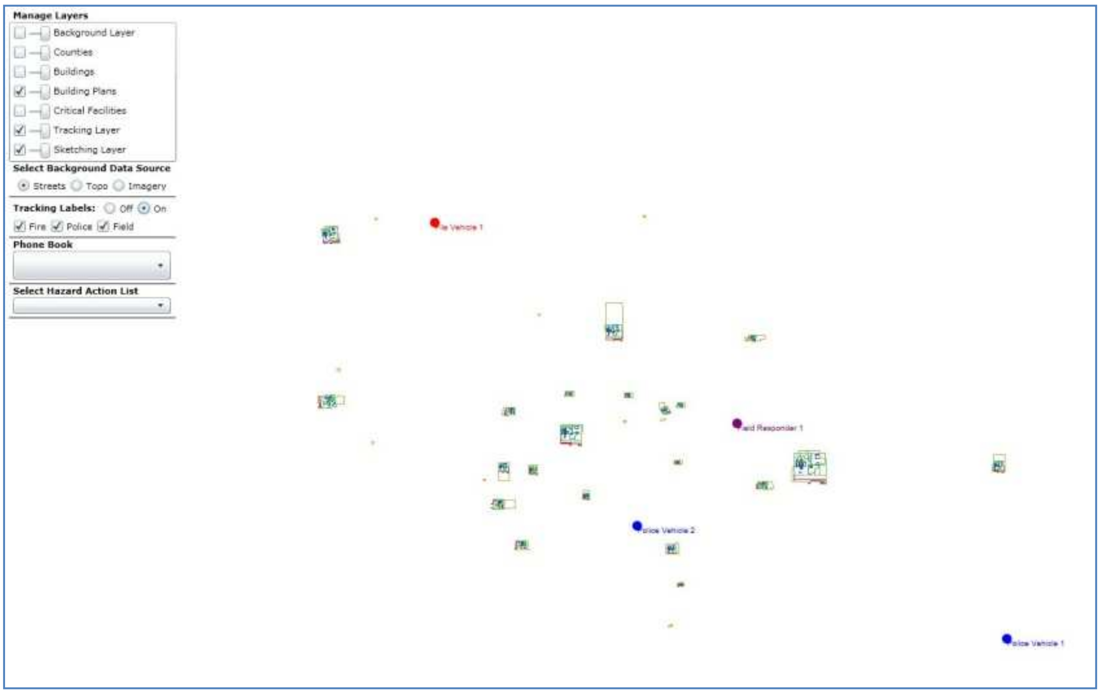

FIGURE 4-4 APPLICATION INTERFACE WITH SOME LAYERS TURNED OFF

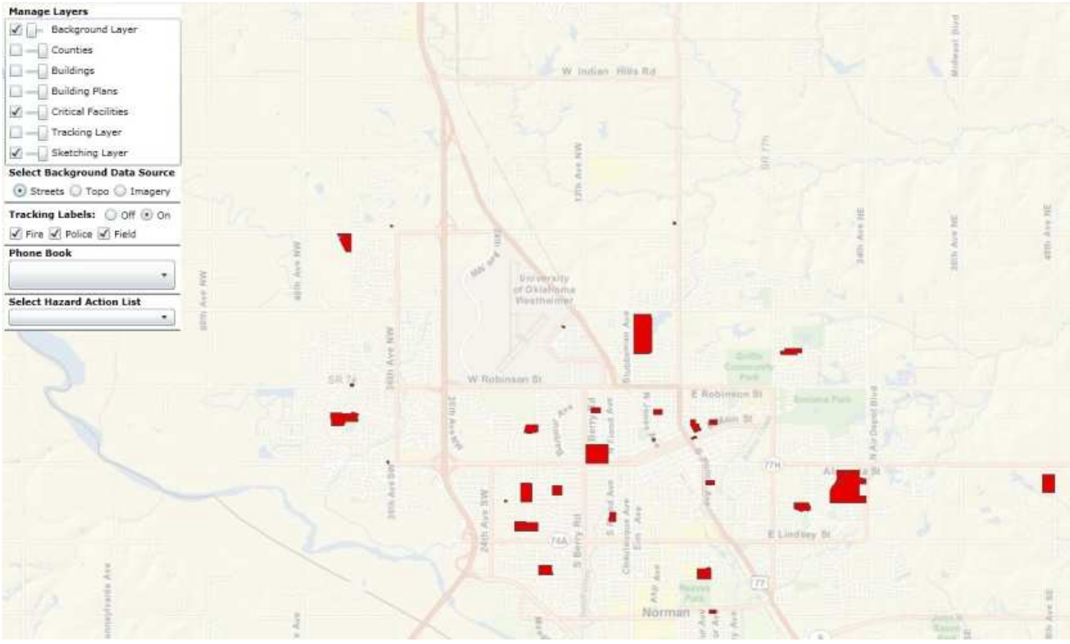

FIGURE 4-5 TRANSPARENCY FOR THE BACKGROUND LAYER ADJUSTED

The ability to turn on and off layers as in Figure 4-3 **Manage Layers Panel**Figure 4-3 lets user to focus on certain spatial information, and filter out other information that may distract him/her. This ability is exemplified in Figure 4-4 with turning off all the layers except building plans and tracking features of emergency vehicles and personnel. The ability to adjust transparency as in Figure 4-5 is one of alternative methods to focus on certain features while decreasing the visual salience of certain other spatial features.

Tracking Management (Task #3)

This task refers to the following user stories that were identified in the input sessions in the project.

- User views locations of emergency vehicles on the map real time (prototype) (User Story #12)
- User toggles between different types of emergency vehicles (User Story #13)
- User views locations of emergency managers on the map real time (prototype) (User Story #14)
- User views labels on top of emergency vehicles on the map (User Story #15)

This task was developed as a prototype. The purpose is to bring live spatial information alongside static spatial data to allow users to have a more complete operational picture. Inclusion of live elements is very important from a situational awareness oriented design, because the first two levels of situational awareness which are perception and comprehension of current situation require provision of real time or near real time information. This information is later used to achieve the third and last level of situational awareness, which is projection of future situation. Live spatial information is also introduced in third development cycle, with live cameras and real time weather data features.

Ideally, there would be vehicles with signal emitters sending GPS coordinates of the vehicle to a server to locate them on the application. With this application, the movements of virtual vehicles and emergency personnel on the map were made to be random.

Sketching and Selection Management (Task #4)

This task refers to the following user stories that were identified in the input sessions in the project.

- User identifies spatial features by point selection (User Story #4)
- User draws polygons on the fly during response (User Story #23)
- User views what facilities are in a drawn polygon automatically (User Story #24)
- User ranks the selected facilities based on an attribute (User Story #25)
- User views phone number for a critical building by clicking on it (User Story #47)
- User views what shelters are in a drawn polygon automatically (User Story #48)
- User chooses whether shelters and/or critical facilities will be displayed by clicking on checkboxes when a polygon is drawn (User Story #49)
- User views a shelters capacity, proximity to drawn polygon and contact information (User Story #50)

One of the major changes from the initial user story to the actual implementation in this development cycle was the omission of proximity to drawn polygon as shown in. During the implementation developer decided it was not possible or it would take too much effort for the development cycle. The activity diagram for this task was illustrated by Figure 4-6.

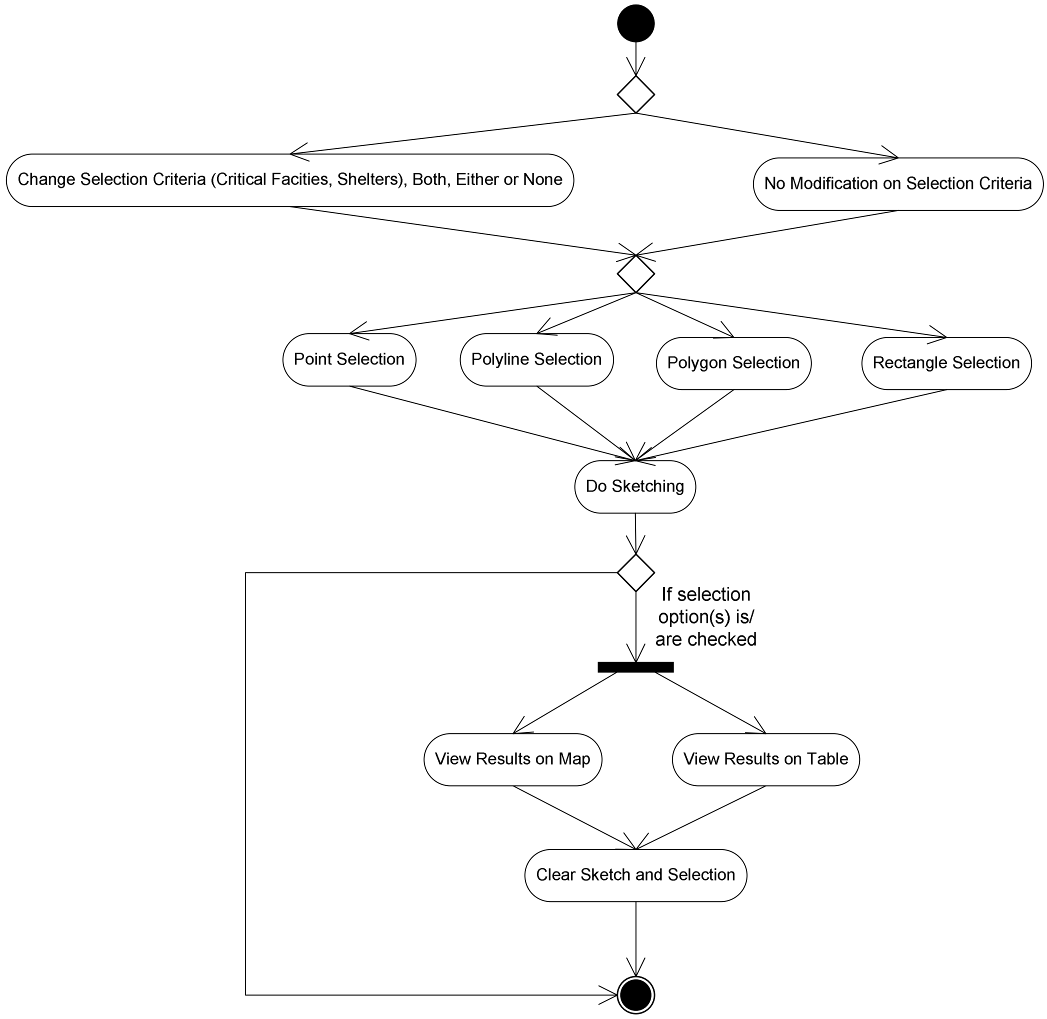

FIGURE 4-6 ACTIVITY DIAGRAM FOR SKETCHING AND SELECTION

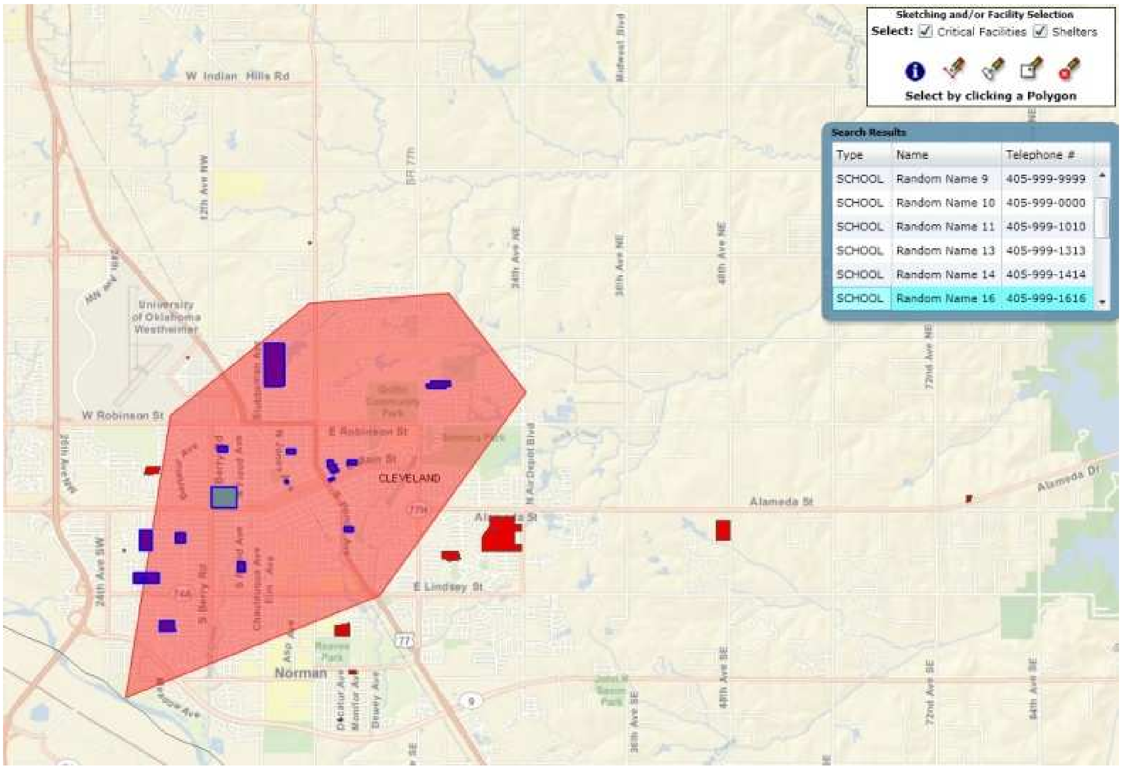

FIGURE 4-7 SKETCHING A FACILITY SELECTION

Figure 4-7 shows an example of using this feature, revealing the graphical sketching, selection and displaying the attributes.

Action Checklist (Task #5)

This task refers to the following user story that was identified in the input sessions in the project.

- User views an action checklist (User Story #7)

This was developed as a prototype task to list a number of actions an emergency manager needs to take when user specifies a certain situation, e.g. a hazard. This task is shown in Figure 4-8.

Address Book (Task #6)

This task refers to the following user stories that were identified in the input sessions in the project.

- User views the address book (User Story #45)

This was developed as a prototype task to see the contact information of necessary entities an emergency manager needs to contact. This task is shown in Figure 4-9.

With the development of the Address Book task, the planned implementation part of first development cycle has ended as seen in Figure 4-10.

FIGURE 4-8 SOME HAZARD ACTION LIST ITEMS CHECKED

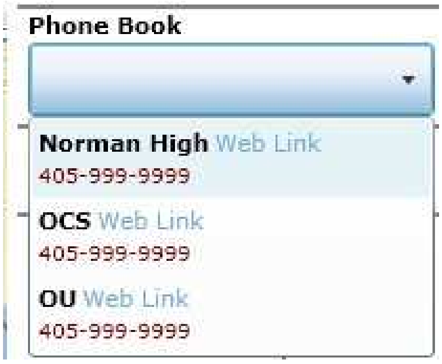

FIGURE 4-9 VIEWING PHONEBOOK ITEMS

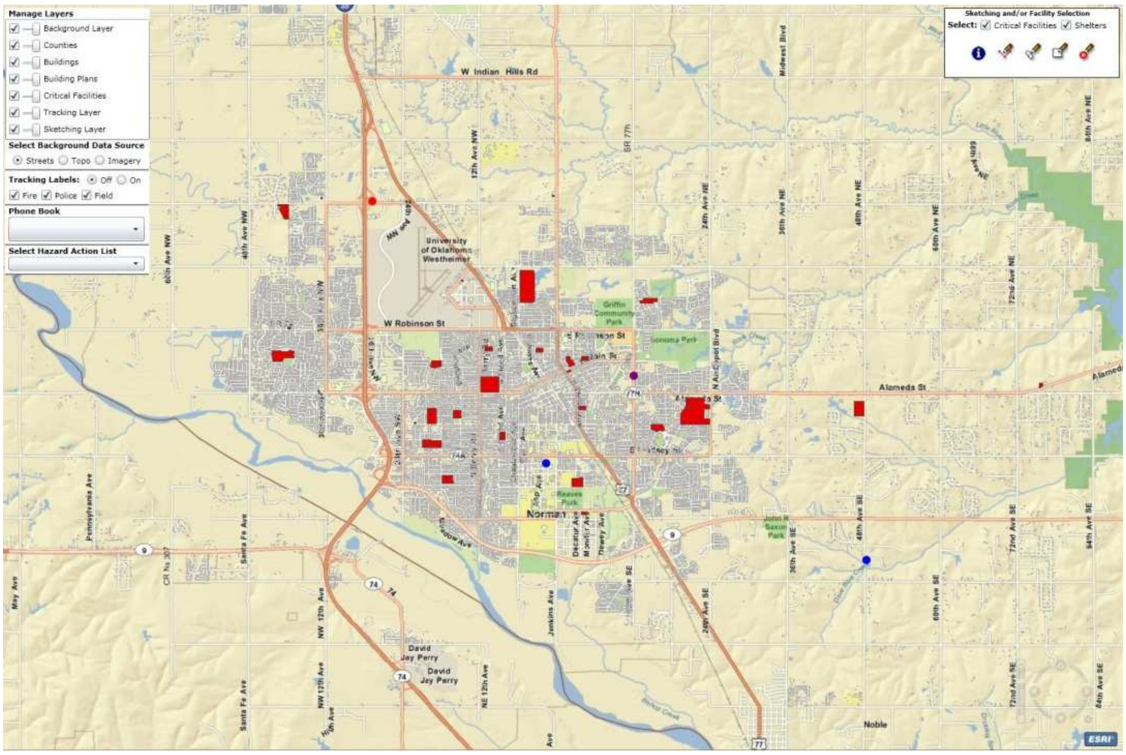

FIGURE 4-10 GENERAL VIEW OF THE APPLICATION AFTER FIRST CYCLE WAS FINISHED

Evaluation of the First Development Cycle

While the application worked well at the local development machine, the developer failed to foresee any integration issues that might arise when transferring the application on an actual server that would host the application to public access. These problems arose towards the end of the first development cycle and caused a delay in the project.

These problems, as explained in Table 4-7, were unseen in advance as the development was done on a local machine with the ASP.NET Development Server since it was more comfortable for the developer to work on a local machine. The planned and actual schedules are shown in Table 4-6. The operator manual for the second development cycle is in Appendix 6.1 and the user manual is in Appendix 6.4.

TABLE 4-6 FIRST DEVELOPMENT CYCLE SCHEDULE IN DETAIL

Task Name	Baseline Estimated Duration	Baseline Estimated Start	Baseline Estimated Finish	Actual Duration	Actual Start	Actual Finish	Duration Variance	Start Variance	Finish Variance
First Cycle	**16 days**	**Wed 9/30/09**	**Thu 10/22/09**	**18 days**	**Thu 11/12/09**	**Mon 12/7/09**	**2 days**	**31 days**	**32 days**
Project Planning (CMMI) / Release Planning (XP)	**0.5 days**	**Tue 10/6/09**	**Tue 10/6/09**	**1 day**	**Thu 11/12/09**	**Thu 11/12/09**	**0.5 days**	**27 days**	**27 days**
Task Description Document (CMMI)	0.5 days	Tue 10/6/09	Tue 10/6/09	1 day	Thu 11/12/09	Thu 11/12/09	0.5 days	27 days	27 days
Detailed Work Estimate Document (CMMI)	0.5 days	Tue 10/6/09	Tue 10/6/09	1 day	Thu 11/12/09	Thu 11/12/09	0.5 days	27 days	27 days
Detailed Project Schedule (CMMI)	0.25 days	Tue 10/6/09	Tue 10/6/09	1 day	Thu 11/12/09	Thu 11/12/09	0.75 days	27 days	27 days
Data Management Plan	0.38 days	Tue 10/6/09	Tue 10/6/09	1 day	Thu 11/12/09	Thu 11/12/09	0.63 days	27 days	27 days
Project Resource Plan (CMMI)	0.13 days	Tue 10/6/09	Tue 10/6/09	1 day	Thu 11/12/09	Thu 11/12/09	0.88 days	27 days	27 days
Technical Solution (CMMI) / Iteration (XP)	**11.25 days**	**Tue 10/6/09**	**Wed 10/21/09**	**17 days**	**Fri 11/13/09**	**Mon 12/7/09**	**5.75 days**	**28 days**	**33 days**
Solution Alternatives Document (CMMI)	0.38 days	Tue 10/6/09	Tue 10/6/09	1 day	Fri 11/13/09	Fri 11/13/09	0.63 days	28 days	28 days
Product Component Solutions Document (CMMI)	0.38 days	Tue 10/6/09	Tue 10/6/09	1 day	Fri 11/13/09	Fri 11/13/09	0.63 days	28 days	28 days
Product Design Document (CMMI)	0.38 days	Tue 10/6/09	Tue 10/6/09	1 day	Fri 11/13/09	Fri 11/13/09	0.63 days	28 days	28 days
Implementation / Coding	10 days	Tue 10/6/09	Tue 10/20/09	15 days	Sat 11/14/09	Fri 12/4/09	5 days	29 days	33 days
User Manual (CMMI)	0.38 days	Tue 10/20/09	Tue 10/20/09	1 day	Mon 12/7/09	Mon 12/7/09	0.63 days	34 days	34 days
Sent 1st Iteration to customers	2 days	Thu 10/22/09	Fri 10/23/09	1 day	Mon 12/7/09	Mon 12/7/09	-1 day	32 days	31 days

TABLE 4-7 RISK AND ISSUE REGISTERS FOR THE FIRST DEVELOPMENT CYCLE

ID		Description	Impact				Response			Follow Up				
	Classification			Severity	Probability (P)	Exposure (SxP)	Action Person	Mitigation Plan	Contingency Plan	Date Raised	Status	Details	Actions Taken	Actual Closure Date
10	Issue	**Tracking Data Inavailability:** Developer does not have access to a data feed for emergency vehicle locations or emergency people	Cannot implement the feature completely	Moderate (3)	Very Likely (5)	15	Naci Dilekli	No realistic mitigation plan can be devised	(1) Develop an algorithm to produce random locations over time to produce a prototype (2) Accept the risk	11/12/2009	Closed		Generated a random algorithm	11/19/2009
11	Issue	**Floor Plans Inavailability:** Developer does not have the floor plans for the actual buildings	Cannot implement the feature completely	Moderate (3)	Very Likely (5)	15	Naci Dilekli	No realistic mitigation plan can be devised	(1) Use a few example floor plans for prototyping (2) Accept the risk	11/12/2009	Closed		Put arbitrary floor plans	11/16/2009
12	Issue	**Critical Facilities or Shelters information Inavailability:** Developer does not have the critical facilities or shelters locations	Cannot implement the feature completely	Moderate (3)	Very Likely (5)	15	Naci Dilekli	No realistic mitigation plan can be devised	(1) Use a few example or randomly drawn critical facilities and shelters for prototyping (2) Accept the risk	11/12/2009	Closed		Put arbitrary information	11/17/2009
13	Issue	**Cannot sort GraphicAttributeColumn:** ArcGIS Silverlight API's (V.1.1) GraphicAttributeColumn cannot be sorted	Cannot sort the records according to attributes	Critical (5)	Very Likely (5)	25	Naci Dilekli	NONE	(1) Create another table and use MS's GraphicAttributeColumn instead for sorting (2) Accept the risk	11/12/2009	Open		Excluded the sort functionality so that benefits of GraphicAttributeColumn could be used (such as highligting of features)	11/19/2009
14	Issue	**Integration of Separate Applications:** It takes considerable time to put together all the separate applications.	Cannot finish on time	Moderate (3)	Very Likely (5)	15	Naci Dilekli	NONE	(1) Extend the development cycle	11/19/2009	Closed		Took the effort to integrate the separate applications	11/24/2009
15	Issue	**Local services could not be added to the Silverlight application:** This caused errors on the run time	Cannot implement the feature completely	Critical (5)	Very Likely (5)	25	Naci Dilekli	NONE	(1) Look for the error messages and find the solution (2) Do the implementation without any local services (decreasing functionality dramatically)	11/25/2009	Closed		It was found out that the server had to be added as a GIS Server with a www.webaddress.com address rather than defining it as a localhost	11/26/2009
16	Issue	**The URL option could not be added under GIS servers:** This was necessary for adding the local services	Cannot implement the feature completely	Critical (5)	Very Likely (5)	25	Naci Dilekli	NONE	(1) Look for the error messages and find the solution (2) Do the implementation without any local services (decreasing functionality dramatically)	11/26/2009	Closed	There was a problem adding local maps on the application. They would not be displayed on the application	(1) Register agsx extension in IIS (2) Do an Arcgis Server Web Post Install	11/26/2009
17	Issue	**Issue # 16 persisted:** A solution could not be found	Cannot implement the feature completely	Critical (5)	Very Likely (5)	25	Naci Dilekli	NONE	(1) Look for the error messages and find the solution (2) Do the implementation without any local services (decreasing functionality dramatically)	11/26/2009	Closed	Developer browsed the ESRI user forums for 4 hours and checked 16 forum threads for possible solutions. Tried everything on the specific thread with http://forums.esri.com/Thread.asp?c=158&f=1702&t=206995&mc=5 68 posts.	called esri support 12/1/09 (incident #: 773416). walked though the problem. it turned out i needed to enter C:\Program Files (x86)\ArcGIS\DotNet>adfutil -registerservices arcgis. I saw it on the user forum as well. However I copied and pasted it and it did not work. Manual	11/26/2009
18	Issue	**Silverlight Server Configuration 1:** Got a Silverlight configutation related problem	Cannot implement the system	Critical (5)	Very Likely (5)	25	Naci Dilekli	NONE	(1) Look for the error messages and find the solution (2) Do the implementation without any local services (decreasing functionality dramatically)	11/27/2009	Closed	Got the "A security exception occured while trying to connect to the REST endpoint. Make sure you have a cross domain policy file available at the root for your server that allows for requests from this application." Error. It turns out a silverlight project will not just work on an IIS. It will need some configuration	Add mime types (based on directions on http://learn.iis.net/page.aspx/262/configuring-iis-for-silverlight-applications/)	11/27/2009
19	Issue	**Silverlight Server Configuration 2:** Got a Silverlight configutation related problem	Cannot implement the system	Critical (5)	Very Likely (5)	25	Naci Dilekli	NONE	(1) Look for the error messages and find the solution (2) Do the implementation without any local services (decreasing functionality dramatically)	11/27/2009	Closed	debugging failed because integrated windows authentication is not enabled. when trying to use the actual IIS instead of the visual studio development environment	enable integrated windows authentication	11/27/2009
20	Issue	**Silverlight Server Configuration 3:** Got a Silverlight configutation related problem	Cannot implement the system	Critical (5)	Very Likely (5)	25	Naci Dilekli	NONE	(1) Look for the error messages and find the solution (2) Do the implementation without any local services (decreasing functionality dramatically)	11/27/2009	Closed	Could not download the silverlight application, check the web server settings: when trying to host silverlight application on the localhost	solved by resetting IIS	11/27/2009
21	Issue	**Issue #15 persisted despite of solving issues #19, #20, and #21.**	Cannot implement the system	Critical (5)	Very Likely (5)	25	Naci Dilekli	NONE	(1) Look for the error messages and find the solution (2) Do the implementation without any local services (decreasing functionality dramatically)	11/30/2009	Closed		(1) Called ESRI support: It turned out the actual problem was different. I was using the soap endpoint instead of rest	11/30/2009
22	Issue	**Issue #15 persisted despite of tying to add REST Services:** current rest endpoint at the arcgis server instance was not readable	Cannot implement the system	Critical (5)	Very Likely (5)	25	Naci Dilekli	NONE	(1) Look for the error messages and find the solution (2) Do the implementation without any local services (decreasing functionality dramatically)	12/1/2009	Closed		I had to create a new arcgis server instance (arcgis2) so that the end point is readable	12/1/2009
23	Issue	**Layer Management Problem:** It is not possible to turn on the layers on the fly	Not user friendly	Minor (2)	Very Likely (5)	10	Naci Dilekli	NONE	(1) Design a layer management panel (2) Don't do anything	12/1/2009	Closed		Added the manage layers panel	12/1/2009
24	Issue	**Spatial Query Failed 1:** Error "Query failed: ESRI.ArcGIS.Client.	Cannot implement the feature completely	Critical (5)	Very Likely (5)	25	Naci Dilekli	NONE	(1) Look for the error messages and find the solution (2) Do the implementation without the spatial query functionality	12/1/2009	Closed		I had to specify the specific layer for the query http://fish-atlas.ou.edu/ArcGIS2/rest/services/CriticalBuildings/MapServer/0. '0' being the first and the only layer. Unfortunately this was explicit enough for the developer in the ESRI resources help	12/1/2009
25	Issue	**Spatial Query Failed 2:** Query fails when there is another layer	Cannot implement the feature completely	Critical (5)	Very Likely (5)	25	Naci Dilekli	NONE	(1) Look for the error messages and find the solution (2) Do the implementation without the spatial query functionality	12/1/2009	Closed		After 6 hours of research and trials, developer realized that if not all the layers in the application have the same spatial reference, query will fail (no dynamic graphic to	12/3/2009
26	Issue	**Borders are not available for Stackpanels in Silverlight :** A border can only be put on a canvas	Not user friendly	Minor (2)	Very Likely (5)	10	Naci Dilekli	NONE	(1) Use canvases instead (2) Put lines in between the stackpanels	12/4/2009	Closed		So developer decided to put lines between panels. Additional margins were put as well so that the panel features could be better distinguished	12/4/2009
27	Issue	**Too Much Project Delay:** The amount of time spent on the first exceeded the planned amount very much.	Loss of time for unnecessary documentation	Minor (2)	Very Likely (5)	10	Naci Dilekli	NONE	(1) Write the documentation anyway (2) Cancel unccessary documentation	12/7/2009	Closed	In the first development phase it was seen that, for almost all the solutions, there is only one component available and therefore it is pointless to document component alternatives.	Developer decided to drop the components related documentation, and regard any components that may come across in the future as solution alternatives.	12/7/2009
28	Issue	**Problem with calculating distances from the drawn polygon and the shelter location:** It did not look like the current Silverlight API supported such functionality	Cannot implement the feature completely	Moderate (3)	Very Likely (5)	15	Naci Dilekli	NONE	(1) Don't do anything	12/8/2009	Closed	A quick survey on ESRI forums and developer resources website did not reveal any potential solutions to calculate those distances.	Developer decided omit this feature given the already delayed cycle schedule.	12/8/2009

All the issues that were faced during the development of first development cycle are shown in Table 4-7.

4.3.2. Second Development Cycle

The second development cycle of implementation took 66 hours. The total lines of XAML code were 569. The total lines of C# code were 1109. Based on another meeting with Dr. McPherson, it was decided that uploading user generated content (such as emergency plans) was a priority. Accordingly, developer selected 6 user stories (as seen in Table 4-8) that were seen as feasible to implement. These user stories were organized under 2 tasks as a project planning was done for this development cycle as shown in Table 4-9.

Supported tasks include zooming to bookmarked spatial features and emergency plans management. For these tasks, then, a project planning document was created shown in Table 4-9. The solution alternatives and the rationale for the selected solution have been discussed in Table 4-10.

TABLE 4-8 USER STORIES SELECTED FOR SECOND DEVELOPMENT CYCLE

Task ID	Story ID	Story	Importance	Ease to Implement	Priority	Clarity	Completeness	Testable	Source
7	6	User zooms to the bookmarked features using a dropdown menu	6	8	48	10	10	Yes	Tarek and Me approved by AR and OK Input Session Meetings
7	44	User views critical facilities by selecting them from a drop down menu	8	8	64	10	10	Yes	First Input Session Meeting
8	7	User draws a hazard response plan	10	1	10	8	4	Yes	Tarek and Me approved by AR and OK Input Session Meetings
8	8	User edits a hazard response plan	10	1	10	8	4	Yes	Tarek and Me approved by AR and OK Input Session Meetings
8	9	User shares hazard response plans	10	1	10	8	8	Yes	Tarek and Me approved by AR and OK Input Session Meetings
8	10	User erases hazard response plans	10	1	10	8	10	Yes	Tarek and Me approved by AR and OK Input Session Meetings

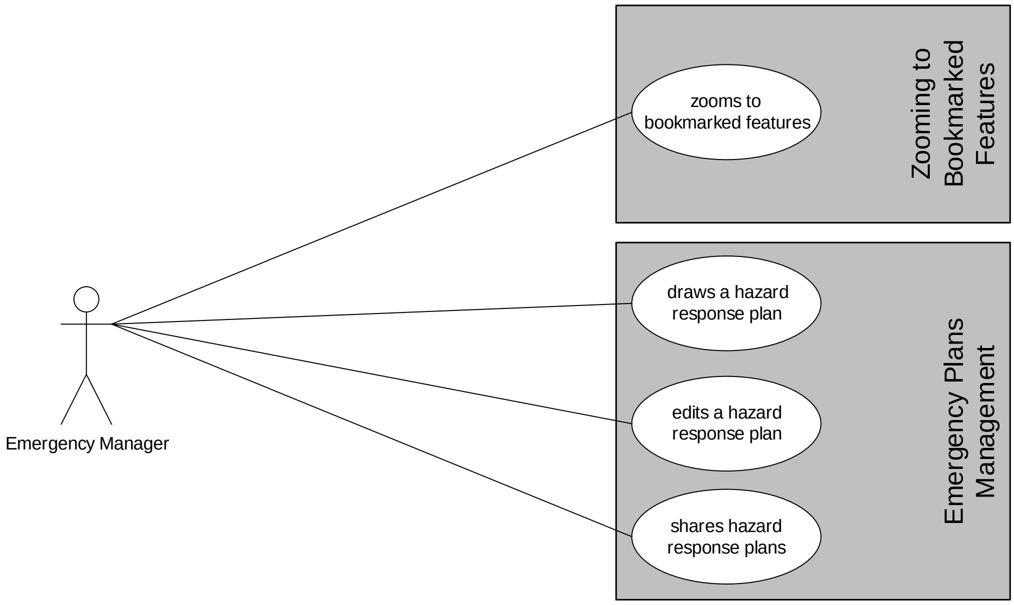

FIGURE 4-11 USE CASE DIAGRAM FOR USER STORIES AND TASKS THAT WERE DEVELOPED IN THE SECOND DEVELOPMENT CYCLE

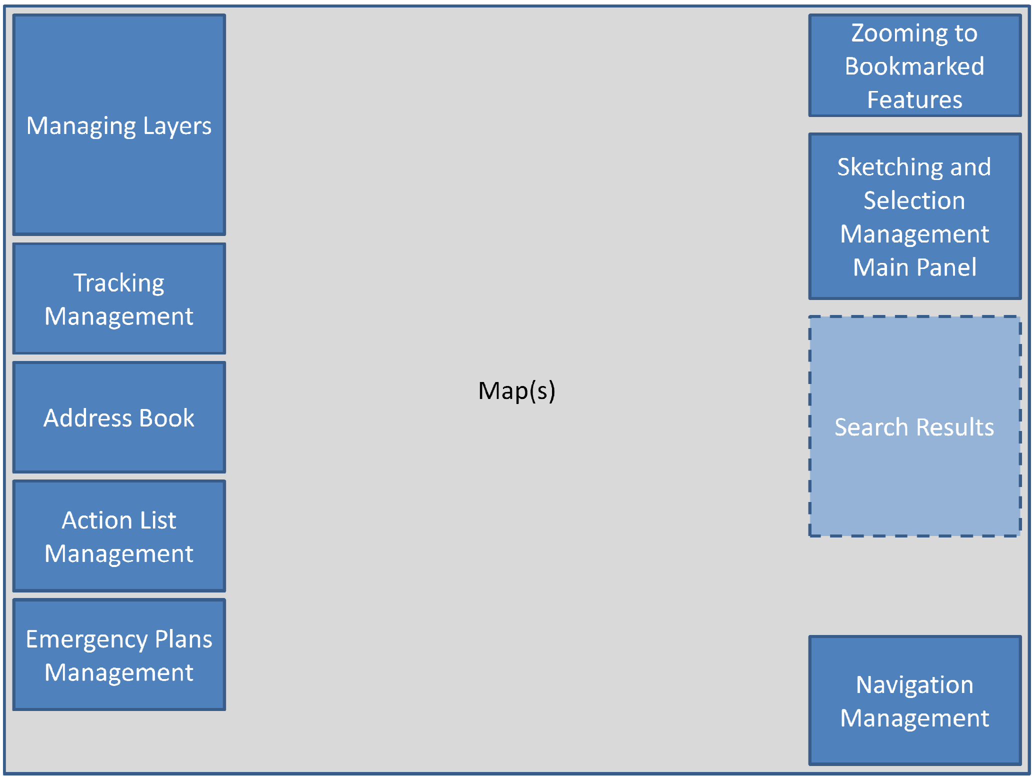

FIGURE 4-12 INTERFACE DESIGN FOR SECOND DEVELOPMENT CYCLE

TABLE 4-9 PROJECT PLANNING FOR THE TASKS DEVELOPED IN SECOND DEVELOPMENT CYCLE

Task ID	Task Description	Relevant Requirements	Estimate Criteria	Size and Complexity of tasks and work products	Work Estimate	Project start and end dates	Actual Time It Took	Actual Date Started and Finished
7	User zooms to the bookmarked features using a dropdown menu	6, 44	Experience from previous cycle	This tasks requires creating a dropdown menu, linking it with the REST service, and placing an event handler when the dropdown menu index is changed	4 hours	This task will be started on 12/17/09 and finished the same day	6 hours	This task was started on 12/17/09 and finished the same day
8	User manages hazard response plans	7, 8, 9, 10	Experience from previous cycle	This task requires writing a fair amount of new code. Developer did not have a clear strategy to develop this feature in the planning process.	80 hours	This task will be started on 12/17/09 and finished on 12/30/09	60 hours	This task was started on 12/18/09 and finished on 12/28/09

Task ID	Task Description	Schedule Assumptions	Task Dependencies	Potential Risks	List of Managed Data	Schedule of collection of project data	Critical facilities and equipment list	Problems Encountered
7	User zooms to the bookmarked features using a dropdown menu	There will not be any major interferences to this work	This task does not depend on any other tasks	None	Critical facilities data (arbitrarily generated)	Task implementation will start with the data generation	Computer with Visual Studio.NET, Silverlight for Visual Studio.NET, Silverlight API for ArcGIS Server and ArcGIS Server.	Graphic Selection Problem (Issue 30)
8	User manages hazard response plans	There will not be any major interferences to this work	This task does not depend on any other tasks	None	None	NA	Computer with Visual Studio.NET, Silverlight for Visual Studio.NET, Silverlight API for ArcGIS Server and ArcGIS Server.	None

TABLE 4-10 SOLUTION ALTERNATIVES IN THIRD DEVELOPMENT CYCLE WITH SELECTED SOLUTIONS HIGHLIGHTED

User Story ID	Task ID	Task Title and Desciption	Solution Alternative	Cost estimate	Technical Limitations	Risks	Advantages and Disadvantages	Complexity
6, 44	7	Zooming to bookmarked features using a dropdown menu	Writing the code from scratch	4 hours	None	None	Short development time	Very simple
Rationale for Selecting the Solution Alternative: There was only one alternative to choose from								
7, 8, 9, 10	8	Creating, editing and sharing hazard response plans	Writing an application from scratch to create and modify features within the spatial database online.	Unknown hours	It may not be technically possible to develop this solution alternative with Silverlight API	While this is possible doing in .NET Web Application Developer Framework, there were not any available samples to do this using the Silverlight API. It may be not possible to develop due to complexity or may not be accomadated within the available time frame	Best solution for usability and functionality.	Very complex
			Saving graphics of hazard plans as text files using JavaScript Object Notation format.	Probably over 160 hours	None	It may be not possible to develop due to complexity or may not be accomadated within the available time frame	While not as usable as the previous solution alternative, users would be able to share maps using a single text file.	Very complex
			Saving created hazard plans as raster images along with their coordinates to a separate text file	Probably over 120 hours	None	While there weren't any readily available examples to save drawn graphics only on a map based on a quick survey, it is relatively easier than the previous alternatives.	Easier than the two previous solution alternatives, however users would not be able to share the maps/plans they create	Fairly complex
			Creating shapefiles on desktop GIS and uploading them to the application	80 Hours	None	After a quick survey, developer found a library that allows users uploading a shapefile, however it did not allow uploading multiple files	This is the easiest solution due to availability of an existing library. Developer however would need to modify consiredably to make it usable for the application	Fairly simple

Zooming to Bookmarked Features (Task #7)

This task refers to the following user stories that were identified in the input sessions in the project.

- User zooms to bookmarked spatial features using dropdown menu (User Story #6)
- User zooms to critical facilities using dropdown menu (User Story #44)

The zooming to bookmarked features task was developed using prototype data. This ability lets user to focus on certain spatial information by zooming to its extents, and therefore filtering out other information that may distract him/her. Arbitrarily generated critical facilities and shelters were used as bookmarks for this task.

FIGURE 4-13 FACILITY ZOOMED AND ITS ATTRIBUTES SHOWN

Emergency Plans Management (Task #8)

This task refers to the following user stories that were identified in the input sessions in the project.

- User draws a hazard response plan (User Story #7)
- User edits a hazard response plan (User Story #8)
- User shares a hazard response plan (User Story #9)
- User shares hazard response plans (User Story #10)

There were different development alternatives to be chosen for emergency plans management. These were, editing emergency plans on the fly, saving them as flat text records, saving them as image files (along with a text file including the coordinates of the plan's extent), or uploading shapefiles into the application. A brief survey showed them among these alternatives, the quickest and safest solution was uploading a shapefile into ArcGIS Silverlight application was using an extension called EsriSLContrib (http://esrislcontrib.codeplex.com/).

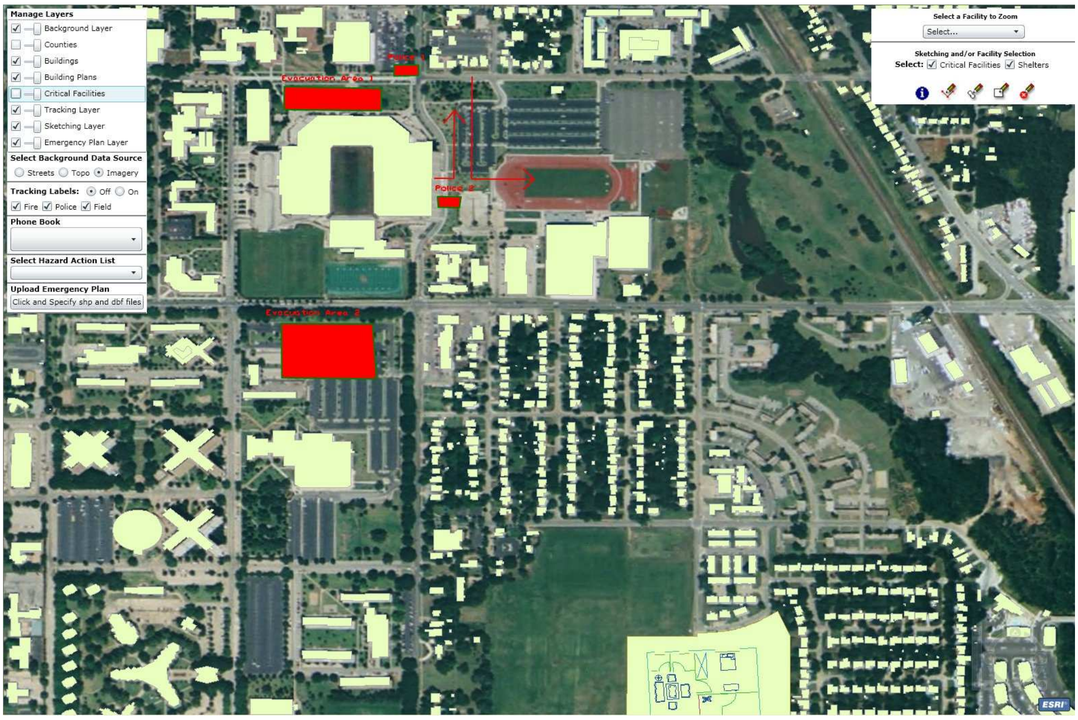

FIGURE 4-14 AN ARBITRARY EMERGENCY PLAN NEAR OKLAHOMA MEMORIAL STADIUM ATTACHED TO THE APPLICATION

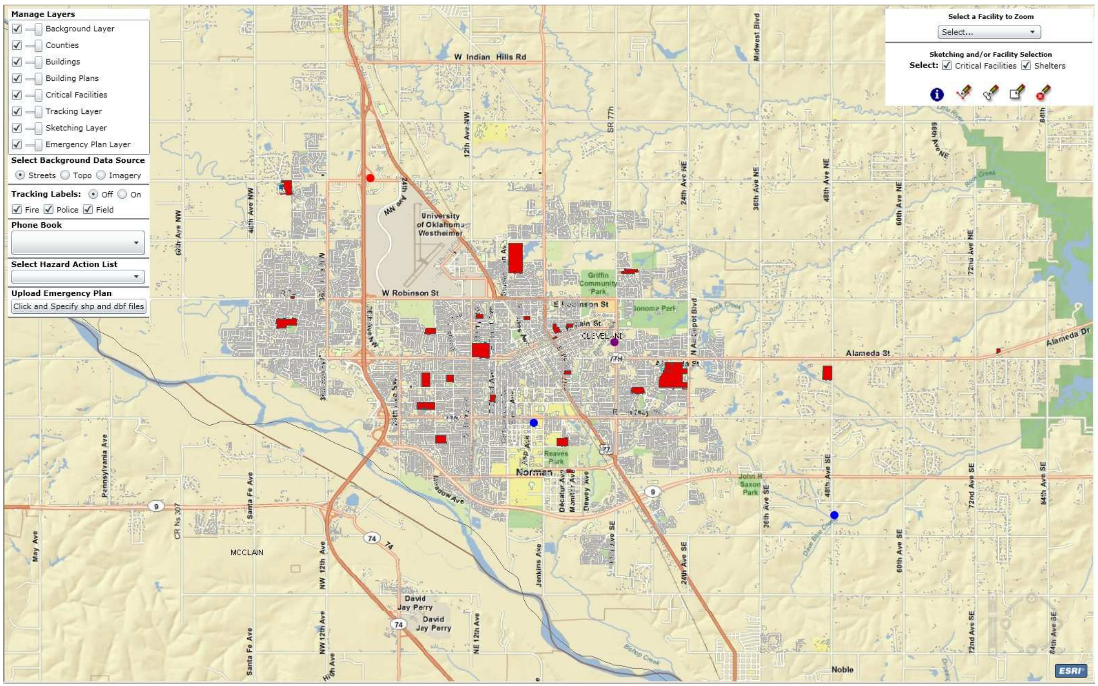

FIGURE 4-15 GENERAL VIEW OF THE APPLICATION AFTER SECOND CYCLE WAS FINISHED

Evaluation of the Second Development Cycle

While the coding portion of second development cycle was finished before the expected date 12/29/10, it was not finished immediately. The email to emergency managers was sent on 1/4/10 instead. This was because developer considered that emergency managers would not be able to access their emails during vacation time.

With the development of the emergency plans management task, the implementation portion of the second development cycle was finished. The operator manual for the second development cycle is in Appendix 6.2 and the user manual is in appendix 6.5.

TABLE 4-11 SECOND DEVELOPMENT CYCLE IN DETAIL

Task Name	Baseline Estimated Duration	Baseline Estimated Start	Baseline Estimated Finish	Actual Duration	Actual Start	Actual Finish	Duration Variance	Start Variance	Finish Variance
Second Cycle	**15.63 days**	**Wed 10/21/09**	**Wed 11/11/09**	**16 days**	**Mon 12/14/09**	**Mon 1/4/10**	**0.37 days**	**38 days**	**38 days**
Project Planning (CMMI) / Release Planning (XP)	**0.5 days**	**Mon 10/26/09**	**Mon 10/26/09**	**1 day**	**Mon 12/14/09**	**Mon 12/14/09**	**0.5 days**	**35 days**	**35 days**
Task Description Document (CMMI)	0.5 days	Mon 10/26/09	Mon 10/26/09	1 day	Mon 12/14/09	Mon 12/14/09	0.5 days	35 days	35 days
Detailed Work Estimate Document (CMMI)	0.5 days	Mon 10/26/09	Mon 10/26/09	1 day	Mon 12/14/09	Mon 12/14/09	0.5 days	35 days	35 days
Detailed Project Schedule (CMMI)	0.25 days	Mon 10/26/09	Mon 10/26/09	1 day	Mon 12/14/09	Mon 12/14/09	0.75 days	35 days	35 days
Data Management Plan	0.38 days	Mon 10/26/09	Mon 10/26/09	1 day	Mon 12/14/09	Mon 12/14/09	0.63 days	35 days	35 days
Project Resource Plan (CMMI)	0.13 days	Mon 10/26/09	Mon 10/26/09	1 day	Mon 12/14/09	Mon 12/14/09	0.88 days	35 days	35 days
Technical Solution (CMMI) / Iteration (XP)	**11.25 days**	**Mon 10/26/09**	**Tue 11/10/09**	**11 days**	**Tue 12/15/09**	**Tue 12/29/09**	**-0.25 days**	**36 days**	**35 days**
Solution Alternatives Document (CMMI)	0.38 days	Mon 10/26/09	Mon 10/26/09	1 day	Tue 12/15/09	Tue 12/15/09	0.63 days	36 days	36 days
Implementation / Coding	10 days	Mon 10/26/09	Mon 11/9/09	8 days	Thu 12/17/09	Mon 12/28/09	-2 days	38 days	35 days
User Manual (CMMI)	0.38 days	Mon 11/9/09	Tue 11/10/09	1 day	Tue 12/29/09	Tue 12/29/09	0.63 days	36 days	35 days
Sent 2nd Iteration to customers	2 days	Wed 11/11/09	Fri 11/13/09	1 day	Mon 1/4/10	Mon 1/4/10	-1 day	38 days	36 days

TABLE 4-12 RISK AND ISSUE REGISTERS FOR THE SECOND DEVELOPMENT CYCLE

ID		Description	Impact				Response			Follow Up				
	Classification			Severity	Probability (P)	Exposure (SxP)	Action Person	Mitigation Plan	Contingency Plan	Date Raised	Status	Details	Actions Taken	Actual Closure Date
29	Issue	**Developer had some other obligations:** Developer had to catch up on the RA project	Project Delay	Minor (2)	Very Likely (5)	10	Naci Dilekli	NONE	(1) Delay the research project (2) Work on the research project	12/15/2009	Closed		Developer delayed the research project	12/21/2009
30	Issue	**Graphic Selection Problem:** Clearing selection and making another selection causes a crash on the VS virtual server	Buggy application	Serious (4)	Very Likely (5)	20	Naci Dilekli	NONE	(1) Fix the problem (2) Do nothing	12/25/2009	Closed		Developer discovered that problem goes away when the machine is restarted. It seemed like it was a memory conflict problem	12/28/2009
31	Issue	**Emergency Managers not available:** Developer realized if he sent the email containing manuals, it was likely that emergency managers would miss it since they were busy around that time	Insufficient user input	Minor (2)	Very Likely (5)	10	Naci Dilekli	NONE	(1) Delay the email (2) Send it anyway	12/29/2009	Closed		Developer decided to wait and delay the email to improve the chances of getting more response	1/4/2010

All the issues that were faced during the development of first development cycle are shown in Table 4-12. In general, there were not as many and problematic issues compared to the first development cycle, mainly because the integration issues were taken care of in the first development cycle.

4.3.3. Third Development Cycle

The third development cycle of implementation took 96 hours. The total lines of XAML code were 886. The total lines of C# code were 1472.

After the second development cycle was finished, another input session was organized with 4 emergency managers. In this session, they were asked about their opinions on the application, as well as what tasks should be developed next. Their responses focused on integration of weather data and hazmat information. Those emergency managers have been using CAMEO, MARPLOT and ALOHA software packages for a long time, and discussed that similar functionalities need to be included in this WEM-DSS as well. Consequently, the developer decided to focus on these two tasks for the third and last development cycle. Project planning for these tasks as well as traffic cameras management were shown and discussed in Table 4-14. Additionally, solution alternatives and the rationale for the selected solution were discussed in Table 4-15.

.

TABLE 4-13 USER STORIES SELECTED FOR THIRD DEVELOPMENT CYCLE

Task ID	Story ID	Story	Importance	Ease to Implement	Priority	Clarity	Completeness	Testable	Source
7	6	User zooms to the bookmarked features using a dropdown menu	6	8	48	10	10	Yes	Dr. Rashed and I, and approved at first two input session meetings
7	44	User views critical facilities by selecting them from a drop down menu	8	8	64	10	10	Yes	First Input Session Meeting
8	7	User draws a hazard response plan	10	1	10	8	4	Yes	Dr. Rashed and I, and approved at first two input session meetings
8	8	User edits a hazard response plan	10	1	10	8	4	Yes	Dr. Rashed and I, and approved at first two input session meetings
8	9	User shares hazard response plans	10	1	10	8	8	Yes	Dr. Rashed and I, and approved at first two input session meetings
8	10	User erases hazard response plans	10	1	10	8	10	Yes	Dr. Rashed and I, and approved at first two input session meetings

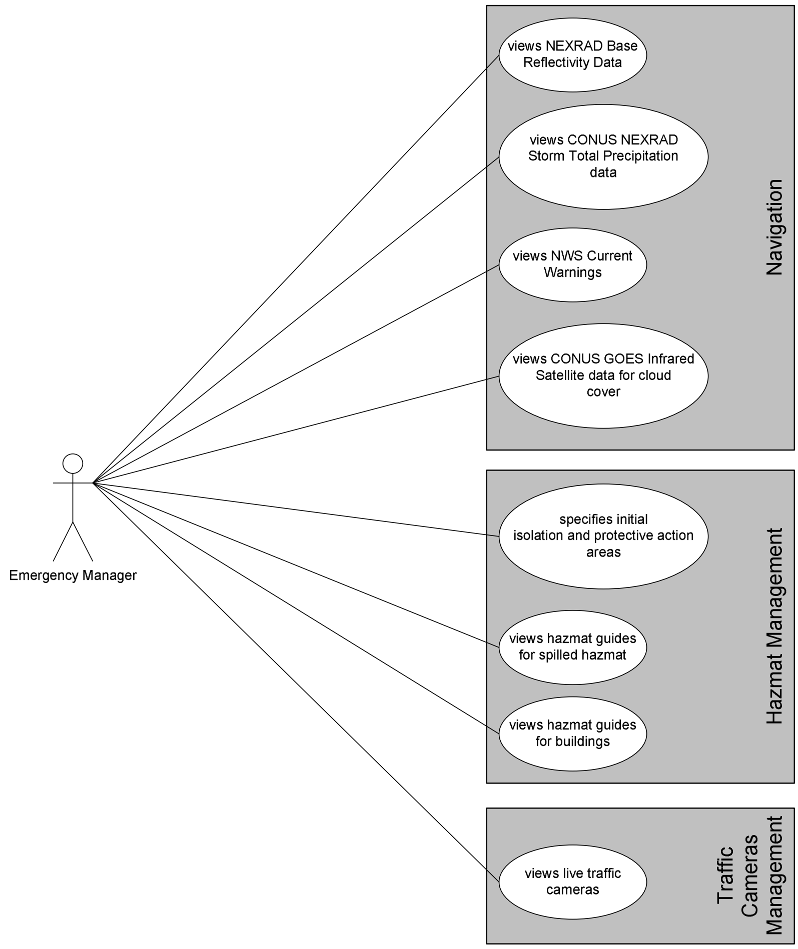

FIGURE 4-16 USER STORIES AND TASKS THAT WERE DEVELOPED IN THE THIRD DEVELOPMENT CYCLE

TABLE 4-14 PROJECT PLANNING FOR THE TASKS DEVELOPED IN THIRD DEVELOPMENT CYCLE

Task ID	Task Description	Relevant Requirements	Estimate Criteria	Size and Complexity of tasks and work products	Work Estimate	Project start and end dates	Actual Time It Took	Actual Date Started and Finished
9	Real time weather data	62, 63, 64, 65	Experience from previous cycles	This task requires either radar services rest format, or extending the existing Silverlight library to use existing WMS services	40 hours	This task will be started on 1/15/10 and finished on 1/20/10	30 hours	This task was started on 1/15/10 and finished on 1/19/10
10	Live cameras on the map	30	Experience from previous cycles	This task requires installation of a camera overlooking a street, installation of Microsoft Expression Encoder to convert the video stream for Silverlight	5 hours	This task will be started on 1/21/10 and finished the same day	6 hours	This task was started on 1/20/10 and finished the same day
11	Hazmat mapping	51, 54, 59, 60, 66	Experience from previous cycles	This task requires creation of a new panel, entering hazmat information and creation of geometric shapes that will be manipulated by the user input	50 hours	This task will be started on 1/22/10 and finished on 1/29/10	60 hours	This task will be started on 1/21/10 and finished on 1/29/10

Task ID	Task Description	Schedule Assumptions	Task Dependencies	Potential Risks	List of Managed Data	Schedule of collection of project data	Critical facilities and equipment list	Problems Encountered
9	Real time weather data	There will not be any major interferences to this work	This task does not depend on any other tasks	None	None	NA	Computer with Visual Studio.NET, Silverlight for Visual Studio.NET, Silverlight API for ArcGIS Server and ArcGIS Server.	None
10	Live cameras on the map	There will not be any major interferences to this work	This task does not depend on any other tasks	None	None	NA	Cameras with connection, permission from necessary authorities, Computer with Visual Studio.NET, Silverlight for Visual Studio.NET, Silverlight API for ArcGIS Server and ArcGIS Server.	Authorities did not allow placing a webcam on campus
11	Hazmat mapping	There will not be any major interferences to this work	This task does not depend on any other tasks	None	Sample data gathered from Emergency Response Guide book	Task implementation will start with the data generation	Computer with Visual Studio.NET, Silverlight for Visual Studio.NET, Silverlight API for ArcGIS Server and ArcGIS Server.	None

TABLE 4-15 SOLUTION ALTERNATIVES IN THIRD DEVELOPMENT CYCLE WITH SELECTED SOLUTIONS HIGHLIGHTED

User Story ID	Task ID	Task Title and Desciption	Solution Alternative	Cost estimate	Technical Limitations	Risks	Advantages and Disadvantages	Complexity
62, 63, 64, 65	9	Displaying real time weather title. This task involves researching what WMS data are available as well.	Reading WMS services into ArcMap to publish them as REST services.	50 hours	None	None	While this is very simple to implement, it is not a fast and efficient implementation to display WMS data	Very simple
			Directly reading WMS services into the application	40 hours	None	None	A short survey revealed that there is a Silverlight library to directly read WMS services as if they were REST services	Fairly simple
Rationale for Selecting the Solution Alternative: Due to performance related issues developer decided to direcly access to WMS services								
30	10	Displaying real time traffic camera streams	Placing a camera on a building on OU campus and streaming from it using	6 hours	It may not be possible to install a camera that is close enough to an available web server	Developer is not experinced much with installing cameras and streaming videos from them. Previous trials involved problems with connecting to video server	This alternative lets use of actual locally acquired data	Fairly simple
			Using arbitrary video streams from Web to show examplify the potential benefit of this function	5 hours	None	None	This alternative is easier to implement since there is not a need to deal with hardware	Very simple
Rationale for Selecting the Solution Alternative: This alternative was chosen because developer wanted to avoid using random data as much as possible								
51, 54, 59, 60, 66	11	Hazmat Mapping	Writing the code from scratch and using partial guides from the Hazmat Guidebook for Emergency Management	50 hours	None	None	This is the only alternative	Fairly complex
Rationale for Selecting the Solution Alternative: There was only one alternative to choose from								

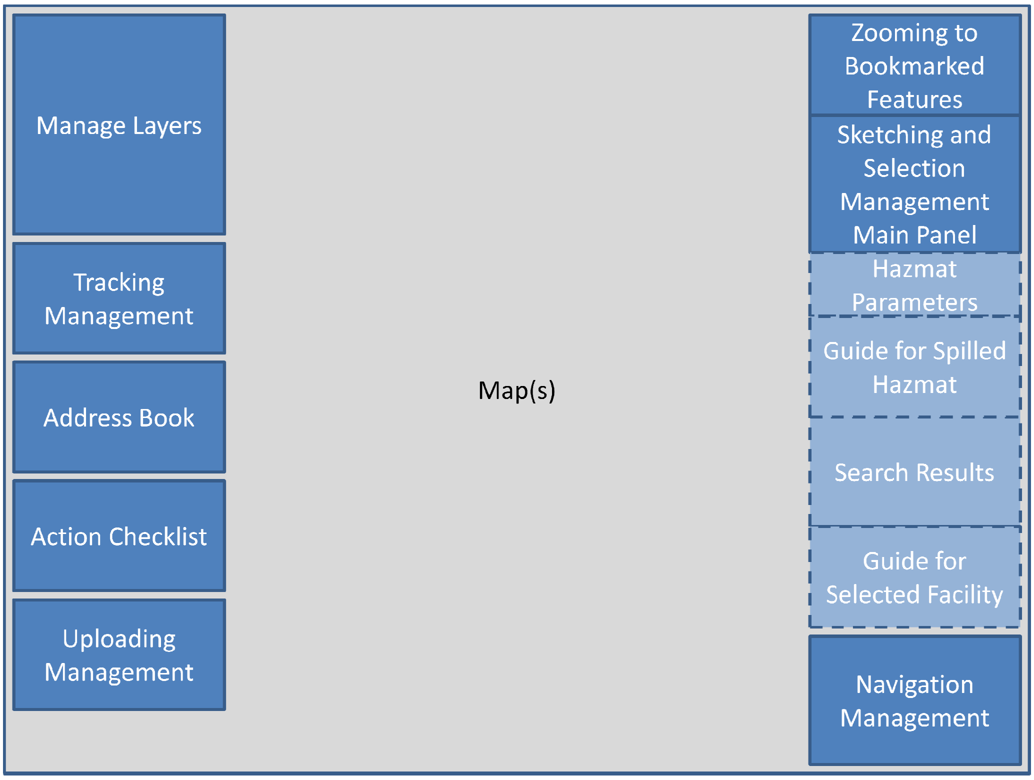

FIGURE 4-17 INTERFACE DESIGN FOR THIRD DEVELOPMENT CYCLE

WMS Data Integration (Task #9)

- User views NEXRAD Base Reflectivity Data (User Story #62)
- User views CONUS NEXRAD Storm Total Precipitation data (User Story #63)
- User views NWS Current Warnings (User Story #64)
- User views CONUS GOES Infrared Satellite data for cloud cover (User Story #65)

Initially, user story #19, which was "user views real time radar data" was going to be implemented within this task. However, developer decided that this user story was too general and the term "radar data" could be interpreted as vague. Therefore, this particular user story was transformed into four other user stories that were unambiguous and mentioned above.

WMS layers were added to the layer management container as seen in Figure 4-18. Silverlight API is originally designed to display ArcGIS services with REST endpoints. In order to be able to display WMS maps, a new component ESRI.ArcGIS.Samples.WMS, was added to the project.

Traffic Cameras Management (Task #10)

- User views live traffic cameras (prototype) (User Story #30)

This was developed as a prototype task, that the cameras shown on the map do not stream the actual locations. By default Traffic Cameras layer is turned on. There are 3 traffic cameras, and these cameras may not always be available based on their maintenance and general network issues.

Three layers containing traffic cameras were added to the layer management container as seen in Figure 4-19. These cameras were not recording media from the actual area on the map. Arbitrary video streams were used for this prototype.

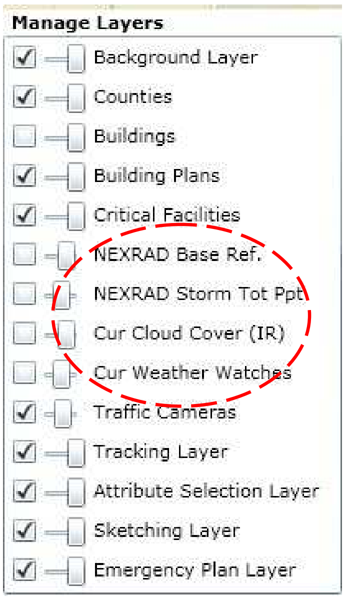

FIGURE 4-18 LAYERS MENU WITH ADDED WMS LAYERS HIGHLIGHTED

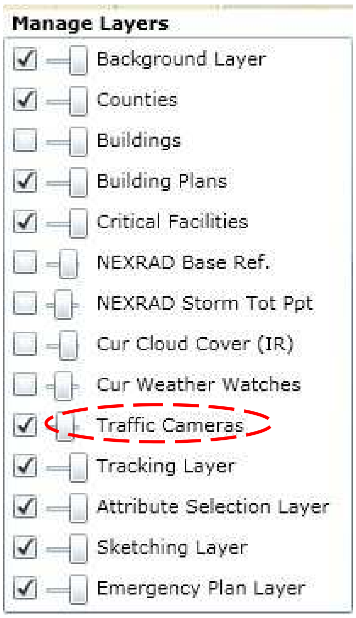

FIGURE 4-19 LAYERS MENU WITH TRAFFIC CAMERAS LAYER HIGHLIGHTED

Hazmat Management (Task #11)

- User specifies initial isolation and protective action areas (User Story #60)
- User views hazmat guides for spilled hazmat (User Story #54)
- User views hazmat guides for buildings (User Story #66)

The hazmat shape generation button was added among the other sketching/selecting tools as shown in Figure 4-20.

With this release, the hazmat drawing tool was added among the sketching and selection tools. When clicked on the hazmat icon, the panel would be expanded so that the user can specify hazmat parameters. The information on the right side of the interface may occupy much of the screen. In this case, user can close either or both of the hazmat guides as shown in Figure 4-21. This was done so that user can opt to decrease the amount of detail and complexity in the interface to avoid data overload and complexity creep, which are deterrents of situational awareness.

FIGURE 4-20 HAZMAT ICON HIGHLIGHTED AND CLICKED

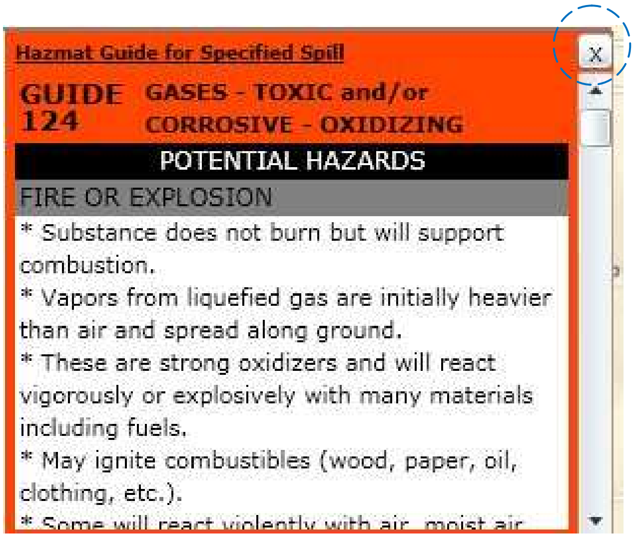

FIGURE 4-21 USER CAN CLOSE A HAZMAT GUIDE BY THE "X" BUTTON

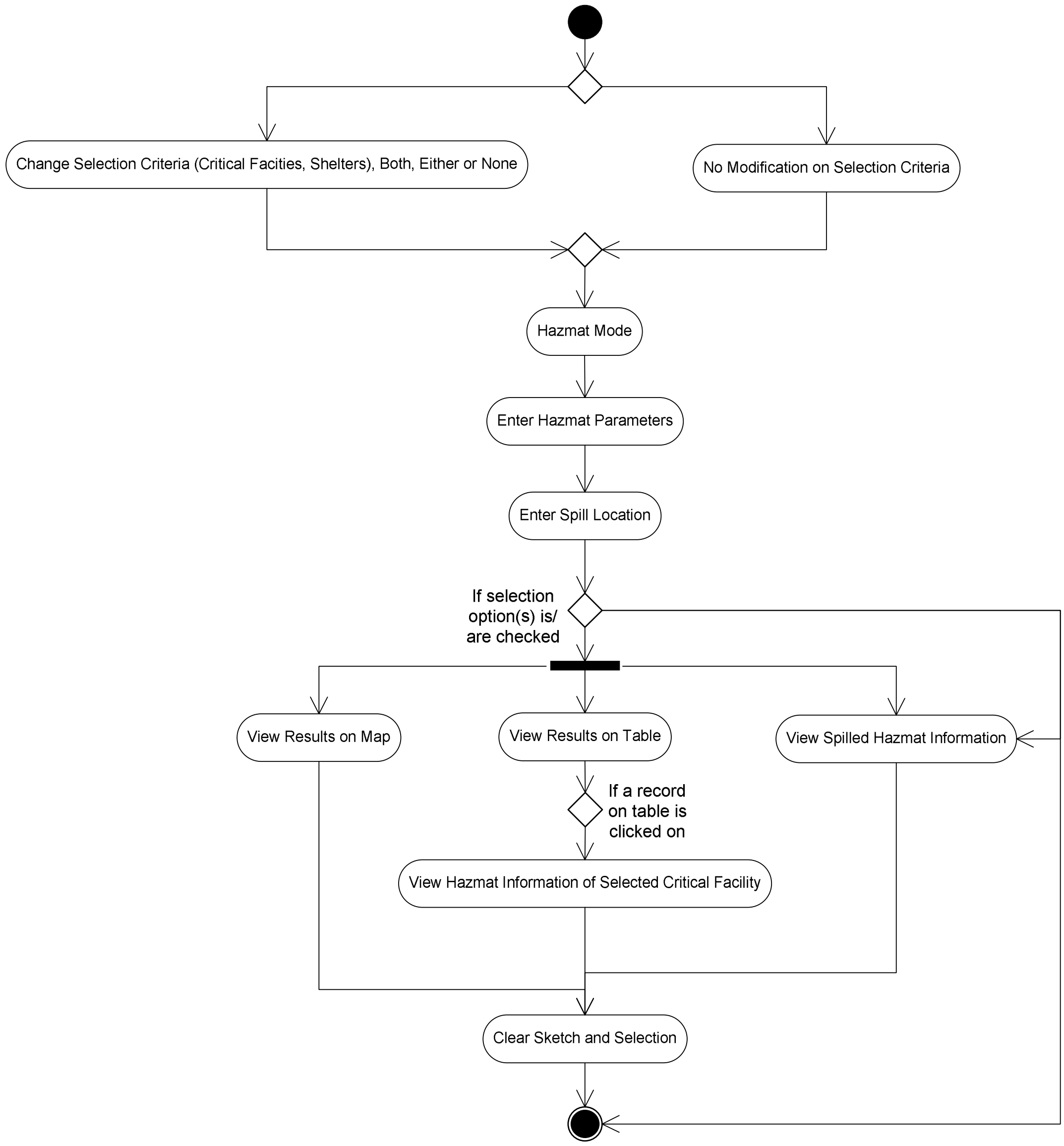

FIGURE 4-22 ACTIVITY DIAGRAM FOR HAZMAT MANAGEMENT

Evaluation of the Third Development Cycle

With the development of hazmat management task, the implementation part of the third and last development cycle was finished. The operator manual for the third development cycle is in Appendix 6.3 and the user manual is in appendix 6.6.

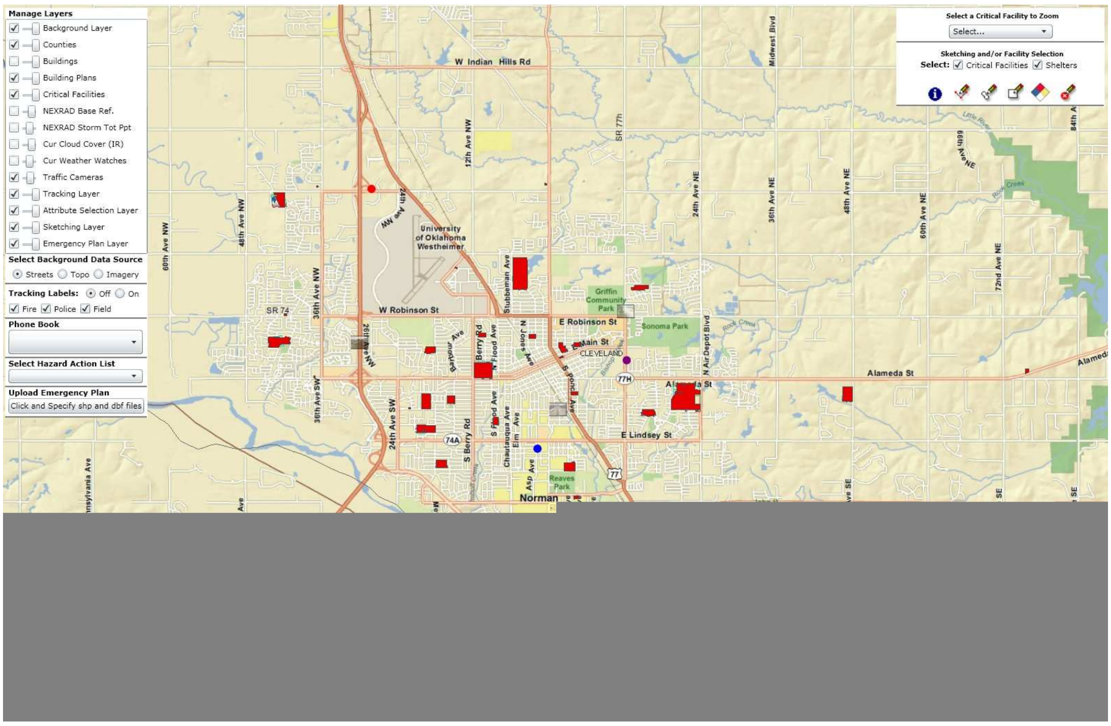

FIGURE 4-23 GENERAL VIEW OF THE APPLICATION AFTER THIRD DEVELOPMENT CYCLE

TABLE 4-16 THIRD DEVELOPMENT CYCLE SCHEDULE IN DETAIL

Task Name	Baseline Estimated Duration	Baseline Estimated Start	Baseline Estimated Finish	Actual Duration	Actual Start	Actual Finish	Duration Variance	Start Variance	Finish Variance
Third Cycle	**21.38 days**	**Mon 11/16/09**	**Tue 12/15/09**	**19 days**	**Tue 1/12/10**	**Fri 2/5/10**	**-2.38 days**	**41 days**	**38 days**
User Stories Collection	**3.75 days**	**Mon 11/16/09**	**Fri 11/20/09**	**1 day**	**Tue 1/12/10**	**Tue 1/12/10**	**-2.75 days**	**41 days**	**37 days**
User Stories Arrival	3 days	Mon 11/16/09	Thu 11/19/09	1 day	Tue 1/12/10	Tue 1/12/10	-2 days	41 days	38 days
Product Requirement Form (CMMI)	0.75 days	Thu 11/19/09	Fri 11/20/09	1 day	Tue 1/12/10	Tue 1/12/10	0.25 days	38 days	37 days
Project Planning (CMMI) / Release Planning (XP)	**0.5 days**	**Fri 11/20/09**	**Fri 11/20/09**	**1 day**	**Wed 1/13/10**	**Wed 1/13/10**	**0.5 days**	**38 days**	**38 days**
Task Description Document (CMMI)	0.5 days	Fri 11/20/09	Fri 11/20/09	1 day	Wed 1/13/10	Wed 1/13/10	0.5 days	38 days	38 days
Detailed Work Estimate Document (CMMI)	0.5 days	Fri 11/20/09	Fri 11/20/09	1 day	Wed 1/13/10	Wed 1/13/10	0.5 days	38 days	38 days
Detailed Project Schedule (CMMI)	0.25 days	Fri 11/20/09	Fri 11/20/09	1 day	Wed 1/13/10	Wed 1/13/10	0.75 days	38 days	38 days
Data Management Plan	0.38 days	Fri 11/20/09	Fri 11/20/09	1 day	Wed 1/13/10	Wed 1/13/10	0.63 days	38 days	38 days
Project Resource Plan (CMMI)	0.13 days	Fri 11/20/09	Fri 11/20/09	1 day	Wed 1/13/10	Wed 1/13/10	0.88 days	38 days	38 days
Technical Solution (CMMI) / Iteration (XP)	**16.25 days**	**Fri 11/20/09**	**Tue 12/15/09**	**17 days**	**Thu 1/14/10**	**Fri 2/5/10**	**0.75 days**	**39 days**	**38 days**
Solution Alternatives Document (CMMI)	0.38 days	Fri 11/20/09	Mon 11/23/09	1 day	Thu 1/14/10	Thu 1/14/10	0.63 days	39 days	38 days
Implementation / Coding	15 days	Mon 11/23/09	Mon 12/14/09	15 days	Fri 1/15/10	Thu 2/4/10	0 days	39 days	38 days
User Manual (CMMI)	0.38 days	Mon 12/14/09	Mon 12/14/09	1 day	Fri 2/5/10	Fri 2/5/10	0.63 days	39 days	39 days
Sent 3rd Iteration to customers	2 days	Wed 12/16/09	Thu 12/17/09	1 day	Fri 2/5/10	Fri 2/5/10	-1 day	37 days	36 days

TABLE 4-17 RISK AND ISSUE REGISTERS FOR THE THIRD DEVELOPMENT CYCLE

ID		Description	Impact				Response			Follow Up				
	Classification			Severity	Probability (P)	Exposure (SxP)	Action Person	Mitigation Plan	Contingency Plan	Date Raised	Status	Details	Actions Taken	Actual Closure Date
32	Issue	**Hazmat Buffer not working:** The graphical hazmat selection required transforming coordinate systems, however, this results with getting NullReferenceException with the ProjectAsynch method.	Cannot implement the feature completely	Moderate (3)	Very Likely (5)	15-Jan-1900	Naci Dilekli	NONE	(1) Fix the problem (2) Do nothing	1/13/2010	Closed		Developer contacted ESRI support and solved the problem working with the support	1/22/2010
33	Issue	**Placing a web cam not allowed on school property:** It was not possible to place a webcam on a school property due to regulations.	Cannot implement the feature completely	Minor (2)	Very Likely (5)	10-Jan-1900	Naci Dilekli	NONE	(1) Develop a prototype instead (2) Don't develop the feature	1/14/2010	Closed		Developer decided to have a prototype feature instead, and used video streams from California to show functionality	2/2/2010
34	Issue	**Cannot put hyperlinks along with spatial results:** Cannot make it go to the specified URL in XML data source. Note that myButton is not recognized as an object in the CS part	Cannot implement the feature completely	Moderate (3)	Very Likely (5)	15-Jan-1900	Naci Dilekli	NONE	(1) Fix the problem (2) Do nothing	1/15/2010	Closed		Instead of putting hyperlinks or buttons, developer made clicking on anywhere on the row open the hazmat details	2/4/2010
35	Issue	**School and office closed:** School was closed due to ice storm	Project Delay	Minor (2)	Very Likely (5)	10-Jan-1900	Naci Dilekli	NONE	(1) Delay the project	1/18/2010	Closed		Developer delayed the research project	1/19/2010

4.4. Changes throughout the Project

The developer tried to follow the intended project plan and schedule as much as possible. There were several requirements changes that arose due to impossibilities or ambiguities with user stories as shown in Table 4-18. These were documented in a requirements change form as required by requirements management process area of CMMI.

Additionally, since the research project involves application of CMMI approach, the developer had the freedom modify the project plan when following the original plan would not help finishing the project, or did not bring any viable benefits.

The changes occurred include:

- *Stakeholder commitment acquisition:* This was a requirement to make sure stakeholders would commit certain time necessitated by this project. This commitment was necessary before beginning the project. However, due to poor user response at the beginning of the project, this requirement has been removed.
- *Product Component Forms:* This is a change that occurred after first development cycle. In the first development phase it was seen that, for almost all the solutions, there is only one component available.
- It was foundthat emergency managers contributed little via emails. While the developer tried to get user stories at the beginning of each development cycle, there was only one email received for the user stories.
- *Product Requirement Form:* Since there was not sufficient user input, no product requirement forms were developed.
- *Validation / Acceptance testing:* After realizing users do not contribute much for the user requirements / user requirements,the developer decided it was unrealistic to expect users to submit acceptance tests which are expected to take considerably more time.
- *Verification:* The first development cycle of the project took more time than expected for the developer. Multiple issues during the coding process took significant time to fix, the developer decided to drop the verification process to save time.

- *Technical Data Package:* This element was eliminated since it did not have a high priority after the first development cycle, which was a lengthy process.
- Programmer's Manual: The first development cycle of the project took more time than expected for the developer. Multiple issues arose during the coding process that took significant time to fix, the developer decided not to write a manual for programmers. On the other hand, user's manuals were always produced because they were necessary for the users to be able to operate the application.
- *Customer Approval:* This is an Extreme Programming practice, but was dropped since it was anticipated that users would not spend any time for this process either.
- *Evaluation of the Project by Users:* After the third development cycle was finished, emails requesting feedback from users were sent. These emails include
    - An email on 2/2/10: To all emergency managers that have shown interest to my study before (20 people)
    - An email on 2/2/10: To my committee members (6 people)
    - An email on 2/3/10: A particular emergency manager for having face to face meetings with a few emergency managers to go over the application.
    - An email on 3/3/10: To all emergency managers that have shown interest to my study before
    - An email on 3/4/10: A particular emergency manager
    - An email on 3/19/10: A particular emergency manager

  There was only one response (received on 3/19/10) for project evaluation after these 6 emails.

TABLE 4-18 REQUIREMENTS CHANGE FORM

Change Date	User Story ID	Original User Story	Modified User Story	Description of the Change
12/3/2009	50	User views a shelters capacity, proximity to drawn polygon and contact information	User views a shelters capacity and contact information	Distances from the drawn polygon and the shelter or facility location needed to be calculated. However, this part of the user story could not be implemented
1/12/2010	19	User views real time radar data	User views NEXRAD Base Reflectivity Data	Before the implementation, it was decided that the original user story was too generic and ambigious. Therefore it was broken into four user stories that were specific and unambigious.
1/12/2010	19	User views real time radar data	User views CONUS NEXRAD Storm Total Precipitation data	Before the implementation, it was decided that the original user story was too generic and ambigious. Therefore it was broken into four user stories that were specific and unambigious.
1/12/2010	19	User views real time radar data	User views NWS Current Warnings	Before the implementation, it was decided that the original user story was too generic and ambigious. Therefore it was broken into four user stories that were specific and unambigious.
1/12/2010	19	User views real time radar data	User views CONUS GOES Infrared Satellite data for cloud cover	Before the implementation, it was decided that the original user story was too generic and ambigious. Therefore it was broken into four user stories that were specific and unambigious.

4.5. Evaluation of the Entire Implementation

I incorporated situational awareness oriented principles into the interface design while developing the application. The principles that influenced the design included:

- *Presenting and organizing information based on spatial proximity.* The application was designed to be able to retrieve information based on spatial proximity. This was the case when users could identify hazmat spill location and size to retrieve the information belonging to critical facilities that were in the affected area.
- *Attentional shifts required and number of separate displays should be minimized.* Accordingly, for the sake of simplicity and usability the designed application had a single display.
- *There should be functionalities that allow users to increase or decrease the level of detail.* Level of detail can be managed by zooming in and out to spatial features. Certain spatial information and detail is shown only at certain scales to realize this task.
- *As attentional constraints may be present, the system should be able to filter the abundant information based on the relevance and importance.* This principle was mainly realized using a layer management task that was addressed in the first development cycle. A layer management task allow users to turn on and off desired spatial information, basically by turning on and off layers. These layers contains vector features (some of which included labels and attributes), raster maps and streaming videos. All prominent GIS (such as ArcGIS, Mapinfo, Geomedia, TNT Mips), CAD (such as AutoCAD) and EM-DSS (such as Hazmat, Marplot, OK-First) utilizes similar functionalities.
- *Long term storage should be able to be accessed as quickly as possible with information organization and object categorization.*
- *Attentional narrowing should be avoided. This can be improved by providing simultaneous access to secondary information which will not interfere with primary tasks.* The designed application is capable of multi-tasking. This way, users can still access the less important functions at the same screen without shutting down other functions.

- *The most important and relevant information should stand out the most perceptually.* Accordingly, spatial information, which is the most important information were always presented in the center of the interface. Additionally, some panels can be added and can later be removed to allow users to limit visual elements that can distract them.
- *Verbal information should be minimized, especially regarding spatial data.* Verbal information was either stored in spatial objects, or they were mainly displayed at the periphery of the user interface.
- *Storing multiple attribute information in spatial objects.* This was done with the primary data displayed in this application. The graphical selection was done by selecting spatial objects by drawing points, lines and polygons to display detailed multiple attribute information.
- *The system should provide information regarding the trends and rates of changes in conditions.* The application environment was chosen as MS Silverlight, which is capable of handling graphical animations and moving objects. The information provided however depended on user input. As discussed later, the system included a prototype application to display emergency personnel and vehicle positions.
- *Peripheral vision can be utilized to input some of the non-crucial information.* Accordingly, while panels that user interacts with to organize data were placed on the periphery of the interface not to distract the user from spatial information.
- *Use of parallel systems:* As opposed to serial systems whose output are requirements for taking actions, This WEM-DSS has been designed as a system that provides information and recommendations that are optional.

While I tried to incorporate all the principles of situational awareness, I was not able to integrate several into my research for various reasons. I was not able to integrate the design principle regarding making cues that are more important to long term memory to stand out in order to provide rapid pattern matching for comprehension. This was because I did not conduct any survey on what cues were more important to long term memory. Another design principle was utilizing additional modes of input such as auditory and tactile modes along with the visual input. This was not considered due to technical and time limitations of this study. Additionally, the situational awareness design principle regarding providing visual redundancy for more important information was not considered as this was regarded as a secondary design goal. The primary goal with respect to information design was to present all important information, especially spatial information. If this primary goal could have been achieved, I would then try to incorporate redundant information. Another principle was that system should provide means for the users to relate himself or herself to the displayed information spatially. This was not achieved since it requires the application to capture user's location, requiring additional hardware and possibly a considerable amount of programming.

It was noticed during the development that evaluation of user stories was not as influential as it would be in an ideal development project. As mentioned before, many of the user stories actually complemented each other or the development of one may have been a requirement for developing another. This evaluation could be further enhanced with additional criteria to structure requirements selection rigorously. This can be done by including a co-dependence index to indicate which requirements depend on each other, in which case implementing one requirement is a prerequisite to implement another. Another index might be one called "complement index", which would be used to evaluate the easiness to implement one future when another one is already implemented. It should be noted that including these two indices and finding ways to structure requirements selection process might be an exhaustively long process, as both indices require cross tabulation of all the requirements in a project (in this project (3721 individual evaluations for 61 initial requirements) and a methodology needs to be devised for a rigorous selection process.

There are other reasons as to why the requirements with the highest priorities were not always selected to be implemented. Throughout the process, it was also realized that when different solution alternatives were considered, the ease of implementing these requirements, and therefore their priorities, could change as well. In order to be able to incorporate this, I would need to think about solution alternatives for each requirement even if I would not necessarily end up implementing said alternatives. This would increase the planning process drastically. Additionally, it was noticed that the importance and ease of implementation requirements changed. Change of importance was due to additional user input during the development. As the project progressed, I became more accurate in predicting the ease of implementation, and this resulted in changes in the evaluation of requirements from my point of view. This however was not reflected in managing requirements change, as modification of requirements parameters were not initially planned in the methodology section.

During the project, there were many delays. One of the primary causes of delays for the project was insufficient response from the users. It will be discussed why the response was so limited in Chapter 5. One of the biggest delays occurred with the first cycle of the development. The main reason was developer's limited exposure to the specific programming environment (ArcGIS Server API for Silverlight) and publication of Silverlight Applications on Windows Server.

For the first development cycle, the actual implementation took 15 working days, while the estimation was 10 working days. For the second development cycle, the actual implementation took 8 working days, while the estimation was 10 working days. This was due to the fact that, the developer was far more cautious with the amount of user stories to be developed. Therefore, the developer chose a more conservative development strategy. However, the developer has not noticed that an important amount of the time in the implementation process in the first development cycle was spent on integration of the application on an actual web server. Specifically, Silverlight needed to be configured to run on the Internet Information Systems. Another problem with the integration included creating the proper GIS services for the web server, which also took a considerable amount of time.

TABLE 4-19 PROJECT EXECUTION ESTIMATED AND ACTUAL TIMELINES SUMMARIZED

Task Name	Baseline Estimated Duration	Baseline Estimated Start	Baseline Estimated Finish	Actual Duration	Actual Start	Actual Finish	Duration Variance	Start Variance	Finish Variance
Project Execution	**57.63 days**	**Wed 9/30/09**	**Fri 12/18/09**	**62 days**	**Thu 11/12/09**	**Fri 2/5/10**	**4.37 days**	**31 days**	**35 days**
First Cycle	**16 days**	**Wed 9/30/09**	**Thu 10/22/09**	**18 days**	**Thu 11/12/09**	**Mon 12/7/09**	**2 days**	**31 days**	**32 days**
Second Cycle	**15.63 days**	**Wed 10/21/09**	**Wed 11/11/09**	**16 days**	**Mon 12/14/09**	**Mon 1/4/10**	**0.37 days**	**38 days**	**38 days**
Third Cycle	**21.38 days**	**Mon 11/16/09**	**Tue 12/15/09**	**19 days**	**Tue 1/12/10**	**Fri 2/5/10**	**-2.38 days**	**41 days**	**38 days**

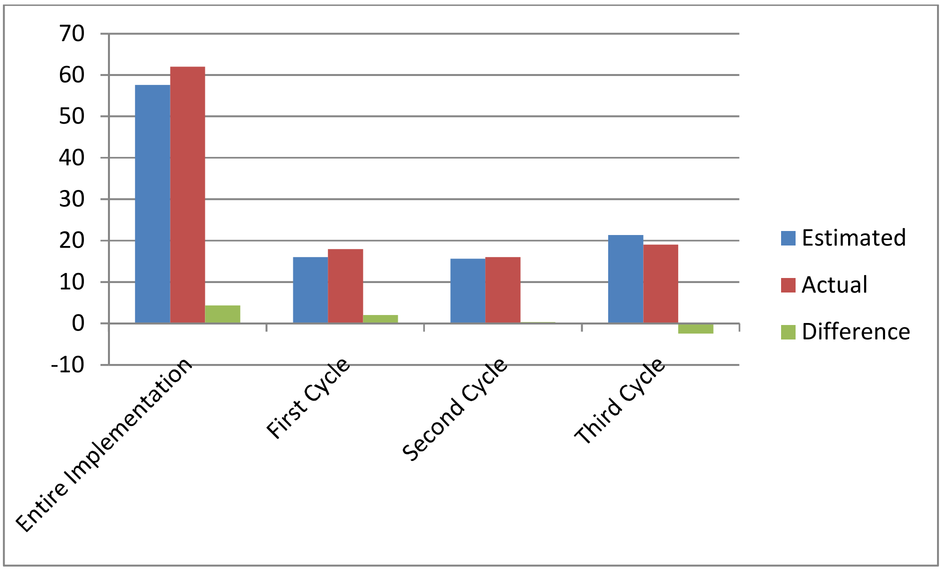

FIGURE 4-24 PROJECT EXECUTION ESTIMATED AND ACTUAL TIMELINES

Looking at Table 4-19 and Figure 4-25, it may seem as if there was not much deviation from the estimated schedule if development cycles are inspected only. This is due to the elimination of some practices such as the verification process that were mentioned in the previous section.

While there were many more issues observed in the first development cycle, as shown in Figure 4-25, the maximum amount of time to close an issue did not exceed 6 days. This figure reflects the conservative approach the developer adopted when selecting user stories to develop in the second development cycle, after the first development cycle where there were too many issues. This adaptive strategy reflects the philosophy of process improvement oriented development suggested by CMMI. The conservative approach however resulted in the completion of tasks in less time than planned.

In second development cycle, there were a total of only 3 issues, and while two of them took five days to close, which was also the maximum amount of time to close an issue in this development cycle.

The third development cycle was designed to be the longest one in the initial estimated plan. It was finished 2 and half working days earlier than expected, due to eliminated practices.

There is a total of 35 risk and issue registers. There were 4 risks identified before the development has begun. 24 issues were recorded during the first development cycle. 3 issues were identified in the second development cycle, and 5 issues were identified in the third development cycle. The reason for the higher number of issues in the first development cycle was unanticipated problems with the installment of Silverlight on the actual server.

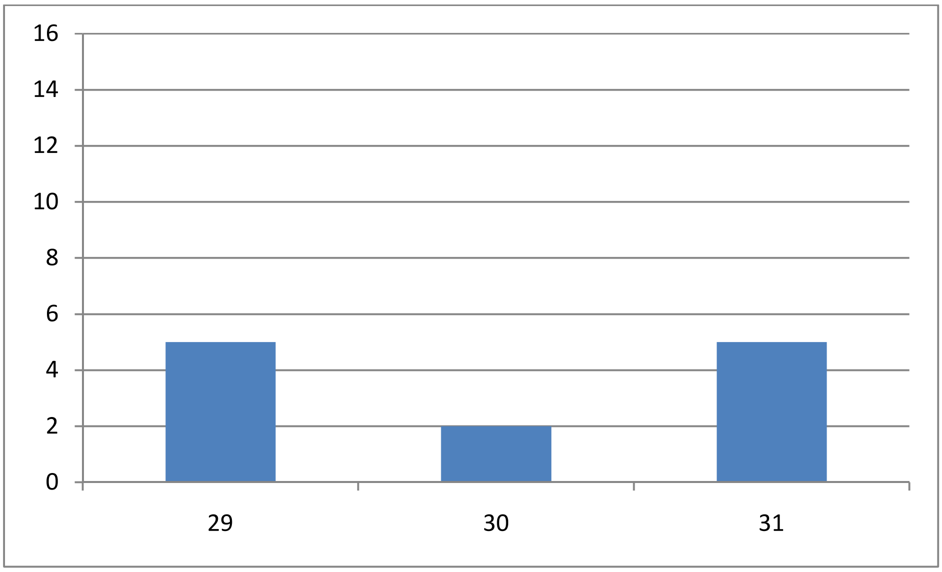

FIGURE 4-25 NUMBER OF DAYS IT TOOK TO CLOSE ISSUES IN THE SECOND DEVELOPMENT CYCLE

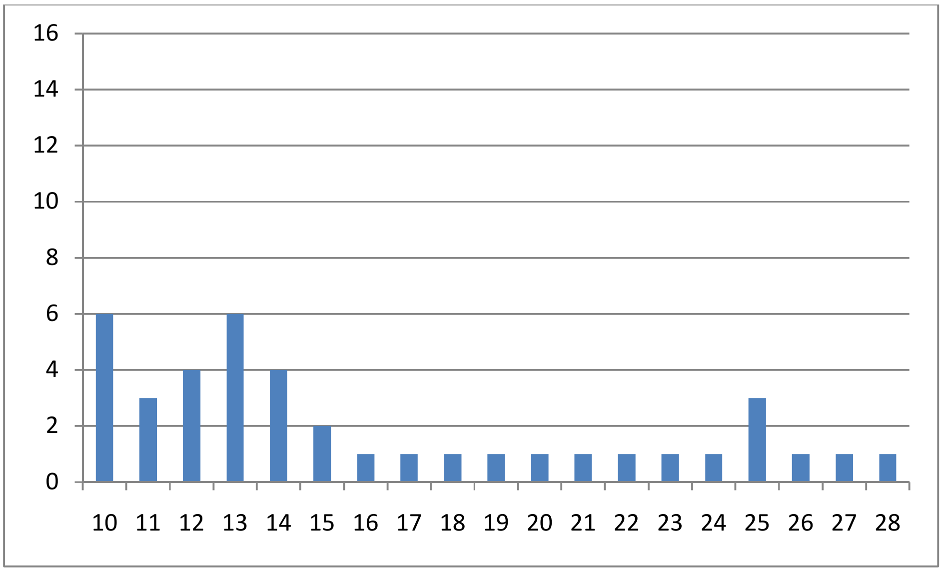

FIGURE 4-26 NUMBER OF DAYS IT TOOK TO CLOSE ISSUES IN THE FIRST DEVELOPMENT CYCLE

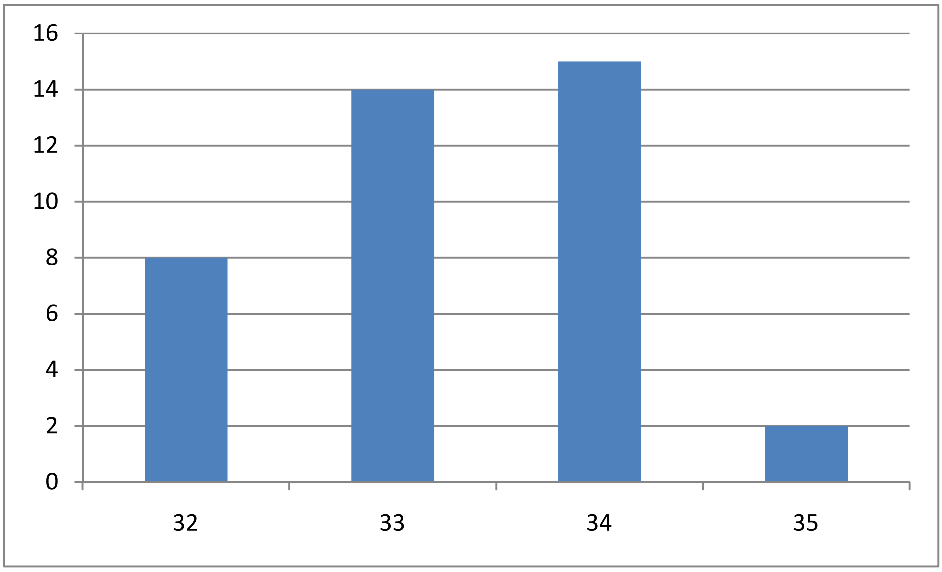

FIGURE 4-27 NUMBER OF DAYS IT TOOK TO CLOSE ISSUES IN THE THIRD DEVELOPMENT CYCLE

Chapter 5 Conclusions and Recommendations

5.1. Summary

The dual purposes of this concluding chapter are to (1) summarize the research that has been carried out and assess the degree to which this research succeeded in answering its guiding questions, and (2) highlight the contributions of this research, its limitations, and directions of future research.

The primary goal of this research was to develop and test a consolidated design methodology for web-based emergency management decision support systems (WEM-DSS). This dissertation, demonstrates how this goal was accomplished.

Chapter 1 included a number of research challenges and intermediate objectives that were set forth to accomplish the goal of this research that was aforementioned. It was argued that WEM-DSS are tools to assist emergency managers for the *entire* emergency management practice. Based on this argument, identification of elements of WEM-DSS, its comparison with other decision support systems and the identification of efficient development strategies were established as intermediate research objectives.

This work continued with a literature review in Chapter 2. The review began with the observation of a paradigm shift from single hazard to multi hazard oriented emergency management, and then addressed its implications in operational emergency management and information needs and systems. Discussion of decision support systems included an examination of user-centered design for decision making and geographic information systems. Then, information systems for emergency management were discussed in particular, with their characteristics, requirements and examples. A long review was devoted to the examination of information systems development methodologies. This portion of the review ranged from traditional sequential models to agile and flexible development methodologies. The review concluded with a comparison of information systems development methodologies.

Chapter 3 built on the theoretical discussions drawn from the literature review. It includes a methodology which was a particular integration of Extreme Programming and

Capability Maturity Model Integration that was laid out and was to be implemented. The details of this integration were discussed especially in regards to how the agility and discipline would be balanced in this new methodology. This chapter was finished with step by step explanation of the methodology and an anticipated timeline.

Chapter 4 included discussion of the application of the proposed development methodology as a case study. This methodology was carried out with project initiation, and then three development cycles in an iterative manner. For each development cycle, a number of user stories were implemented. Usually, a number of user stories are collected under a "task". Use case diagrams and activity diagrams accompanied the tasks. Since the methodology was an agile one, there were changes throughout the project. These changes, along with the justifications were explained in this chapter. In addition to the methodological modifications, there were some variations from the proposed time schedule. These variations were discussed in a quantitative manner. Additionally, the modifications, issues and risks throughout the project were documented during the project and they were discussed in this chapter.

This dissertation ends with Chapter 5 presenting the conclusions of this dissertation, in which an evaluation of the proposed methodology for WEM-DSS is undertaken in the light of qualitative and quantitative analyses conducted in Chapter 5. In addition, the theoretical and practical contributions, along with the future research directions are discussed in this chapter.

5.2. Contributions

The contributions of this study stem from addressing the research questions and challenges. The first research question of this study was:

> What are the key elements of an Emergency Management Decision Support System (EM-DSS)? How has DSS technology been used in the arena of emergency management?

In the literature review and for the justification for the proposed methodology, it was emphasized that it is important to let the users determine the functionalities of the system as well as the presented spatial and non-spatial information. These functionalities may include key elements of a WEM-DSS as well. Still, some features of WEM-DSS can be identified as very important as discovered in literature review, examination of existing WEM-DSS and the user input in this study. These include: (a) Easy to use and fast user interface; (b) Ability to integrate various map services across various platforms; (c) Ability to work (to an extent) without internet connection: This was one of the issues raised by the emergency managers in input sessions I conducted. It was understood that emergency managers can be in areas without internet connection at times; (d) Ability to add user maps; (e) Ability to integrate real time data; (f) Ability to integrate a range of maps and other information to give a sense of situational awareness; (g) Ability to increase and decrease amount of information and functionality detail to avoid data overload, which is a deterrent of situational awareness; (h) Ability to simplify and complicate the user interface according to user needs to avoid complexity creep, which is a deterrent of situational awareness.

The second research question of this study was:

> What additional benefits does a Web Based Emergency Management Spatial Decision Support System (WEM-SDSS) offer over an EM-DSS? Does a WEM-DSS intrinsically have different requirements and challenges than an ordinary EM-DSS? If so, what are the differences?

A web based systems offer crucial advantages over non web systems. These include ability to use service oriented architectures, which entails consuming map and geoprocessing services from non local sources. With this advantage comes a caveat as well. Web based systems, especially service oriented architectures (SOA), may not be the most secure option for emergency management since they are prone to malicious attacks. With SOA, some services may be unavailable at times.

Another major advantage is the ability to potentially not have to install an application on a computer. The product which is a result of this research can be run from any internet browser, for example. While most of the non web based systems have to be written in separate operating systems (such as Windows, MacOS and Linux), web based systems are usually interoperable and platform independent.

The third research question of this study was:

> What is the optimal strategy for the design and implementation of a WEM-DSS to support holistic planning and management of emergencies? How does a designer evaluate the effectiveness of such strategy and ensure it precisely captures end user needs?

As I concluded after the literature review chapter, I suggested that striking a balance between agility and discipline is important. For larger projects, I would suggest producing more documentation, more planning, larger developer teams as well as ensuring user participation either through incentives or having it mandated through their organization.

Answering this research question also helped me to address the first challenge identified in Chapter 1: “There is a need for a systematic approach to develop, evaluate and identify which technology is best suited to a particular type of decision situation during an emergency”. This was specifically done: (a) In Chapter 2 by investigating and evaluating methodological and technological aspects of developing emergency management decision support systems; (b) In Chapter 3 by proposing a new methodology for development of WEM-DSS and (c) In Chapter 4 by implementing and evaluation this particular methodology.

In addition to addressing the research questions and challenges, several other contributions emerged during this research. The contributions of this study range across academic fields. Its contributions to geography stem from utilization and exploration of hazards research, situational awareness and information systems development methodologies in emergency management. It was discussed in the literature review that geographic information systems development processes did not make use of the latest advancements in software engineering. Across the facets of geography, including geographic information systems and geographic education and research, there is a need to incorporate more information technologies and software engineering principles. In GIS education, the conventional focus is towards managing data and designing spatial databases. With the advancements in modern software, extensibility of information systems has become an important aspect. Scope and utilization of GIS can be greatly extended by customizations and extensions that require planning. This newly emerging need requires understanding development processes for an entire GIS or parts of it.

One important contribution to the GIS field originates from the gap identified by Pick (2008) who states that the use of software development methodologies and tools has been limited in GIS industry. This study, which was designed and geared towards developing a spatial information system, is an example to how state of the art software engineering tools and methodologies can be applied in the field of GIS.

The second field my work contributes to is the discipline of information systems and software engineering. My dissertation is one of the few research examples emphasizing the importance of subject matter when designing software in a combined XP and CMMI approach. The methodology I designed addressed incorporating user input from various efforts such as input sessions, user stories collection and validation. However, the utility of user input was limited due to this particular social context, due to a lack of consistent input into the development process. This methodology helped me create and shape software products in areas dealing with spatial based solutions. The results demonstrate the importance of balancing planning and documentation (as represented by CMMI approach in this study) against agility and flexibility (as represented by XP approach in this study) and melding these two approaches despite apparent contradictions...

Thirdly, my work contributes to the field of emergency management. The collected user requirements and their implementation into the WEM-DSS incorporated common needs for all hazards and particular emergency functions. Therefore, the argument that developing multi-hazards emergency management decision support systems is more practical and feasible than developing emergency management decision support systems separately was proved valid.

Another strong point of the proposed methodology is its structure that welcomed frequent user input for developing computerized systems to deliver geographic information to users. Although the user input was limited (as discussed in limitations and future work section), potential users were given frequent chances to drastically change the direction of the development process. This is another fact showing that CMMI can accommodate Extreme Programming principles.

The proposed methodology has not only been applied, but also has been well documented for evaluation purposes. This documentation itself is a major contribution of this study, as it describes the entire application of methodology step by step along with diagrams, solution alternatives and issues that were raised. Documentation allowed explanation of why there were certain delays in the project schedule and where there were changes in the actual implantation of the methodology.

The application produced as a result of this research showcases a holistic web based emergency management decision support system. This holistic system is an example of service oriented architecture as it utilizes services from various sources, including background maps from ArcGIS Online, weather and radar maps from Iowa State Mesonet, maps loaded on the local server and traffic camera stream. These services were mainly utilized by using existing libraries and components. Some of the components were modified to better integrate to the application. In addition to existing and modified components, new tools were developed. It was demonstrated in this research that adopting a service oriented architecture can empower GIS tools and components under a centrally managed holistic system.

5.3. Limitations of the research and Future Work

The purpose of this research was not to come up with the best methodology for developing decision support systems for emergency management. Rather, it was to utilize one case study to see the applicability of an integrated CMMI and XP approach in this particular study which draws input from emergency managers in the state of Oklahoma. The specific social and economic settings of Oklahoma, the particular settings of the Oklahoma emergency managers' community might be generalizable to some extent. However, such a generalization requires the application of this approach in similar contexts.

While the first challenge indentified in the introduction chapter was addressed, the second and third challenges were not addressed. These challenges are:

> There is a need to resolve methodological issues that confound the widespread application of WEM-DSS across different kinds of emergencies; and
> There is a need to demonstrate how WEM-DSS can be integrated into the process of emergency management

These challenges can be addressed in future research. The challenge of resolving methodological issues that confound the widespread application of WEM-DSS across different kinds of emergencies can be addressed by inspecting emergency management organizations that use WEM-DSS. Particularly, various development strategies and their performances need to be analyzed for that purpose. The challenge of demonstrating how WEM-DSS can be integrated into the process of emergency management requires studies that reveal appropriation of technological (particularly those of WEM-DSS) structures for emergency management. Specifically, there needs to be observation methods to monitor how particular methodologies are used, and to what extent they are used for their intended designs. A study that compares the appropriation of various WEM-DSS with various adoption mechanisms which involve alternative techniques (such as instructor led training, interactive training, mandatory training or optional training) can reveal ideal strategies for integrating WEM-DSS into emergency management.

In the introduction chapter, the issue of adaptability was raised, especially across organizations that are not similar to each other. Usability of the resultant product can only be evaluated according to the input session that was carried out after the completion of the second development cycle. In this session, it was observed that users found the web based product “very slick” and easy to use (Third Input Session Notes, 2010). A thorough evaluation of usability of the application is not possible, since a preliminary evaluation requires use of the product by the emergency managers until they use it in their operations and until they become very familiar with it.

The poor user involvement makes the measurement of effectiveness of development approaches difficult. One of the purposes of adopting XP was to observe effectiveness of an agile methodology for this particular setting. However, many of the aspects of XP have not been utilized, including user testing, request of new and modified tasks during the development. Therefore, one significant item in the future research list is using similar methodologies while motivating users to participate in the process with incentives.

This research relied on volunteer input and feedback from emergency managers. There were three meetings with individual emergency managers, and a meeting with a group of emergency managers before the implementation. In order to reach a large group of emergency managers, email was chosen as the principal means of communication with emergency managers in order to have them examine the application during the development after each development cycle was finished, test it, and request new tasks in the form of user stories. However, poor user participation was observed throughout this study. Initially, higher user participation was anticipated by the investigator, since it was likely that the developed product would be helpful to emergency managers.

Lack of user participation has been addressed by several academic works especially regarding user participation in online communities, such file sharing and social networking communities. According to Kollock (1999), there are four types of motivations for online cooperation. First possible motivation is expectation to receive useful help and information in return. The second possibility is to gain reputation through contribution. A third kind was identified as having a sense of efficacy on the environment or society. Lastly, attachment or commitment to a community can motivate to contribute.

Burgahain *et al.* (2003) and Golle *et al.* (2001) suggested using micro-payments to reward individual contributions as incentive mechanisms for peer to peer online communities. Vassileva *et al.* (2004) suggested increasing user participation by rewarding them a higher status in the community, and providing them a higher quality of service. Cheng and Vassileva (2005) found out that while this type of motivation increased the quantity of user participation, it also caused reduced quality as many users tried to maximize their benefits with minimum effort. This finally caused a decrease in the user participation as a consequence of decreased quality of resources shared. Similarly, a study by Farzan *et al.* (2008) revealed that top status focused users in an online community continually added content to a social networking site to stay at a top level. Level focused users however, slowed down or altogether stopped their contributions once they reached a certain level of status.

It should also be noted that the term “customer” is often used when referring to users in information systems development methodologies. There is usually an underlying assumption that the users pay for the software that is developed, therefore their participation is ensured through their organization’s workings (e.g. users being mandated to participate by managers). It is only natural to observe more participation if the user side has paid for the software. In this study however, the participants were emergency managers working at state institutions. They were not compensated for their participation in addition to the fact that they were usually busy with conferences and emergencies.

If a similar project was to be conducted in the future, I would first secure a certain number of participants that agree to be a continuous part of the project until the end of the project. For future research, in the light of these works and especially this particular research, it is apparent that using incentives for user input especially when developing WEM-DSS for nonprofit, research or public organizations would be helpful for generating more user input. Use of incentives is likely to increase the quantity of the user input. However it is important to note that use of incentives might negatively affect the overall quality of input if participants are not motivated. Additionally, comparative studies conducted on different demographics may give a good idea on how to get more user participation both in quantity and quality as well.

Another change in future research I would adopt would be using a development team that involved at least two programmers and a project manager. My contention is such a structure is closer to programming industry standards. A project manager is needed to organize user participation and to give directions when there are scheduling changes and technical issues. A second programmer would be needed to progress faster, as well as to comply with the pair programming principle suggested by Extreme Programming.

In the introduction chapter, the issue of adaptability was raised, especially across organizations that are not similar to each other. Adaptability of the product in this study can only be evaluated according to the input session that was carried out after the completion of the second development cycle. In this session, it was observed that users found the web based product easy to use in the third input session. The fact that there was not sufficient time and that arrangements could not be done to observe emergency managers using the product are other limitations of this study. For future studies, the aforementioned secured user participation should be scoped to include observation of their interaction and use of the product. The analysis of deviation of the actual schedule from the expected schedule can be dramatically affected by the programming and development environment. While the implementation was done using Microsoft .NET, Silverlight, and ArcGIS API for Silverlight, it is quite possible using other environment (such as open source GIS) for future research might have an effect on the learning curve, availability of libraries and tools and performance of the product.

Technical problems have not been large issues throughout the project. Most of the time the issues were overcome by developing prototype functionalities rather than developing real and functional ones. For difficult tasks, the functionalities have been simplified as well. There were several technical issues that can be addressed in future studies. The tool for uploading emergency plans currently only supports uploading shapefiles with WGS84 coordinate system. Additional libraries need to be developed to convert coordinates systems on the fly. Another limitation with this tool is that, annotation layers cannot be uploaded, as they are not available in shapefile format. However, it is possible to convert an annotation layer into a polyline shapefile, and upload it as a part of emergency plan. Additional libraries can be developed that can read text files which include text and their spatial information (e.g. xmin, ymin and xmax, ymax coordinates).

6. Appendix

6.1. Operator Manual for the First Development Cycle

Navigation (Feature #1)
A navigation tool was placed on the application using XAML code, simply using the Navigation component in ESRI ArcGIS Silverlight Toolkit.

Managing Layers (Feature #2)
For each layer, there is a row containing a checkbox, a slider box and layers name. Layer management is achieved through XAML code. In this XAML code, first a list for all the layers is created. For each layer, a checkbox is created to turn it on and off. For each layer, a slider is created to change its transparency. For each layer, a textbox is created to show copyright information when user hovers over the layer. For each layer, a textbox is created to display name or description of the layer.

The radio buttons and their container were written in XAML. A C# code was added to handle the click radio button event. Accordingly, as soon as the radio button is clicked, an event is fired and the URL of the layer is updated.

Tracking Management (Feature #3)
The radio buttons, checkboxes and their containers were written in XAML. The event handlers and the functions to generate the random movements were added to the C# portion of the application.

Sketching and Selection Management (Feature #4)
Sketching / selection menu and the results pane were specified using XAML code. There are 10 functions and a helper class to manage the all the operations within this feature. The most three important functions include the events esriTools_ToolbarItemClicked and MyDrawSurface_DrawComplete and QueryTask_ExecuteCompleted. With esriTools_ToolbarItemClicked event, application it put into a certain drawing mode (e.g. point, polyline, polygon, rectangle) to create a graphic or the graphic already drawn is cleared. With MyDrawSurface_DrawComplete event, the drawn graphic is added into a graphics layer, a query task is created, using the graphic and using the options provided by user (limiting the query to shelters only, limiting the query to critical facilities only or

no limitation) and using the QueryTask_ExecuteCompleted event, the query is executed, and the results are put into a separate graphics layer and their tabular data are also binded to the results datagrid. In case there is a problem with the query, QueryTask_Failed event is fired.

Remaining 6 functions include events:

- GraphicsLayer_MouseEnter is activated when user hovers over a selected feature to highlight the corresponsing row in the results data grid.
- GraphicsLayer_MouseLeave is activated when the mouse cursor leaves a selected feature's graphic to turn off the highlight of the corresponding row in the results data grid.
- Row_MouseEnter is activated when the mouse cursor hovers over to a row, highlighting the corresponding graphic.
- Row_MouseLeave is activated when the mouse cursor hovers to leave a row, turning off the highlight of the corresponding graphic.
- QueryDetailsDataGrid_LoadingRow activates events Row_MouseEnter and Row_MouseLeave.
- QueryDetailsDataGrid_SelectionChanged is activated when the mouse cursor hovers over from one row to another changing the selection, highlighting the corresponding graphic.

Action Checklist (Feature #5)

The combobox and its container were written in XAML. A single event handler was written to generate the list of actions once a hazard type is specified in the C# portion of the application. Once an item from the combobox is selected, the list items are added below the combobox.

Address Book (Feature #6)

The combobox the contact information and the container were written in XAML. No event handles were written in the C# portion of the application.

6.2. Operator Manual for the Second Development Cycle

Zooming to Bookmarked Features (Feature #7)

The combobox, the datagrid and their container were written in XAML. Three event handlers were written to display the information of the selected critical facility or shelter in the C# portion of the application.

They include the events AttributeQueryComboBox_SelectionChanged, AttributeQueryTask_ExecuteCompleted and AttributeQueryTask_Failed. With AttributeQueryComboBox_SelectionChanged event, a query task is created, using combobox item selected by user and using the QueryTask_ExecuteCompleted event, the query is executed, and the results are put into a separate graphics layer, the selected graphic features are zoomed in and their tabular data are also binded to the attribute results datagrid. In case there is a problem with the query, QueryTask_Failed event is fired.

Emergency Plans Management (Feature #8)

The button and its container were written in XAML. An event handler was written to load specified shapefiles in the C# portion of the application. This event handler uses a component, Vishcious.ArcGIS.SLContrib. An example application using this component was configured to upload a single shapefile at a time. The code was modified so that multiple shapefiles could be uploaded at the same time. To do this, user needs to specify all the dbf and shp files of corresponding shapefiles.

6.3. Operator Manual for the Third Development Cycle

WMS Data Integration (Feature #9)

WMS layers were added to the layer management container using XAML code. A WMS layer can be added as if adding a REST service by utilizing ESRI.ArcGIS.Samples.WMS component.

Traffic Cameras Management (Feature #10)

Three layers containing traffic cameras were added to the layer management container using XAML code.

Hazmat Management (Feature #11)

The hazmat shape generation button was added among the other drawing tools using XAML code. Hazmat specification controls were added below the sketching and selection management panel using XAML code. Two hazmat information panels, one for spilled material, one for the hazmat in critical facility within the spill area, were added below the hazmat specification controls using XAML code.

Several changes were made to the C# code for sketching / selection management.

- A new drawing mode in esriTools_ToolbarItemClicked event was prepared.
- Control mechanism in MyDrawSurface_DrawComplete event to check if user entered an azimuth value correctly and if the type of hazmat was selected was created.
- GeometryService_Failed and GeometryService_ProjectCompleted events are invoked through MyDrawSurface_DrawComplete to transform coordinates. The hazmat center point has coordinates in latitude and longtidues. However, since no accurate measurements can be done to draw hazmat initial isolation and protective action areas. In order to draw these geometric shapes accurately, the hazmat center point needs to be transformed into UTM (zone 35N for this study) coordinate system using an ArcGIS geometry service. If hazmat area generation tool was selected QueryTask_ExecuteCompleted event is not fired yet, since the initial isolation and protective distance areas need to be generated first.
- GeometryService_ProjectCompleted event was created to draw initial isolation distance and protection distance shapes according to the hazmat parameters

specified by the user. After these polygons are drawn, they are transformed back to the original latitute langtitude coordinate system using another using an ArcGIS geometry service. Once these geometry service is finished transforming coordinates back, GeometryService2_ProjectCompleted event is fired.

- GeometryService2_ProjectCompleted event is used put three graphics on the map; isolation, protective area as well as a query area graphic that emcompasses the both isolation and protective areas in order to query the features using the options provided by user (limiting the query to shelters only, limiting the query to critical facilities only or no limitation) and using QueryTask_ExecuteCompleted event. After the query is executed, the results are put into a separate graphics layer and their tabular data are also binded to the results datagrid. In case there is a problem with the query, QueryTask_Failed event is fired. Additionally, loadSpillHazmat event is fired in the end.
- loadSpillHazmat is automatically fired by GeometryService2_ProjectCompleted event. Based on the hatmat type/name, hazmat guide information as described in Emergency Response Guidebook (US Department of Transportation, 2008) is displayed in a very similar format to the book below the hazmat parameter specification panel and above search results panel.
- QueryDetailsDataGrid_MouseLeftButtonUp event is activated when user clicks on a row on the data grid to fire loadCriticalHazmat event to display hazmat information of the hazmat material of the selected critical facility or shelter. loadCriticalHazmat event is activated by QueryDetailsDataGrid_MouseLeftButtonUp to display hazmat information of the hazmat material of the selected critical facility or shelter. CriticalHazmatDisplay_Close_Click event is used to close the critical hazmat information display SpillHazmatDisplay_Close_Click event is used to close the spilled hazmat information display.

6.4. Interface Description and User Manual for the First Development Cycle of the Project

This application is developed by Naci Dilekli, a PhD student in University of Oklahoma, Department of Geography for his research.

Warning: This is a prototype application, and the purpose of it is to show functionality provided that there are correct and updated information in it. For the study, Norman, OK was chosen, and data was collected and generated accordingly. Data in this application is mostly arbitrarily and/or randomly generated, and is not to be used for any actual decision making.

1. Introduction

The purpose of this study is to develop and test a consolidated design methodology for web-based emergency management decision support systems (WEM-DSSs). A WEM-DSS is a decision support system utilizing recent developments in communications, especially Internet technology, for holistic and effective emergency management. Accordingly, an emergency management decision support system (EM-DSS) is a tool to assist emergency managers in all elements related to the holistic planning and management of emergencies, from efforts aiming to prevent emergencies, to preparing for the emergencies, to the management of the actual emergency response.

This application addresses a group of user stories that are collected through user surveys conducted by the researcher. This application was developed using ArcGIS Server and the ArcGIS API for Microsoft Silverlight. User needs to install Silverlight to run the application. Note that the application has only been tested in the MS Windows.

- For Windows: http://www.microsoft.com/silverlight/get-started/install/default.aspx
- For Mac OS: http://www.apple.com/downloads/macosx/development_tools/silverlight.html
- For Linux and UNIX: http://www.go-mono.com/moonlight/

The application consists of a panel on the left for managing the data, a panel for sketching and selecting on top right, and a panel for navigation on the bottom left of the interface. A complete list of supported features (user stories) is provided at the end of this document.

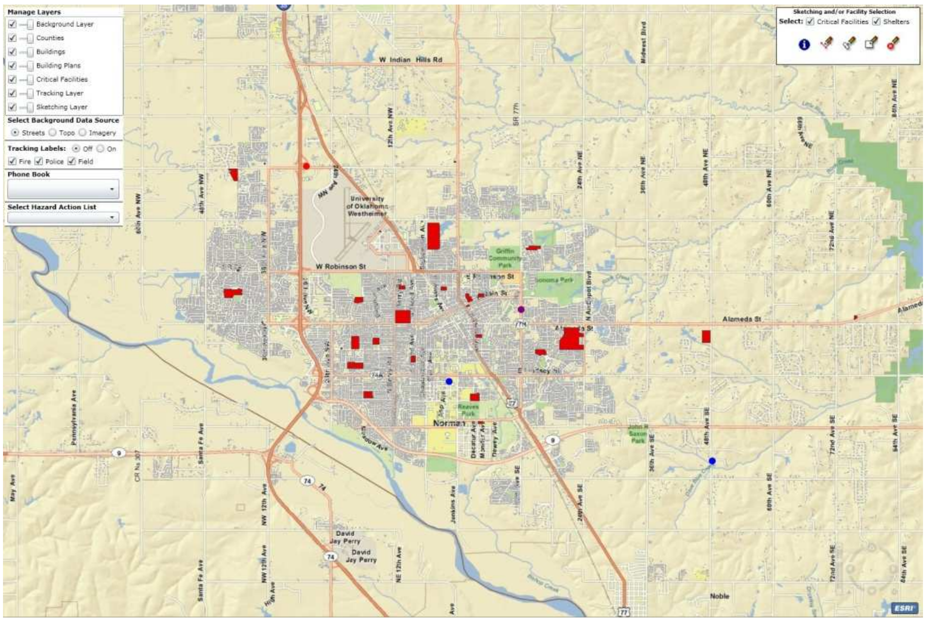

Figure. General View of the Application

2. Supported Features / User Stories

- User zooms in to map
- User zooms out of map
- User pans across map
- User zooms to the map extent
- User identifies features
- User views locations of emergency vehicles on the map real time (prototype)
- User toggles between different types of emergency vehicles
- User views locations of emergency managers on the map real time (prototype)
- User views labels on top of emergency vehicles on the map
- User views topographic maps
- User views land cover satellite imagery
- User views building floor plans
- User toggles between different data source
- User can view county names on the map
- User draws polygons on the fly during response
- User views what facilities are in a drawn polygon automatically
- User views what shelters are in a drawn polygon automatically
- User views phone number for a critical building after selection
- User chooses whether shelters and/or critical facilities will be displayed by clicking on checkboxes when a polygon is drawn
- User views a checklist for actions to do for certain events

- User views the address book to contact people in critical facilities and other agencies

3. Managing Layers

Layers can be managed through the top portion of the panel on the left, titled: Manage Layers. For each layer, there is a row containing a checkbox, a slider box and layers name.

Figure. Manage Layers Panel

A layer can be turned on and off by the checkbox on the left. Transparency of a layer can be adjusted by the slide bar in the middle. Currently, there are 7 layers available for managing, including:

- Background Layer: A layer showing either the street map, topographical map or the satellite imagery.
- Counties: A layer showing the boundaries and names of counties in Oklahoma.
- Buildings: A layer showing the buildings in Norman, OK.
- Building Plans: A prototype layer showing the building plans of the critical facilities in Norman, OK.
- Critical Facilities: A prototype layer showing the critical facilities (Police stations, fire stations, schools, hospitals and shelters) in Norman, OK.
- Tracking Layer: A prototype layer showing the locations for emergency vehicle and people locations. It includes fire vehicles, police vehicles and field responders.
- Sketching Layer: A layer that controls the visibility of the sketching graphics. Sketching is done through the top right panel.

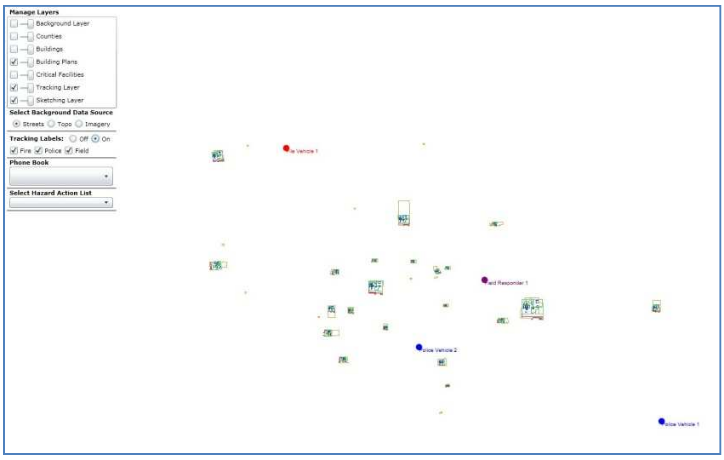

Figure. Some layers turned off

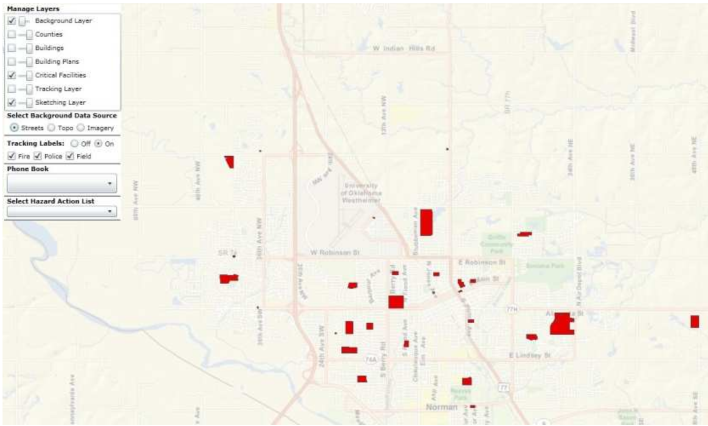

Figure. Transparency for the background layer adjusted

User can change the contents of the background layer by choosing one of the options through the radio buttons. The options are street data, topography data and satellite imagery.

Figure. Background Data Source Panel

4. Managing Tracking Symbols and Labels

User can turn on and turn off labels for the tracked features by the radio buttons next to "Tracking Labels:". Categories of certain tracked features can be turned on and off by the checkboxes next to "Fire", "Police" and "Field".

Figure. Managing Tracking Symbols and Labels Panel

5. Phonebook

This is a panel for viewing the web sites and phone numbers of the listings in the application. User can click on the combobox and can view the entries. Also, by clicking on the web link, user will be directed to the relevant institution’s web site.

Figure. Viewing Phonebook Items

6. Viewing Hazard Action List

This is a panel for viewing the action lists for certain events. When the user clicks on the combobox, and selects a category, a list of actions will pop up below. Note that checking any of the actions will not cause anything on the user interface. The list is a reminder for the emergency manager for what he/she needs to do under certain circumstances.

Figure. Selecting Hazard Action Lists

Figure. Some Hazard Action List Items Checked

7. Sketching and/or Facility Selection

This is a panel for drawing features on the map. User can choose to select critical facilities and/or shelters based on the drawn features as well. For this, user needs to choose which group of features will be selected by the checkboxes next to “Select”. User then can make the selection based on a single point (Identify), drawing a polyline, drawing a polygon or drawing a rectangle. To do this, user first needs to click on the selection option (identify, polyline, polygon or rectangle) and then do the drawing / selection operation on the screen. User can then erase the drawing by clicking on the clear selection button. For example, using the polygon sketching tool, user may draw a rough plume area on the map, and see what facilities are in this area.

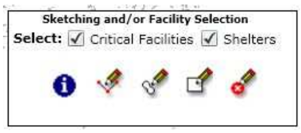

Figure. Sketching and/or Facility Selection Menu

If any selection is made, then the results (information regarding the critical facilities and shelters) will be displaying below the Sketching and Selection Panel. When user hovers on to a selected feature, that graphical feature as well as its entry in the results table will be highlighted. The same effect happens when the user hovers on any entry in the results table.

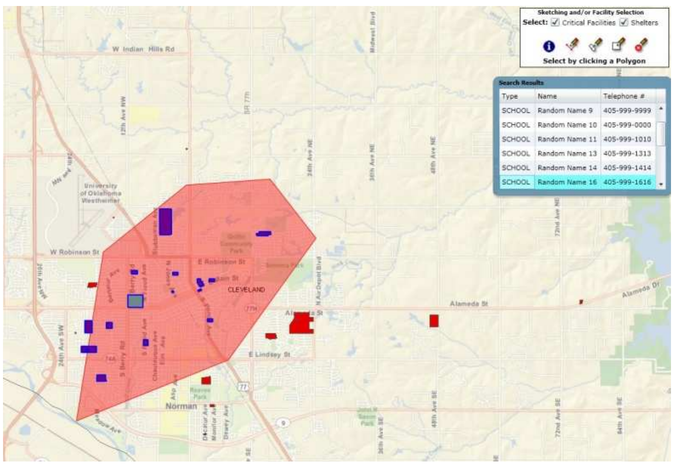

Figure. Sketching an Facility Selection

8. Navigation and Navigation Panel

Navigation on the map can be done via the mouse gestures, which are similar to Google Earth controls. To pan the map, user can hold the left mouse button down, and move the mouse. User can also use the arrow keys on his/her computer to move left, right, up and down as well. To zoom in and out of the map, user can use the mouse wheel.

On the navigation panel at the right bottom of the map, user can pan, zoom in and out, rotate the map, reset the rotation, and zoom to the extents of the map.

By clicking on the arrow buttons around the ring, user can pan the map. User can zoom in and out to the map using the zoom slider or the zoom in and out buttons on the left side of the navigation panel. User needs to click and drag the ring, and then move the mouse to upward or downward direction to rotate the view. User needs to click the north-up button to reset the view so that north is at the top of the screen.

Figure. Rotating Using the Navigation Panel

6.5. Interface Description and User Manual for the Second Development Cycle of the Project

This application is developed by Naci Dilekli, a PhD student in University of Oklahoma, Department of Geography for his research.

Warning: This is a prototype application, and the purpose of it is to show functionality provided that there are correct and updated information in it. For the study, Norman, OK was chosen, and data was collected and generated accordingly. Data in this application is mostly arbitrarily and/or randomly generated, and is not to be used for any actual decision making.

1. Introduction

The purpose of this study is to develop and test a consolidated design methodology for web-based emergency management decision support systems (WEM-DSSs). A WEM-DSS is a decision support system utilizing recent developments in communications, especially Internet technology, for holistic and effective emergency management. Accordingly, an emergency management decision support system (EM-DSS) is a tool to assist emergency managers in all elements related to the holistic planning and management of emergencies, from efforts aiming to prevent emergencies, to preparing for the emergencies, to the management of the actual emergency response.

This application addresses a group of user stories that are collected through user surveys conducted by the researcher. This application was developed using ArcGIS Server and the ArcGIS API for Microsoft Silverlight. User needs to install Silverlight to run the application. Note that the application has only been tested in the MS Windows.

- For Windows: http://www.microsoft.com/silverlight/get-started/install/default.aspx
- For Mac OS: http://www.apple.com/downloads/macosx/development_tools/silverlight.html
- For Linux and UNIX: http://www.go-mono.com/moonlight/

The application consists of a panel on the left for managing the data, a panel for sketching and selecting on top right, and a panel for navigation on the bottom left of the interface. A list of supported features (user stories) that are developed specifically for this release is provided at the next section.

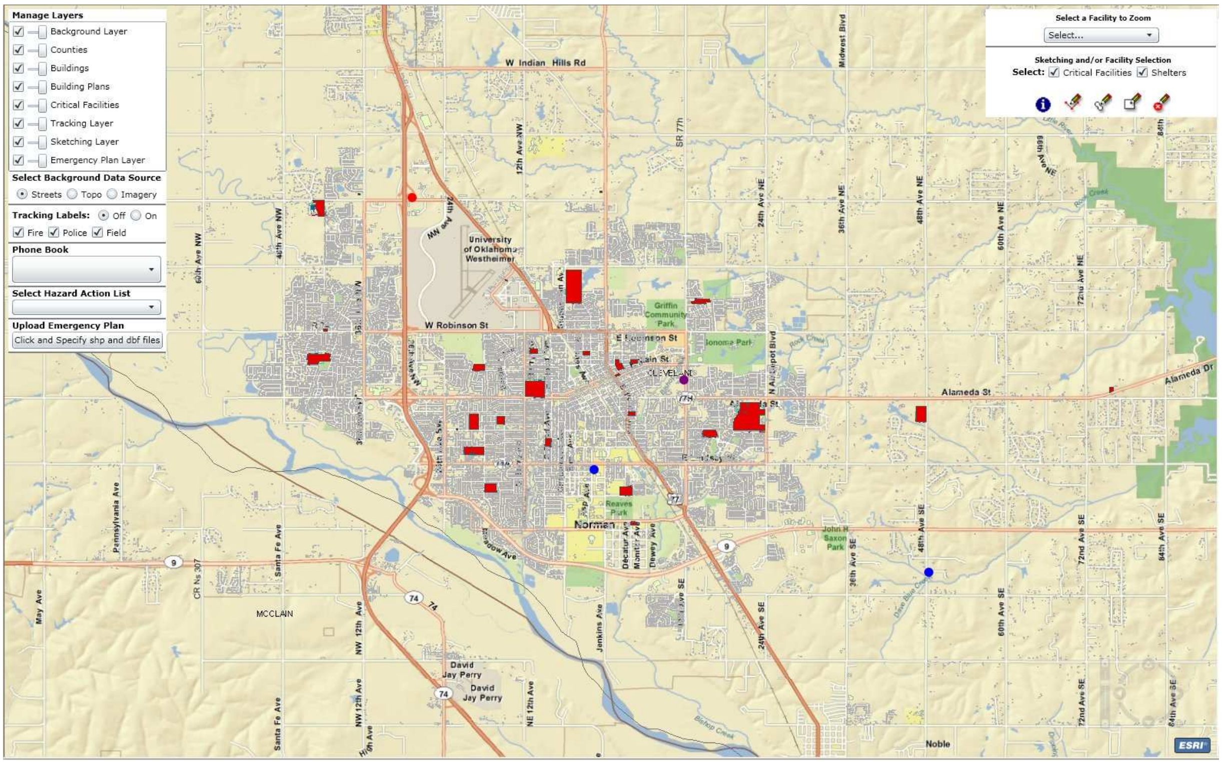

Figure. General View of the Application

2. Supported Features / User Stories

- User zooms to the bookmarked features using a dropdown menu
- User draws a hazard response plan
- User edits a hazard response plan
- User shares hazard response plans

3. Zooming to Bookmarked Features

Features can be zoomed in by selecting their names from the dropdown menu on the right.

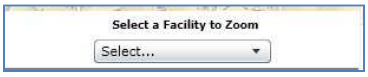

Figure. Zooming to Features Menu

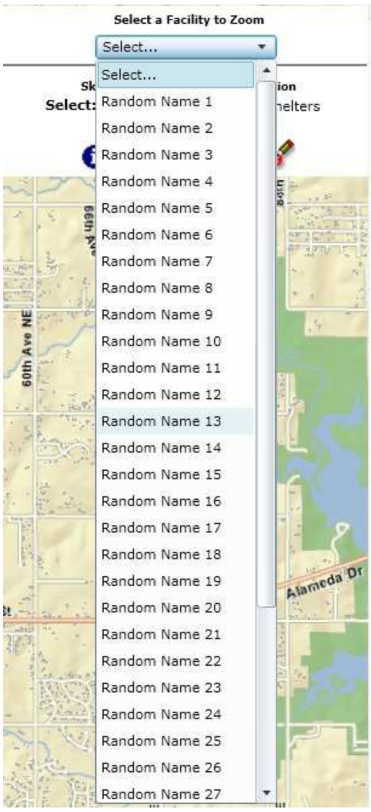

Figure. Zooming to Features Menu Expanded

After clicking on a facility name, it will automatically zoom to that feature and it will show its attributes below.

Figure. Facility Zoomed and Its Attributes Shown

4. Uploading Emergency Plans

User will draw / edit and share emergency plans using ArcGIS or any other software (including free open source software) that is capable of managing and saving features in ESRI shapefile format. The shapefiles need to have WGS 1984 coordinate system in order to be properly displayed since underlying layers have this specific coordinate system. User can then upload multiple shapefiles of an emergency plan using the “Click and Specify shp and dbf files” button under “Upload Emergency Plan” section on the left side of the interface.

Figure. Upload Emergency Plan Panel

After user clicks on “Click and Specify shp and dbf files” button, user will have a dialog to specify the shapefile(s).

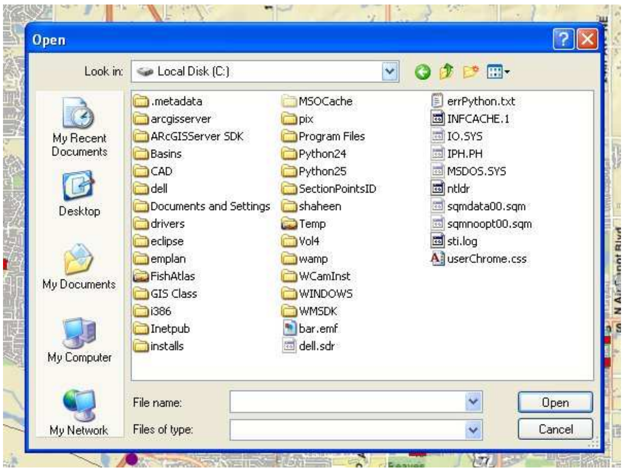

Figure. Dialog to upload shapefiles

User then browses into the folder containing emergency plan shapefiles, and selects them. User only needs to specify the files with ‘dbf’ and ‘shp’ extensions (user can do this by holding the control key down and selecting individual files); however it will also work if user selects all files for required shapefiles.

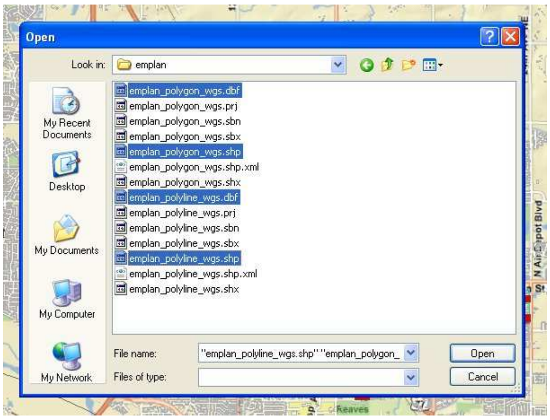

Figure. Files with Extensions ‘dbf’ and ‘shp’ Selected

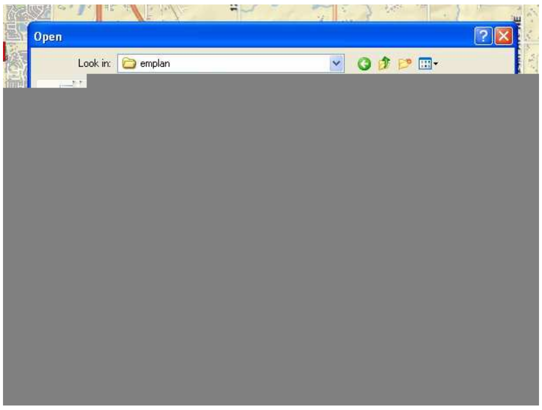

Figure. All Files of Required Shapefiles Selected

After selection, user needs to click “Open”, and the emergency plan will be uploaded to the web application. User can use the sample data (contained in emplan.zip file archive), to attach a plan that involves some arbitrary drawings around Oklahoma Memorial Stadium.

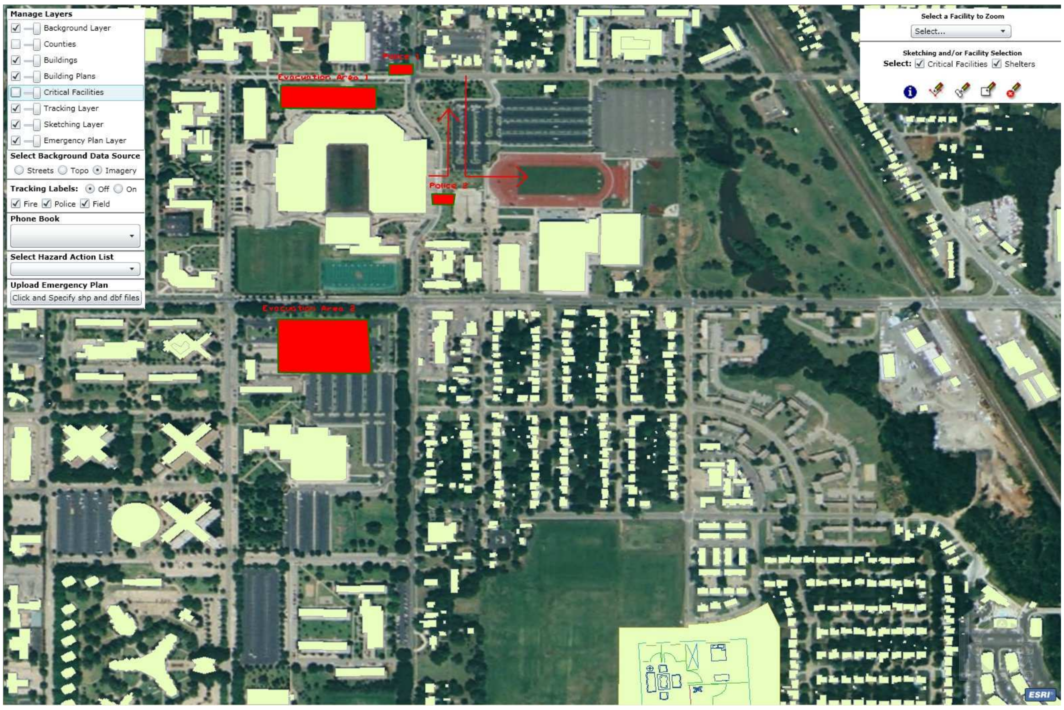

Figure. An Arbitrary Emergency Plan near Oklahoma Memorial Stadium Attached to the Application

6.6. Interface Description and User Manual for the Third Development Cycle of the Project

This application is developed by Naci Dilekli, a PhD student in University of Oklahoma, Department of Geography for his research.

Warning: This is a prototype application, and the purpose of it is to show functionality provided that there are correct and updated information in it. For the study, Norman, OK was chosen, and data was collected and generated accordingly. Data in this application is mostly arbitrarily and/or randomly generated, and is not to be used for any actual decision making.

1. Introduction

The purpose of this study is to develop and test a consolidated design methodology for web-based emergency management decision support systems (WEM-DSSs). A WEM-DSS is a decision support system utilizing recent developments in communications, especially Internet technology, for holistic and effective emergency management. Accordingly, an emergency management decision support system (EM-DSS) is a tool to assist emergency managers in all elements related to the holistic planning and management of emergencies, from efforts aiming to prevent emergencies, to preparing for the emergencies, to the management of the actual emergency response.

This application addresses a group of user stories that are collected through user surveys conducted by the researcher. This application was developed using ArcGIS Server and the ArcGIS API for Microsoft Silverlight. User needs to install Silverlight to run the application. Note that the application has only been tested in the MS Windows.

- For Windows: http://www.microsoft.com/silverlight/get-started/install/default.aspx
- For Mac OS: http://www.apple.com/downloads/macosx/development_tools/silverlight.html
- For Linux and UNIX: http://www.go-mono.com/moonlight/

The application consists of a panel on the left for managing the data, a panel for sketching and selecting on top right, and a panel for navigation on the bottom left of the interface. A list of supported features (user stories) that are developed specifically for this release is provided at the next section.

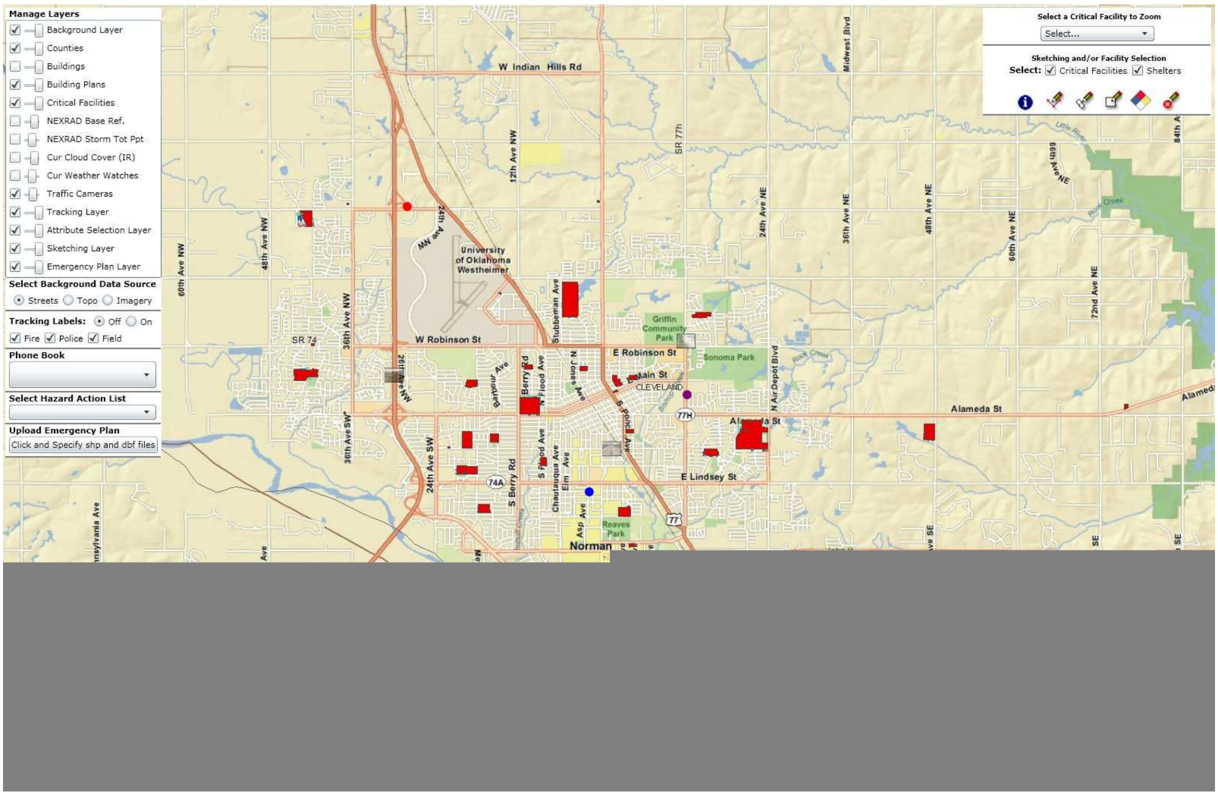

Figure. General View of the Application

2. Supported Features / User Stories in this Cycle

- User views NEXRAD Base Reflectivity Data
- User views CONUS NEXRAD Storm Total Precipitation data
- User views NWS Current Warnings
- User views CONUS GOES Infrared Satellite data for cloud cover
- User views live traffic cameras (prototype)
- User specifies initial isolation and protective action areas
- User views hazmat guides for spilled hazmat
- User views hazmat guides for buildings

3. NEXRAD Base Reflectivity Data

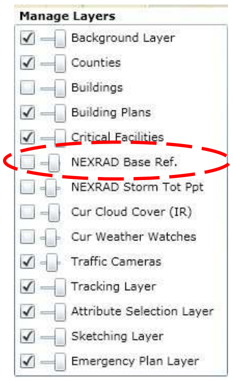

Figure. Layers Menu with NEXRAD Base Reflectivity layer highlighted

NEXRAD Base Reflectivity Data can be viewed by turning on this layer in the layers menu.

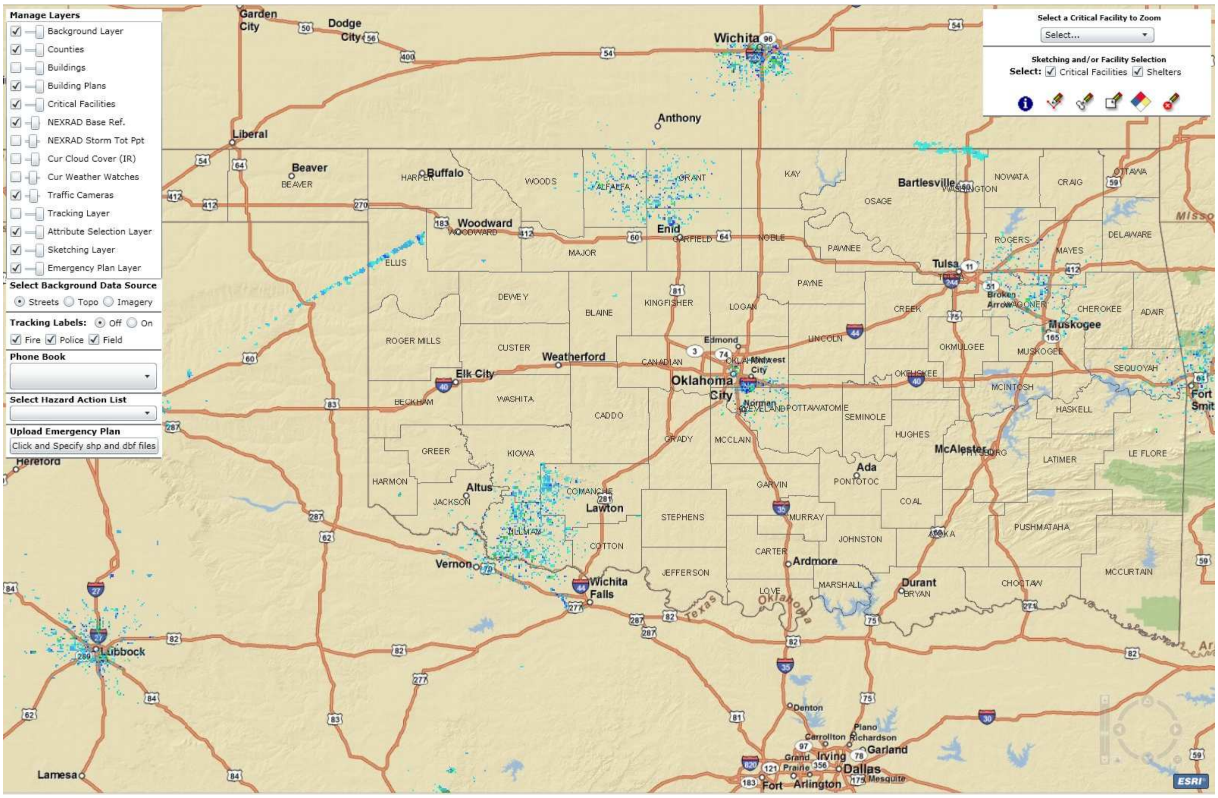

Figure. NEXRAD Base Reflectivity Data shown on the map

4. CONUS NEXRAD Storm Total Precipitation data

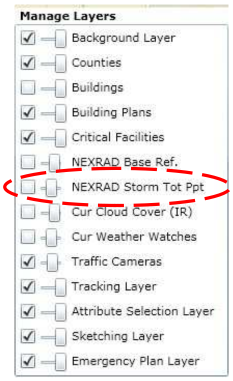

Figure. Layers Menu with CONUS NEXRAD Storm Total Precipitation layer highlighted

CONUS NEXRAD Storm Total Precipitation layer can be viewed by turning on this layer in the layers menu.

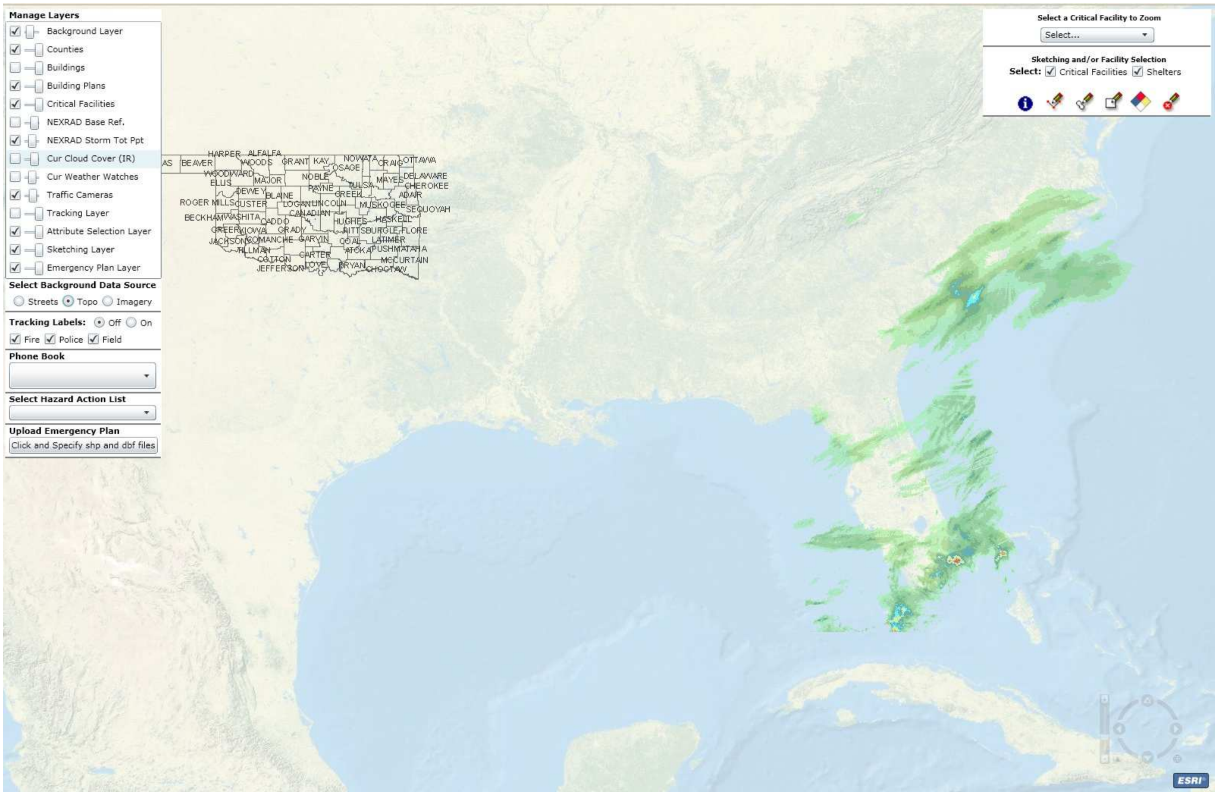

Figure. CONUS NEXRAD Storm Total Precipitation Data shown on the map

5. CONUS GOES Infrared Satellite Data

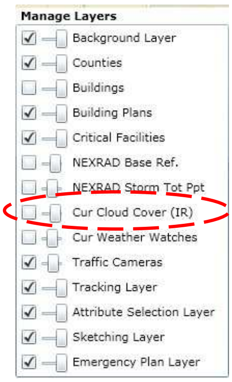

Figure. Layers Menu with CONUS GOES Infrared Satellite layer highlighted

CONUS GOES Infrared Satellite layer can be viewed to see the cloud cover by turning on this layer in the layers menu.

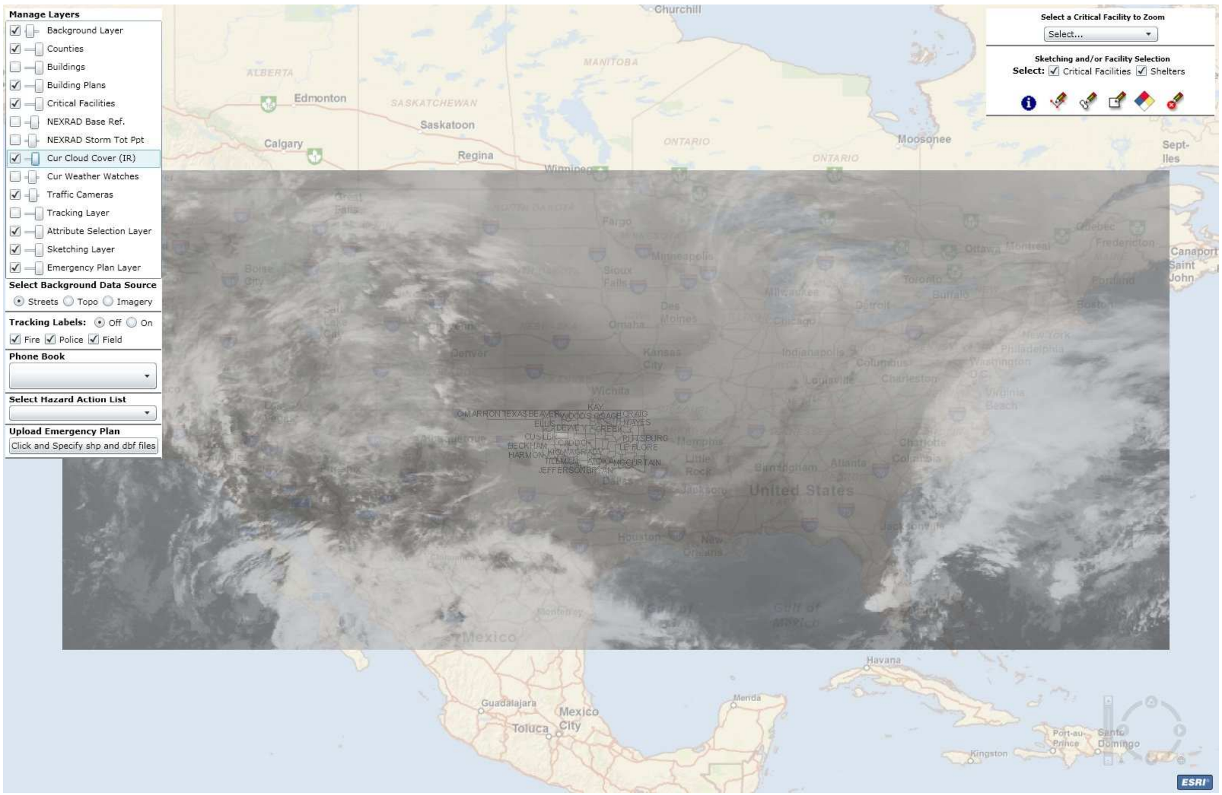

Figure. CONUS GOES Infrared Satellite Data shown on the map

6. NWS Current Warnings

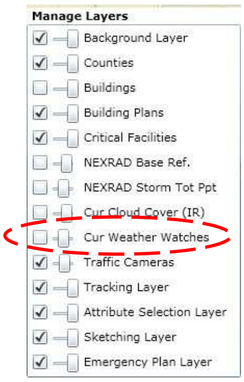

Figure. Layers Menu with NWS Current Warnings layer highlighted

NWS Current Warnings layer can be viewed by turning on this layer in the layers menu.

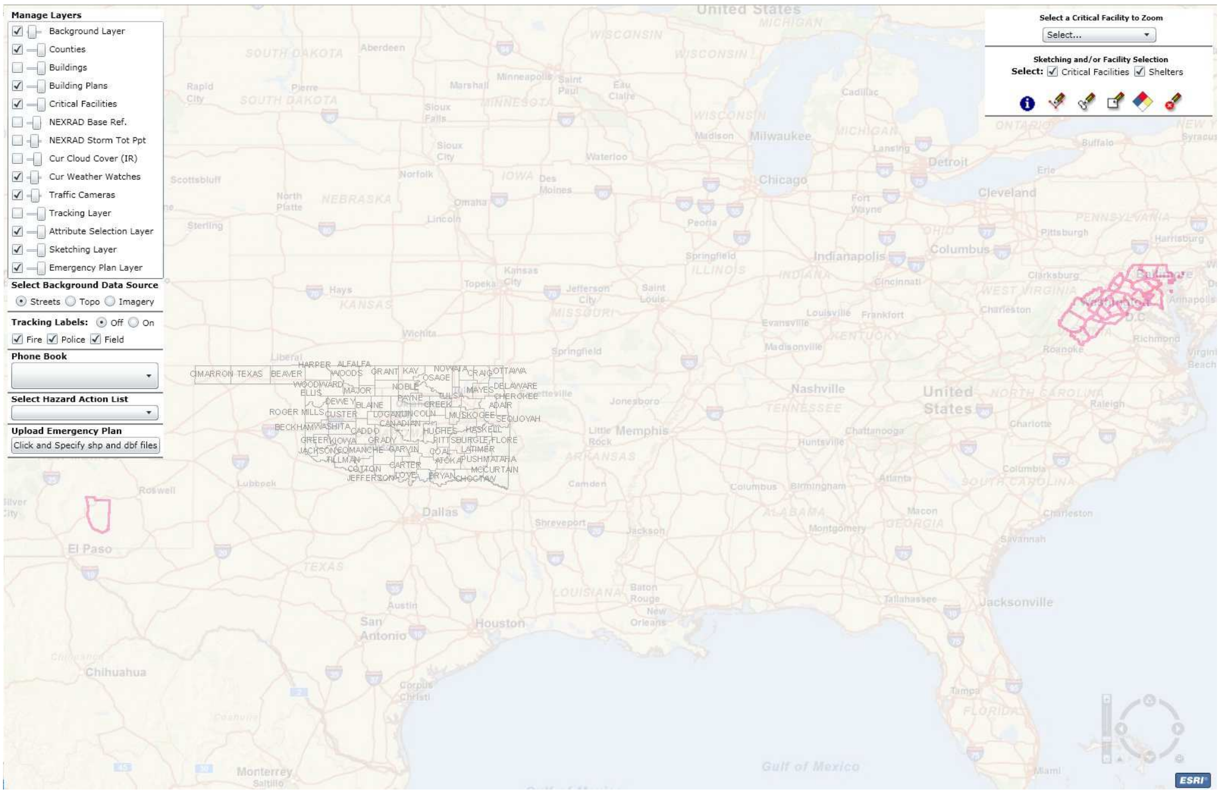

Figure. NWS Current Warnings shown on the map

7. Live Traffic Cameras

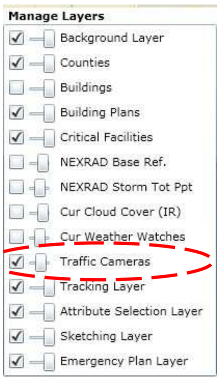

Figure. Layers Menu with Traffic Cameras layer highlighted

This is a prototype feature, that the cameras shown on the map do not stream the actual locations. By default Traffic Cameras layer is turned on. There are 3 traffic cameras, and these cameras may not always be available based on their maintenance and general network issues.

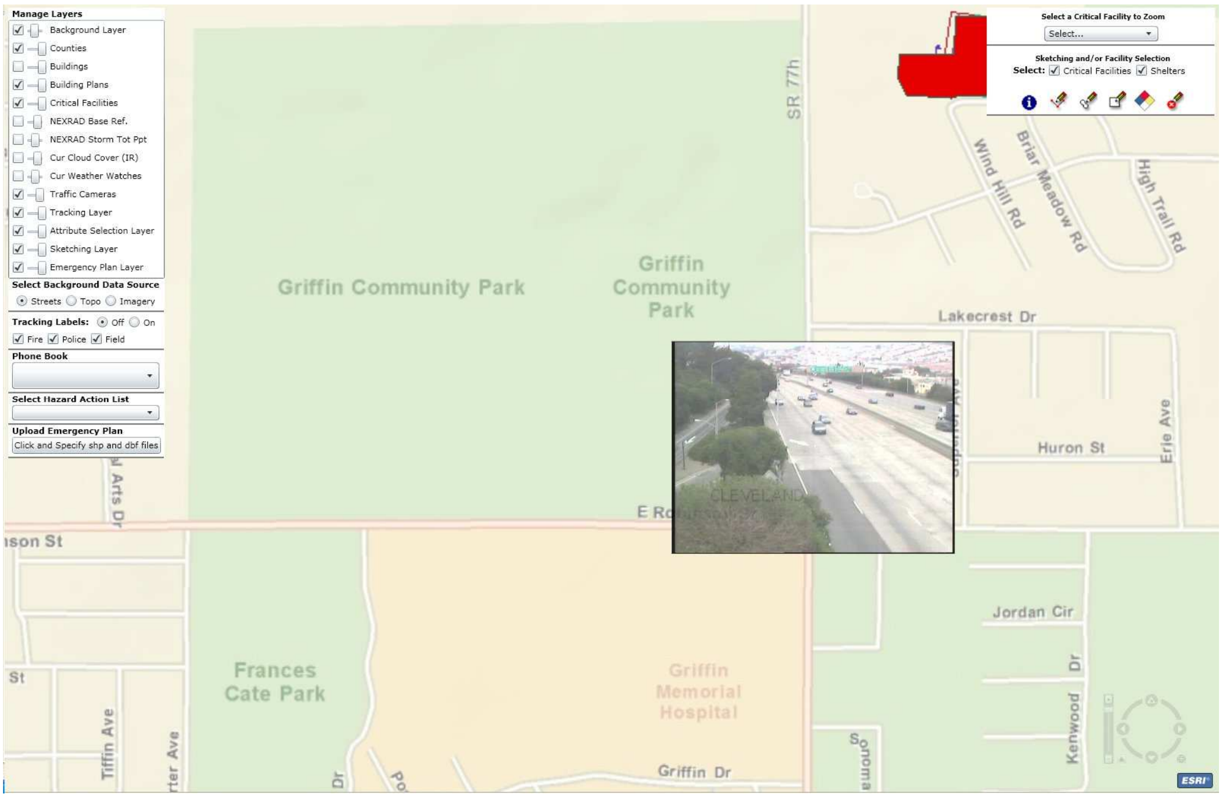

Figure. One camera stream is zoomed in

8. Managing Hazmat Features

With this release, the hazmat drawing tool is added among the sketching and selection tools. When clicked on the hazmat icon, the panel will be expanded so that the user can specify hazmat parameters.

Figure. Hazmat Icon highlighted and clicked

After user specifies the hazmat parameters, user can click on the map to specify the origin of the hazmat spill. After this, the origin location specified by a hazmat icon, a graphic indicating the initial isolation (red) area and the protective action (blue) area will be drawn. A guide (orange colored) that corresponds to the specified hazmat is also automatically displayed below the hazmat parameters. Like the other selection tools, the records for the critical facilities inside the drawn graphic are shown automatically in the graphical search results. These results are placed under the spilled hazmat guide.

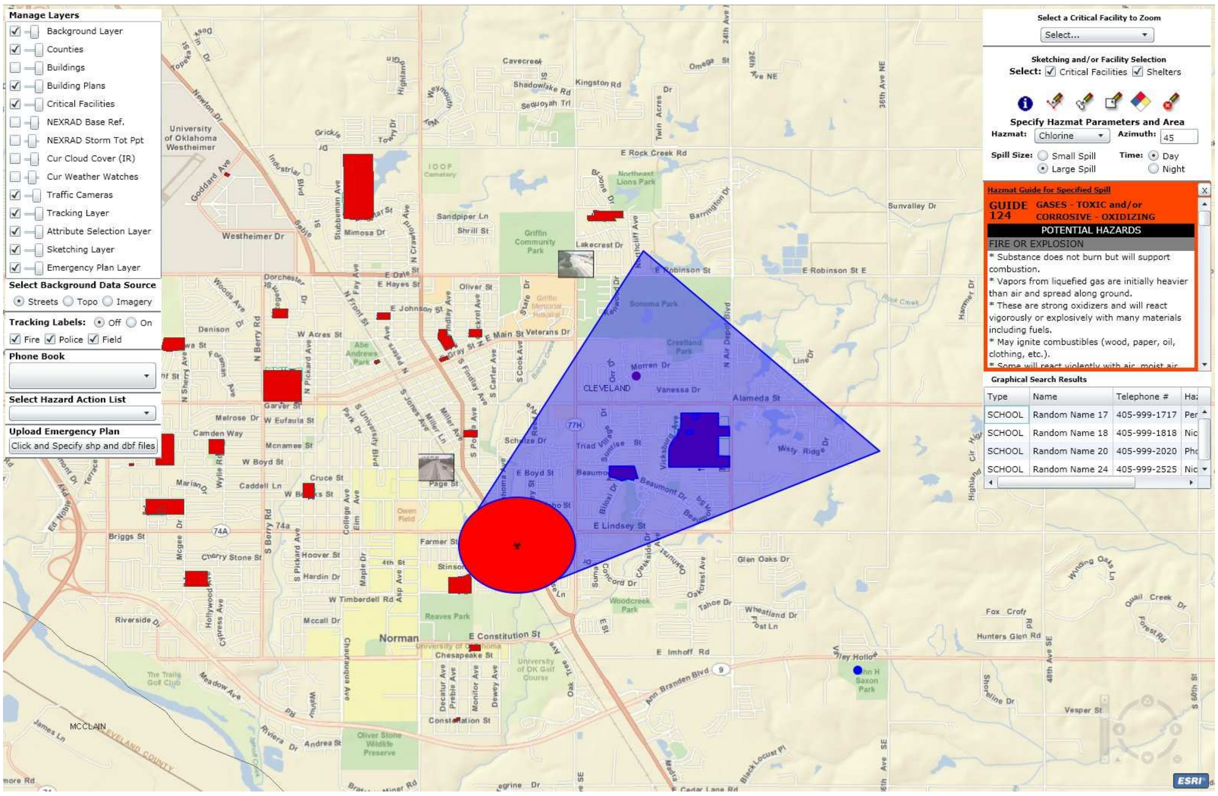

Figure. Hazmat area drawn, guide displayed, and overlaying critical facilities selected

User can also access the hazmat information that is contained in a building. To do this, user needs to click on a record in the graphical search results box, and the corresponding guide will be displayed under the graphical search results. Note that, graphical search results can be accessed with identify, polyline, polygon and rectangle selection tools as well. User can also access the hazmat information for the critical facilities using these tools.

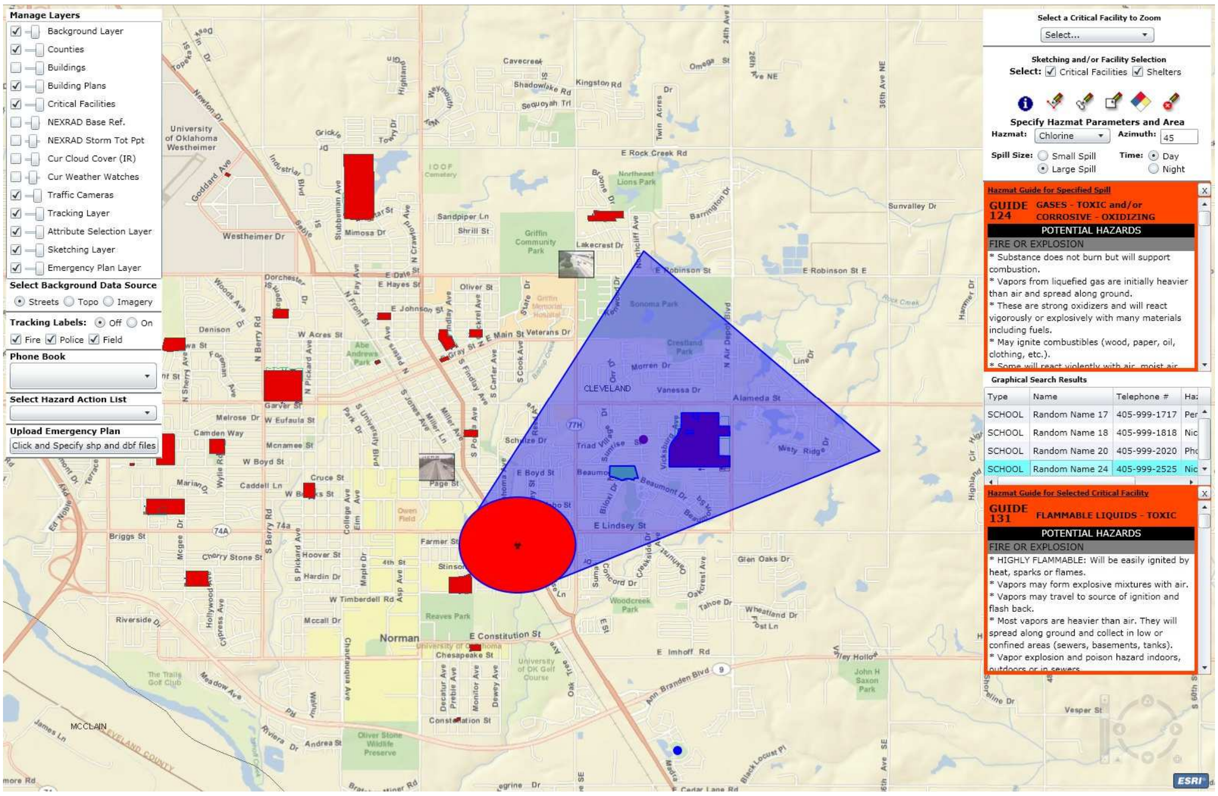

Figure. A critical facility record clicked, and corresponding hazmat guide displayed

The information on the right side of the interface may occupy much of the screen. In this case, user can close either or both of the hazmat guides.

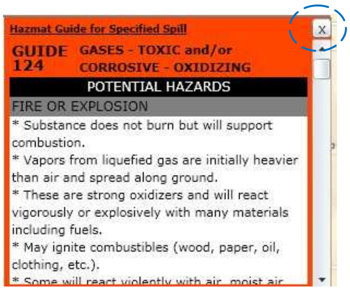

Figure. User can close a hazmat guide by the “X” button

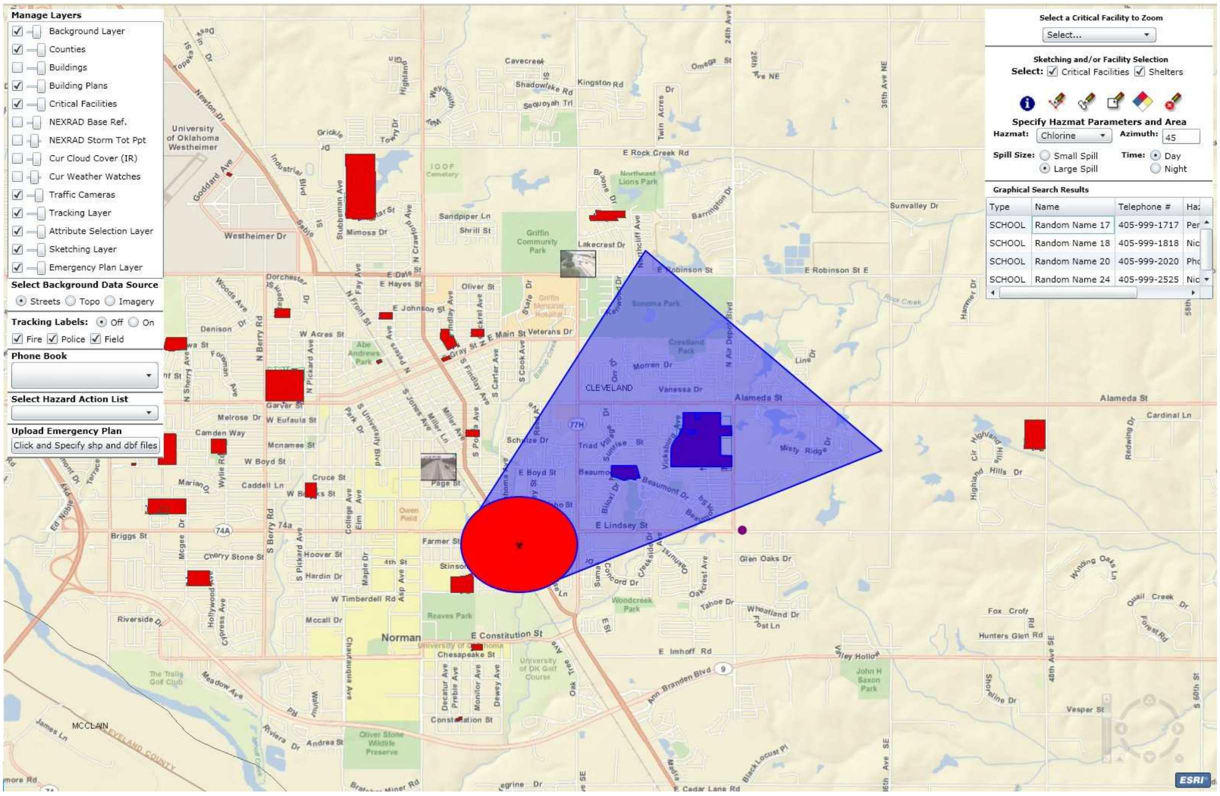

Figure. User interface after hazmat guides are closed

7. References

Alexander, D. (2000). Confronting Catastrophe: New Perspectives on Natural Disasters. New York, N.Y., Oxford University Press.

Anderson, D. J. (2005). Stretching Agile to fit CMMI Level 3 - the story of creating MSF for CMMI; Process Improvement at Microsoft Corporation. Proceedings of the Agile Development Conference, IEEE Computer Society.

Arnold, R. S. and S. Bohner (1996). Software Change Impact Analysis, IEEE Computer Society Press.

Aurum, A. and C. Wohlin (2005). Engineering and Managing Software Requirements, Springer-Verlag New York, Inc.

Avison, D. and G. Fitzgerald (2008). Information Systems Development: Methodologies, Techniques and Tools Maidenhead (Great Britain), McGraw-Hill Education.

Avison, D. and V. Taylor (1997). "Information Systems Development Methodologies: A Classification According to Problem Situation." Journal of Information Technology **12**: 73-81.

Beck, K. (1999). Extreme Programming Explained: Embrace Change. Boston, Addison-Wesley.

Beck, K., M. Beedle, et al. (2001). "Manifesto for Agile Software Development." Retrieved August 22, 2010, from http://www.agilemanifesto.org.

Bell, S. (1996). Learning with Information Systems: Analysis and Design in Developing Countries, Routledge.

Benediktsson, O., D. Dalcher, et al. (2006). "Comparison of software development life cycles: a multiproject experiment." Software, IEE Proceedings - **153**(3): 87-101.

Berander, P. and A. Andrews (2005). Requirements Prioritization. Engineering and Managing Software Requirements. A. Aurum and C. Wohlin. New York, Springer-Verlag**:** 69-94.

Berry, D. M. (2004). The Inevitable Pain of Software Development: Why There Is No Silver Bullet. Radical Innovation of Software and Systems Eng. in the Future, Proc. 2002 Monterey Conf., LNCS 2941, Springer.

Boehm, B. (2002). "Get Ready for Agile Methods, with Care." Computer **35**(1): 64-69.

Boehm, B. and W. Hansen (2001). "The Spiral Model as a Tool for Evolutionary Acquisition." Retrieved August 2, 2010, from http://www.stsc.hill.af.mil/crosstalk/2001/05/boehm.html.

Boehm, B. W. (1988). "A spiral model of software development and enhancement." Computer **21**(5): 61-72.

Bozheva, T. and M. Gallo (2005). Framework of Agile Patterns. Software Process Improvement**:** 4-15.

Brooks, F. P., Jr. (1987). "No Silver Bullet Essence and Accidents of Software Engineering." Computer **20**(4): 10-19.

Buragohain, C., D. Agrawal, et al. (2003). A Game Theoretic Framework for Incentives in P2P Systems. Proceedings of the 3rd International Conference on Peer-to-Peer Computing, IEEE Computer Society.

Carreño, M. L., O. Cardona, et al. (2007). "Urban Seismic Risk Evaluation: A Holistic Approach." Natural Hazards **40**(1): 137-172.

Cheng, R. and J. Vassileva (2006). "Design and evaluation of an adaptive incentive mechanism for sustained educational online communities." User Modeling and User-Adapted Interaction **16**(3-4): 321-348.

CMMI Product Team (2006). CMMI for Development, Version 1.2. Technical Report CMU/SEI-2006-TR-008. Pittsburgh, PA, Carnegie Mellon University/Software Engineering Institute.

Cohn, M. (2004). User Stories Applied: For Agile Software Development, Addison Wesley Longman Publishing Co., Inc.

Currion, P., C. d. Silva, et al. (2007). "Open source software for disaster management." Commun. ACM **50**(3): 61-65.

Cutter, S. L. (2001). "A research agenda for vulnerability science and environmental hazard." Newsletter for the International Human Dimensions Programme on Global Environmental Change **2**(1): 8-9.

Dahlstedt, Å. and A. Persson (2005). Requirements Interdependencies: State of the Art and Future Challenges. Engineering and Managing Software Requirements. A. Aurum and C. Wohlin. New York, Springer-Verlag**:** 95-116.

Deek, F. P., J. A. M. McHugh, et al. (2005). Strategic Software Engineering: An Interdisciplinary Approach. Boca Raton, FL, Auerbach Publications.

Denger, C. and T. Olsson (2005). Quality Assurance in Requirements Engineering. Engineering and Managing Software Requirements. A. Aurum and C. Wohlin. New York, Springer-Verlag**:** 163-185.

Department of Homeland Security (2008). National Response Framework Washington, DC.

Ehrhardt, J. (1997). "The RODOS System: Decision Support for Off-Site Emergency Management in Europe." Radiat Prot Dosimetry **73**(1-4): 35-40.

Endsley, M., B. Bolte, et al. (2003). Designing for situation awareness: an approach to user-centered design, London; New York: Taylor Francis.

Endsley, M. R. (1988). Design and Evaluation for Situation Awareness Enhancement, Human Factors Society.

Endsley, M. R. (1993). "A Survey of Situation Awareness Requirements in Air-to-Air Combat Fighters." International Journal of Aviation Psychology **3**(2): 1-23.

Endsley, M. R. (1995). "Toward a theory of situation awareness in dynamic systems: Situation awareness." Human factors **37**(1): 32-64.

Erickson, P. A. (2006). Emergency Response Planning: For Corporate and Municipal Managers. Burlington, MA, Elsevier Butterworth-Heinemann.

Farzan, R., J. M. DiMicco, et al. (2008). Results from deploying a participation incentive mechanism within the enterprise. Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems. Florence, Italy, ACM.

Goddard, S., S. K. Harms, et al. (2003). "Geospatial decision support for drought risk management." Commun. ACM **46**(1): 35-37.

Goldfinch, S. (2007). "Pessimism, Computer Failure, and Information Systems Development in the Public Sector." Public Administration Review **67**(5): 917-929.

Golle, P., K. Leyton-Brown, et al. (2001). Incentives for Sharing in Peer-to-Peer Networks. Proceedings of the Second International Workshop on Electronic Commerce, Springer-Verlag.

Grünbacher, P. and N. Seyff (2005). Requirements Negotiation. Engineering and Managing Software Requirements. A. Aurum and C. Wohlin. New York, Springer-Verlag**:** 143-162.

Hanna, M. (1995). Farewell to waterfalls? Software Magazine. **15:** 38-ff.

Holsapple, C. W. and A. B. Whinston (2001). Decision Support Systems: A Knowledge Based Approach. Cincinnati, OH, Thomson Learning Inc.

IEEE (1990). IEEE standard glossary of software engineering terminology. IEEE Std 610.12-1990.

Jayaratna, N. (1994). Understanding and Evaluating Methodologies: NIMSAD, a Systematic Framework, McGraw-Hill, Inc.

Jianshe, D., W. Shuning, et al. (1994). "Computerized support systems for emergency decision making." Annals of Operations Research **51**(1): 315-325.

Jönsson, P. and M. Lindvall (2005). Impact Analysis. Engineering and Managing Software Requirements. A. Aurum and C. Wohlin. New York, Springer-Verlag**:** 117-142.

Kähkönen, T. and P. Abrahamsson (2004). Achieving CMMI Level 2 with Enhanced Extreme Programming Approach. Product Focused Software Process Improvement**:** 378-392.

Kershaw, P. J. and B. Mason (2006). The Indian Ocean Tsunami Disaster: Implications for U.S. and Global Disaster Reduction and Preparedness. Summary of the June 21, 2005 Workshop of the Disasters Roundtable N. R. Council. Washington, DC, The National Academies.

Kollock, P. and M. Smith (1999). The Economies of Online Cooperation: Gifts and Public Goods in Cyberspace. Communities in Cyberspace, Routledge**:** 220-239.

Kurbel, K. E. (2008). The Making of Information Systems: Software Engineering and Management in a Globalized World, Springer.

Levy, J. K., J. Hartmann, et al. (2007). "Multi-Criteria Decision Support Systems for Flood Hazard Mitigation and Emergency Response in Urban Watersheds¹." JAWRA Journal of the American Water Resources Association **43**(2): 346-358.

Lianfa, L., m. J. Wang, et al. (2005). "Typhoon insurance pricing with spatial decision support tools." International Journal of Geographical Information Science **19**(3): 363-384.

Manktelow, K. and J. Jones (1987). Priniciples from the psychology of thinking and mental models. Applying cognitive psychology to user-interface design. M. M. Gardiner and B. Christie. New York, NY, John Wiley & Sons, Inc.**:** 83-117.

Marçal, A., B. de Freitas, et al. (2008). "Blending Scrum practices and CMMI project management process areas." Innovations in Systems and Software Engineering **4**(1): 17-29.

Martinsson, J. (2003). Maturing XP through the CMM. Proceedings of the 4th international conference on Extreme programming and agile processes in software engineering. Genova, Italy, Springer-Verlag.

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

McLoughlin, d. (1985). "A Framework for Integrated Emergency Management." Public Administration Review **45**(Special Issue: Emergency Management: A Challenge for Public Administration): 165-172.

McPherson, R. A., C. A. Fiebrich, et al. (2007). "Statewide Monitoring of the Mesoscale Environment: A Technical Update on the Oklahoma Mesonet." Journal of Atmospheric and Oceanic Technology **24**(3): 301-321.

Mileti, D. S. (1999). Disasters by Design: A Reassessment of Natural Hazards in the United States. Washington, D.C., Joseph Henry Press.

Mitchell, J. K. (1999). "Megacities and Natural Disasters: A Comparative Analysis." GeoJournal **49**: 137-142.

Morris, D. A. (1998). The Design and Implementation of a Web-Based Decision-Support System for Public Safety Agencies. Preprints, 14th International Conference on Interactive Information and Processing Systems for Meteorology, Oceanography, and Hydrology, American Meteorological Society, Phoenix, Arizona.

Morris, D. A., K. C. Crawford, et al. (2002). "OK-FIRST: an example of successful collaboration between the meteorological and emergency response communities on 3 May 1999." Wea. Forecasting **17**: 567-576s.

Morris, D. A., K. C. Crawford, et al. (2001). "OK-FIRST: A meteorological information system for public safety." Bulletin of the American Meteorological Society **82**(9): 1911-1923

Naur, P. and B. Randell (1969). Software Engineering: Report of a conference sponsored by the NATO Science Committee, Garmisch, Germany, 7-11 Oct. 1968. Brussels, Scientific Affairs Division, NATO.

Nogueira, J. C., C. Jones, et al. (2000). Surfing the Edge of Chaos: Applications to Software Engineering. Proceedings of 2000 Command and Control Research and Technology Symposium. Monterey, CA**:** 1-13.

Paulk, M. C. (2001). "Extreme Programming from a CMM Perspective." IEEE Softw. **18**(6): 19-26.

Pick, J. B. (2008). Geo-Business: GIS in the Digital Organization. Hoboken, N.J. , John Wiley & Sons.

Pikkarainen, M. and A. Mäntyniemi (2006). An Approach for using CMMI in Agile Software Development Assessments: Experiences of Three Case Studies. SPICE. Luxemburg.

Power, D. J. (2004). Decision Support Systems: Frequently Asked Questions. Lincoln, NE, iUniverse.

Pressman, R. S. (2005). Software Engineering: A practitioner's Approach. New York, McGraw-Hill.

Quarantelli, E. (1999). "Implications for programmes and policies from future disaster trends." Risk Management: An International Journal **1**(1): 9-19.

Reeve, D. E. and J. R. Petch (1999). GIS Organizations and People: A Socio-technical Approach. London, UK, Taylor and Francis.

Reifer, D. J. (2003). "XP and the CMM." IEEE Software **20**: 14-15.

Royce, W. (1970). Managing the Development of Large Software Systems.

Santos, P. (2007). "SCRUM Meets CMMI -- Agility and discipline combined." Dr. Dobbs Magazine. Retrieved August 2, 2010, from http://www.drdobbs.com/architecture-and-design/201202684.

Schenker-Wicki, A. (1997). "The Swiss decision support system : State of the art and experiences." Radiation protection dosimetry **73**(1-4): 119-122.

Schneider, P. J. and B. A. Schauer (2006). "HAZUS---Its Development and Its Future." Natural Hazards Review **7**(2): 40-44.

Shine Technologies (2003). "Agile Methodologies Survey." Retrieved August 2, 2010, from http://www.shinetech.com/download/attachments/98/ShineTechAgileSurvey2003-01-17.pdf.

Silva, F. N. d. (2001). "Providing spatial decision support for evacuation planning: a challenge in integrating technologies." Disaster Prevention and Management **10**(1): 11-20.

Stephens, M. and D. Rosenberg (2003). Extreme Programming Refactored: The Case Against XP. Berkeley, CA, Apress.

Tolentino Jr., A. S. (2007). "The Challenges of Tsunami Disaster Response Planning and Management." International Review for Environmental Strategies: Best Practice on Environmental Policy in Asia and the Pacific **7**(1): 147-154.

Tralli, D. M., R. G. Blom, et al. (2005). "Satellite remote sensing of earthquake, volcano, flood, landslide and coastal inundation hazards." ISPRS Journal of Photogrammetry and Remote Sensing **59**(4): 185-198.

Turban, E. and J. Aronson, E. (2000). Decision Support Systems and Intelligent Systems. Upper Saddle River, N.J., Prentice Hall PTR.

Turner II, B. L., P. A. Matson, et al. (2003). "Science and Technology for Sustainable Development: Illustrating the coupled human-environment system for vulnerability analysis: Three case studies." Proceedings of the National Academy of Sciences **100**(5): 8080-8085.

Turner, R. (2002). Agile Development: Good Process or Bad Attitude? Product Focused Software Process Improvement**:** 134-144.

Turner, R. and A. Jain (2002). Agile Meets CMMI: Culture Clash or Common Cause? Extreme Programming and Agile Methods — XP/Agile Universe 2002**:** 153-165.

Turoff, M. (2002). "Past and future emergency response information systems." Commun. ACM **45**(4): 29-32.

US Department of Transportation (2008). 2008 Emergency Response Guidebook: A Guidebook for First Responders During the Initial Phase of a Dangerous Goods/Hazardous Materials Incident, Department of Transportation.

Vassileva, J., R. Cheng, et al. (2004, August 2, 2010). "Stimulating User Participation in a File-Sharing P2P System Supporting University Classes " P2P Journal. Retrieved August 2, 2010, from http://madmuc.usask.ca/Papers/P2PJ.pdf.

Wailgum, T. (2007). From Here to Agility ; Study after study indicates that agile methodologies produce better results in software development and project management. So why have so few CIO's adopted them? Read to find out how you can change your organization's culture of development. CIO Magazine. **20:** 1.

Wells, D. (2006). "Extreme Programming: A gentle introduction." Retrieved August 2, 2010, from http://www.extremeprogramming.org/.

Wybo, J. L. (1998). "FMIS: a decision support system for forest fire prevention and fighting." Engineering Management, IEEE Transactions on **45**(2): 127-131.

Zowghi, D. and C. Coulin (2005). Requirements Elicitation: A Survey of Techniques, Approaches, and Tools. Engineering and Managing Software Requirements. A. Aurum and C. Wohlin. New York, Springer-Verlag**:** 19-46.