Security Quality Requirements Engineering (SQUARE): Case Study Phase III

Lydia Chung
Frank Hung
Eric Hough
Don Ojoko-Adams

Advisor
Nancy R. Mead

May 2006

SPECIAL REPORT
CMU/SEI-2006-SR-003

Carnegie Mellon
Software Engineering Institute

Pittsburgh, PA 15213-3890

Security Quality Requirements Engineering (SQUARE): Case Study Phase III

CMU/SEI-2006-SR-003

Lydia Chung
Frank Hung
Eric Hough
Don Ojoko-Adams

Advisor
Nancy R. Mead

May 2006

Networked Systems Survivability Program

Unlimited distribution subject to the copyright.

This report was prepared for the

SEI Administrative Agent
ESC/XPK
5 Eglin Street
Hanscom AFB, MA 01731-2100

The ideas and findings in this report should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

This work is sponsored by the U.S. Department of Defense. The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense.

Copyright 2006 Carnegie Mellon University.

NO WARRANTY

THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

Use of any trademarks in this report is not intended in any way to infringe on the rights of the trademark holder.

Internal use. Permission to reproduce this document and to prepare derivative works from this document for internal use is granted, provided the copyright and "No Warranty" statements are included with all reproductions and derivative works.

External use. Requests for permission to reproduce this document or prepare derivative works of this document for external and commercial use should be addressed to the SEI Licensing Agent.

This work was created in the performance of Federal Government Contract Number FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.

For information about purchasing paper copies of SEI reports, please visit the publications portion of our Web site (http://www.sei.cmu.edu/publications/pubweb.html).

Table of Contents

List of Figures

List of Tables

Abstract

This special report is the third in a series by the Software Engineering Institute focusing on the practical application of the Security Quality Requirements Engineering (SQUARE) process. In this report, a student team presents their results of working with three clients over the course of a semester. Each client was developing a large-scale software application and worked with the students to generate security requirements. The students' main contribution to the SQUARE process was to determine how existing software requirements-elicitation techniques could be applied to software security requirements (as opposed to end-user requirements).

With each client, the students implemented a different structured requirements-elicitation technique: Issue-Based Information Systems with an information technology firm, Joint Application Development (JAD) with the Delta client, and the Accelerated Requirements Method (ARM) with the Beta client. The ARM technique, which is a variant of JAD, held the most promise for inclusion in future applications of SQUARE. In addition to an analysis of the three elicitation techniques, the student team also generated feedback and recommendations on different steps of the SQUARE process, such as requirements prioritization and inspection. They found the Analytic Hierarchy Process to be highly useful for prioritizing requirements quickly; however, they did not find a requirements inspection technique that was well suited for any of the clients.

1 Introduction

1.1 The SQUARE Process

The System Quality Requirements Engineering (SQUARE) methodology is a process developed by Dr. Nancy Mead at the Networked Systems Survivability (NSS) Program in the Carnegie Mellon® Software Engineering Institute. The goal of the nine-step process is to improve the integration of security requirements in the product development life cycle. Briefly, the steps are as follows:

1. Agree on definitions.
2. Identify safety and security goals.
3. Develop artifacts to support requirements definition.
4. Perform risk assessment.
5. Select requirements-elicitation technique(s).
6. Elicit security requirements.
7. Categorize requirements by level (system, software, or other) and type (whether they are requirements or other kinds of constraints).
8. Prioritize requirements.
9. Perform requirements inspection.

A more complete description of the steps can be found in Appendix A on page 69. Since its inception, the SQUARE process has been used in four case studies with real-world industry clients. Through analyzing these case studies and their results, two teams of graduate students at Carnegie Mellon have provided meaningful feedback to the NSS Program and helped to refine the SQUARE methodology.

In this report, we describe our experience using the SQUARE process with three clients during the summer of 2005. We are concerned with Steps 5–9 of the process; in particular we have attempted to analyze several structured requirements-engineering techniques as they apply to security requirements.

® Carnegie Mellon is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

1.2 Case Study Clients

1.2.1 Acme Group

The Acme Group is a privately owned company that provides technical and managerial consulting services to a wide array of governmental and private clients. Headquartered in Pittsburgh, the company employs a staff of over 1,000 in multiple sites across the United States.

One of four subsidiaries of the Acme Group, the subsidiary the team worked for specializes in emergency planning and asset management tools. Through this subsidiary, the Acme Group offers an asset management system used for emergency planning and disaster handling. This system offers a graphical interface for users communicating among their peers and monitoring the status of emergencies and disasters. This system integrates Archibus/FM, AutoCAD, and Geographical Information Systems (GIS) to offer enterprise-level emergency management services.

1.2.2 Delta Project

To help users understand and reduce software problems, the Delta client and others are working together to develop a Web site to provide information that can improve software engineering. This project will include a broad range of information to help software developers and architects.

1.2.3 Beta

The Beta team is part of a national center located in the U.S.

The Beta team receives, analyzes, and coordinates reports. Because its current system has limited capability to process information, the Beta team plans to enable collaborative work and greater data exchange capabilities. The Beta team is in the process of reengineering core business processes and will refashion existing tools to support the new processes.

2 Step 5: Select Elicitation Techniques

Our student team's work in Step 5 of the SQUARE process consisted of researching and selecting several elicitation techniques that would be experimented with over the summer. Our objective was to select a tool that would provide a comprehensive framework to cull the necessary requirements from the client and to compare how a particular elicitation technique worked with its designated project. This step was completed without input from the client, and the student team selected the elicitation techniques. By completing this step, the team became familiar with the structured elicitation techniques and gained an understanding of the data it would need to request from the client. The team researched many methodologies and selected three elicitation techniques—one to be implemented with each project over the summer term.

2.1 Methodology

The team began by researching the established software-engineering requirements methodologies in a literature review of well-known elicitation techniques provided by the team's faculty advisor. The team researched the methodologies separately and transmitted copies of the most promising methodologies to an online portal site used as a communications hub. After several days of research, the team convened to discuss and rate each of the following elicitation techniques (see Table 1 for the ratings):

- misuse cases (used to generate requirements in the previous year's project)
- soft systems methodology (SSM)
- quality function deployment (QFD)
- controlled requirements expression (CORE)
- issue-based information systems (IBIS)
- joint application development (JAD)
- feature-oriented domain analysis (FODA)
- critical discourse analysis (CDA)
- accelerated requirements method (ARM)

To evaluate the elicitation techniques, the team used the following criteria:

- **adaptability—**the ability to generate requirements in multiple environments (Would the elicitation work well with a software package that is near completion and a project in the planning stages?)

- **computer-aided software engineering (CASE) tool—**"a computer-based product aimed at supporting one or more software engineering activities within a software development process" [SEI 04]
- **client acceptance—**the likelihood that the clients would agree to the elicitation technique in analyzing their requirements (Is the implementation too invasive in a business environment? Can the elicitation technique be implemented in a reasonable amount of time?)
- **complexity—**the degree of difficulty in understanding and properly executing the elicitation technique (Can the requirement engineers and clients easily understand the elicitation technique?)
- **graphical output—**the ability of the elicitation technique to produce readily understandable visual artifacts
- **implementation duration—**the length of time the requirements engineers and clients need to fully execute the elicitation technique
- **learning curve—**the pace at which the requirements engineers and clients can fully comprehend the elicitation technique
- **maturity—**the time, exposure, and analysis the elicitation technique has experienced in its vetting by the requirements-engineering community
- **scalability—**the ability of the elicitation technique to address the requirements of enterprise-level systems and small-scale applications

Table 1: Comparison of Elicitation Techniques

3 = Very Good, 2 = Fair, 1 = Poor

	Misuse Cases	SSM	QFD	CORE	IBIS	JAD	FODA	CDA	ARM
Adaptability	3	1	3	2	2	3	2	1	2
CASE Tool	1	2	1	1	3	2	1	1	1
Client Acceptance	2	2	2	2	3	2	1	3	3
Complexity	2	2	1	2	3	2	1	1	2
Graphical Output	2	2	1	1	2	1	2	2	3
Implementation Duration	2	2	1	1	2	1	2	2	3
Learning Curve	3	1	2	1	3	2	1	1	1
Maturity	2	3	3	3	2	3	2	2	1
Scalability	1	3	3	3	2	3	2	1	2
Total Score	**18**	**18**	**17**	**16**	**22**	**19**	**14**	**14**	**18**

The team ultimately decided to use JAD, ARM (a variation of JAD), and IBIS for the three projects undertaken for the summer term. As Table 1 illustrates, these three techniques were subjectively ranked to be the most suitable candidates for the case studies. As indicated by the total

score for each technique, ARM tied with misuse cases and SSM. However, the team chose ARM due to its more structured, stepwise approach to requirements elicitation.

2.2 Client Feedback

The client was not involved in this step of the project.

2.3 Recommendations

We recommend that future teams examine the possibility of combing IBIS and CORE due to their respective strengths in information gathering and analysis: the maturity and scalability of CORE combined with the graphical output, CASE tools, and ease of use of IBIS. We also recommend developing many artifacts—such as architecture diagrams, end-user requirements documents, and preliminary documentation—to assist with the selection and implementation of an elicitation technique.

We advise future teams to consider the time necessary to implement an elicitation technique. In the same vein, the team should consider the time needed to learn a new tool in an implementation of a technique. The team should also select an elicitation technique that meets the needs of a diverse group of stakeholders, in order to address a broader range of security requirements.[1]

[1] In our context, a stakeholder is a person or group that could affect or be affected by the software or system product.

3 Step 6: Elicit Security Requirements

After completion of Step 5, the team decided to use IBIS, ARM, and JAD to elicit security requirements for Acme, Beta, and Delta, respectively. In this section, we describe in detail the team's experience in the application of each technique, providing recommendations when possible.

3.1 IBIS

IBIS was developed in the 1970s to improve the definition, discussion, and resolution of "wicked" problems. That is, the methodology works best with issues that are ill defined or hotly contentious among stakeholders.

In IBIS, all problems are decomposed into *issues* that are phrased in the form of open questions to the stakeholders. One question might be, for instance, "How should the system guard against insider threats, if at all?" Each issue is then resolved by proposed *positions* that are resolutions to the issue put forth by the stakeholders. Every position has corresponding *arguments* that either support or oppose the position.

The requirements engineer is tasked with recording the articulation of issues, positions, and arguments. The results are then presented in the form of an IBIS *map* (IM). Figure 1 is an example of such a map. The IBIS maps are analyzed by the requirements-engineering team and client to elicit the actual security requirements.

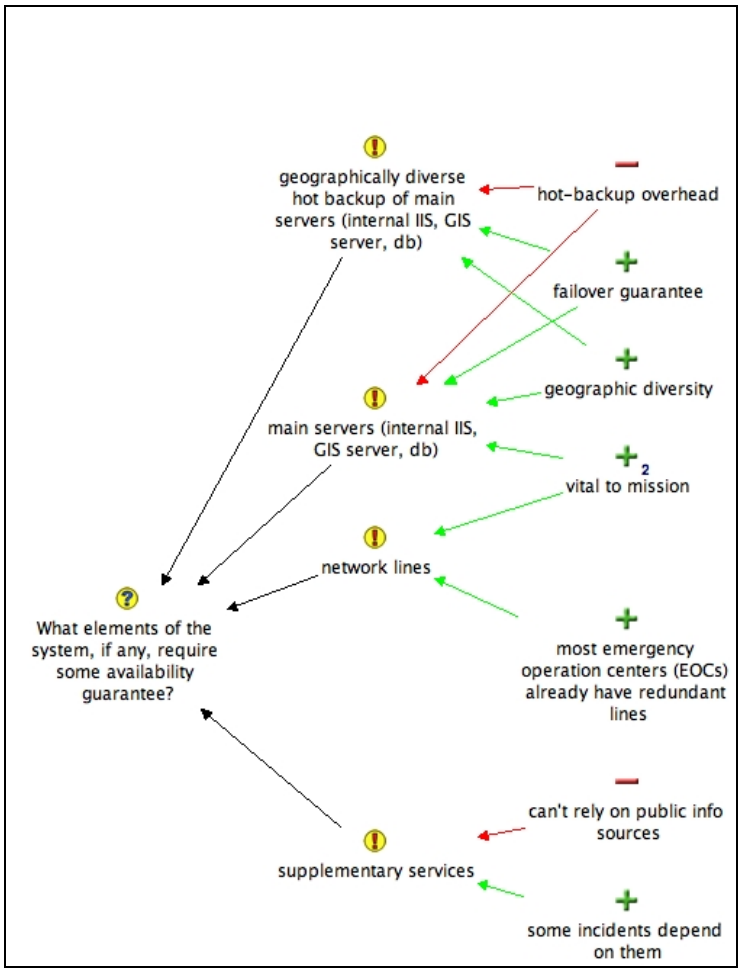


Figure 1: Example of an IBIS Map Generated with Compendium

3.1.1 Methodology

First, the team formulated a set of questions that would likely cover every aspect of security that could affect the system. We tried to identify any and all questions that would cover confidentiality, availability, and integrity of the system. Many of our questions were based on the artifacts that were collected by the previous SQUARE team. Then, after addressing these questions with Acme Group stakeholders in our first face-to-face meeting, we discovered overlaps and gaps in the scope of our questions. This led us to revise our set of questions. The initial questions, along with the corresponding revisions, can be found in Appendix C on page 79.

Using our notes from the initial meeting, we then created a set of IBIS maps using the Compendium CASE tool, which is freely available from the Compendium Institute's Web site.[2] This software was very easy to use, and we were able to create our set of maps quickly.

[2] For more information, visit http://www.compendiuminstitute.org/.

Since the maps didn't contain any inherent identification information, we attached a table to each map to identify and reference it. The tables, which we call IBIS map descriptors, contain information such as the positions, affected security attributes, and related security requirements. Figure 2 shows one of our IBIS map descriptor tables. The full list of IBIS maps and map descriptors can be found in Appendices D and E, beginning on page 83.

Number	IMD-04
Issue	What type of authentication, if any, should the system utilize?
Positions	Integrated Windows logon SSL-protected login screen
Security Attributes Affected	x Confidentiality _ Integrity _ Availability
Survivability Attributes Affected	x Resist _ Evolve _ Recognize _ Recover
Related IBIS Map Descriptors	IMD-15, IMD-16
Related Security Requirement(s)	SR-1, SR-2, SR-3, SR-5, SR-6

Figure 2: Example IBIS Map Descriptor

Using these maps, along with feedback from Acme, we were able to generate an initial set of security requirements for the system. Unfortunately, IBIS does not provide a mechanism for translating the maps into requirements, so our requirements were based on feedback from Acme Group during the meetings and our own recommendations. (See our categorized list of security requirements on page 28.)

3.1.2 Client Feedback

In general, the client was not satisfied with the IBIS maps. Aside from visually identifying the related pros and cons of a position, the maps were difficult to follow due to their large size and cluttered appearance. The client also commented that they were unclear about the purpose of the maps, and they did not like the inconsistency in the level of detail on each map.

Perhaps more importantly, the client was unable to see a connection between the IBIS maps and the security requirements that we generated. This is interesting to note, since the team had precisely this problem in generating the security requirements.

Aside from the negative experiences with the IBIS process itself, the client seemed satisfied that the interview generated discussion between stakeholders and raised security issues that otherwise would not have been addressed.

3.1.3 Recommendations

Given our experience with IBIS in eliciting security requirements, we do not recommend using it in the future.

Although IBIS maps are extremely effective in documenting complex discussions, they do not provide a structured means to generate security requirements. We found that, for many issues, the client only considered one position. In turn, we were forced to create an arbitrary number of alternative positions, even if the client had never considered them. By creating additional positions, we may have offered new solutions to the client, but we do not feel that this course of action produces the most accurate and complete set of requirements.

In the remainder of this section, we provide our recommendations in case another team decides to experiment with IBIS in the future.

3.1.3.1 Choose Interview Questions Wisely

The effectiveness of IBIS in eliciting security requirements depends on the quality of the interview questions. To the greatest extent possible, the scope of questions must cover the entire range of security requirements that could possibly involve the system. Most importantly, we found that the interviewer must be persistent in encouraging the stakeholders to explain their rationale when proposing a solution to an issue. By explaining *why* they have chosen such a position, the stakeholders can naturally discuss the pros and cons among themselves. The IBIS interview, then, simply needs to record the statements made during the discussion. In our case study with Acme, we had some difficulty in the creation of IBIS maps because we failed to ask consistently for the rationale of a particular viewpoint.

Questions that utilize the word *should* are well suited for IBIS interviews, since they force the stakeholder to provide answers that aid in eliciting the actual security requirements. For instance, "What type of intrusion detection mechanisms, if any, should the system utilize?" The IBIS question should be less concerned with the implementation—that is, with how the requirement will be met. However, using the word *how* is appropriate in cases where there is no doubt among all stakeholders of a particular requirement. For example, "How quickly should the system recover from failure?" implies that all stakeholders agree that the system must recover from failure. The IBIS interview must be certain, likely through artifact collection, that there is no ambiguity among stakeholders on this requirement.

We found that direct, nonleading questions work very well. For instance, we recommend asking questions such as "What measures should be in place to protect against configuration errors, if any?" rather than "How should the system respond to configuration errors?" The latter question implies an agreed-upon requirement, even though there may be debate as to whether the topic of the question is a requirement at all. Much like direct examination in judicial proceedings, direct questions allow the stakeholders to provide an honest, unbiased, and complete response to each question.

The IBIS questions must strike a balance between being generic enough for reuse in subsequent projects and too general to elicit specific responses. For example, the question "What type of security mechanisms should be used?" can be reused from project to project, but it is far too general to elicit a useful and complete response from the stakeholders. "Should the system use the Snort intrusion detection system?" is a question that would lead to a detailed response, but this question is overly specific. Instead, the IBIS interviewer should ask, "What type of intrusion detection system, if any, should the system utilize?"

The requirements engineer should also consider artifacts collected from Steps 1–4 of SQUARE when formulating IBIS questions. Upon examination of our first draft of IBIS questions for the client, we discovered that many of them would have been answered in the artifact collection phase of the process. Due to time constraints, and to our existing general knowledge of the client's project, we mistakenly included some artifact-related questions such as "What are the minimum computing requirements to run the software?" We suggest that IBIS questions be formulated carefully to exclude questions that were answered in earlier stages of SQUARE.

3.1.3.2 Include a Variety of Stakeholders

In addition to proper question selection, we found that the success of IBIS is directly proportional to the variety and level of participation of stakeholders in the project. In fact, IBIS works best when different stakeholders present opposing viewpoints, which is common in large-scale projects. By presenting differing viewpoints, the stakeholders are forced to discuss and analyze one another's positions while, we hope, reaching a consensus. Examples of stakeholders include software developers, hardware developers, network engineers, security managers, executives, end users, and marketing and finance managers. During the setup phase, well ahead of the actual interview session, the requirements engineer should emphasize the need for the client to include as many stakeholders as possible in the discussion.

3.1.3.3 Avoid Presenting Cluttered IBIS Maps

The Compendium software tool was easy to use and effective in generating IBIS maps. In order to avoid displaying extremely large maps (which our client found difficult to read), we recommend exploiting the *nested maps* feature in Compendium. This feature enables the team to "hide" some of the lower level details of the maps by nesting them inside other map elements, while maintaining the ability to drill down into the details if requested. In fact, a comment we received from the client indicated that such a hierarchical map structure would have been more beneficial in handling some of the larger maps.

Organize the maps to reduce clutter and orient each map in a consistent manner when printing out the maps for client review. We found that the client had some difficulty in determining the correct orientation of the pages when reviewing the maps.

3.2 ARM

ARM is designed to elicit, categorize, and prioritize security requirements. Therefore, ARM spans over Steps 6, 7, and 8 in the SQUARE methodology. The team spent two weeks completing the ARM process with Beta using their application as a testbed.

3.2.1 Methodology

The ARM methodology includes the three phases shown in Figure 3. We discuss each phase in detail. To present our experience with the ARM methodology in the clearest way, we will describe the entire process in this section on SQUARE Step 6 and reference portions of it in later discussions of Steps 7 and 8.

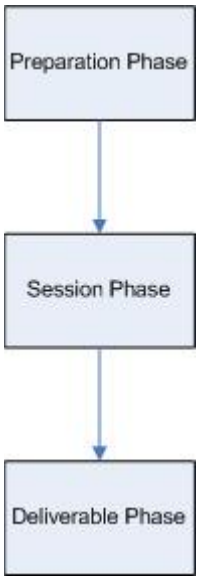


Figure 3: Three Phases of ARM

3.2.1.1 Preparation Phase

As the name implies, this phase is used to prepare for the Session Phase. There are six steps in the Preparation Phase:

1. Define goals, objectives, and project success criteria (PSC) of the project.
2. Define objectives and preliminary scope of the session.
3. Establish partitions and identify participants.
4. Determine environmental and logistical aspects.
5. Establish expectations for participants.
6. Communicate with participants.

During the initial steps of the Preparation Phase, the team prepared a feedback form composed of seven questions for the client:

1. What are the goals of this project?
   The goal statement usually describes the purpose of the project in one sentence.
2. What are the objectives of this project?
   The objective statement is derived from the goal and can be treated as a goal with a detailed statement. A project can typically have five to seven objectives.
3. What are the PSC for this project?
   PSC are used to describe the factors that can promote success. PSC can be both business and functional criteria.
4. What is the scope for this project?
   "In" and "out" are two kinds of scope for a project. "In" scope items are topics that are suitable to discuss in the Session Phase. "Out" of scope items are those topics unsuitable to discuss in the Session Phase.
5. What are the partitions of this project?
   According to the goal, objective, PSC, and scope, a partition breaks the project into small pieces that are correlated with one another and highly cohesive.
6. Who are the participants?
   According to the goal, objective, PSC, scope, and partition, the participants are those who are suitable to join the meeting.
7. What are the environmental aspects?
   To have a successful meeting, the team should prepare environmental and logistical arrangements that include room selection, technology arrangements, and refreshments.

The client took two business days to complete and return the form. While the client worked on the form, the team prepared a memorandum containing goals, objectives, PSC, preliminary scope, partition definitions, participants, and logistical arrangements. Participants were asked to read the memorandum before the Session Phase in order to understand the content, expectation, and goals of the methodology. The overall goal of the memorandum is to increase the quality of the Session Phase. (The memorandum is in Appendix G beginning on page 125.)

After reviewing the client's feedback forms, the team decided that there were no outstanding questions, issues, or confusion with the process and that another meeting with the client was not necessary before beginning the Session Phase.

3.2.1.2 Session Phase

The Session Phase is the heart of the ARM process, and it entails the following six steps:

1. executive sponsor commentary
2. scope closure
3. brainstorm, organize, and name (BON)
4. details

5. prioritization
6. participant feedback

Before the Session Phase meeting, the team made logistical arrangements to ensure that the meeting would go smoothly. The following is a brief summary of the arrangements:

- Supply note cards for participants to write more sentences.
- Supply tape to attach cards to the wall.
- Provide the following items to aid team members:
  - –detailed individual job assignments
  - –security requirement form (The words written in the cards could have been too small to read, so the team prepared a form that could project the content of the cards on the screen.)
  - –grouping form (In the Organize step of the process [Step 3], the team could show the categorized result on the screen immediately.)
  - –details form (In the Detail step [Step 4], the team could fill in the outputs from the participants on the form.)
- Supply package for each participant including
  - –memorandum
  - –prioritization form
  - –feedback form
  - –Session Phase slides
  - –Preparation Phase slides
  - –scratch paper
  - –note cards (four)

The instruction for the team is in Appendix H beginning on page 129.

Executive Sponsor Commentary

Due to time constraints, the team decided to omit this step. Also, the team understood that the participants already possessed the information, which would have been conveyed during the Preparation Phase. However, the team did provide a brief introduction to ARM and the procedures of the Session Phase meeting.

Scope Closure

The team also decided to omit this step because its primary purpose is to prepare the participants for the following steps. However, since the participants work closely together in the same department, they required little preparation time to familiarize themselves with security issues.

BON: Brainstorm, Organize, and Name

The BON step provided an efficient way to elicit the candidate requirements from participants. First, the team asked the participants the *focus question*, which was crafted to tie to the goals, objectives, and scope of the project. In this case, we chose the following focus question:

> *An important security requirement of the application is __.*

Based on their professional experience and security knowledge, the participants were asked to write down seven important security requirements on scratch paper within the time limit of seven minutes.

Afterwards, the team asked the participants to write down his or her top three or four security requirements on cards within three minutes. The team then collected the cards and put the candidate security requirements on the wall. The 24 candidate security requirements produced are listed below in Table 2.

Table 2: Initial Requirements Produced in ARM

1	The ability to securely transmit data to remote sources	13	Accountability (who did what, when, how...)
2	The preservation of data integrity	14	Integrity (assurance in data protection and validity)
3	The enforcement and usability of an access control system	15	Indelibility (deletions and retractions are noted/logged)
4	Security must be manageable and not hinder business (where possible).	16	Integrity
5	There must be a strong/reliable authentication process.	17	Access control
6	Information must be kept private from the outside world.	18	Confidentiality (encryption, etc.)
7	Consistent application program interfaces (APIs)	19	Partitioned data store (public read only and private read/write)
8	Data integrity	20	Selectively secure communication with outside entities.
9	Authentication and access control	21	Represent and support segmented disclosure.
10	Strong authentication	22	Role-based restricted views/edit/action access (e.g., summary report information)
11	Reduce/eliminate risks of inappropriate behavior.	23	Available 24/7 via remote authenticated access and secure
12	Granular access to data for users (operators) and customers	24	Key action audit (e.g., attribution of who/from where the publish button was pressed and what changes were made)

In the Organize step, all the participants reviewed the candidate security requirements generated during the brainstorming session to see whether any duplicate or inadequate security requirements

were included. Then the participants discussed thoroughly what they thought were important requirements. This step provided an opportunity for the participants to share their security concerns about the project. After a period of discussion and debate, they deleted seven candidate security requirements as redundant or inappropriate.

Specifically the participants removed requirements 1, 2, 5, 9, 14, 16, and 17. The remaining requirements are shown in Table 3.

Table 3: The Remaining Requirements after Initial Eliminations

3	The enforcement and usability of an access control system	15	Indelibility (deletions and retractions are noted/logged)
4	Security must be manageable and not hinder business (where possible).	18	Confidentiality (encryption, etc.)
6	Information must be kept private from the outside world.	19	Partitioned data store (public read only and private read/write)
7	Consistent APIs	20	Selectively secure communication with outside entities.
8	Data integrity	21	Represent and support segmented disclosure.
10	Strong authentication	22	Role-based restricted views/edit/action access (e.g., summary report information)
11	Reduce/eliminate risks of inappropriate behavior.	23	Available 24/7 via remote authenticated access and secure
12	Granular access to data for users (operators) and customers	24	Key action audit (e.g., attribution of who/from where the publish button was pressed and what changes were made)
13	Accountability (who did what, when, how...)		

In the Name step, the participants were instructed to group the selected security requirements and create names for each group. However, the participants instead engaged in a spirited discussion that combined grouping, naming, and categorizing. Thus, security requirements, groups, and names were generated together. In the end, the participants categorized security requirements into six groups, each containing one to four security requirements. Table 4 lists the groups and the requirements contained in each.

Table 4: Grouped Requirements

Group A: Confidentiality	• Information must be kept private from the outside world. • Selectively secure communication with outside entities.
Group B: Access Control	• Role-based restricted views/edit/action access (e.g., summary report information) • The enforcement and usability of an access control system • Granular access to data for users (operators) and customers • Represent and support segmented disclosure.
Group C: Data integrity	• Partitioned data store (public read only and private read/write) • Indelibility (deletions and retractions are noted/logged)
Group D: Manageability	• Accountability (who did what, when, how...) • Key action audit (e.g., attribution of who/from where the publish button was pressed and what changes were made) • Auditing capabilities
Group E: Usability	• Security must be manageable and not hinder business (where possible). • Available 24/7 via remote authenticated access and secure • Consistent APIs • Reduce/eliminate risks of inadvertent behavior.
Group F: Authentication	• Strong authentication

Requirements 8 and 18 were deleted in this step, and a third requirement was added to Group D.

Details: Benefits, Proof, Assumptions, Issues, and Action Items

In Step 4, the participants were asked to evaluate the requirements using the following 10 questions:

1. Is the candidate requirement a fragment or duplicate of anything that has already been discussed?
2. Is the candidate requirement fragment in scope (according to the contributor and the group)?
3. Would you like to change the title?
4. If you had this capability, how would it help the business?
5. What will you consider acceptable evidence that the envisioned capability has been successfully delivered to the business?
6. Are there any special constraints on the requirement?
7. Are there any assumptions made regarding the requirement?
8. What are the remaining issues and actions items for the requirement?
9. Are there any related notes or comments?

10. Is there anything that needs to be clarified by the supplier of the requirement?

Due to the time constraints, the team skipped questions 6, 8, 9, and 10. However, the participants found it difficult to ask the 10 questions of each requirement in turn. Instead, the participants reviewed all the security requirements together, not individually. In other words, they reviewed the assumption of the all security requirements, not the assumption of each security requirement.

The participants took an extended period of time to review the questions. Each question prompted a series of discussions and generated significant feedback from the participants. In the course of their discussion, the participants reviewed, reworded, and redefined those incomplete, incorrect, or ambiguous security requirements. The participants also encouraged one another to discover the true security requirements, not just recommendations. Their basic assumptions are provided below:

- All equipment exists and is physically secure.
- Selected external parties have read/write access.
- Base system security is installed, current, and active.
- Classified information exists on the system and must be protected.
- Different levels of access within the development team are supported.

Prioritization

In the BON step of ARM, the participants generated the candidate security requirements of their project, and then they further modified and redefined the projected security requirements in the Details step to ensure that the requirements were unambiguous, clear, and concise.

The Prioritization Phase of the ARM process began with the team providing instructions to guide participants to label each requirement *A, B*, or *C*, where *A* stood for most important, *B* stood for very important, and *C* stood for important. The rankings had to be assigned equally across the security requirements: One-third of the security requirements had to be assigned to *A*, one-third assigned to *B*, and one-third assigned to *C*.

Participants prioritized each security requirement based on their professional knowledge and the importance of the requirement to the project. The participants completed the phase in 10 minutes. Table 5 shows the raw result from the participants. In this table and in Table 6, an identifier is listed in the left-most column for each requirement listed in Table 4 above. Identifier A1, then, refers to the first requirement under Group A; A2, to the second one under Group A; and so on.

Table 5: Participant's Prioritization of Security Requirements

	Par. 1	Par. 2	Par. 3	Par. 4	Par. 5	Par. 6	Par. 7
A1	A	A	C	B	A	A	B
A2	A	A	A	C	A	B	C
B1	B	B	A	B	C	B	B
B2	B	A	B	C	A	A	B
B3	A	C	A	B	B	B	C
B4	C	C	B	A	C	C	A
C1	C	C	A	C	B	C	B
C2	C	B	C	C	C	C	A
D1	B	A	C	C	C	B	B
D2	B	B	B	B	B	B	C
D3	B	B	B	C	C	B	A
E1	C	C	C	C	A	B	C
E2	A	C	B	C	B	C	B
E3	C	C	C	C	C	C	C
E4	B	B	B	C	B	A	A
F1	A	A	A	A	A	A	A

After the session concluded, the team calculated the scores. First, the team substituted the rankings (*A*, *B*, and *C*) with numeric values 9, 3, and 1, respectively. Then, the team calculated the average score of each requirement. The results are shown in Table 6 and Figure 4.

Table 6: Ranked Score of the Requirements

	Par. 1	Par. 2	Par. 3	Par. 4	Par. 5	Par. 6	Par. 7	Average
A1	9	9	1	3	9	9	3	6.14
A2	9	9	9	1	9	3	1	5.86
B1	3	3	9	3	1	3	3	3.57
B2	3	9	3	1	9	9	3	5.29
B3	9	1	9	3	3	3	1	4.14
B4	1	1	3	9	1	1	9	3.57
C1	1	1	9	1	3	1	3	2.71
C2	1	3	1	1	1	1	9	2.43
D1	3	9	1	1	1	3	3	3.00
D2	3	3	3	3	3	3	1	2.71
D3	3	3	3	1	1	3	9	3.29
E1	1	1	1	1	9	3	1	2.43
E2	9	1	3	1	3	1	3	3.00
E3	1	1	1	1	1	1	1	1.00
E4	3	3	3	1	3	9	9	4.43
F1	9	9	9	9	9	9	9	9.00

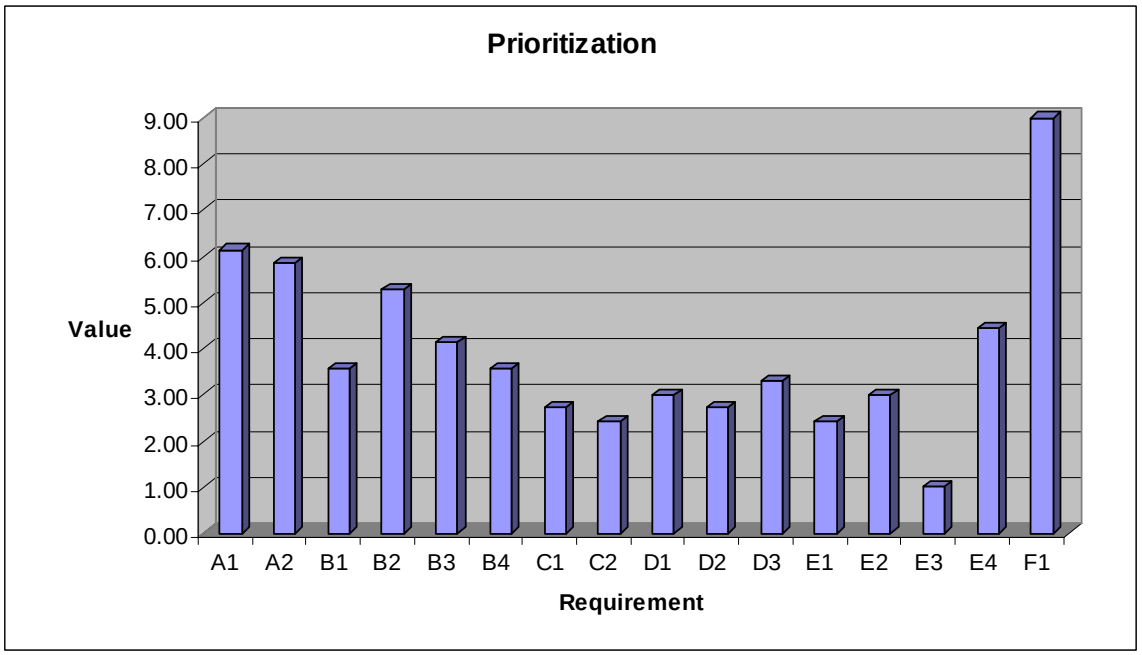


Figure 4: Ranked Score of the Requirements

Listed in order of priority and stated in a verifiable manner, the final requirements are:

1. The system shall utilize cryptographically strong authentication.
2. The information in the system must be kept private from unauthorized users.
3. The system shall implement selectively secure communication with outside entities.
4. The system shall utilize and enforce an access control system.
5. The system will attempt to reduce or eliminate risks of inadvertent behavior.
6. The system shall provide granular access to data for users (operators) and customers.
7. The system shall provide role-based restricted views/edit/action access (e.g., summary report information, public information for particular people).
   (tied with)
   The system shall represent and support segmented disclosure.
8. The system shall implement auditing capabilities.
9. The system shall provide accountability of users' actions.
   (tied with)
   The system will be available 24/7 via remote authenticated access.
10. The system shall maintain a partitioned data store (public read only and private read/write).
    (tied with)

The system shall implement a key action audit (e.g., attribution of who/from where the publish button was pressed, what changes were made).

11. The system shall implement indelibility.
    (tied with)
    The system's security features must be manageable and not hinder business (where possible).
12. The system shall expose consistent APIs to developers.

It is interesting to note that all participants agreed that Requirement F1, "strong authentication," was a high priority and that Requirement E3, "consistent APIs," was a low priority. Moreover, the participants thought the requirements in Category A, "confidentiality," and F, "authentication," were the most important ones since all the requirements in those categories were ranked highest.

Based on the result of the prioritization, the participants could then plan to implement their security requirements. Therefore, they could use their limited resources effectively and maximize their satisfaction of the security of the Web site within time and budget constraints.

Client Feedback

In the final portion of the Session Phase, the team requested that the participants fill out the feedback form that was used to collect useful information to improve the methodology. Through the feedback forms, the team hoped to elicit the pros and cons of the Session Phase, areas for improvement, the experience of the participants during the meeting, and most importantly, how well the phase was able to elicit adequate security requirements. On the feedback form, three questions were asked:

1. What did you like or not like about the Session Phase?
2. What did you think was the most important part of the Session Phase?
3. What would you change about the Session Phase?

The participants mentioned that the meeting time was not long enough to permit them to generate detailed, useful security requirements. Also, some participants thought that narrowing down to three security requirements might make those requirements too high level to be useful, and that more security requirements would be better for the project.

Many participants agreed that the Organize step was very important. In the Organize step, they generated and revised many ideas. Categorizing requirements allowed them to determine what was sensitive, critical, and less important. Additionally, organizing ideas seemed to promote a final demarcation of proposed requirements. The participants suggested that the session would have been more productive if more structure had been added to the categorization step and terms had been defined in advance.

The client's reaction to this phase was mixed. Some of the participants admitted that they didn't know exactly how to define a requirement and seemed to appreciate that the Session Phase was flexible enough to capture a diverse grouping of requirements. On one hand, most of the participants thought that the Session Phase promoted discussion and brought consensus regarding what

they were trying to build. On the other hand, some of the participants regarded the process as less structured than they had anticipated.

3.2.2 Recommendations

Overall, ARM was extremely effective and, true to its name, a rapid method of collecting requirements. We found that by simply choosing the correct focus question, the process was very easily adapted to elicit security requirements.

In retrospect, loose time management was the greatest flaw in this process. Due to the large number of questions that must be asked for each requirement, we recommend that future requirements-engineering teams enforce strict time management and proactively guide the discussions among the stakeholders.

Our results for the ARM methodology may be slightly biased given that the participants were all security experts already. As such, they were able to easily generate a comprehensive set of security requirements. In future sessions, it's unlikely that all of the participants will have such a background; thus the requirements-engineering team may need to review some security concepts with the participants before the session begins.

3.3 JAD

JAD brings users and technical professionals together to discover the functional (end user) requirements in software development. The centerpiece of JAD is a structured workshop known as the JAD Session, in which all stakeholders design a system or piece of software. JAD includes five distinct phases:

1. project definition
2. research
3. preparation
4. the JAD session
5. the final document

3.3.1 Project Definition Phase and Research Phase

The Project Definition Phase identifies the purpose, scope, objectives, assumptions, and open issues of the project by interviewing the managers from the users' departments.

The Research Phase focuses on collecting more detailed information about user requirements. In this phase, the work flow and preliminary specifications (data elements, screens, and reports) are obtained by interviewing users.

The team decided to combine the Project Definition and Research Phases because only one client could attend both meetings, and the other two clients were able to attend only one of the meet-

ings. Moreover, there was no distinction regarding their positions in the project. The users of the project were intended to be the general public. Thus, our client could play both the manager and user roles.

In the first two phases of JAD, instead of interviewing the clients, as intended by the process, the team prepared a list of questions for our clients. The team did this because contact with the client was by teleconference, and the client had existing documents answering all the questions the team asked.

The client was asked the following questions:

- What is the purpose of the project?
- What is the scope of the project?
- What are the management objectives of the project?
- What are the security objectives of the project?
- What are the functions of the project?
- What are the constraints of the project?
- What are the assumptions of the project?
- What are the open issues of the project?
- Who are the participants in the project?
- What is the work flow of the project?

The results are in Appendix I, which begins on 135.

3.3.2 Preparation Phase

The team decided not to use any visual aids in the Preparation Phase, because the main client was in teleconference with the team.

3.3.3 The JAD Session Phase

The JAD Session Phase, the heart of JAD, addresses work flow, data elements, screens, reports, and open issues. Because JAD is designed mainly for functional (end user) requirements, there are some steps not suitable for discussing security requirements. Thus, the team decided not to go over the work flow, data elements, screens, and reports steps in the phase. Therefore, the only remaining step was to discuss open issues.

The client initially provided 16 open issues, but only 8 of them were security related. Based on the document from the first 2 steps, the team generated 11 additional open issues:

- How can you provide high availability? In the Web tier? In the database tier?
- What is your approach to version control?
- What is your approach to configuration management?

- What is your approach to defect and issue management?
- What is the testing methodology to be used in the proposed project?
- Do you maintain a separate environment for testing, or is testing performed on development servers?
- What is the software and system architecture?
- What are the security model and security configuration/implementation details of the software architecture?
- Why are you considering secure sockets layer (SSL) to authenticate users and provide privacy?
- What is the difference between HTTP and Remote Method Invocation (RMI) in regards to privacy?
- How should the integrity of the site be protected?
- How can unauthorized changes to the project affect the mission?
- What are the best procedures to guarantee these actions are recorded?
- Does the stipulation that all or most of the code be open source present any potential security issues?
- What are the differences between clustering and active/active failover in regard to availability?
- How will you manage users and authorization?
- Is 100% uptime necessary for the project?
- Why don’t you need version control to be built into the project?
- Why do you prefer to use the Subversion version-control system?

The team conducted an interview and generated the security requirements based on the answers provided from our client. Table 7 lists the requirements that were generated.

Table 7: Security Requirements Produced from the JAD Session

SR1	The Web site shall provide reliable information to the users who have legitimate access to the Web site.
SR2	The Web site shall ensure that only authenticated users can access the protected content of the Web site.
SR3	The Web site shall protect the authenticated users' privacy by securing the communication channel.
SR4	The Web site shall ensure the integrity of content that is provided to the users by using authentication, authorization, and access control.
SR5	The Web site shall enable version control in both the content of the Web site and the development software.
SR6	The Web site shall enable auditing features that log all content modifications, work flow state transitions, access failures, and authentication attempts.
SR7	The Web site shall set up clustering to make the service sustainable when disaster occurs.

3.3.4 Client Feedback

The client participants thought the most important parts of the JAD Session Phase related to the architecture of the system. Unfortunately, neither the requirements document they provided nor the issues they discussed resulted in a clear presentation of the architecture.

The participants said that they need a real architecture description, most likely in graphical form, which clarifies what the major parts of the system are, what function they serve, and how they interact with one another.

3.3.5 Recommendations

By not defining the work flow, data elements, screens, and reports of the project, the JAD method turned out to be very similar to an unstructured interview process. In essence, the team just asked the client some questions about the project. The team thus did not use the full capability of the JAD method. The JAD Session Phase was designed for developing functional (end user) requirements; there was no specific way to discuss quality requirements such as security. Therefore, the team spent a lot of time researching other methods to assist in obtaining better security requirements during the JAD Session. The team suggests that JAD be used with an additional methodology to deal with security requirements or that a future iteration of SQUARE use a completely different methodology.

The quality of the security requirements generated relied on the quality of the questions asked during the interview. This relationship posed a great risk for the JAD method. In this case, the team produced a number of different questions simply because the clients at Delta were security professionals and had already considered the kinds of questions used in the Acme Group and Beta cases. If the client had been a "normal," less security-conscious set of stakeholders, the results would have been very different from this case. The team may not have been able to obtain as much information and may not have grasped the core security issues of the project.

The variety of the participants in the JAD Session Phase is very important as well. In this case, there were two clients participating in the phase, but only one of them answered the questions and there was no discussion between them. They didn't exchange their ideas at all, so it was likely that they didn't understand the each other's needs. Therefore, the results may be inaccurate and biased. The team recommends that the JAD Session should involve all the stakeholders and that the facilitator should encourage them to share their opinions.

4 Step 7: Categorize Requirements

Step 7 in the SQUARE process is to categorize security requirements elicited from the previous step. The purpose of this step is to help requirements analysts understand the difference between essential/necessary requirements, goals, and recommendations. The team used the "three R's" of survivability (resistance, recognition, and recovery) as security requirements categories. Using this process, the team produced defined, structured, and logically connected security requirements.

4.1 Acme Group

4.1.1 Methodology

The process of categorizing security requirements consisted of three stages.

1. The SQUARE team conducted a literature review to select a suitable methodology to organize security requirements into groups with similar features.
2. The team held a work session to determine which methodology to use and decided to apply the "three R's" of survivability: resistance, recognition, and recovery. Resistance is the ability of a system to repel attacks. Recognition is the ability of a system to recognize attacks. Recovery is the ability of a system to restore essential services during attacks and recover full services after attacks. We determined that the concept of survivability provided a comprehensive approach to analyzing and categorizing the needs of the client systems.
3. The team held an additional work session to categorize security requirements. Initially, the team produced 16 security requirements based on the IBIS outputs. However, after categorizing the requirements, we found that some did not qualify as requirements. For example, two of the security requirements were goals and five were recommendations. In the end, the team generated nine security requirements: four of them were categorized into resistance; two, into recognition; and three, into recovery. Table 8 shows the categorized security requirements.

Table 8: Categorized Security Requirements

Category	No.	Requirement
Resistance	SR-1	The system shall implement access control via a secure login screen.
	SR-2	The system shall identify and authenticate all the users who attempt to access it.
	SR-3	The server-side components and files contained therein shall have their access restricted to authorized personnel.
	SR-4	Fault tolerance shall be provided for the asset management system's essential services (IIS server, GIS server, and network lines).
Recognition	SR-5	The system shall maintain data integrity via logged modifications and user access control.
	SR-6	An access control system shall be configured for optimal information gathering for auditing purposes (access log and application log).
Recovery	SR-7	The system shall recover from attacks, failures, and accidents in less than one minute.
	SR-8	A backup shall consist of a complete reproduction of every file on the server.
	SR-9	The system shall be able to provide full functionality from backup.

4.1.2 Client Feedback

The client was not involved in this step of the project.

4.1.3 Recommendations

The group relied on its knowledge and experience to distinguish security requirements from goals and recommendations. The team lacked a formal, objective method to categorize the requirements and evaluate the results. In the future, it would be beneficial to clarify the definition of goals, requirements, and recommendations before categorizing the requirements.

Also, some requirements did not fit neatly into the survivability categories we selected and some requirements spanned more than one category. For example, SR-6, "An access control system shall be configured for optimal information gathering for auditing purposes (access log and application log)," can be put in both the "resistance" and "recognition" categories. In the future, the team suggests that requirements engineers conduct a more wide-ranging review of categorization methodologies to elicit more precise categories for requirements classification.

4.2 Beta

The categorization developed with this client is shown in Table 4.

4.3 Delta

4.3.1 Methodology

The team used the generally accepted perspective of information security—which is to break down the security requirements into three categories: confidentiality, integrity, and availability (CIA).

Confidentiality refers to limiting information access and disclosure to a set of authorized users and deters unauthorized users from accessing or disclosing information. Both authentication mechanisms that identify users and access-control mechanisms that limit user access support the goal of confidentiality.

Integrity refers to the trustworthiness of information resources. It consists of two concepts. One is data integrity; the other is source integrity. Data integrity means that data has not been changed inappropriately, whether by accident or intentional activity. Source, or origin, integrity means that data came from the people or entity you think provided it, rather than from an imposter.

Availability refers to the delivery of information to the right person when it is needed. Availability is the ability of a component or information technology (IT) service to perform its required function at a stated instant or over a stated period of time.

The team put four of the security requirements in the category of integrity, two of them in confidentiality, and one in availability. Table 9 shows the result.

Table 9: Delta Security Requirements

Category	No.	Security Requirement
Integrity	SR1	The Web site shall provide reliable information to the users who have legitimate access to the Web site.
	SR4	The Web site shall ensure the integrity of content that is provided to the users by using authentication, authorization and access control.
	SR5	The Web site shall enable version-control in both the content of the Web site and the development software.
	SR6	The Web site shall enable auditing features that log all content modifications, work flow state transitions, access failures, and authentication attempts.
Confidentiality	SR2	The Web site shall ensure that only authenticated users can access the protected content of the Web site.
	SR3	The Web site shall protect the authenticated users' privacy by securing the communication channel.
Availability	SR7	The Web site shall set up clustering to make the service sustainable when disaster occurs.

4.3.2 Client Feedback

The client was not involved in this step of the project.

4.3.3 Recommendations

When conducting the literature review, the team found a variety of ways to categorize the security requirements. The team had difficulty in selecting candidate categorization methods, since there was no way to compare them. In short, the team could not make a decision about which technique was superior to the others. Thus, the team chose the categorization technique arbitrarily.

In order to deliver more useful categorization results, the team suggests two avenues for improvement. One way to improve the process is to find or develop a standardized taxonomy to break down the security requirements. For example, provide users with a template that contains security requirements, their description, and a list of categories. With predefined categories, users won't have to generate or define requirement categories. The clear definition for each category can prevent the confusion and the assignment of security requirements to incorrect categories. Of course, predefined categories must to be broad enough to cover all security-related fields. This way, each candidate security requirement is required to group into at least one of the categories. However, the disadvantage of fixed categories is that users cannot expand their ideas, which will lead to limiting the effect of categorization. For instance, in the Beta project, the participants produced a new category: manageability. If the categorization method is not flexible enough, it may not stimulate discussion and feedback from the stakeholders. If a taxonomy cannot be developed, then finding or developing an organized way to categorize security requirements would be helpful. This categorization technique should provide a way to separate the security requirements from goals and implementations. As mentioned previously, it would have been beneficial for the team to have had a structured approach to delineate security goals and solutions from security requirements. If the security goals and solutions cannot be separated from security requirements, the results of categorization may be biased.

During the categorization process, the team learned that one security requirement can be placed in more than one category. For example, one security requirement can be in both the availability and integrity categories. Any categorization method must permit security requirements to be placed in one or more of the categories.

A second avenue for improvement is to assign the categorization task to stakeholders. During the categorization process, stakeholders can clarify, verify, and reconsider the security requirements. This technique should provide a step-by-step open discussion session to have all stakeholders involved.

5 Step 8: Prioritize Requirements

In this step of the SQUARE process, we prioritized the security requirements that were categorized in Step 7. The purpose of prioritizing the requirements was to ensure that the identified security requirements were the most essential ones. Moreover, due to time and budget constraints, it is extremely difficult to implement all the requirements elicited for a system. To prioritize the security requirements, the team adopted the Analytic Hierarchy Process (AHP) methodology to determine which requirements were the most important security concerns for the system.

By applying the AHP prioritizing methodology, the team utilized a quantitative scale to rank the security requirements, instead of evaluating them through the subjective viewpoint of the requirements engineers.

5.1 Literature Review

Before settling on AHP, the team conducted a literature review of the various prioritization methodologies applicable to the project. The methodologies reviewed were Numeral Assignment Technique, Theory-W, and AHP. We discuss each technique briefly.

5.1.1 Numeral Assignment Technique

The Numeral Assignment Technique provides a scale for each requirement. Brackett proposed dividing the requirements into three groups: mandatory, desirable, and inessential [Brackett 90]. According to Joachim Karlsson's method, participants assign each requirement a number on a scale of 1 to 5 to indicate the importance of those requirements [Karlsson 95]. The numbers carry the following meaning:

1. does not matter (the customer does not need it)
2. not important (the customer would accept its absence)
3. rather important (the customer would appreciate it)
4. very important (the customer does not want to be without it)
5. mandatory (the customer cannot do without it)

The final ranking is the average of all stakeholders' rankings for each requirement.

5.1.2 Theory-W

Theory-W was developed at the University of Southern California by Barry W. Boehm and Rony Ross in 1989 [Boehm 89, Park 99]. It has two principles:

1. Plan the flight and fly the plan.
2. Identify and manage your risks.

The first principle seeks to build well-structured plans that meet predefined standards for easy development, classification, and query. “Fly the plan” ensures that the progress follows the original plan. The second principle, “Identify and manage your risks,” involves risk assessment and risk handling. It is used to guard the stakeholders’ “win-win” conditions from infringement. Theory-W has four steps:

1. Separate the people from the problem.
2. Focus on interests, not positions.
3. Invest options for mutual gain.
4. Insist on using objective criteria.

5.1.3 AHP

AHP was developed by Thomas Saaty and applied to software engineering by Joachim Karlsson and Kevin Ryan in 1997 [Saaty 80, Karlsson 96, and Karlsson 97]. AHP is a method for decision making in situations where multiple objectives are present. This method uses a “pair-wise” comparison matrix to calculate the relative value and costs of security requirements. By using AHP, the requirements engineer can also confirm the consistency of the result. AHP can prevent subjective judgment errors and increase the likelihood that the results are reliable. There are five steps in the AHP method:

1. Review candidate requirements for completeness.
2. Apply the pair-wise comparison method to assess the relative value of the candidate requirements.
3. Estimate the relative cost of implementing each candidate requirement.
4. Calculate each candidate requirement’s relative value and implementation cost and plot each on a cost-value diagram.
5. Use the cost-value diagram as a map for analyzing the candidate requirement.

5.1.4 Prioritization Technique Comparison

The team decided to use AHP as a prioritizing technique in SQUARE. Factoring into the rationale behind choosing AHP were the team members’ familiarity with the technique, its quantitative outputs, and its structure in providing definite steps for implementation. The comparison matrix is shown in Table 10. The description of evaluation criteria are

- **clear-cut steps—**the degree of definition between stages within the prioritization methodology for progress and implementation
- **quantitative measurement—**the prioritization methodology's numerical output that clearly displays the clients' priorities for all requirements
- **maturity—**the time, exposure, and analysis the prioritization technique has experienced in its vetting by the appropriate community
- **labor-intensity—**the number of hours needed to properly execute the prioritization technique.
- **learning curve—**the pace at which the requirements engineers and clients can fully comprehend the elicitation technique

Table 10: Comparison of Prioritization Techniques

3= Very Good, 2= Fair, 1= Poor

	Numeral Assignment Technique	Theory-W	AHP
Clear-cut steps	3	2	3
Quantitative measurement	3	1	3
Maturity	1	3	3
Labor-intensity	2	1	2
Learning curve	3	1	2

5.2 Methodology for Acme Group Case

The team followed the five steps of the AHP methodology to prioritize the security requirements. Three stakeholders were involved in the AHP prioritization process. In this step, the team held one meeting to give instructions and a follow-up meeting to clarify some ambiguous parts of the AHP methodology.

5.2.1 Review Candidate Requirements for Completeness

The team reviewed and reanalyzed the requirements to ensure they were correct, complete, and clear. After meeting with the client, the team revised the security requirement based on the feedback received.

5.2.2 Apply Pair-Wise Comparison Method

In this step, the stakeholders implemented the pair-wise comparison method of AHP. The team provided brief instructions for using the AHP methodology to the participants. (The instructions are provided in Appendix J that begins on page 149.) Because the team generated 9 security requirements, the method should produce a matrix with 81 (9 x 9) cells. However, the participants needed to fill the upper half of the matrix only, and each requirement had a value of "1" when

compared to itself. Consequently, each participant had to respond to 36 cells.[3] The team highlighted the cells that required feedback from the participants. A sample of the feedback is shown in Table 11.

Table 11: Prioritization Feedback of Acme Group

	A	B	C	D	E	F	G	H	I	J
1		**SR-1**	**SR-2**	**SR-3**	**SR-4**	**SR-5**	**SR-6**	**SR-7**	**SR-8**	**SR-9**
2	**SR-1**	1	8	1/5	3	1	2	2	3	1
3	**SR-2**	1/8	1	1/5	1/7	1/7	1/7	1/7	1/9	1/9
4	**SR-3**	5	5	1	1	2	1	3	1	1
5	**SR-4**	1/3	7	1	1	1/2	1/2	3	1/2	1
6	**SR-5**	1	7	1/2	2	1	3	3	1	1/3
7	**SR-6**	1/2	7	1	2	1/3	1	1/3	1	1
8	**SR-7**	1/2	7	1/3	1/3	1/3	3	1	3	2
9	**SR-8**	1/3	9	1	2	1	1	1/3	1	1/6
10	**SR-9**	1	9	1	1	3	1	1/2	6	1

The prioritization matrix is also shown in Appendix J.

AHP uses a pair-wise comparison matrix to determine the relative value and cost between security requirements. An arbitrary entry in row i and column j of the matrix, labeled a_{ij}, indicates how much higher (or lower) the value/cost for requirement i is than that for requirement j. The value/cost is measured on an integer scale from 1 to 9, with each number having the interpretation shown in Table 12 and Table 13.

Table 12: Interpretation of Values in a Matrix

Intensity of Value	Interpretation
1	Requirements i and j are of equal value.
3	Requirement i has a slightly higher value than j.
5	Requirement i has a strongly higher value than j.
7	Requirement i has a very strongly higher value than j.
9	Requirement i has an absolutely higher value than j.
2, 4, 6, 8	These are intermediate scales between two adjacent judgments.
Reciprocals	If Requirement i has a lower value than j

[3] In general, a matrix with *n* requirements must have responses in *n* * (*n*-1) cells. This rule reduced the participants' workload in this step.

Table 13: Interpretation of Costs in Matrix

Intensity of Value	Interpretation
1	Requirements i and j are of equal cost.
3	Requirement i has a slightly higher cost than j.
5	Requirement i has a strongly higher cost than j.
7	Requirement i has a very strongly higher cost than j.
9	Requirement i has an absolutely higher cost than j.
2, 4, 6, 8	These are intermediate scales between two adjacent judgments.
Reciprocals	If Requirement i has a lower cost than j

The participants filled in the cells of the prioritization matrix to demonstrate the level of concern expressed for the candidate security requirements. The results for each participant are in Appendix K.

5.2.3 Determine the Priority of Requirements

In this section, we provide instructions for creating cost-value diagrams based on the Excel spreadsheet shown in Figure 5. The formulas that we used in Excel to calculate our results are mentioned throughout this section.

First, the team filled in the lower half of prioritization matrix based on the participants' feedback from the upper half of the matrix. The team then averaged the data over normalized columns to estimate the *eigenvector* of the matrix, which represents the criterion distribution.[4] To do this, we first computed the sum of the columns in the matrix. Then, we divided each value in the matrix by the column sum. The output is the normalized matrix show in Figure 5. (In figures 5 and 6, columns B through K are presumed to contain the raw user feedback and are omitted for clarity of presentation.)

[4] Mathematically, an eigenvector of a transformation is a nonzero vector whose direction remains unchanged; only its magnitude is scaled by some factor.

	A	L	M	N	O	P	Q	R	S	T
1		SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9
2	SR-1	0.1021	0.1333	0.0321	0.2405	0.1074	0.1582	0.1503	0.1806	0.1314
3	SR-2	0.0128	0.0167	0.0321	0.0115	0.0153	0.0113	0.0107	0.0067	0.0146
4	SR-3	0.5106	0.0833	0.1604	0.0802	0.2148	0.0791	0.2254	0.0602	0.1314
5	SR-4	0.0340	0.1167	0.1604	0.0802	0.0537	0.0395	0.2254	0.0301	0.1314
6	SR-5	0.1021	0.1167	0.0802	0.1603	0.1074	0.2373	0.2254	0.0602	0.0438
7	SR-6	0.0511	0.1167	0.1604	0.1603	0.0358	0.0791	0.0250	0.0602	0.1314
8	SR-7	0.0511	0.1167	0.0535	0.0267	0.0358	0.2373	0.0751	0.1806	0.2628
9	SR-8	0.0340	0.1500	0.1604	0.1603	0.1074	0.0791	0.0250	0.0602	0.0219
10	SR-9	0.1021	0.1500	0.1604	0.0802	0.3223	0.0791	0.0376	0.3612	0.1314

Figure 5: Normalized Comparison Matrix

The formula of cell L2 is "= B2/ Sum (B$2: B$10)." To generate the remaining values, drag the cursor from L2 to T10.

To determine the score of each requirement, average the row in the normalized matrix by dividing each row sum by the number of requirements (Figure 6). The formula of V2 is "= Average (L2:T2)." To generate all the values in the row, drag the cursor from V2 to V10.

The score of each requirement is the percentage that the requirement adds to the requirements' total value. In this case, SR-1 comprises 2.82% of the requirements' total value. The computed results for each participant are in Appendix K, pages 157–159.

	A	L	M	N	O	P	Q	R	S	T	V
1		SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9	Scores
2	SR-1	0.1021	0.1333	0.0321	0.2405	0.1074	0.1582	0.1503	0.1806	0.1314	0.1373
3	SR-2	0.0128	0.0167	0.0321	0.0115	0.0153	0.0113	0.0107	0.0067	0.0146	0.0146
4	SR-3	0.5106	0.0833	0.1604	0.0802	0.2148	0.0791	0.2254	0.0602	0.1314	0.1717
5	SR-4	0.0340	0.1167	0.1604	0.0802	0.0537	0.0395	0.2254	0.0301	0.1314	0.0968
6	SR-5	0.1021	0.1167	0.0802	0.1603	0.1074	0.2373	0.2254	0.0602	0.0438	0.1259
7	SR-6	0.0511	0.1167	0.1604	0.1603	0.0358	0.0791	0.0250	0.0602	0.1314	0.0911
8	SR-7	0.0511	0.1167	0.0535	0.0267	0.0358	0.2373	0.0751	0.1806	0.2628	0.1155
9	SR-8	0.0340	0.1500	0.1604	0.1603	0.1074	0.0791	0.0250	0.0602	0.0219	0.0887
10	SR-9	0.1021	0.1500	0.1604	0.0802	0.3223	0.0791	0.0376	0.3612	0.1314	0.1582

Figure 6: Scores for Requirements

5.2.4 Check for Consistency

The ability of AHP to test for consistency is one of the method's greatest strengths. Consistency is based on the idea of *cardinal transitivity*. For example, if Requirement A is considered to be two times more important than Requirement B, and Requirement B is considered to be three times more important than Requirement C, then perfect cardinal consistency would imply that Requirement A be considered six times more important than Requirement C. In this way, if the participants judge Requirement A to be less important than Requirement C, it implies that a judgmental error exists and the prioritization matrix is inconsistent. [5]

In this section, the team used consistency index/random index (CI/RI) ratio to check the consistency of the results (Figure 7).[6] To compute the CI/RI ratio, the team took the following steps:

1. Calculate the product of the pair-wise comparison matrix and the vector of scores. Make sure that the user data is in decimal form (i.e., "1/5" is now represented as "0.2"). Highlight cells B2 to J10, and type the formula "mmult (B2:J10, V2:V10)." Press Control-Shift-Enter, which applies the formula to the entire highlighted matrix.
2. Calculate the ratios. In cell Y2, calculate the ratio of the score and product values with the formula "=X2/V2" and copy this to the range "Y3:Y10."
3. Calculate the CI value. In cell Y11, calculate the consistency index with the formula "=(average (Y2:Y10) – 9) /8." The value *9* is the number of requirements and *8* is the number of requirements minus one.
4. Calculate the CI/RI score. The RI is the average value of the CI, if the entries in the pair-wise comparison matrix were chosen at random. If the CI/RI score is sufficiently small, then the participants' comparisons are probably consistent enough to be useful. Thomas Saaty suggests that if the CI/RI is smaller than 0.10, then the degree of consistency is satisfactory; however, if the CI/RI is larger than 0.10, inconsistencies exist and the AHP method may not yield meaningful results [Saaty 80]. To calculate the CI/RI score, the team first gets the standard RI value from Saaty's information; a few of those RI values are listed in Table 14. Because the number of security requirements is 9, the RI is 1.45. Second, in cell Y12, calculate the CI/RI score with the formula "= Y11/1.45."

Table 14: Random Index Values

Number of Requirements	2	3	4	5	6	7	8	9	10
RI	0	0.58	0.90	1.12	1.24	1.32	1.41	1.45	1.51

[5] Our discussion of cardinal transitivity is based on material from http://rutgersscholar.rutgers.edu/volume04/maurluxh/maurluxh.htm.

[6] In Figure 7, columns not significant to the display of CI/RI results have been omitted from the chart.

	A	B	C	D	E	F	G	H	I	J	V	X	Y
1		SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9	Scores	Product	Ratio
2	SR-1	1	8	1/5	3	1	2	2	3	1	0.1373	1.5427	11.2344
3	SR-2	1/8	1	1/5	1/7	1/7	1/7	1/7	1/9	1/9	0.0146	0.1549	10.5917
4	SR-3	5	5	1	1	2	1	3	1	1	0.1717	1.9647	11.4415
5	SR-4	1/3	7	1	1	1/2	1/2	3	1/2	1	0.0968	1.0743	11.0955
6	SR-5	1	7	1/2	2	1	3	3	1	1/3	0.1259	1.4065	11.1681
7	SR-6	1/2	7	1	2	1/3	1	1/3	1	1	0.0911	0.9550	10.4813
8	SR-7	1/2	7	1/3	1/3	1/3	3	1	3	2	0.1155	1.2740	11.0301
9	SR-8	1/3	9	1	2	1	1	1/3	1	1/6	0.0887	0.9134	10.2961
10	SR-9	1	9	1	1	3	1	1/2	6	1	0.1582	1.7547	11.0884
11												CI	0.2420
12												CI/RI	0.1669

Figure 7: Data and Results for CI/RI Score

The results of the consistency checks for individual participants are shown in Appendix K beginning on page 153.

5.2.5 Analyze Requirements Using Cost-Diagram Plot

Table 15 displays the value and cost CI/RI scores for each participant in the AHP process. As shown in that table, only one CI/RI score is less than 0.10 (the value CI/RI for participant 3). The average CI/RI is 0.16. According to Saaty's determination, then, inconsistencies exist in the results.

To reduce the impact of these inconsistencies, the team decided to delete the largest value in both the value and cost rows and then calculated the average of the remaining two CI/RIs. This refinement resulted in our basing the requirements' value CI/RI on the average scores of participants 2 and 3 and the requirements' cost CI/RI on the average scores of participants 1 and 2.

Table 15: Average Costs and Value CI/RI Scores for Participants 1-3

CI/RI Type	Participant 1	Participant 2	Participant 3
Value	0.25	0.16	0.07
Cost	0.17	0.15	0.18

The requirements' final value is shown in Figure 8, and the requirements' final cost is shown in Figure 9. The value of each requirement is relative. That is, if the value of a requirement is 20%, this requirement is twice as important as the 10% value of another requirement. The sum of the scores of the requirements should always be 100%. When the value of the requirement is 10%, this requirement consists of 10% of the value of all requirements.

According to Figure 8, the three most valuable requirements are SR-3, SR-5, and SR-6. Together, they constitute 47% of the requirements' total value. The three least valuable requirements are SR-1, SR-4, and SR-8, which constitute 23% of the requirements' total value. Figure 9 shows that requirements SR-4, SR-7, and SR-9 are the three most expensive. Together, they constitute 72% of the requirements' total cost. The three least expensive requirements are SR-1, SR-2, and SR-3 that constitute 7% of the requirements' total cost.

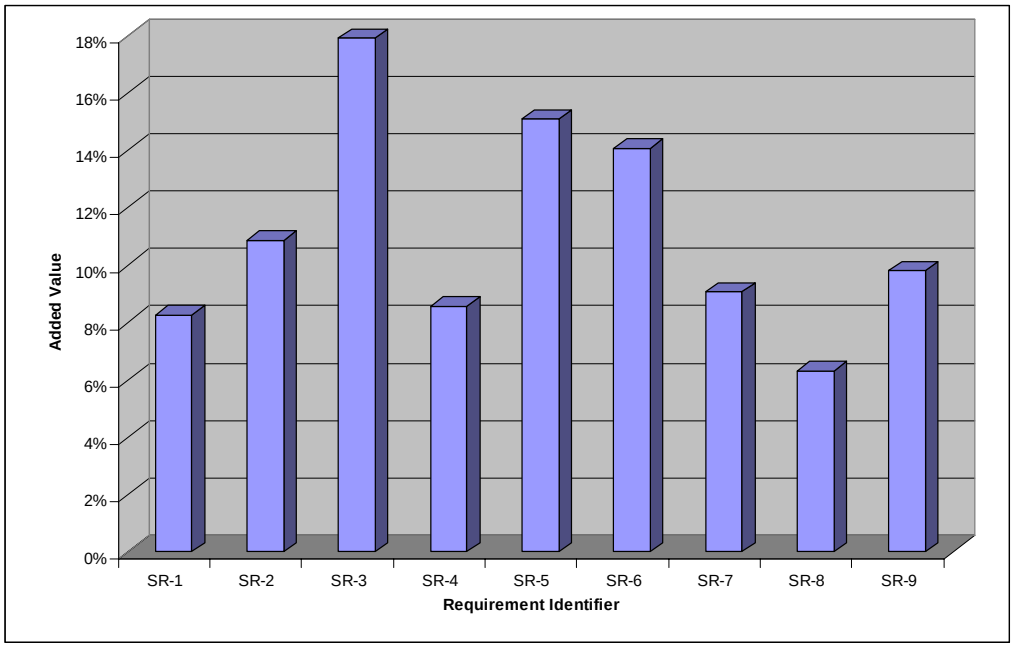


Figure 8: Value Distribution of Requirements

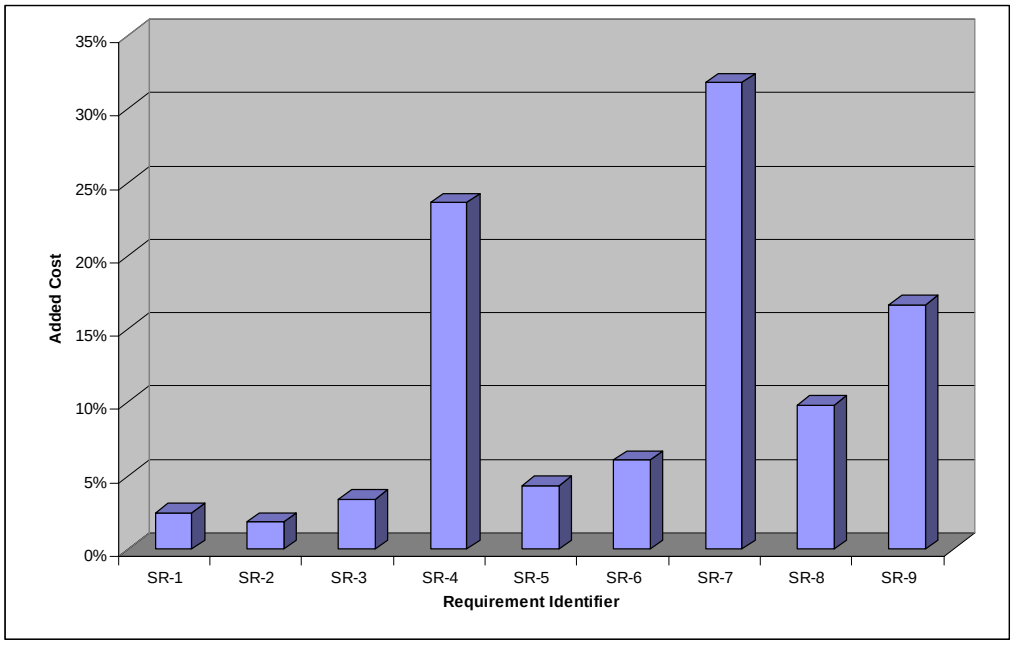


Figure 9: Cost Distribution of Requirements

At this point, the stakeholders had assigned a cost and value to each requirement. The next logical step is to calculate the cost-value ratios for each requirement. This way, the stakeholders can pin-point the requirements that are most valuable and least expensive to implement. The cost-value diagram in Figure 10 is divided into three groups:

1. **high** value-to-cost ratio of requirement (larger than 2.0)
2. **medium** value-to-cost ratio of requirement (between 2.0 and 0.5)
3. **low** value-to-cost ratio of requirement (less than 0.5)

Table 8 contains the requirements and their associated identifiers. Requirements SR-1, SR-2, SR-3, SR-5, and SR-6 are high priority. Requirements SR-4 and SR-7 are low priority. When security requirements are prioritized, the client can implement the security requirement based on their relative priority.

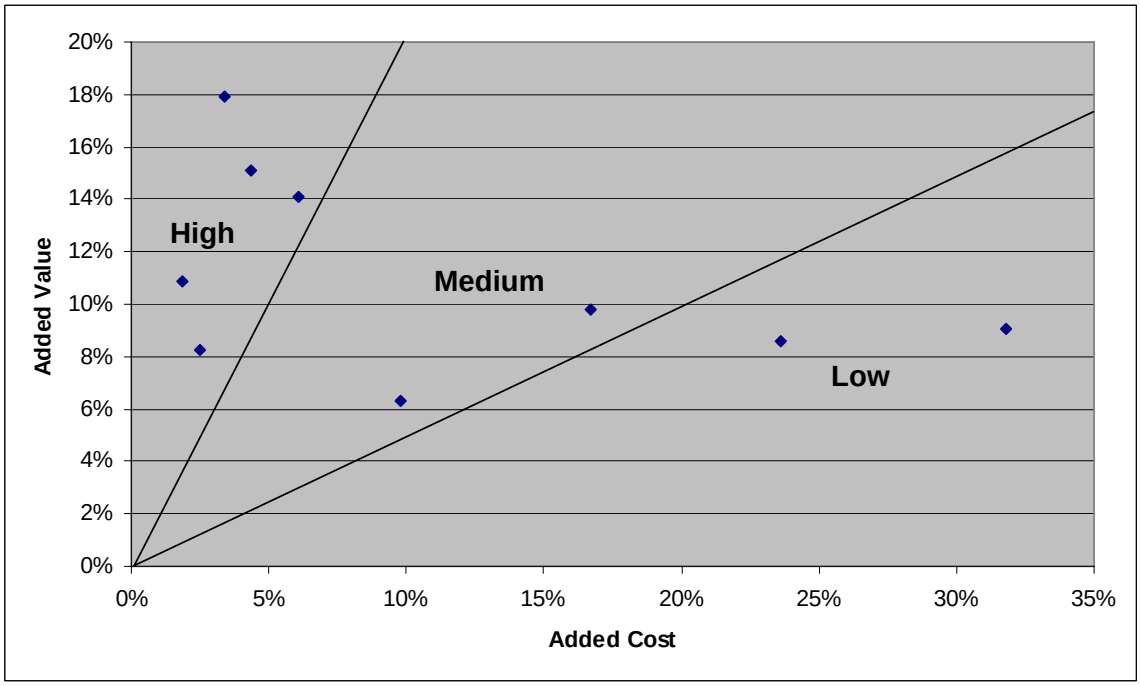


Figure 10: Cost-Value Diagram of Requirements

5.2.6 Reprioritize Security Requirements

During the AHP process, clients were confused with some of the security requirements, and they were not sure about the definitions of *value* and *cost*. For example, some of the clients viewed the costs as the price of implementing these security mechanisms, but other clients thought the costs were the impact of not implementing these security mechanisms.

The team verified and clarified this confusion in a subsequent meeting. The clients then decided to redo the prioritization matrix to get a better result. As a result, the team conducted the AHP

process again to generate new prioritization results. These results are shown in Figure 11, Figure 12 and Appendix L (pages 167–171).

In Figure 11, requirements SR-2 and SR-3 constitute almost half of the value of the security requirements. This means that requirements SR-2 and SR-3 are the most valuable security requirements by far. Compared to the previous prioritization result, the value scores for each security requirement vary.

In Figure 12, the results of the cost assessment are very similar to the previous iteration. Apparently requirements SR-4 and SR-7 are the most expensive security requirements to implement.

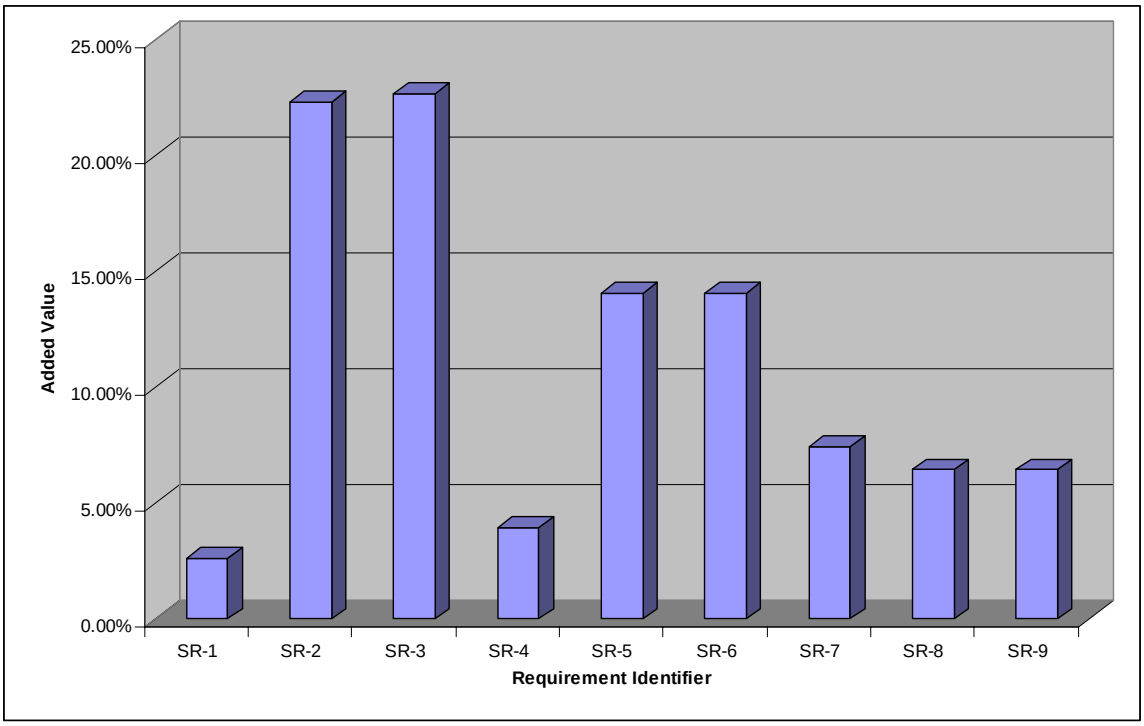


Figure 11: Refined Value Distribution of Requirements

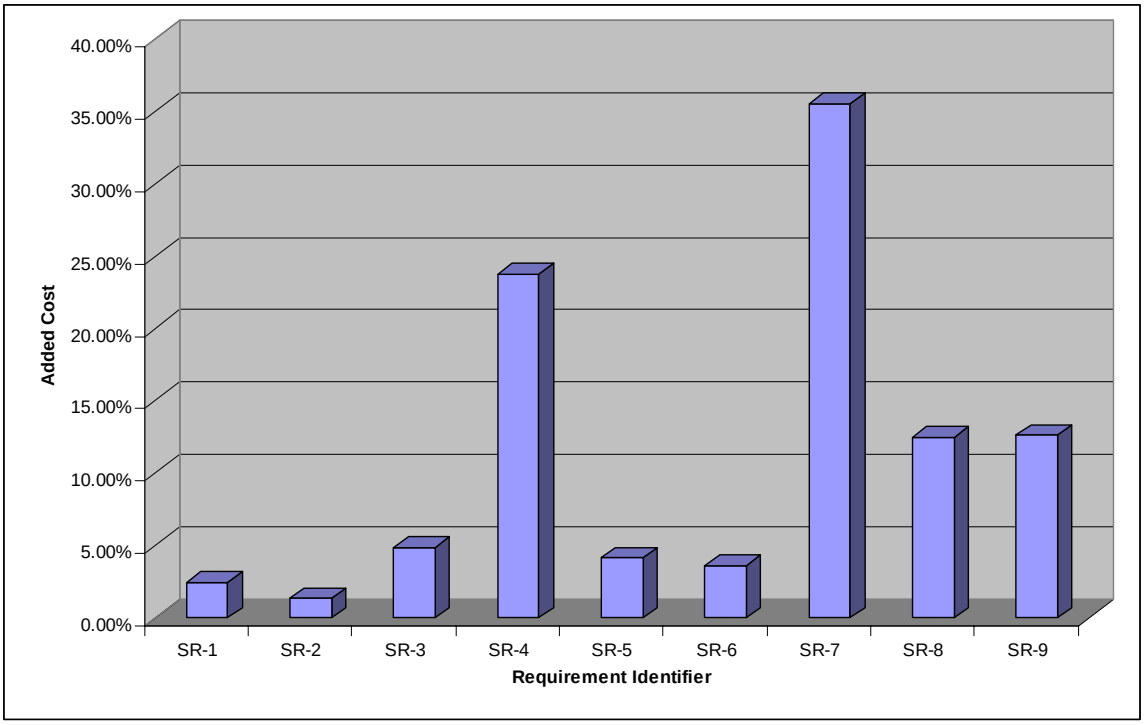


Figure 12: Refined Cost Distribution of Requirements

The CI/RI ratios of the value and cost reports are 0.15 and 0.17, which are quite close to the previous results. Although clients tried to make the new prioritization result consistent, the CI/RI ratios were still larger than 0.10 and judgment errors still exist in the new result.

In Figure 13, we see that the client has four security requirements that fall into the high-priority category, three security requirements in the medium-priority category, and two security requirements in the low-priority category. Those requirements with a high value-cost ratio (such as SR-2 and SR-3) fall into the high-priority area. Likewise, those with a low value-cost ratio (such as SR-4 and SR-7) fall into the low-priority area.

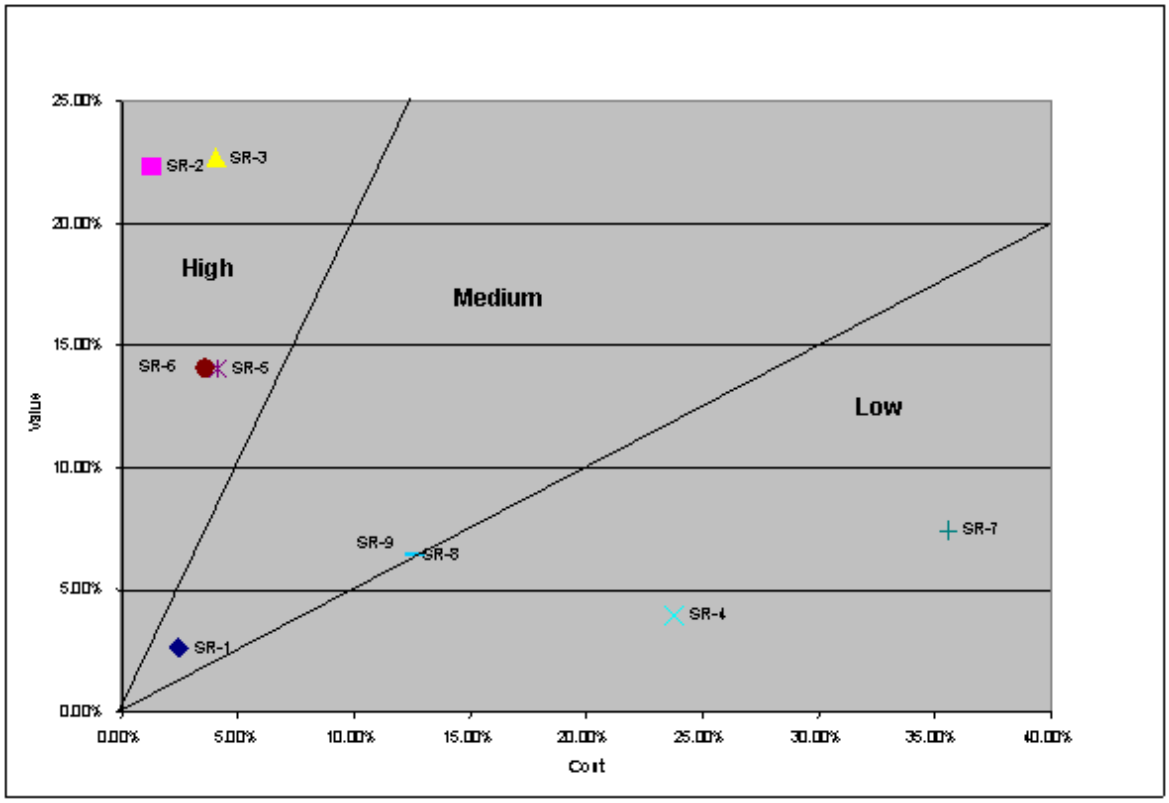


Figure 13: Refined Cost-Value Diagram of Requirements

5.3 Methodology for Beta Case

The process used and the results achieved with the Beta customer are described in Section 3.2.1.2.

5.4 Methodology for Delta Case

The team also used the AHP methodology in this case to prioritize the security requirements. The analyzed results are shown in Appendix M on pages 173–176 .

The team collected the results from Steps 6 and 7 (i.e., security requirements and their corresponding categories) and then sent the outputs and prioritization instructions to the client. The client spent five days completing the analysis.

The CI/RI scores of the value and cost reports are 0.15 and 0.17. Since the CI/RI scores are larger than 0.10, judgment errors probably exist in the prioritization result. Figure 14 and Figure 15 illustrate the final value and cost distributions of the requirements.

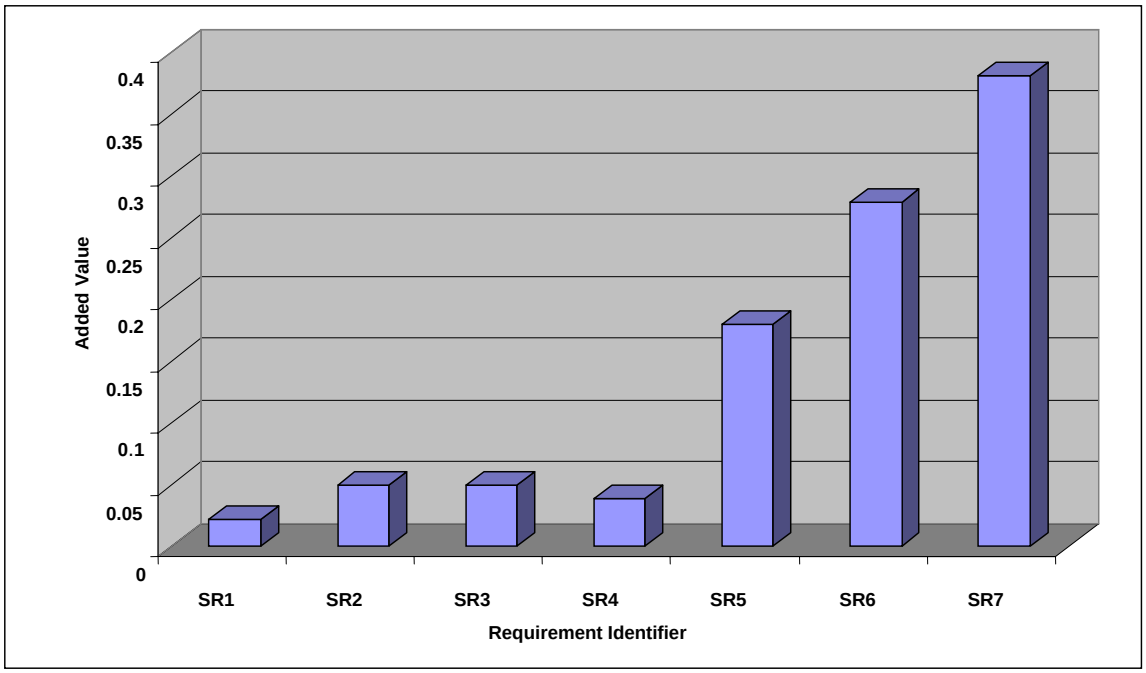


Figure 14: Value Distribution of Requirements

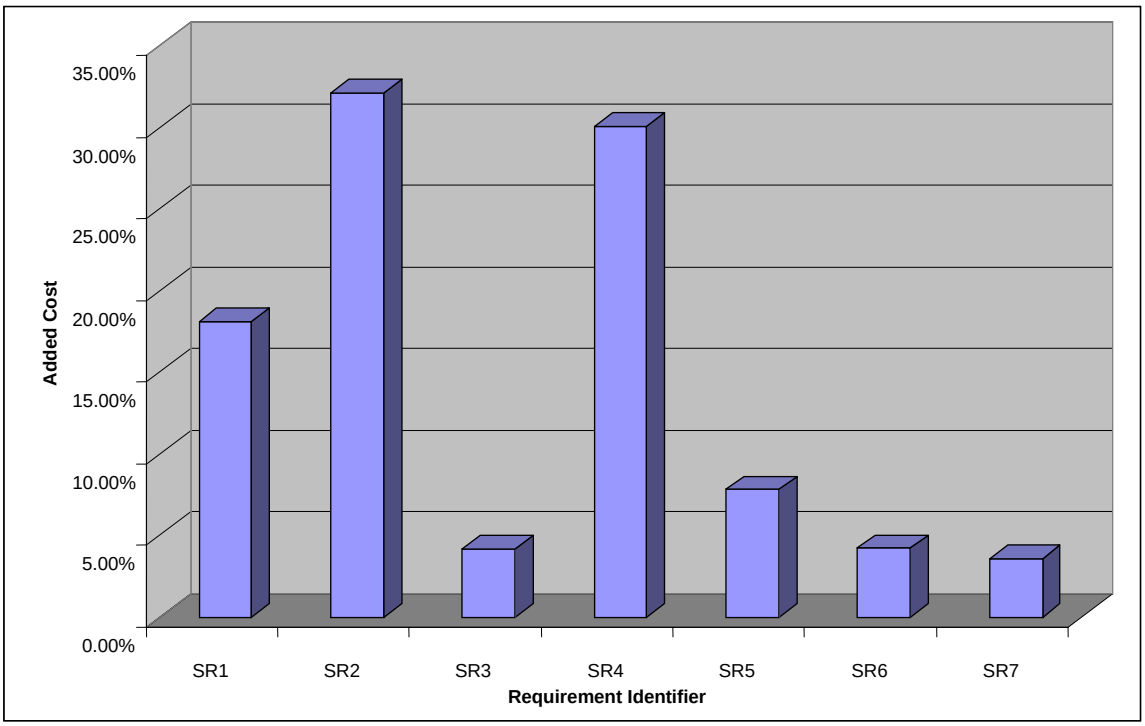


Figure 15: Cost Distribution of Requirements

In Figure 16 , the team divided the security requirements into three levels of priority. As shown in Figure 16, the client has two security requirements that fall into the high-priority category, two security requirements in the medium-priority category, and three security requirements in the

low-priority category. The suggested order for implementing the security requirements is as follows:

1. SR7: The Web site shall set up clustering to make service sustainable when disaster occurs.
2. SR6: The Web site shall enable auditing features that log all content modifications, work flow state transitions, access failures, and authentication attempts.
3. SR5: The Web site shall enable version-control in both the content of the Web site and the development software.
4. SR3: The Web site shall protect the authenticated users' privacy by securing the communication channel.
5. SR2: The Web site shall ensure that only authenticated users can access the protected content of the Web site.
6. SR4: The Web site shall ensure the integrity of content that is provided to the users by using authentication, authorization and access control.
7. SR1: The Web site shall provide reliable information to the users who have legitimate access to the Web site.

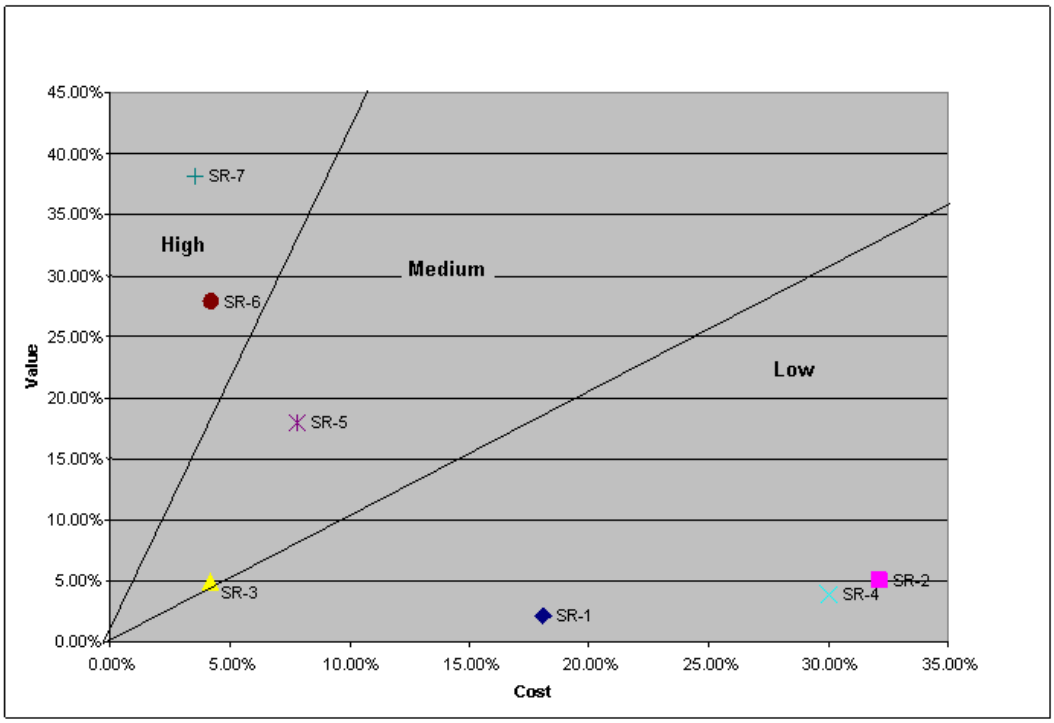


Figure 16: Cost-Value Diagram

5.5 Client Feedback

The client would have preferred to see the consistency checker in action because some of the values were derived through negotiation between the developer, administrator, and marketing team. The difference in value perception between the three stakeholders was also very interesting.

The client generally found AHP to be clear, easily understood, and able to provide a good indication of the cost/value ratio for prioritizing requirements. However, they would have liked to have known whether cost, value, or both were the main drivers in establishing the priority. Moreover, they thought that evaluating some items was like "comparing apples to oranges": two requirements were quite valuable but for very different reasons. Also, the value of the requirement makes the person filling out the survey take many variables into account such as the following:

- frequency of occurrence
- danger of occurrence
- marketability
- system robustness

Different stakeholders may place very different weights on these variables, yet their relative weighting is not taken into account anywhere. It is simply summarized as "value."

The clients felt that the range of values available in assigning to a comparison could be trimmed down to three or four values and their reciprocals.

It is difficult to assess the value of the AHP methodology with so few requirements to prioritize. It is relatively easy to evaluate a few requirements at the same time. In a small set, too, it is difficult to objectively consider requirements in pairs without considering others as well, turning the matrix completion process into something closer to a ranking.

The client thought that the AHP methodology needs to be supported by a tool that presents only two requirements at a time to solicit a comparison. That process should assign only positive integers to the winner. In addition, there should be some cognitive subversion to ensure true responses. One possibility is to trick the user into ranking the pairs multiple times by presenting them in random orders.

Lastly, the clients felt that the consistency-check result is a direct reflection of how the requirements are interpreted. The better defined the requirements are, the more consistent the outcome should be.

5.6 Recommendations

Generally speaking, the AHP methodology was a straightforward method for prioritizing requirements. However, it was difficult to define the value and cost of each security requirement because the value and cost could be very complicated and could vary dramatically due to of the stakeholders' different viewpoints. Each participant had an opinion about the value and cost of the security requirements. For example, a developer may view the value of privacy protection as very low and the cost of privacy protection as very high—simply because he or she doesn't feel strongly about privacy issues. On the other hand, a user may think the value of privacy is very high and have no idea about the cost of the technology to ensure it.

In its first attempt to prioritize the requirements, the team asked each participant to prioritize separately so that everyone had a prioritization matrix. Then the team came up with all the participants' scores and averaged those scores. However, prioritization wasn't that easy. Prioritization is an iterative process, and clients repeated the negotiation and concession again and again. So, it may not have been a good idea to simply average all the participants' scores. The team recommends that all the participants come together to discuss the priority of the security requirements in a session instead of doing the prioritization individually. During the prioritization process, the stakeholders can ascertain that everyone has the same understanding about the security requirements and further examine any ambiguous requirements. After everyone reaches a consensus, the result of prioritization will be more reliable.

6 Step 9: Requirements Inspection

Step 9 in the SQUARE process focuses on inspecting security requirements and correcting the defects found. There are two reasons for doing this step: (1) requirements inspection in the early stages of the software development cycle can reduce the cost of fixing defects, and (2) inspecting requirements and making corrections in the early stage can improve the quality of software substantially. The team used a peer review method to examine the security requirements.

6.1 Literature Review

Requirements inspection can be done with a wide degree of formality. The team researched requirements inspection methodologies and found that peer review has long been viewed as a powerful method to enhance the quality of requirements. Basically, peer review involves consulting peers in reviewing and examining work product and discovering improvement. Peer review has many forms, including inspection, team reviews, walkthroughs, pair programming, peer desk checks, and pass-arounds. Common methods used to inspect requirements are

- Fagan inspection
- walkthrough
- checklist

We will discuss each of these methods briefly.

6.1.1 Fagan Inspection

Fagan inspection is a structured review developed by Michael Fagan in 1970s [Fagan 86, Fagan 01, Eickelmann 03, Answers 06, Doolan 92]. Fagan used statistical quality control to perform software inspection. The Fagan inspection methodology follows specific processes, and the participants play predefined roles. Those roles are moderator, author, reader, recorder, and inspector.

There are six steps in the Fagan inspection process:

1. **Planning**
   - –Determine if material to be inspected meets entry criteria.
   - –Arrange appropriate participants.
   - –Schedule meeting time and location.
2. **Overview**
   - –Educate participants.
   - –Assign roles to participants.

3. **Preparation**

   –Have participants learn their roles and study the materials.

4. **Meeting**

   –Start to look for defects and document any defects that are found.

5. **Rework**

   –Correct those defects found in the previous step.

6. **Follow-up**

   –Verify the corrected defects and search for other unfound defects.

6.1.2 Walkthrough

Walkthrough is another category of peer review. A walkthrough involves an individual describing his or her own work to colleagues and asking for comments [Wiegers 05a, Wiegers 05b, and Powell 03] in a meeting. The signature feature of this methodology is that the owner of the work plays a dominant role in the process while other roles are not clearly defined. There is usually very little documentation, and the procedures are not specific. As a result, it is difficult to track the defects. Also, the participants usually are not asked to prepare for the meeting.

6.1.3 Checklist

The checklist technique is an informal method of reviewing requirements. The goal of the checklist technique is to assist the stakeholders by providing questions to ask when deciding whether to trust security requirements. By using the checklist technique, stakeholders can judge the extent to which they can safely use the security requirements to build their system or software. Furthermore, the checklist technique can be used to motivate and guide stakeholders toward improving their software/requirements-engineering process and practices [Park 95a and Park 95b].

To improve the effectiveness of this methodology, the reviewer usually implements a known-defects checklist and a causal analysis [Wiegers 05a, Wiegers 05b, and Powell 03].

6.1.4 Comparison of Inspection Techniques

The team rated the three methodologies—Fagan inspection, walkthrough, and checklist—on five selected criteria:

1. **defined procedure—**the clarity of the methodology and implementation of the inspection technique
2. **number of participants—**the number of contributors needed for the implementation of the inspection technique
3. **complexity—**the degree of difficulty in understanding and properly executing the inspection technique
4. **documentation—**the organized collection of records that describe the structure, purpose, operation, maintenance, and data requirements for the inspection technique

5. **duration—**the length of time the requirements engineers and clients need to execute the inspection technique fully

The checklist technique is neither extremely easy nor extremely difficult to implement. Although checklist is less structured than other procedures, it has advantages. On the one hand, unlike Fagan inspection, the checklist technique doesn't require a long implementation time or a large number of complicated steps. As a result, we could receive feedback from clients immediately. On the other hand, unlike walkthrough, the checklist technique generates a reasonable amount of documentation and thus enhances the traceability of defects.

Given the balanced set of features, the group decided to use the checklist technique (see full comparison results in Table 16).

Table 16: Comparison of Inspection Techniques

3= Very Good, 2= Fair, 1= Poor

	Fagan Inspection	Walkthrough	Checklist
Defined procedure	3	1	2
Number of participants	1	3	3
Complexity	1	3	2
Documentation	3	1	2
Duration	1	3	3
Total Score	**9**	**11**	**12**

6.2 Methodology

6.2.1 Acme Group Inspection

In the checklist for security requirements inspection, there are four groups of questions:

1. organization
2. correctness
3. traceability
4. special issues

Table 17 shows the groups and their associated questions.

Table 17: Checklist for Security Requirements

Organization
• Are all requirements written at a consistent and appropriate level of detail? • Is the implementation priority of each requirement included? • Do the requirements include all of the known customer or system needs? • Is any necessary information missing from a requirement? • Are there areas not addressed in the document that need to be? • Is the document well organized? • Are there requirements that contain an unnecessary level of design detail?
Correctness
• Do any requirements conflict with or duplicate other requirements? • Is each requirement written in clear, concise, an unambiguous language? • Is each requirement verifiable by testing, demonstration, review, or analysis? • Is each requirement in scope for the project? • Is each requirement free from content and grammatical errors? • Can all of the requirements be implemented within known constraints?
Traceability
Is each requirement uniquely and correctly identified?
Special Issues
Are all requirements actually requirements, not design or implementation solutions?

Before the inspection, the team informed clients about the rules of filling out this checklist. The rules were very simple:

1. Check the box only if you agree completely with the statement.
2. Leave the box unchecked if you feel that some part of the statement is incomplete or inaccurate, and write any comments below the statement.

There were four participants in this process. The results of the inspection are shown in Table 18.

Table 18: Results of the Security Requirements Inspection of the Asset Management System

Category	Question		No. of Checks	Comments
Organization	**1**	Are all requirements written at a consistent and appropriate level of detail?	4	
	2	Is the implementation priority of each requirement included?	3	Don't agree with values, but the matrix changes should take care of that.
	3	Do the requirements include all of the known customer or system needs?	3	
	4	Is any necessary information missing from a requirement?	3	SR-7 is incomplete
	5	Are there areas not addressed in the document that need to be?	2	Cost/value results for each requirement would be nice. That way you could determine what's driving the priority.
	6	Is the document well organized?	4	
	7	Are there requirements that contain an unnecessary level of design detail?	3	SR-4 incomplete.
Correctness	**8**	Do any requirements conflict with or duplicate other requirements?	4	Other than what we discussed today, no.
	9	Is each requirement written in clear, concise, unambiguous language?	4	Backup is used without reservation. As revised during meeting.
	10	Is each requirement verifiable by testing, demonstration, review, or analysis?	4	
	11	Is each requirement in scope for the project?	4	
	12	Is each requirement free from content and grammatical errors?	4	
	13	Can all of the requirements be implemented within known constraints?	3	Not sure.
Traceability	**14**	Is each requirement uniquely and correctly identified?	4	
Special Issues	**15**	Are all requirements actually requirements, not design or implementation solutions?	4	

6.2.2 Beta Inspection

The team used the Acme Group checklist described in Table 17 to inspect the security requirements of the Beta application. There was only one participant in this step who required two days to complete the inspection checklist. As Table 19 shows, we are representing the checked boxes from the sole participant as "Yes/No" responses.

Table 19: Results of the Security Requirements Inspection of the Beta Application

Category	Question		Yes/No	Comments
Organization	**1**	Are all requirements written at a consistent and appropriate level of detail?	No	In particular, E3 and E4 are good examples of items that have very different scopes of detail.
	2	Is the implementation priority of each requirement included?	Yes	
	3	Do the requirements include all of the known customer or system needs?	Yes	
	4	Is any necessary information missing from a requirement?	Yes	
	5	Are there areas not addressed in the document that need to be?	Yes	Many of the requirements use language with no shared understanding among all of the participants (e.g., Indelibility).
	6	Is the document well organized?	Yes	
	7	Are there requirements that contain an unnecessary level of design detail?	Yes	B1 and D2, for example, have more detailed than needed.
Correctness	**8**	Do any requirements conflict with or duplicate other requirements?	Yes	A1 and A2 appear to conflict.
	9	Is each requirement written in clear, concise, unambiguous language?	No	B3 and B4, for example, are ambiguous. What's the difference between users (operators) and customers?
	10	Is each requirement verifiable by testing, demonstration, review, or analysis?	No	E1 and E4, for example, are not verifiable.
	11	Is each requirement in scope for the project?	No	The project scope is not well defined, so determining what is and is not in scope becomes problematic. E4 seems to be out of scope though.
	12	Is each requirement free from content and grammatical errors?	No	E2 is wrong. It should probably read "Available 24/7 via remote authenticated channel."
	13	Can all of the requirements be implemented within known constraints?	No	E4 has some built-in assumptions about how the information in the system is used by end users. We cannot control how that information is used by a customer.

Table 19: Results of the Security Requirements Inspection of the Beta Application (cont.)

Category	Question		Yes/No	Comments
Traceability	**14**	Is each requirement uniquely and correctly identified?	Yes	
Special Issues	**15**	Are all requirements actually requirements, not design or implementation solutions?	No	E3 is not a requirement.

The ARM process does provide the opportunity for participants to review and check the correctness and completeness of security requirements. In the Step 4 of the Session Phase (Details), participants are asked to evaluate each security requirement as a duplicate or in scope. At that time, the participant who answered the checklist did cross out some of the duplicates and out-of-scope security requirements.

However, there were still some duplicated and out-of-scope security requirements undetected. For instance, using the checklist technique, the participants found that security requirements A1 (information must be kept private from the outside world) and A2 (selectively secure communication with outside entities) were in conflict and that E4 (reduce/eliminate risks of inadvertent behavior) was out of scope because the scope of the project hadn't been clearly defined. By using the checklist, the security requirements came under closer scrutiny.

6.2.3 Delta Inspection

By the time we focused on Delta results, we had feedback on our checklist from Acme. Based on that feedback, we modified our instructions and removed the check box from the evaluation field.

We changed the instruction to be as follows: "Please answer the questions and write any comments in the spaces below." Also, we deleted the check boxes and left that area empty. In this way, participants would not be confused by answering "agree" or "disagree" to yes/no questions. By deleting the check boxes, we permitted participants to write in a text answer such as "yes," "no," or "unknown."

There was only one Delta participant involved in this step. The participant required five days to complete the security requirements inspection, the results of which are shown in Table 20.

Table 20: Results of the Security Requirements Inspection for Delta Web Site

Category		Question	Yes/No	Comments
Organization	**1**	Are all requirements written at a consistent and appropriate level of detail?	Yes	
	2	Is the implementation priority of each requirement included?	No	I don't understand this question.
	3	Do the requirements include all of the known customer or system needs?	Yes	
	4	Is any necessary information missing from a requirement?	Unknown	We'll know this as the project progresses.
	5	Are there areas not addressed in the document that need to be?	Unknown	Which "document"? I assume "Delta Security Requirements" document.
	6	Is the document well organized?	N/A	Being just a list, I don't think "organization" applies.
	7	Are there requirements that contain an unnecessary level of design detail?	No	If "detail" includes mention of particular implementations, then several do. See the comments below.
Correctness	**8**	Do any requirements conflict with or duplicate other requirements?	Some do	SR1, SR2, and SR3 all relate to a common "access control" requirement. It seems these are not orthogonally stated. In addition, SR5 merges requirements for the Web site with requirements for its development. They should be separated because they need to be prioritized separately.
	9	Is each requirement written in clear, concise, unambiguous language?	Yes	
	10	Is each requirement verifiable by testing, demonstration, review, or analysis?	Mostly	SR1 and SR8 depend on a subjective interpretation of "reliability" and "availability."
	11	Is each requirement in scope for the project?	Yes	
	12	Is each requirement free from content and grammatical errors?	Yes	
	13	Can all of the requirements be implemented within known constraints?	Unknown	We will see when we are done. It seems—now—that they can be.
Traceability	**14**	Is each requirement uniquely and correctly identified?	Yes	I am not sure what the meaning of this question is.
Special Issues	**15**	Are all requirements actually requirements, not design or implementation solutions?	Mostly	SR5, SR6, and SR7 all contain the word "utilize," which seems to imply a particular implementation, rather than the actual requirement. In addition, SR4 is followed by a "using" clause that implies an implementation as well.

6.3 Client Feedback

The clients did not appreciate being given a limited period of time to fill out the checklist, and they didn't feel confident about totally agreeing with almost any item. They thought it might be better to add some "gray" area in the response or some response between "agreed" and "disagreed." If this had been a real-world situation rather than a simulation, the clients would have wanted to go through every point meticulously in an iterative process.

The clients thought that most of the entries were not statements; they were questions answerable by a "Yes" or a "No." They found the entries, then, to be different from the way they were described in the instructions. Consequently, they would have preferred that the items had been reworded as statements to be agreed with, disagreed with, or commented upon. Alternatively, the instructions should have been reworded as questions that required a "Yes" or "No" response.

Of the entries with which they agreed, the clients thought that some entries had a positive connotation and others a negative one. The clients were uncomfortable about that inconsistency and felt that their results may not have been taken as intended.

Generally, the clients found value in the requirements inspection checklist. However, they also felt that some questions were ambiguous. It would be beneficial to have an accompanying document that describes each question in more detail, perhaps with examples. For example, new users of the checklist technique might not recognize the issues this client did, such as scope, orthogonality, and appropriate level of abstraction.

6.4 Recommendations

On the one hand, perhaps the team didn't properly convey the purpose of filling out the checklist; indeed, one client checked all the boxes even while not agreeing with the statements. The team made the assumption that the clients would actually be performing the inspection by reading the statements, and, once done with the inspection, they would check exactly one box for each question. However, some of the clients might have seen their check marks as an indication of agreement with the statements, not as completion of the inspection.

On the other hand, when filling out the checklist form, each client had an interpretation of the statements. Due to varied interpretations, two clients may have had different understandings of the statement and may have filled out the forms differently—one stakeholder checking the box in agreement and the other not checking the box, even if they have the same standpoint. Consider, for example, Question 5, "Are there areas not addressed in the document that need to be?" When two clients think there are no further areas that need to be addressed in the document, one client will perceive that as a positive sentence and will check the box; the other client may not check the box to convey agreement with the statement, saying in essence that there are no other areas that need to be addressed.

Because of the scenario described above, the team recommends that the statements be worded and arranged more clearly to avoid future misunderstandings. In crafting the statements, the team should make them declarative sentences with "agree" and "disagree" options or questions with "yes" and "no" answers.

At the end of this inspection process, the team could do no more than read the comments from our clients. So the team recommends that there should be a systematic method of evaluating the results of the checklist—a rule, perhaps, such as "If there are more than three boxes unchecked, the security requirements are not qualified." By some means, a future team needs to establish a threshold to qualify the inspection results.

7 Comparison

According to Pohl, “Requirements Engineering can be defined as the systematic process of developing requirements through an iterative co-operative process of analyzing the problem, documenting the resulting observations in a variety of representation formats, and checking the accuracy of the understanding gained” [Pohl 93].

This definition describes the essential elements in requirements engineering. Even if they know this definition, requirement engineers find it difficult to avoid failures during the requirements-engineering process.

Macaulay proposed that the causes of information systems failure can be classified into five types [Macaulay 96]:

1. lack of a systematic process
2. poor communication between people
3. lack of appropriate knowledge or shared understanding
4. inappropriate, incomplete, or inaccurate documentation
5. poor management of people or resources

When the objective of the requirements-engineering process is to describe a successful system clearly and exactly, Macaulay thought, the requirements engineers must have knowledge of the possible reasons of failure and use techniques to avoid failure. For this reason, we need structured elicitation techniques.

7.1 Five Dimensions of Elicitation Techniques

Based on those causes, Macaulay suggested a classification of the criteria for elicitation techniques into five different dimensions [Macaulay 96]:

1. the requirement engineering process
2. human communication within requirements
3. knowledge development
4. documentation of requirements
5. management

7.1.1 The Requirements-Engineering Process

The requirements-engineering process can be viewed as a series of activities that include problem analysis, feasibility study, and documentation. After going through these steps, the requirements can be more complete.

There are three "musts" in the requirements-engineering process: (1) the process has to produce standardized and easily managed, maintained, and documented output; (2) the process has to provide a way to evaluate its efficiency and effectiveness (i.e., we should be able to judge the outputs and improve the process based on the analyzed results); and (3) the process must provide automated means, such as CASE tools, to reduce labor and maintenance costs and increase productivity.

By considering the three aspects mentioned above, the team developed these 11 needs for requirements-elicitation techniques:

1. Support preliminary investigation.
2. Support problem definition.
3. Support feasibility study.
4. Support cost-benefit analysis.
5. Support documentation of requirements.
6. Support a step-by-step approach.
7. Support a systematic approach to record the outputs.
8. Support a systematic approach to manage the outputs.
9. Provide requirement inspection.
10. Provide CASE tools to support the requirement-elicitation process.
11. Provide easy adaptability to security requirements.

7.1.2 Human Communication within Requirements

Human communication is a very important component in the requirements-engineering process. Human communication consists of four aspects.

1. Users are consulted by means of interviews, questionnaires, and observation of behaviors. As a result, users are passively involved and the outputs are founded mainly on the professional knowledge of consultants. The more professional the consultants are, the better the output they will generate.
2. Users participate, providing the inputs of the process in an active way. The more effort and time users contribute to the process, the more successful the outcome will be. The professional background of the users is very helpful in developing requirements. Users will contribute to the process based on their professional knowledge and work content—experience that outside consultants cannot have.

3. Stakeholders participate. A responsible decision-making process asks for consideration of the effects felt by all stakeholders. One caveat for consultants: Usually, all stakeholders are not entitled to deliberate equally on all aspects of their welfare. For example, a software development decision may influence or be influenced by employees. The greater the variety of employees involved in its requirements-engineering process, the more effective the design for the software will be. While that is certainly true, stakeholders should discuss broad topics and avoid getting mired in a small range of subjects. Moreover, since stakeholders come from different backgrounds and have different viewpoints, conflicts are likely to break out. Therefore, consultants have to carefully deal with those conflicts and build consensus among stakeholders.
4. Stakeholders communicate. A good process requires that stakeholders not only passively participate in the process but also actively discuss and share their opinions and ideas with one another. Better communication among stakeholders reduces the time spent resolving conflicts—that alone can reduce effort and cost significantly.

Based on these four aspects, the team generated 11 questions:

1. Have we provided instructions on interviewing users?
2. Have we provided instructions on designing interview questions?
3. Have we provided instructions on identifying conflicts?
4. Have we provided an opportunity for users to review the outputs?
5. Have we provided an opportunity for users to analyze their own problems?
6. Do we support stakeholder identification?
7. Do we encourage communication between participants with a variety of backgrounds?
8. Do we support term definition?
9. Do we support domain familiarization?
10. Do we encourage creative thinking?
11. Do we support facilitated meetings with predefined agendas?

7.1.3 Knowledge Development

The requirements-engineering process should not only generate the requirements but also acquire knowledge from people with different backgrounds. Participants with distinct backgrounds involved in the process will provide a variety of knowledge that can be reused in the future.

The team developed the following requirements for elicitation techniques:

- Help the project manager identify the skills needed to complete the requirements-engineering process.
- Support traceability of requirements from concept to the requirement document.

7.1.4 Requirements Documentation

The requirements documentation must conform to the needs of the users and organizations. The purpose of requirements documentation is to integrate and document the outputs of the requirements-engineering process, human communication, and knowledge development. The final documentation should be consistent, complete, easily modified, traceable, and maintainable. The team distilled these principles for effective requirements documentation:

- Support identification of current system architecture.
- Support identification of constraints.
- Support identification of objectives.
- Encourage the writing of unambiguous statements.
- Encourage a complete specification to be written.
- Encourage the writing of nonconflicting and nonduplicated requirement statements.

7.1.5 Management

Management is an essential element in the requirements-engineering process. A well-managed project limits direct costs, such as labor and effort. For example, holding a discussion session will generate a cost because of the time, resources, and materials involved; but holding an ineffective discussion session will waste time and generate unwanted effort costs.

In the requirements-engineering process, the goals and requirements of the project have to be supervised. Too many goals will lead to an unfinished project, and too many or unnecessarily detailed requirements will strangle the productivity of the project.

The team developed these needs of the requirements-engineering techniques:

- Support relevant structures on the users' present work.
- Support vision and design proposal.

7.2 Comparison

Using the techniques and questions it generated for Macauley's five groups of elicitation techniques, the team evaluated IBIS, ARM, and JAD (as shown in Figure 17). In all, the team generated 33 needs for requirements-elicitation techniques.

Figure 17: Comparison of Elicitation Techniques

Grouping (Role)	No.	Criteria	Requirements-Elicitation Techniques		
			IBIS	ARM	JAD
The Requirements Engineering Process	01	Support preliminary investigation.		•	•
	02	Support problem definition.		•	•
	03	Support feasibility study.			
	04	Support cost-benefit analysis.			
	05	Support documentation of requirements.		•	
	06	Support a step-by-step approach.	•	•	•
	07	Support a systematic approach to record the outputs.	•	•	•
	08	Support a systematic approach to manage the outputs.			•
	09	Provide requirement inspection.		•	
	10	Provide CASE tools to support the requirement-elicitation process.	•		•
	11	Provide easy adaptability to security requirements.		•	
Human Communication Within Requirements	12	Provide instructions on interviewing users.	•	•	•
	13	Provide instructions on designing interview questions.		•	
	14	Provide instructions on identifying conflicts.	•	•	•
	15	Provide instructions on solving conflicts.	•	•	•
	16	Provide opportunity for users to review the outputs.	•	•	•
	17	Provide opportunity for users to analyze their own problems.	•	•	•
	18	Support stakeholder identification.		•	•
	19	Encourage communication between participants with variety of backgrounds.	•	•	•
	20	Support term definition.			
	21	Support domain familiarization.			
	22	Encourage creative thinking.	•	•	•
	23	Support facilitated meeting with pre-defined agendas.		•	•
Knowledge Development	24	Support relevant structures on the users' present work.			•
	25	Support visions and design proposal.		•	•
Requirement Documentation	26	Support identification of current system architecture.		•	•
	27	Support identification of constraints.		•	•
	28	Support identification of objectives.		•	•
	29	Encourage the writing of unambiguous statements.		•	
	30	Encourage a complete specification to be written.		•	
	31	Encourage the writing of nonconflicting and nonduplicated requirement statements.		•	
Management	32	Help the project manager identify the skills needed to complete the requirements-engineering process.	•	•	
	33	Support traceability of requirements from concept to the requirement document.		•	•

8 Conclusions

In this report, we have described our experience using the SQUARE methodology with three industry clients. Focusing on Steps 5–9 of the process, we have analyzed three structured requirements-engineering techniques in terms of their applicability to eliciting security requirements. Finally, we have experimented with the last three steps of SQUARE, reporting on our experience in categorizing, prioritizing, and inspecting the collected requirements. In this section, we discuss our overall impressions of the case studies and conclusions.

8.1 Choice of Elicitation Technique

Based on our experience, ARM outperformed both IBIS and JAD in efficiency and ability to elicit security requirements. We found that ARM was easily adapted to elicit nonfunctional, as opposed to functional (end user), requirements and that its structured approach was easy to follow.

IBIS, while effective in providing a structured means to document decision making and discussions, was not as effective in eliciting security requirements. The main problem with IBIS is that it did not provide any translation from its output, the IBIS maps, to a set of verifiable security requirements. Instead, we had to create requirements based on our own subjective impressions of the maps. Clearly, this method will not produce consistent and correct requirements.

While JAD is an extremely comprehensive and mature requirements-engineering technique, we discovered that it was nearly impossible to adapt the method to *security* requirements. Much of JAD is based on workflows and "screens," neither of which is particularly relevant in terms of quality attributes of a system. Unfortunately, we were forced to utilize only a small subset of the JAD process to attempt to elicit a set of security requirements. It's unlikely that our use of this method produced the most comprehensive and correct requirements for our client.

However, our results with ARM could be slightly biased. Since our clients for this methodology were security experts, it's possible that their background knowledge expedited the process greatly. In the future, we would like to test the ARM methodology with another project involving stakeholders that have much less security knowledge.

8.2 Categorization

We found that none of the three techniques discussed in this report provide a particularly structured approach to the categorization of requirements. In both IBIS and JAD, the requirements-engineering team is expected to categorize the results. We don't feel that such asymmetry in the

duties is the strongest approach. While ARM does allow the stakeholders to do their own categorization, it is still done in an ad-hoc manner. Given more time, we would have liked to have researched structured categorization methods or tested out a novel method of our own.

8.3 Prioritization

AHP was, by far, the most successful prioritization method that we encountered. The stakeholders were very receptive to the method, and we preferred the mathematical background of this technique.

8.4 Inspection

The checklist technique is an easy and efficient approach for inspecting security requirements. The questions in the checklist are quite straightforward and transparent. Neither consultants nor participants are required to learn this methodology.

The team judged that its failure with this methodology stemmed from spending an insufficient amount of time on the inspection step; the main purpose of our research was to find the most suitable elicitation technique. Generally speaking, the team thought that the checklist technique is quite useful in inspecting security requirements, but we suggest spending two or three sessions to review the security requirements. In these three cases, we did not ask participants to have a discussion during the inspection session. Instead of holding a discussion session, we asked participant to fill out the whole checklist alone. In order to achieve the full capability, the team suggests having iterative discussion sessions during the requirements inspection.

Choosing the checklist technique reflected a tradeoff between the quality of security requirements and time/effort. The checklist technique did not provide a structured approach for reviewing security requirements and documenting the inspection process, but it supported a quick and simple requirements examination. In contrast, Fagan inspection provides an intact and structured way to inspect requirements. Nevertheless, Fagan inspection requires much more effort than the checklist technique does.

During the inspection step, the team found that the result of the checklist technique was too subjective, and the questions in checklist were ambiguous and too general. Therefore, we suggest that future teams find better inspection methods that provide unprejudiced and objective ways to inspect the requirements—such as quantity measurement, specific performance measurement, specific evaluation criteria, or specific evaluation measurement.

8.5 Future Work

Given the success of ARM and AHP, we suggest that a future requirements-engineering team in SQUARE attempt to integrate the two approaches. ARM was very successful in eliciting and categorizing results, but we feel that it would be very easy to add AHP to the ARM process. As

stated previously, we also would like to see a structured approach for categorizing requirements. Perhaps the next SQUARE iteration will include such an approach.

Appendix A The SQUARE Methodology

Table 21: The Nine Steps of the SQUARE Methodology

Step No.	Step	Input	Techniques	Participants	Output
1	Agree on definitions.	Candidate definitions from IEEE and other standards	Structured interviews and/or focus group	Stakeholders and the requirements team	Agreed-to definitions
2	Identify safety and security goals.	Definitions, candidate goals, business drivers, policies and procedures, and examples	Facilitated work session, surveys, and/or interviews	Stakeholders and the requirements engineer	Goals
3	Develop artifacts to support security requirements definition.	Potential artifacts (e.g., scenarios, misuse cases, templates, or forms)	Work session	Requirements engineer	Needed artifacts: scenarios, misuse cases, models, templates, and forms
4	Perform risk assessment.	Misuse cases, scenarios and/or security goals	Risk assessment method and analysis of anticipated risk against organizational risk tolerance, including threat analysis	Requirements engineer, risk expert, and stakeholders	Risk assessment results
5	Select requirements-elicitation technique(s).	Goals, definitions, candidate techniques, expertise of stakeholders, organizational style, culture, level of security needed, cost- benefit analysis, and the like	Work session	Requirements engineer	Selected elicitation techniques
6	Elicit security requirements.	Artifacts, risk assessment results, and selected techniques	JAD, interviews, surveys, model-based analysis, checklists, lists of reusable requirements types, and document reviews	Stakeholders facilitated by requirements engineer	Initial draft of the security requirements

Table 22: The Nine Steps of the SQUARE Methodology (cont.)

Step No.	Step	Input	Techniques	Participants	Output
7	Categorize requirements by level (system, software, or other) and type (whether they are requirements or other kinds of constraints).	Initial requirements and architecture	Work session using a standard set of categories	Requirements engineer and other specialists as needed	Categorized requirements
8	Prioritize requirements.	Categorized requirements and risk assessment results	Prioritization method such as Triage or Win-Win	Stakeholders facilitated by requirements engineer	Prioritized requirements
9	Perform requirements inspection.	Prioritized requirements, candidate formal inspection technique	Inspection method such as Fagan or peer reviews	Inspection team	Initial selected requirements and documentation of decision-making process and rationale

Appendix B Literature Review of Elicitation Techniques

Misuse Case

A use case generally describes behavior that the system/entity owner wants the system to show [Sindre 00]. Use-case diagrams (UCDs) have proven quite helpful for the elicitation of requirements [Jacobson 92, Rumbaugh 94]. However, overlooking significant requirements leads to problems in developing software [Anton 01]. As a result, it is controversial to use UCDs for systems and quality requirements.

Misuse cases apply the concept of a negative scenario—that is, a situation that the system's owner does *not* want to occur—in a use-case context. For example, business leaders, military planners, and game players are familiar with analyzing their opponents' best moves as identifiable threats.

One significant characteristic of misuse cases is that they seem to lead to quality requirements, such as those for safety and security. Use cases describe system behavior in terms of functional (end-user) requirements. Interplay between misuse cases and use cases could improve the efficiency of eliciting all requirements in a system engineering life cycle. Misuse cases and use cases may be developed from system to subsystem levels—and lower as necessary. Lower level cases may draw attention to underlying problems not considered at higher levels and may compel system engineers to reanalyze the system design. Misuse cases are not a top-down method, but they provide opportunities to investigate and validate the security requirements necessary to accomplish the system's mission.

The student team decided not to apply misuse cases as an elicitation technique in the project in order to validate other elicitation techniques. According to the 2004 report on the SQUARE methodology [Mead 04], a prior student team used use/misuse cases to analyze Acme Group's asset management system. One of the current team's objectives was to apply an alternate elicitation methodology to provide data for evaluation against the prior team's output.

Soft Systems Methodology (SSM)

SSM deals with problem situations in which there is a high social, political, and human activity component [Checkland 90]. The SSM can deal with "soft problems" that are difficult to define, rather than "hard problems" that are more technology oriented. Soft problems are the situations in which we know the system is not performing in the desired manner, and we want to find out why and see if we can do anything about it. SSM was developed by Peter Checkland to deal with soft problems; it is composed of seven stages:

1. Find out the problem situation.
2. Express the problem situation through rich pictures (i.e., representations of organizational structure and processes pertinent to the problem situation).
3. Select how to view the situation and produce root definitions.
4. Build conceptual models of what the system must do for each root definition.
5. Compare the conceptual models with the real world.
6. Identify feasible and desirable changes.
7. Make recommendations to improve the problem situation [Checkland 89].

The primary benefit of SSM methodology is that it provides structure to soft problem situations and enables their resolution in an organized manner. It compels the developer to discover a solution that goes beyond technology.

The team decided not to use SSM because it requires analysts to carry out a large number of interviews, workshops, and discussions with the stakeholders to develop ideas and build the richest possible picture [Wilson 90]. Another requirement of SSM gave the team pause: SSM requires analysts to participate in the working process and observe the client on-site. As a result of business concerns, time constraints, and labor-intensive costs, the team concluded they were unable to implement this methodology [Christel 92].

Quality Function Deployment (QFD)

QFD is "an overall concept that provides a means of translating customer requirements into the appropriate technical requirements for each stage of product development and production" [QFD 05]. The distinguishing attribute of QFD is the focus on customer needs throughout all product development activities. By using QFD, organizations can promote teamwork, prioritize action items, define clear objectives, and reduce development time [QFD 05].

Although QFD covers a broad portion of the product development life cycle, the earlier stages of the process are applicable to requirements elicitation for software engineering. These stages include

1. identifying the customer (stakeholders)
2. gathering high-level customer requirements
3. constructing a set of system features that can satisfy customer needs
4. creating a matrix to evaluate system features against satisfaction of customer needs

Since QFD does not specify precisely how to gather customer requirements, we felt that it was not entirely suitable for integration into the SQUARE process. However, we found that some of the planning steps in QFD are similar to the steps of the SQUARE methodology, suggesting that QFD is instead better suited for categorizing and prioritizing product requirements after elicitation.

Controlled Requirements Expression (CORE)

CORE is a requirements analysis and specification method that clarifies the user's view of the services to be supplied by the proposed system and the limitations imposed by that system's operational environment, in conjunction with some degree of performance and reliability investigation [Mullery 79]. CORE provides methods and notations for every phase of "elicitation, specification and analysis of requirements, and results in a structured data flow form of specification" [Finkelstein 92].

Advantages

First, CORE is a mature methodology with a set of guidelines on how to apply the method to a problem [SDS 86]. The method is a flexible approach to requirements elicitation, permitting it to be applied to a wide set of problems.

Second, CORE encourages contributions from many different communities to develop requirements. CORE delineates the tasks of the members of this community (e.g., Viewpoint Authorities) and structures the communication between these groups [Christel 92].

Third, an incremental examination of information flows and processing activities can be executed using CORE, with each previous step providing the foundation for the present step of specification.

Finally, CORE assists in discovering design limitations.

Disadvantages

First, the role of the requirements analyst in CORE seems too passive. The assumptions are that the users will propose solutions and the analyst will make sense of them and that the customer

authority, not the analyst, will resolve conflicting views of the proposed system. The analyst is not likely to be able to take such a passive role and form a coherent system requirements specification.

Second, the steps of the method are not very precise. Some concepts in CORE are not well defined, and the use of information gathered during certain phases of CORE is unclear.

Third, CORE is deficient in representing constraints (e.g., conditions under which an action is performed) and real-time requirements.

Fourth, CORE does not provide any modeling primitives to support the expression of internal functions (e.g., with CORE the analyst cannot decompose actions into subactions).

Finally, CORE does not provide the framework for storing the rationale behind the requirements with the requirements themselves. The rationale for any given viewpoint can be assumed to rest with the Viewpoint Authority, but the individual or set of individuals in that role is likely to change over time. Therefore, it is worthwhile to document the rationale for the requirements and to store that rationale with the requirements [Christel 92].

Evaluation

Elicitation techniques can serve these purposes:

- information gathering
- requirements expression and analysis
- validation [Christel 92]

Since two of the selected techniques (IBIS and JAD) are used for gathering information during requirements elicitation, we thought that we would like to try a method that serves a different function. CORE is intended for slightly different purposes than JAD and IBIS—for expressing and analyzing requirements during requirements elicitation. Also, CORE uses a reasonably rich set of representations and expresses requirements in a structured, diagrammatic notation that fosters communication and is less ambiguous than natural language. However, due to the significant limitations of CORE described above, the team decided against using it for the case studies.

Issue-Based Information Systems (IBIS)

Developed by Horst Rittel, the IBIS method is based on the principle that the design process for complex problems, which Rittel terms “wicked” problems, is essentially an exchange among the stakeholders in which they bring their personal expertise and perspective to the resolution of design issues [Kunz 70].

Any problem, concern, or question can be an issue and may require discussion and resolution in order for the design to proceed. The IBIS model centers on this “give-and-take” that constitutes the design process. The model was developed over 20 years ago and has been implemented effec-

tively in varied design situations from architectural design to planning at the World Health Organization.

The IBIS model focuses on the articulation of the key issues in the design problem. Each issue can have many positions. A position is a statement or assertion that resolves the issue. Often positions will be mutually exclusive of each other, but the method does not require this. Each of an issue's positions, in turn, may have one or more arguments that either support or object to it.

There are several types of links among the concepts in IBIS. For example, a position responds to an issue with a "responds to" link. Arguments must be linked to their positions with either "supports" or "objects to" links. Issues may generalize or more narrowly focus other issues and they may question or be suggested by other issues, positions, and arguments.

Advantages

The flexibility of IBIS allowed the team to generate requirements for the asset management system, which was nearly complete. Many of the other elicitation techniques reviewed by the team assumed that the requirements-engineering team would be present during many of the decisions made in the early design phase. The SQUARE team was not afforded that luxury since the system was nearly complete and was in the final stages of development.

The IBIS model allows stakeholders to reexamine what design decisions were made and why. IBIS diagrams provide a clear picture of the "arguments" taking place and the justifications used.

Disadvantages

The flexibility of IBIS can also lead to a perceived lack of coherence. The team discussed the possibility of producing diagrams detailing every thought or possible issue with the client's product. Examining such a free-ranging set of issues would likely generate IBIS maps that would be far too complicated to analyze in a reasonable period of time.

Joint Application Development (JAD) and Accelerated Requirements Method (ARM)

Unlike QFD, JAD is specifically designed for the development of large computer systems. The goal of JAD is to involve *all* stakeholders in the design phase of the product via highly structured and focused meetings. Typical participants in the session include a facilitator, end users of the product, main developers, and observers.

In the preliminary phases of JAD, the requirements-engineering team is tasked with fact finding and information gathering. Typically, the outputs of this phase, as applied to security requirements elicitation, are security goals and artifacts. The actual JAD session is then used to validate this information by establishing an agreed-upon set of security requirements for the product.

The success of JAD is a result of highly focused sessions combined with the involvement of all stakeholders of a project. In the team's view, this technique had potential for security requirements elicitation, so we implemented the methodology along with a variation of it in this work. Specifically, the variant technique we used is known as ARM. It differs from traditional JAD in that the facilitator is neutral, the brainstorming techniques are slightly different, and there are variations on the group dynamic behaviors.

Feature-Oriented Domain Analysis (FODA)

FODA is a domain analysis and engineering technique that focuses on developing reusable assets [Kang 90]. The FODA methodology was founded on two modeling concepts: abstraction and refinement [Kean 97]. Abstraction is used to create domain products from the specific applications in the domain. These generic domain products abstract the functionality and designs of the applications in a domain. The generic nature of the domain products is created by abstracting factors that make one application different form other related applications. The FODA method advocates that applications in the domain should be abstracted to the level where no differences exist between the applications. Specific applications in the domain are developed as refinements of the domain products.

The FODA method has three phases:

1. **context analysis**
   Information required for various activities is gathered from various sources.
2. **domain modeling**
   Product line requirements are analyzed using a set of domain models. Common and variable requirements are identified using a technique called feature modeling. Feature models consist of diagrams that represent features in a hierarchical structure. These requirements are further analyzed using several structured system-analysis techniques such as data flow diagrams, entity relationship diagrams, and functional diagrams.
3. **architecture modeling**
   Domain models are used to create an architecture model. The architecture model can be instantiated to develop individual applications [Kuloor 02].

Because we did not plan to do other work in this domain, we could not take advantage of the main feature of the FODA method—reusability. Moreover, the development of a domain model was viewed as too costly in terms of time and effort. Since our team has a tight schedule, the FODA method was not suitable for our purpose.

Critical Discourse Analysis (CDA)

CDA uses sociolinguistic methods to analyze verbal and written discourse [Schiffrin 94]. Sociolinguistics assigns special significance to the structure of speech and texts and provides methods

for specifying the linguistic features of different types of discourse units and the way they are tied together into larger units of meaning [Alvarez 02].

Moreover, CDA concerns itself with examining social context along the lines of ideology, power, and inequality. Through discourse examination, topics of power inequalities usually along the lines of race, class, gender, sexuality, and occupation are exposed. Therefore, CDA demystifies what is taken to be common sense by "de-familiarizing" it and signaling its functions and consequences in sustaining the social order.

However, since CDA relies on the knowledge of both linguistic and social theory, it was not possible for our team to implement this methodology. In other words, the emphasis during CDA is on linguistic structure and interaction [Alvarez 02]. Researchers or interviewers must be able to analyze clients' intonation, volume, pacing, and other qualities of speech to capture the mood and feel of the interview. Besides that capability, researchers must be able to use transcription conventions to maintain overlaps, exclamations, questions, pauses, and emphasis, to provide a readable script. The members of our team come from IT and policy backgrounds and have almost no knowledge about linguistics. For this reason, it was impossible for us to apply CDA in SQUARE.

Appendix C IBIS Questions and Revisions

Table 22: Original IBIS Questions, Suggested Actions, Comments, and Revisions[7]

No.	Original Question	Action	Comment or Revision
1	What are the implementations of the asset management system version of Knowledge Center (KC)?	Skip	Not suitable for IBIS: this information should come from Steps 1–4 of SQUARE
2	What level of technological savvy should the client possess to successfully operate and maintain KC?	Reword	What level of training, if any, will users of the system require?
3	What information should be kept confidential from public view and why?	Reword	What elements of the system, if any, require some level of confidentiality?
4	What information must be safe from unauthorized modification and why?	Reword	What data elements, if any, require some level of integrity guarantee?
4	What data elements, if any, require some level of integrity guarantee?	Reword	How should the system implement data integrity guarantees?
5	What elements of the system should have an availability guarantee and why?	Reword	What elements of the system, if any, require some availability guarantee?
6	Should the KC have any built-in redundancy?	Skip	Mostly redundant with Question 5
7	What do you see as the greatest threat to the system and why?	Skip	This information will likely come from Step 4 (Risk Assessment) of SQUARE
8	What are your known, existing vulnerabilities and when can they be expected to be resolved?	Skip	Not suitable for IBIS: this information should come from Steps 1–4 of SQUARE
9	What is the most critical security problem in KC?	Skip	Redundant with Question 7
10	What type of access control will be necessary and why?	Reword	What type of access control, if any, should the system utilize?
11	What measures should be in place against insider threats and why?	Reword	What measures, if any, should the system utilize to protect against insider threats?
12	How are you attempting to integrate security to the product and why?	Skip	While this question worked well for a mature system undergoing development, it's not really relevant to a product undergoing design.
13	What type of secure coding measures should be implemented in developing KC?	Reword	What secure coding techniques, if any, are necessary?
14	How should you secure the essential services from malicious attack?	Reword	How should the system's essential services be protected against threats?
15	What type of incident-detection methods do you have in place and why?	Reword	What type of intrusion-detection measures should be in place, if any?
16	How should you detect an attack on Knowledge Center?	Skip	Redundant with Question 15
17	How should you document the attacks on the Knowledge Center and the corresponding response?	Reword	How should the system record intrusions and intrusion responses, if at all?

[7] Duplicate question numbers indicate that the question underwent multiple revisions.

Table 23: Original IBIS Questions, Suggested Actions, Comments, and Revisions (cont.)[8]

No.	Original Question	Action	Comment or Revision
17	How should the system record intrusions and intrusion responses, if at all?	Skip	Redundant with Question 15
18	What are the essential assets/services of KC?	Skip	Not suitable for IBIS: this information should come from Steps 1–4 of SQUARE
19	In the event of an emergency and KC is down, what features should be present to ensure that service is returned in a timely manner?	Reword	What incident-recovery mechanisms should the system utilize, if any?
20	What would you regard as the most critical element(s) of the system and why?	Skip	Not suitable for IBIS: this information should come from Steps 1–4 of SQUARE.
21	What type of incident-recovery mechanisms are necessary and why?	Skip	Redundant with Question 19
22	How quickly should KC recover and provide essential services?	Reword	How quickly should the system recover from being down?
23	Should the asset management system KC implementation come with configuration requirements for the client?	Reword	What level of individual configuration for each installation does the system require?
23	What level of individual configuration for each installation does the system require?	Reword	What level of individual configuration for each client does the system require?
24	What are the risks posed by a KC implementation in a less-than-ideal system configuration?	Skip	This information will likely come from Step 4 (Risk Assessment) of SQUARE.
25	What should you do to protect KC from system configuration risks (or errors)?	Reword	What measures should be in place to protect against configuration errors, if any?
26	What should be the recommended computing requirements for optimal operation of the asset management system version of KC (i.e., RAM, disk space, or firewall)?	Skip	Not suitable for IBIS: this information should come from Steps 1–4 of SQUARE.
27	What are the costs of not implementing security mechanisms?	Skip	This information will likely come from Step 4 (Risk Assessment) of SQUARE
28	What cost should you suffer, when the essential service is compromised (financial, time, brand, etc.)	Skip	This information will likely come from Step 4 (Risk Assessment) of SQUARE.
29	What platforms should the system be designed for?	Add	
30	What should the application identify before allowing users to use its capabilities (e.g., client application or human user)?	Reword	What should the application identify before allowing users to use its capabilities?
31	Why use windows logon and why not? Why use SSL and why not?	Reword	What authentication mechanisms, if any, should the system utilize?

[8] Duplicate question numbers indicate that the question underwent multiple revisions.

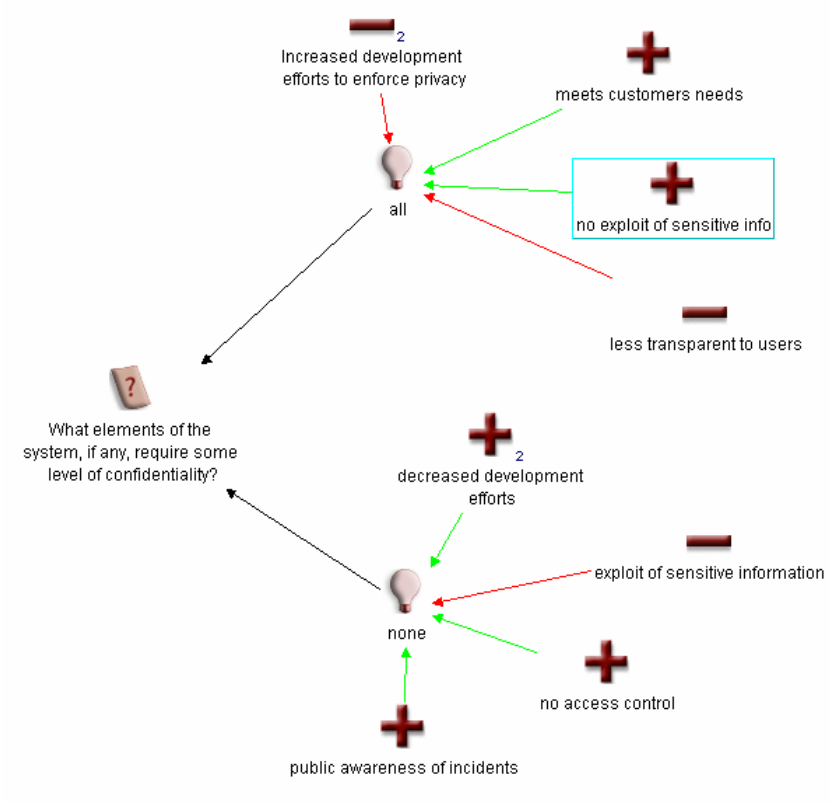
Table 23: Original IBIS Questions, Suggested Actions, Comments, and Revisions (cont.)[9]

No.	Original Question	Action	Comment or Revision
32	What are the reasons not to use Intrusion Detection System?	Skip	This question is combined with Question 15.
33	What are the user privileges for the system?	Reword	What data should be classified?
34	What are the notification methods to inform customer of the new password?	Skip	Asset management system will not provide this function.
35	How should the application respond to the failure of the system?	Skip	Asset management system will not provide this function.
36	Who should assign the access privilege to the system?	Skip	Asset management system will not provide this function.
37	What restriction should be required for simultaneous logins?	Reword	Should the user be allowed to login more than once simultaneously?
38	Should there be a time-out function for login?	Skip	Asset management system will not provide this function.
39	What kind of logs should you keep in the system?	Add	
39	What kind of logs should you keep in the system?	Reword	What kind of logs should you keep in the asset management system?
40	What tamper-proof records should the application create and store?	Reword	How should the tamper-proof records be created and stored?
40	How should the tamper-proof records be created and stored?	Reword	What mechanisms should be in place to provide nonrepudiation service?
41	What are the specific data and communications that require privacy?	Reword	How should you protect specific data and/or communication lines?
41	How should you protect specific data or communication lines?	Skip	
42	What is the security mechanism that needs to be audited?	Skip	Redundant with Question 39
43	What are the password selection criteria?	Reword	What should be the password selection criteria?
43	What should be the password selection criteria?	Skip	Asset management system will not provide this function.
44	What backup method should the system utilize, if any?	Add	
45	Who should be responsible for the backup?	Add	
46	What measures should be in place for ongoing maintenance to prevent configuration error?	Add	
47	What information should be undeniable in the asset management system?	Add	

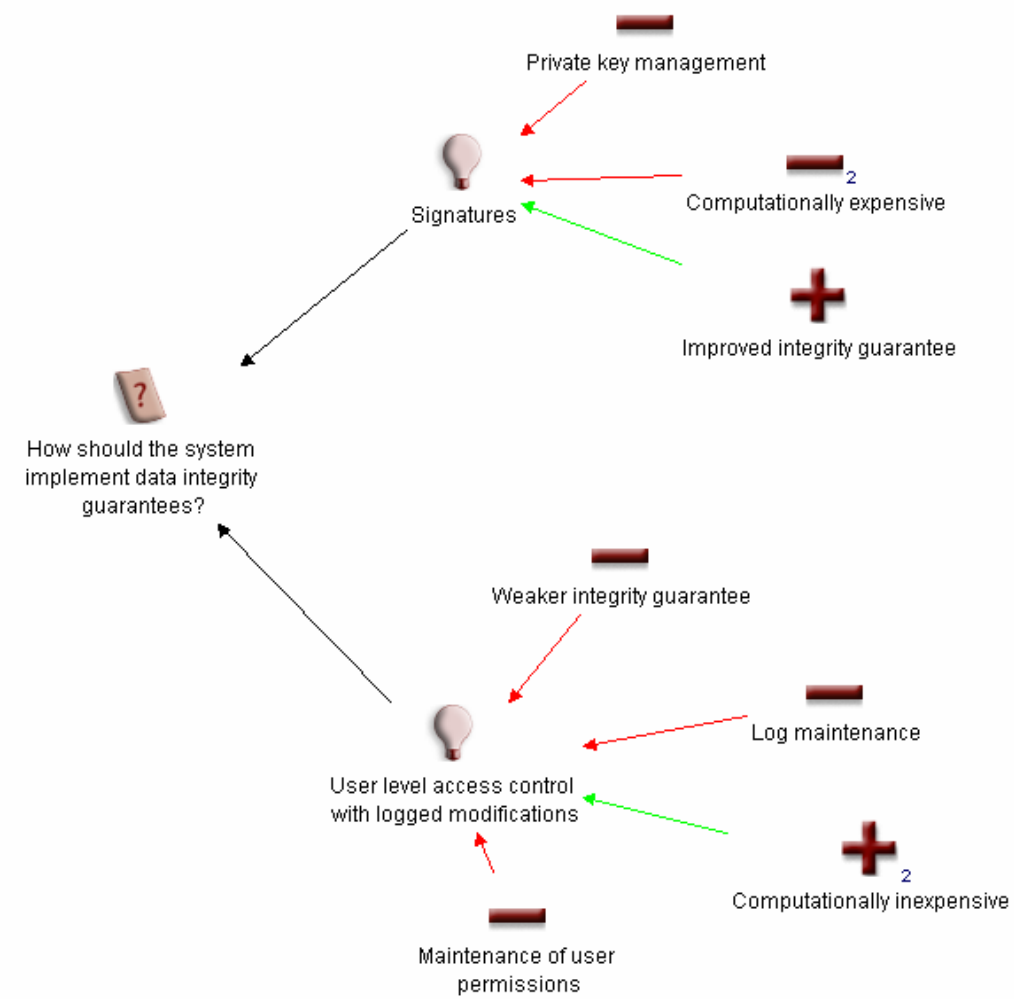
[9] Duplicate question numbers indicate that the question underwent multiple revisions.

Appendix D IBIS Maps

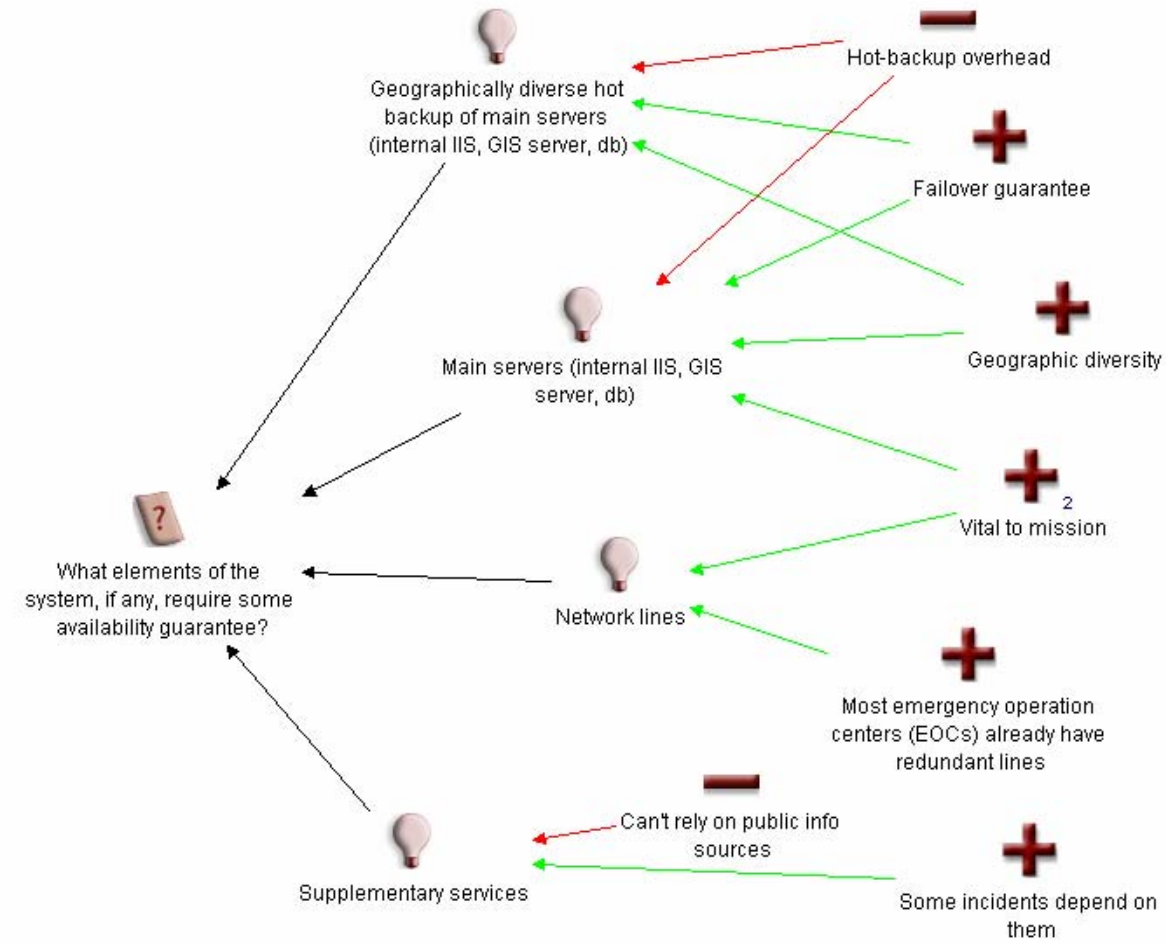
IM-01 (for descriptor map, see Table 23)



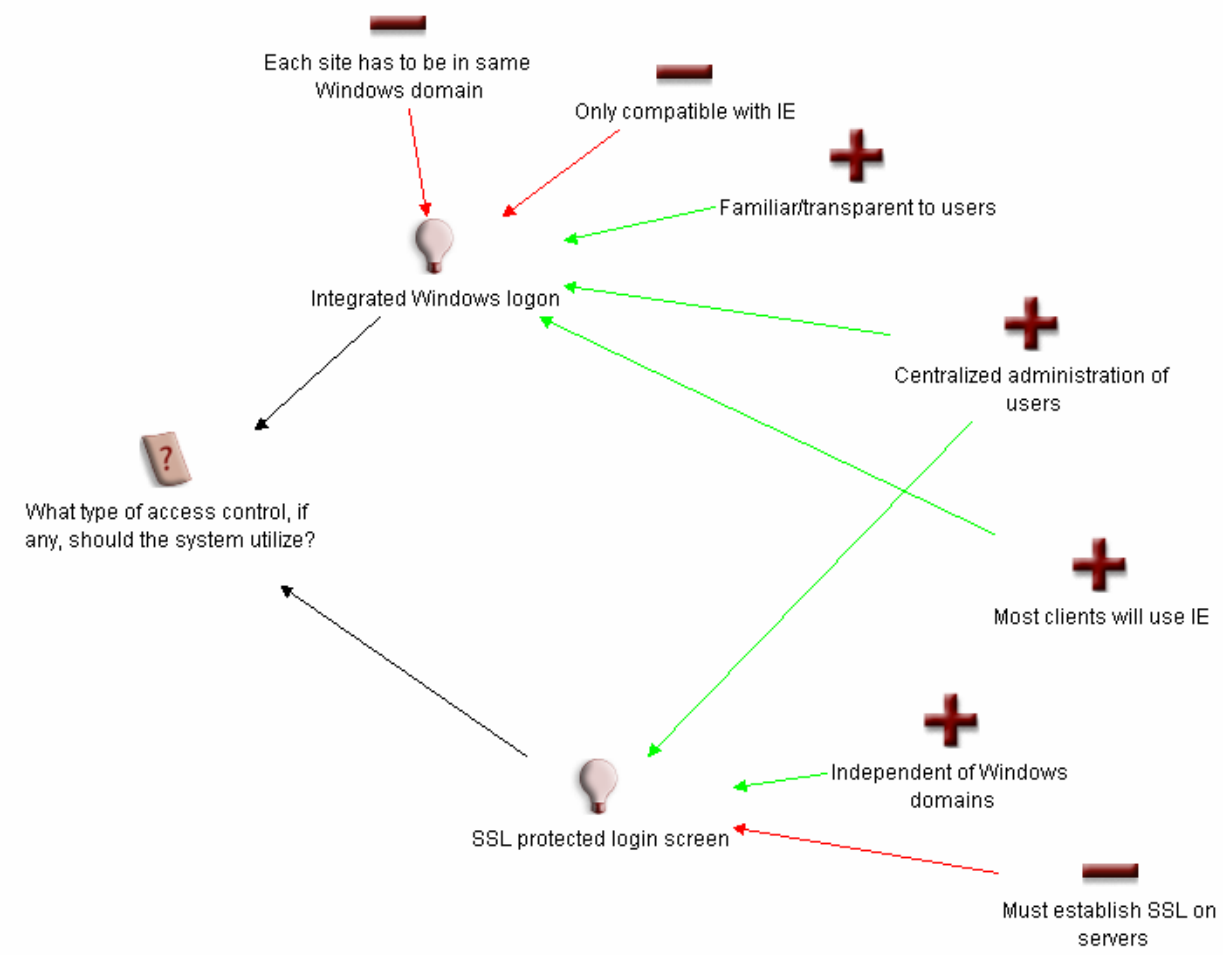
IM-02 (for descriptor map, see Table 24)



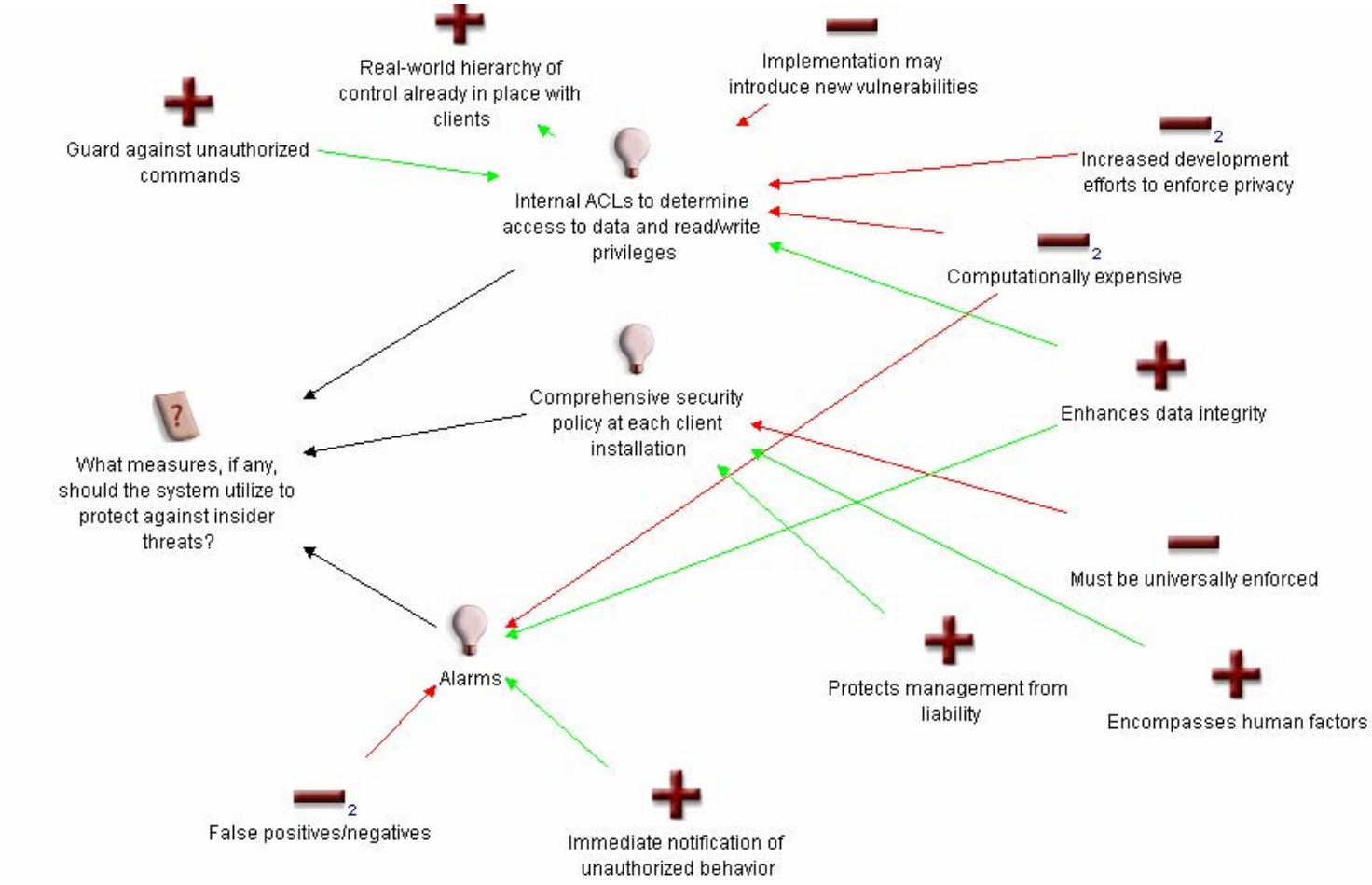
IM-03 (for descriptor map, see Table 25)



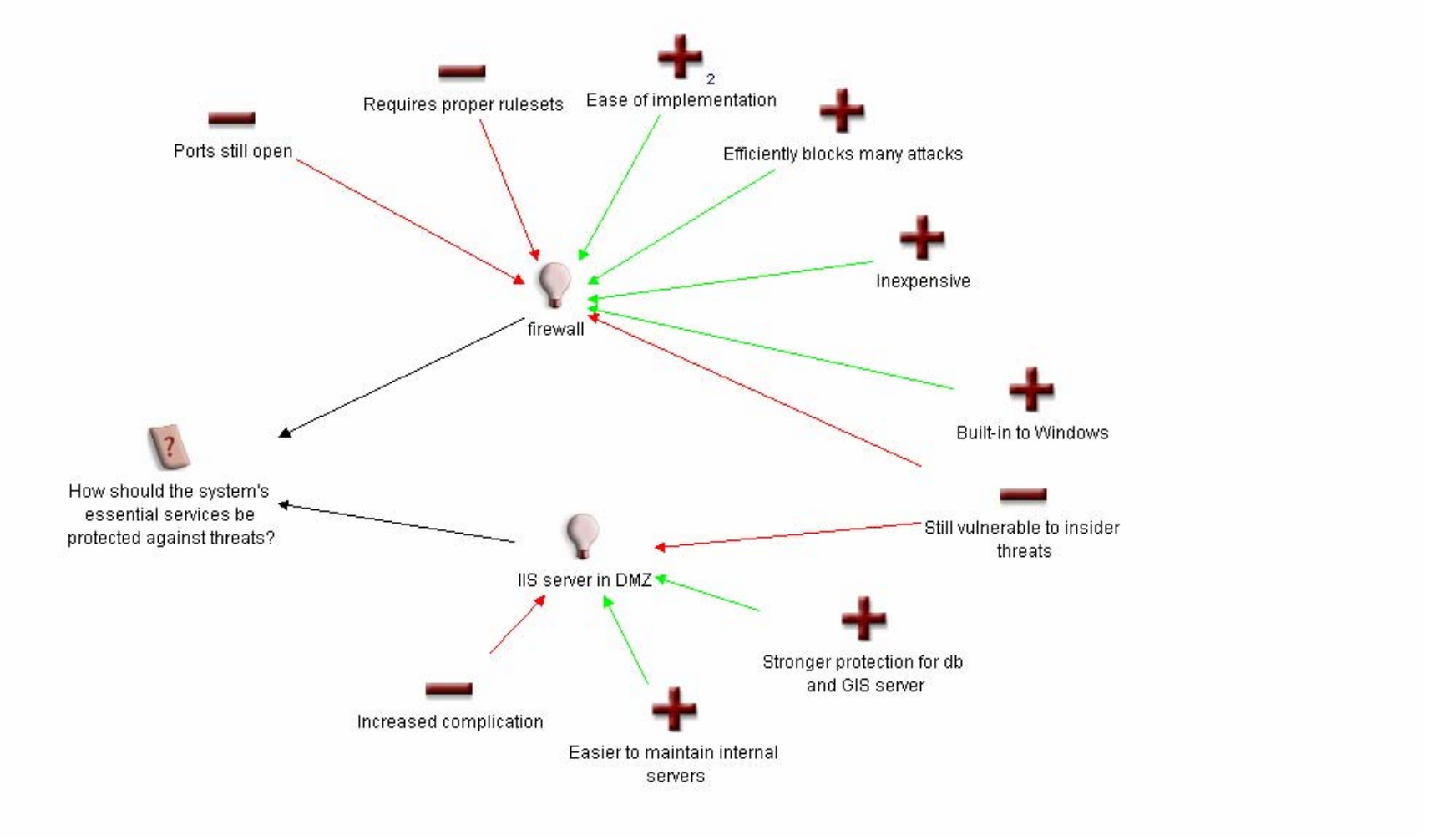
IM-04 (for descriptor map, see Table 26)



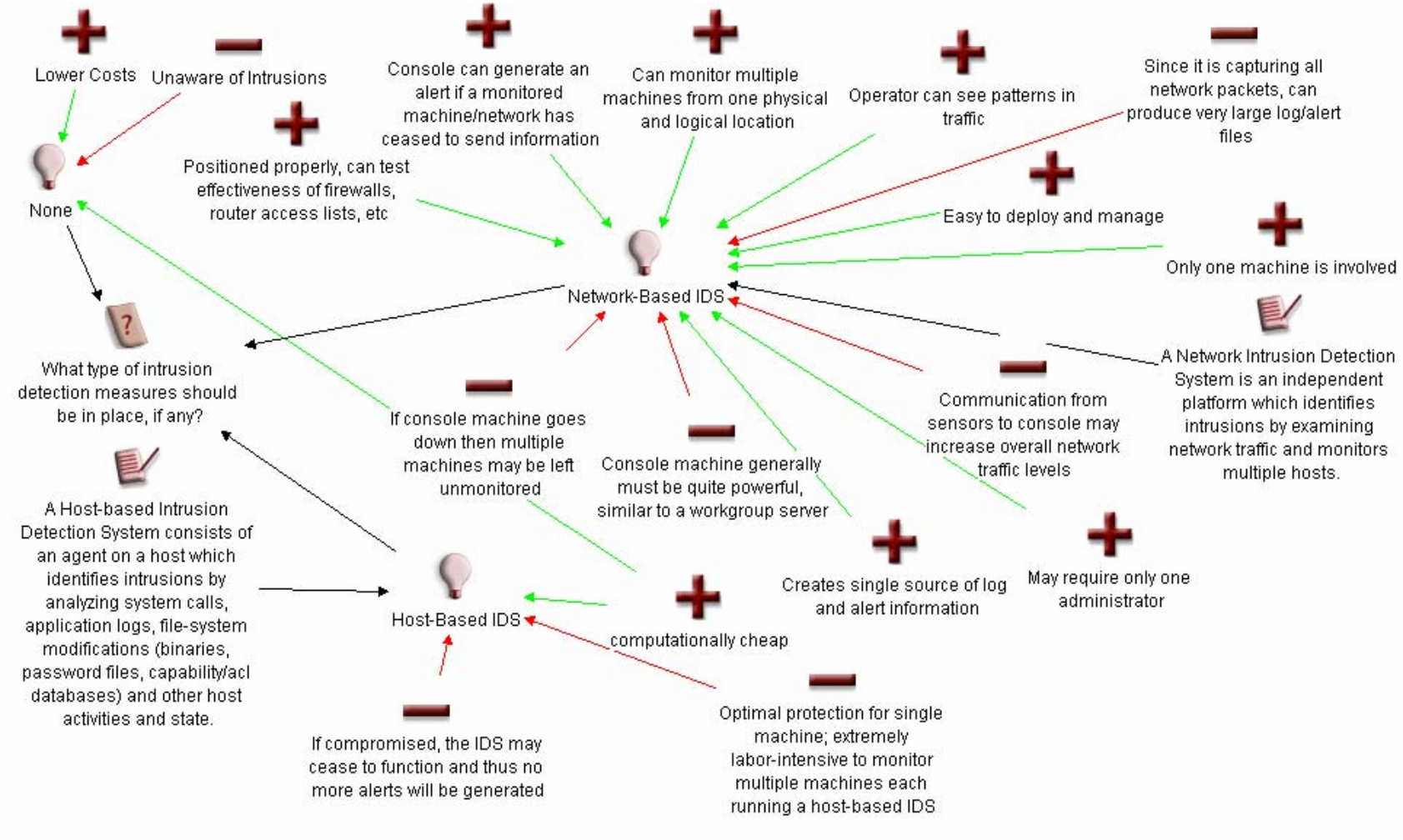
IM-05 (for descriptor map, see Table 27)



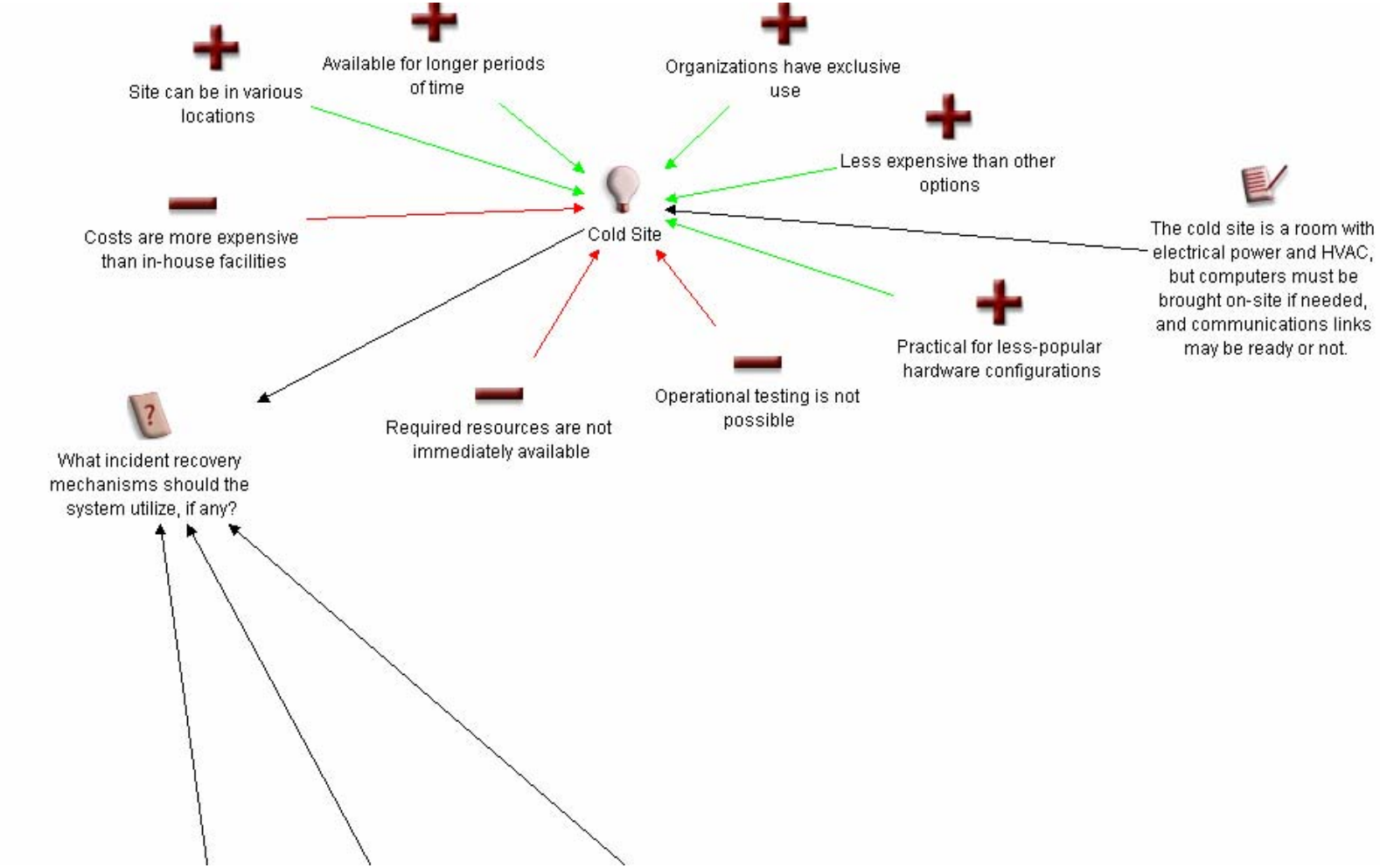
IM-06 (for descriptor map, see Table 28)



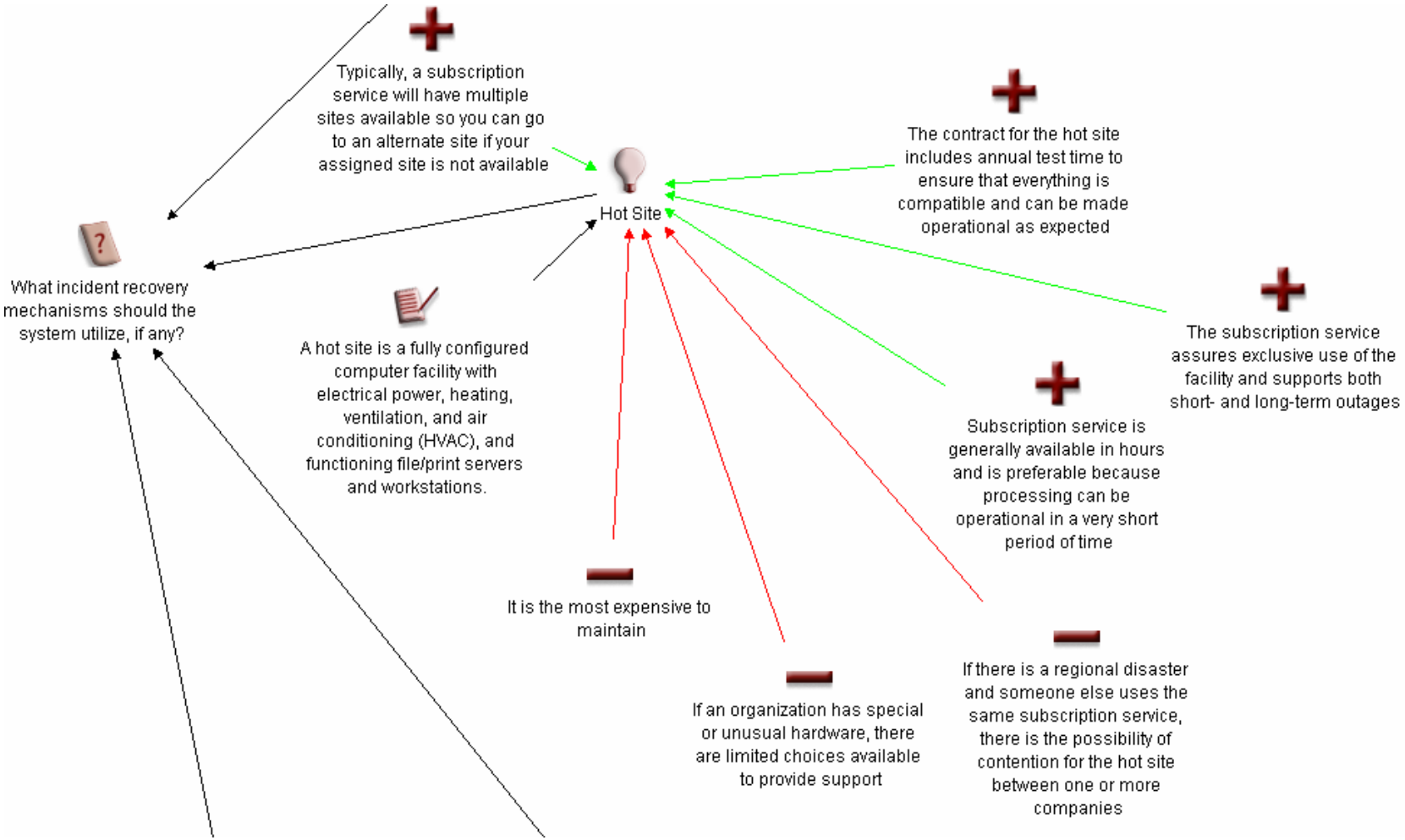
IM-07 (for descriptor map, see Table 29)



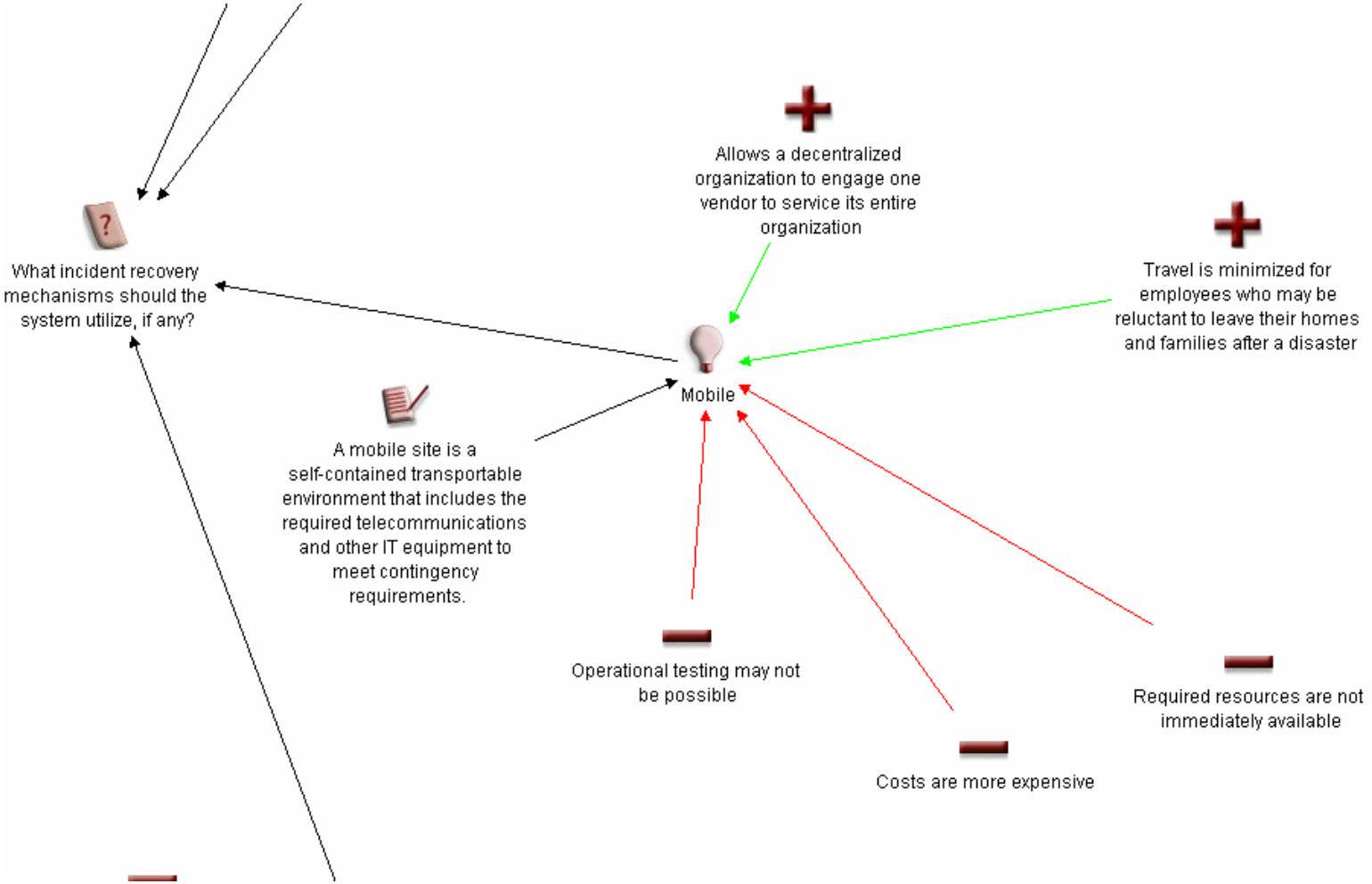
IM-08 (for descriptor map, see Table 30)



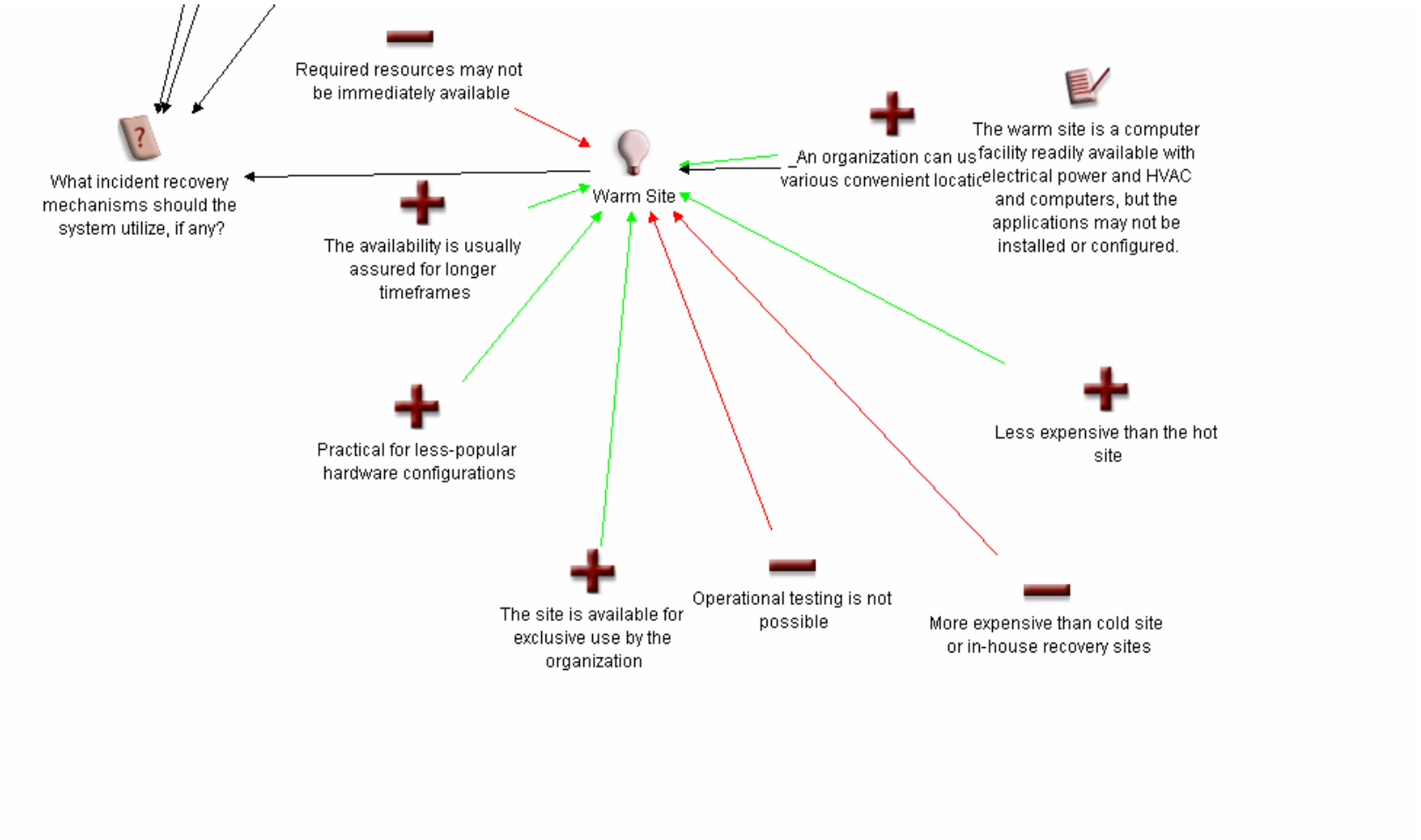
IM-08 (cont.)


Typically, a subscription service will have multiple sites available so you can go to an alternate site if your assigned site is not available
Hot Site
The contract for the hot site includes annual test time to ensure that everything is compatible and can be made operational as expected
What incident recovery mechanisms should the system utilize, if any?
A hot site is a fully configured computer facility with electrical power, heating, ventilation, and air conditioning (HVAC), and functioning file/print servers and workstations.
The subscription service assures exclusive use of the facility and supports both short- and long-term outages
Subscription service is generally available in hours and is preferable because processing can be operational in a very short period of time
It is the most expensive to maintain
If an organization has special or unusual hardware, there are limited choices available to provide support
If there is a regional disaster and someone else uses the same subscription service, there is the possibility of contention for the hot site between one or more companies

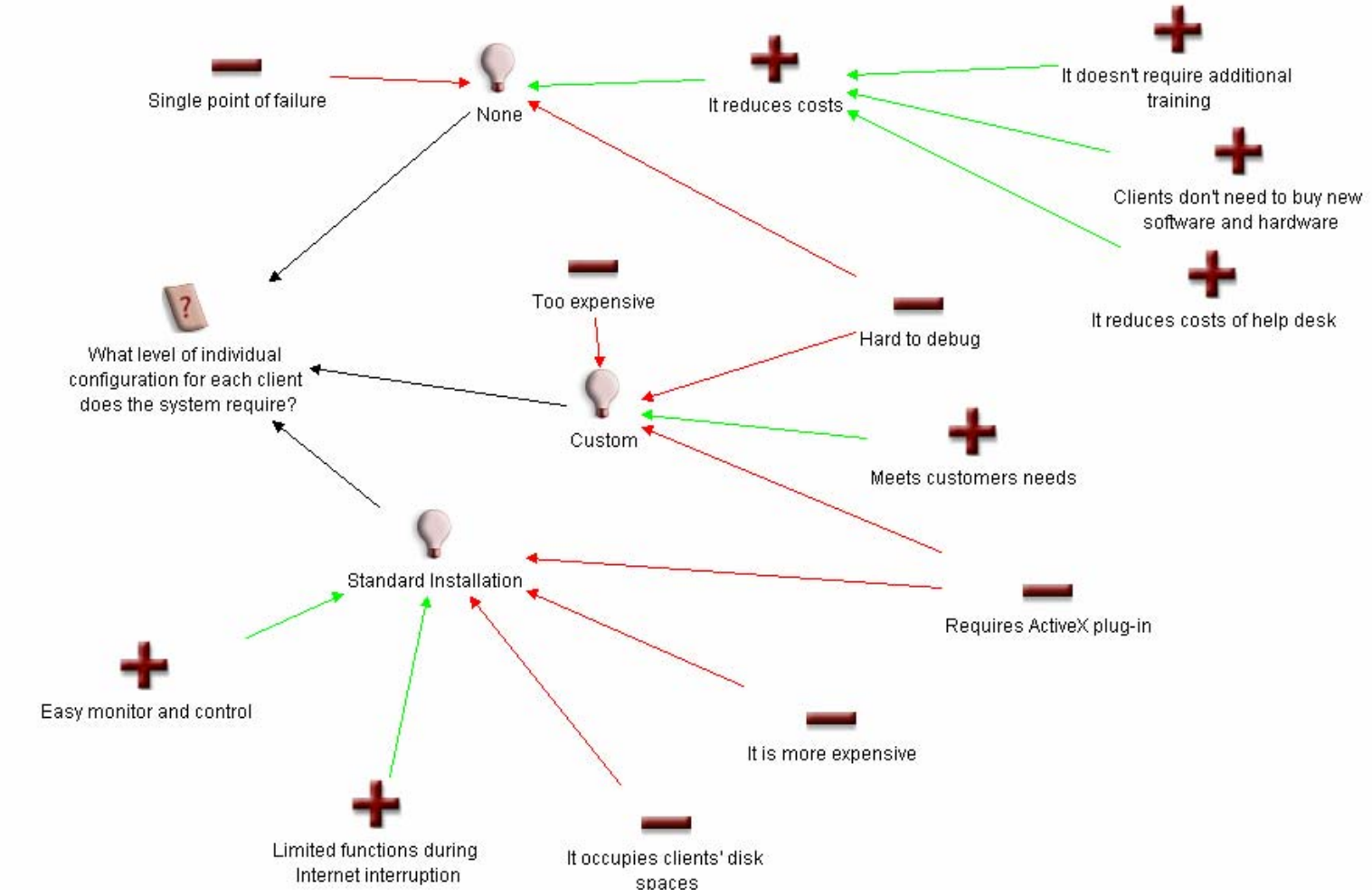
IM-08 (cont.)


What incident recovery mechanisms should the system utilize, if any?
Allows a decentralized organization to engage one vendor to service its entire organization
Travel is minimized for employees who may be reluctant to leave their homes and families after a disaster
Mobile
A mobile site is a self-contained transportable environment that includes the required telecommunications and other IT equipment to meet contingency requirements.
Operational testing may not be possible
Required resources are not immediately available
Costs are more expensive

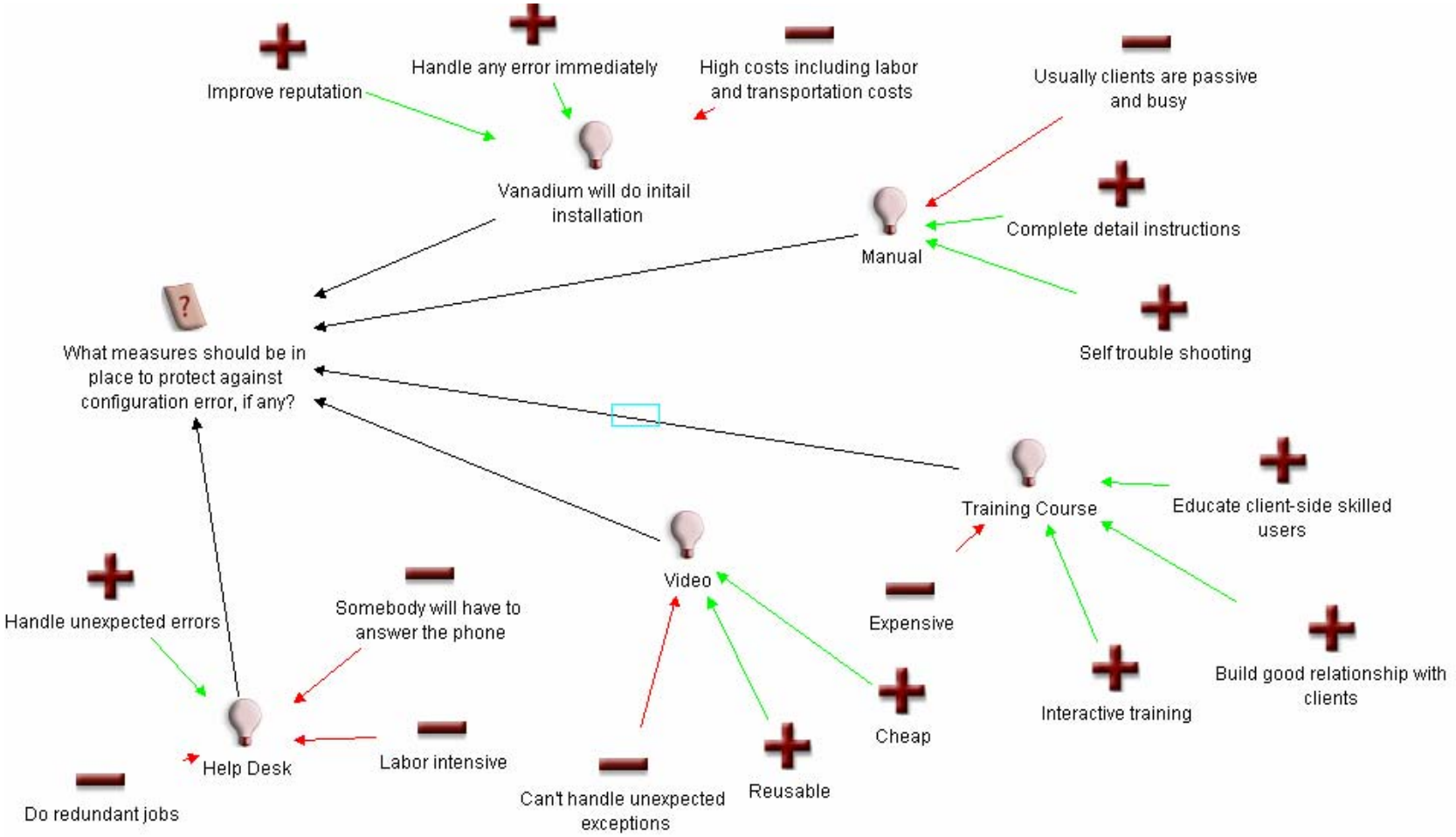
IM-08 (cont.)


Required resources may not be immediately available
What incident recovery mechanisms should the system utilize, if any?
Warm Site
_An organization can us various convenient locatic
The warm site is a computer facility readily available with electrical power and HVAC and computers, but the applications may not be installed or configured.
The availability is usually assured for longer timeframes
Practical for less-popular hardware configurations
Less expensive than the hot site
The site is available for exclusive use by the organization
Operational testing is not possible
More expensive than cold site or in-house recovery sites

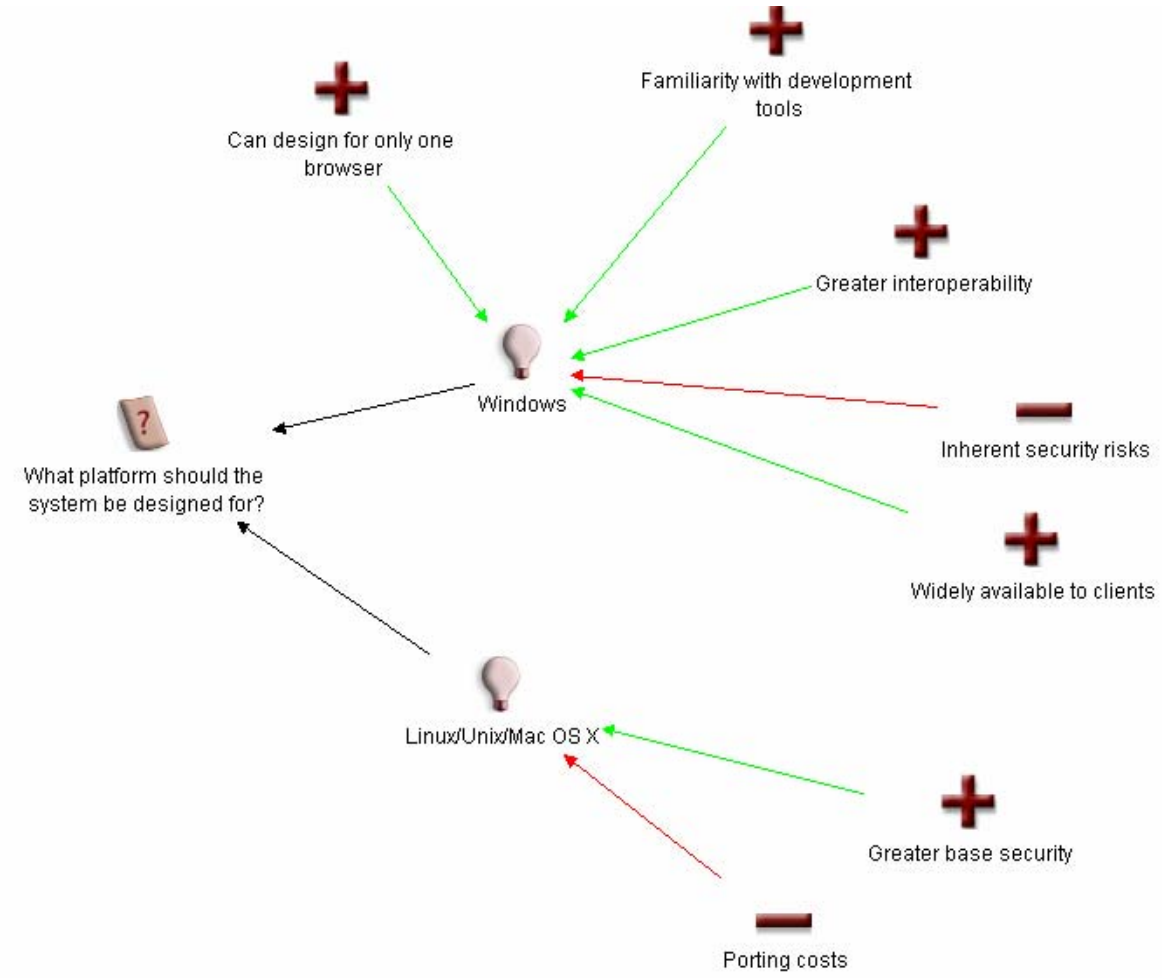
IM-09 (for descriptor map, see Table 31)



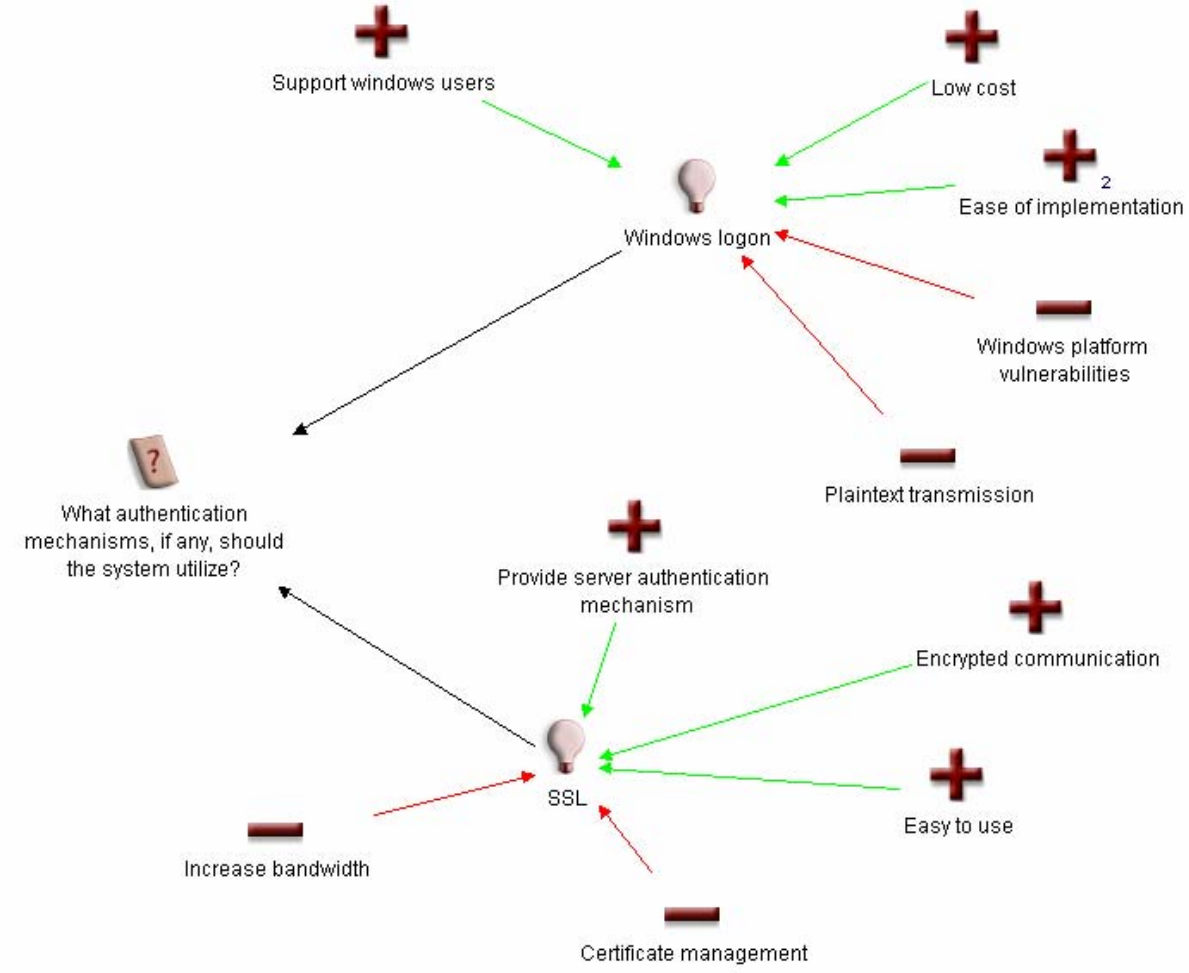
IM-10 (for descriptor map, see Table 32)



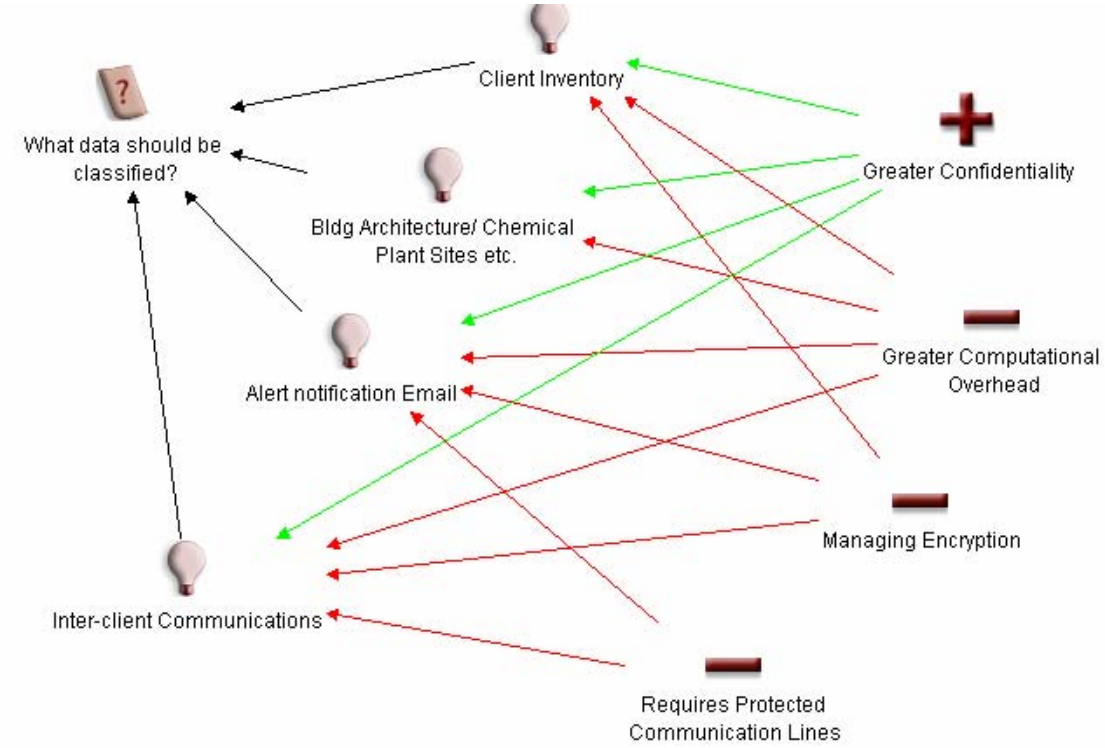
IM-11 (for descriptor map, see Table 33)



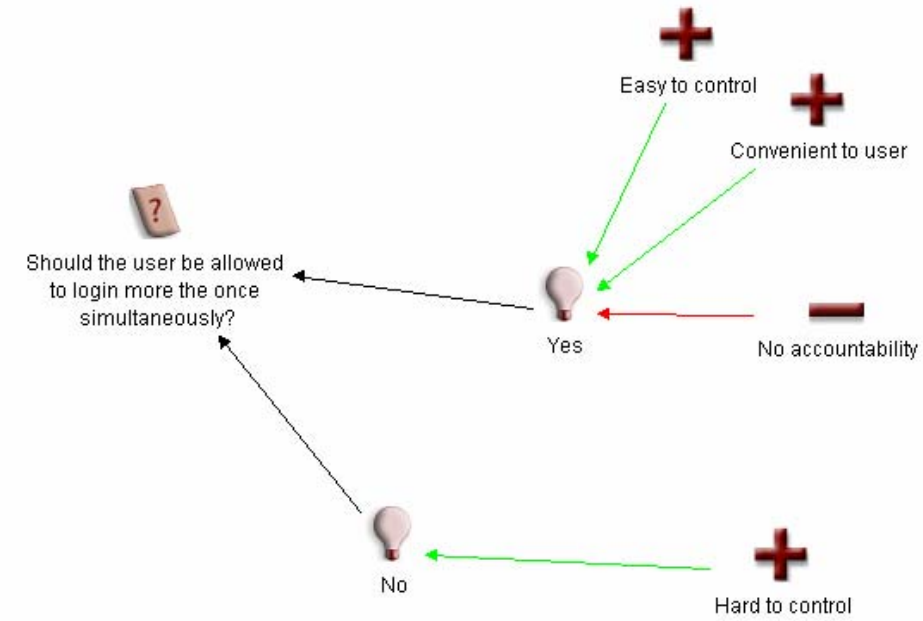
IM-12 (for descriptor map, see Table 34)



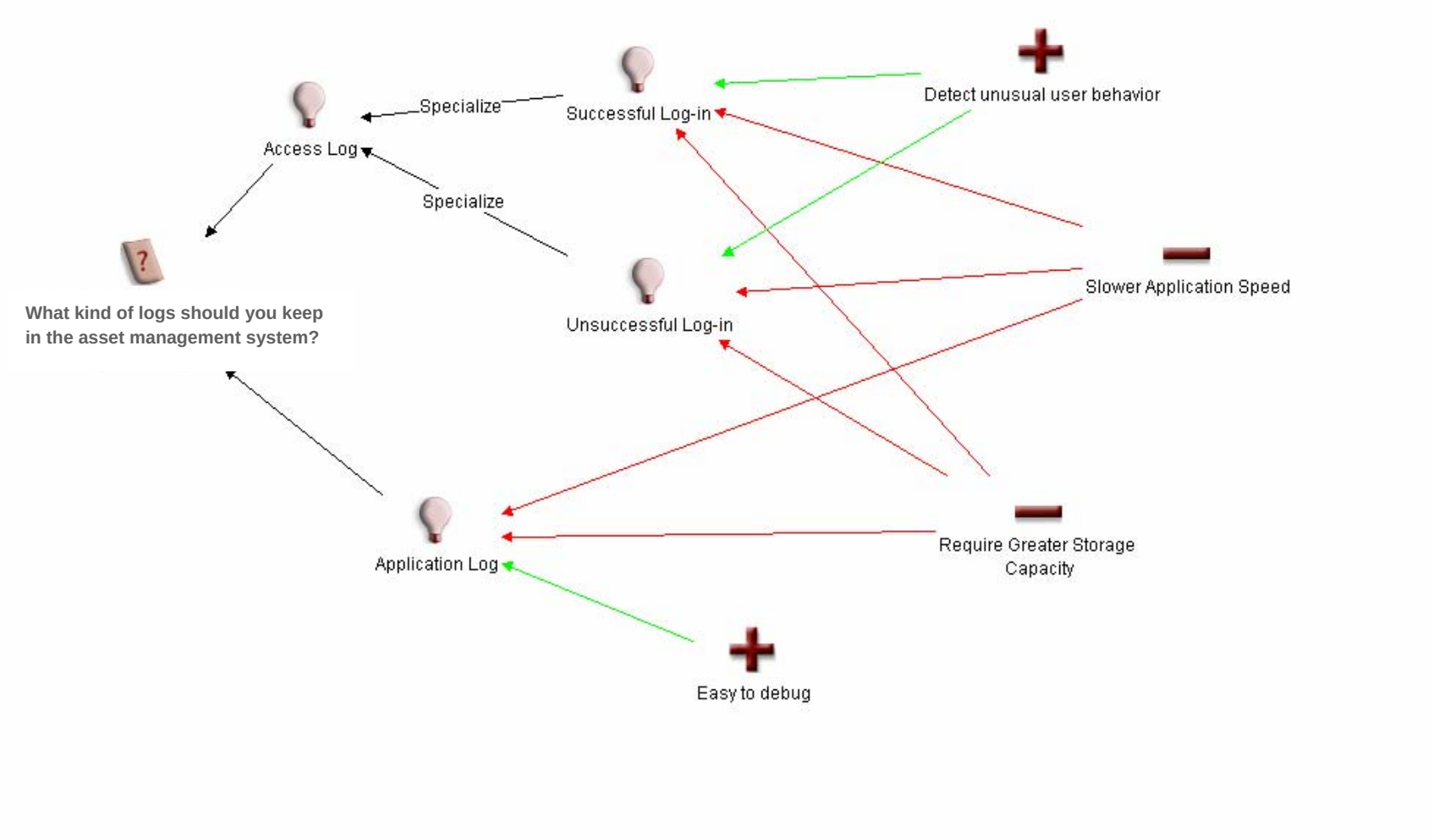
IM-13 (for descriptor map, see Table 35)



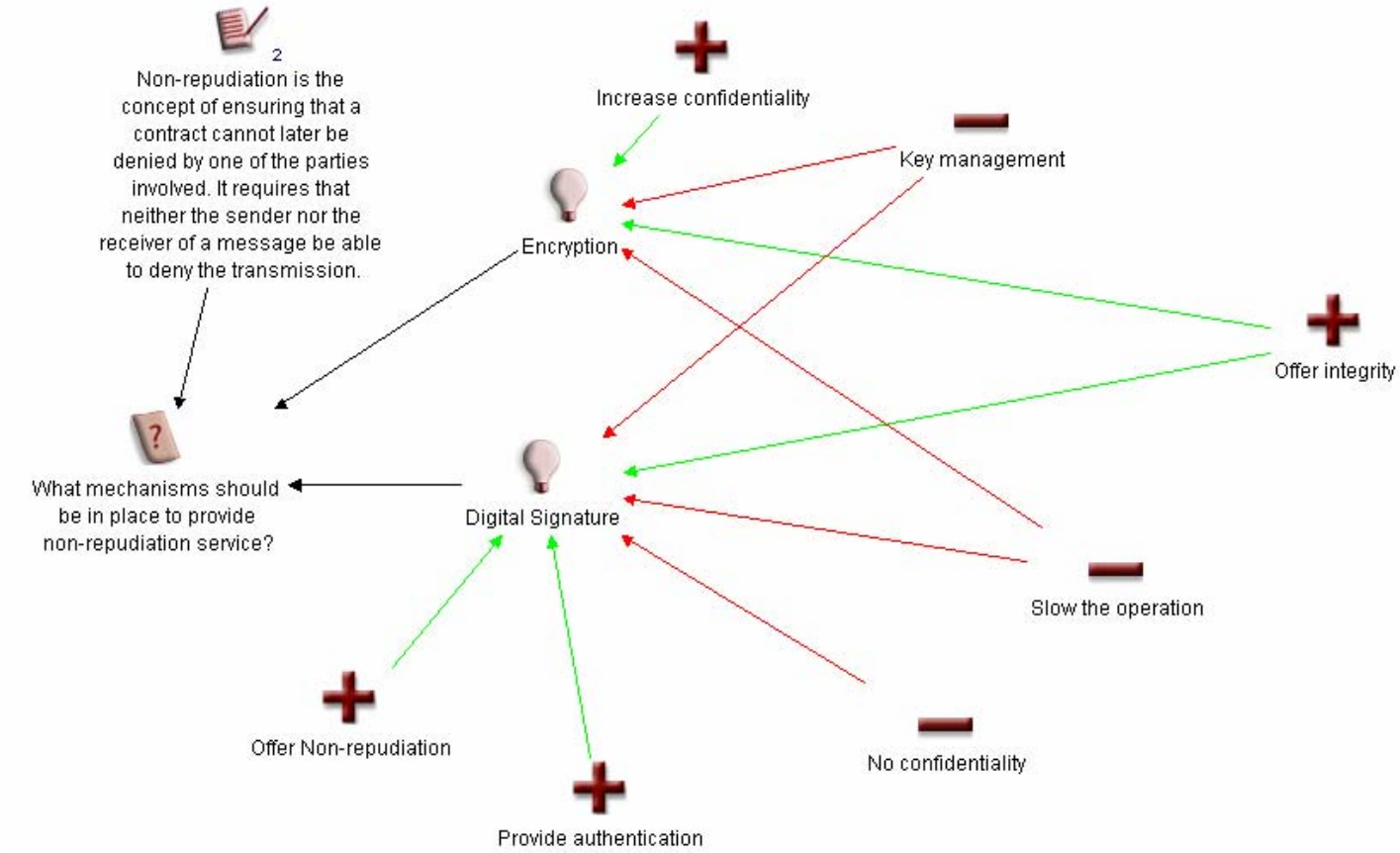
IM-14 (for descriptor map, see Table 36)



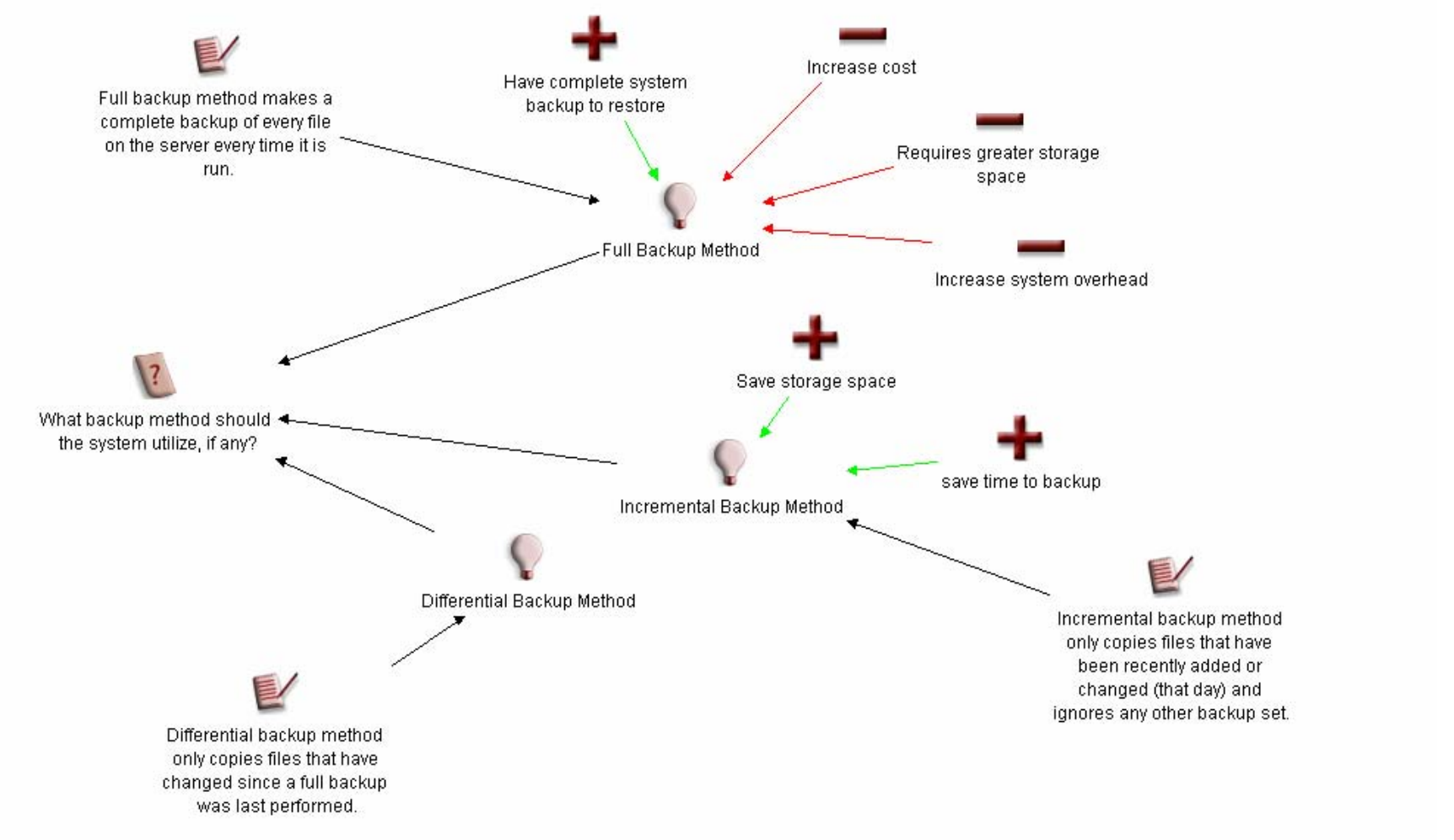
IM-15 (for descriptor map, see Table 37)



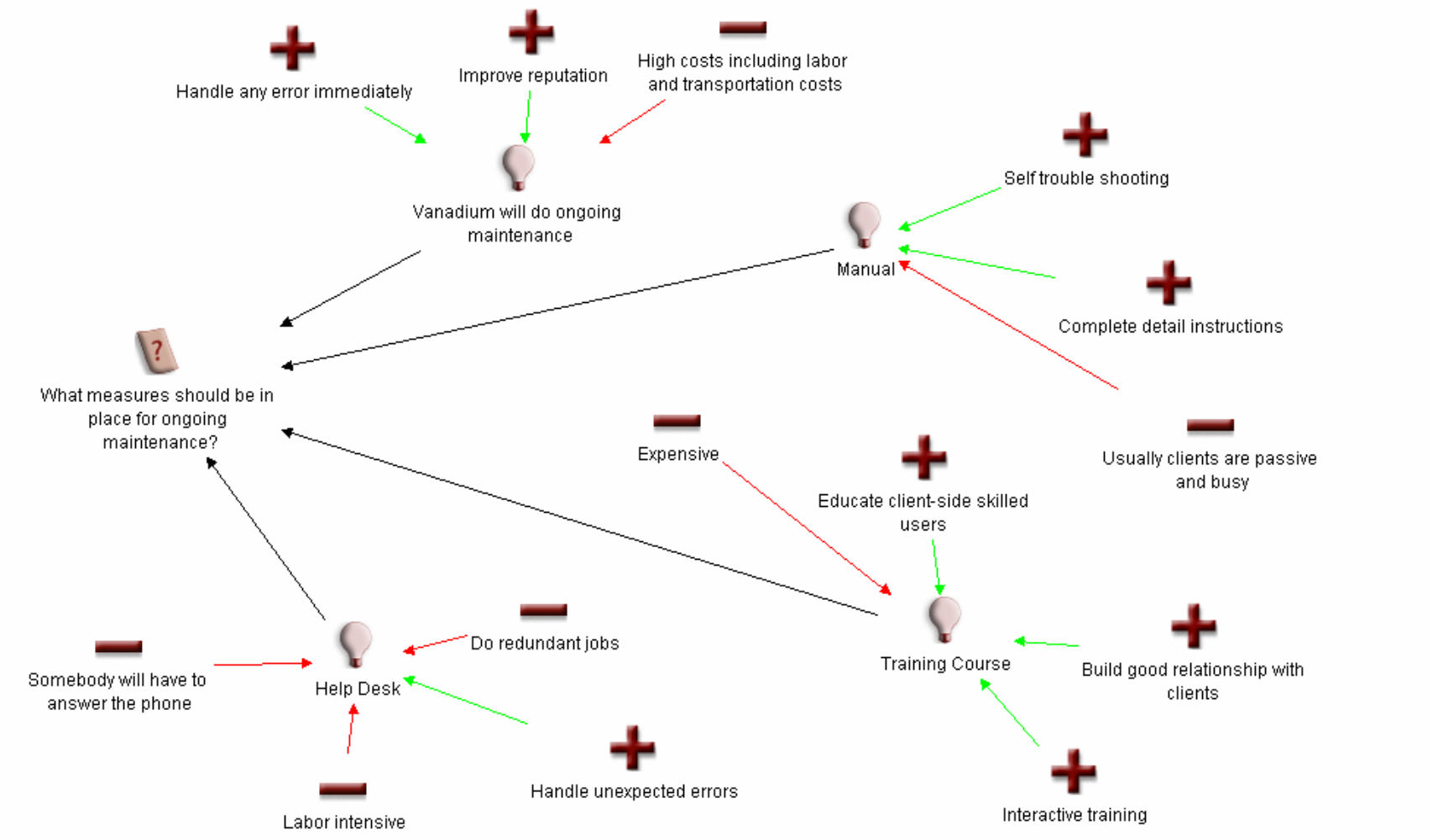
IM-16 (for descriptor map, see Table 38)



IM-17 (for descriptor map, see Table 39)



IM-18 (for descriptor map, see Table 40)



Appendix E IBIS Map Descriptors

Table 23: IBIS Map Descriptor for Map IM-01

Number	**IMD-01**
Issue	What elements of the system, if any, require some level of confidentiality?
Positions	1. All 2. None
Security Attributes Affected	x Confidentiality _ Integrity _ Availability
Survivability Attributes Affected	x Resist _ Evolve _ Recognize _ Recover
Related IBIS Map Descriptors	IMD-13, IMD-14
Related Security Requirement(s)	SR-2, SR-3

Table 24: IBIS Map Descriptor for Map IM-02

Number	**IMD-02**
Issue	How should the system implement data-integrity guarantees?
Position:	1. Cryptographic digital signatures 2. User-level access control with logged modifications
Security Attributes Affected	_ Confidentiality x Integrity _ Availability
Survivability Attributes Affected	_ Resist _ Evolve x Recognize _ Recover
Related IBIS Map Descriptors	IMD-16
Related Security Requirement(s)	SR-5

Table 25: IBIS Map Descriptor for Map IM-03

Number	**IMD-03**
Issue	What elements of the system, if any, require some availability guarantee?
Positions	1. Geographically diverse hot-backup of main servers (internal IIS [a Web server], GIS, and db) 2. Main servers (internal IIS, GIS, and db) 3. Network links 4. Supplementary services
Security Attributes Affected	_ Confidentiality _ Integrity x Availability
Survivability Attributes Affected	x Resist _ Evolve _ Recognize _ Recover
Related IBIS Map Descriptors	IMD-07, IMD-09, IMD-18
Related Security Requirement(s)	SR-4, SR-7, SR-8, SR-9

Table 26: IBIS Map Descriptor for Map IM-04

Number:	**IMD-04**
Issue	What type of access control, if any, should the system utilize?
Positions	1. Integrated Windows logon 2. SSL-protected login screen
Security Attributes Affected	x Confidentiality _ Integrity _ Availability
Survivability Attributes Affected	x Resist _ Evolve _ Recognize _ Recover
Related IBIS Map Descriptors	IMD-12, IMD-13
Related Security Requirement(s)	SR-1, SR-2, SR-3, SR-5, SR-6

Table 27: IBIS Map Descriptor for Map IM-05

Number:	**IMD-05**
Issue	What measures, if any, should the system utilize to protect against insider threats?
Positions	1. Internal access control lists (ACLs) to determine access to data and read/write privileges 2. Comprehensive security policy at each client installation 3. Alarms
Security Attributes Affected	x Confidentiality x Integrity x Availability
Survivability Attributes Affected	x Resist _ Evolve _ Recognize _ Recover
Related IBIS Map Descriptors	IMD-04, IMD-12, IMD-13
Related Security Requirement(s)	SR-1, SR-2, SR-3, SR-5, SR-6

Table 28: IBIS Map Descriptor for Map IM-06

Number	**IMD-06**
Issue	How should the system's essential services be protected against threats?
Positions	1. Firewall 2. IIS Server in DMZ
Security Attributes Affected	x Confidentiality x Integrity x Availability
Survivability Attributes Affected	x Resist _ Evolve x Recognize x Recover
Related IBIS Map Descriptors	IMD-01, IMD-07, IMD-08, IMD-12, IMD-13, IMD-15, IMD-17, IMD-18
Related Security Requirement(s)	SR-1, SR-2, SR-3, SR-4

Table 29: IBIS Map Descriptor for Map IM-07

Number	**IMD-07**
Issue	What type of intrusion detection measures should be in place, if any?
Positions	1. Network-based IDS 2. Host-based IDS 3. None
Security Attributes Affected	x Confidentiality x Integrity x Availability
Survivability Attributes Affected	x Resist _ Evolve x Recognize _ Recover
Related IBIS Map Descriptors	IMD-03, IMD-04, IMD-10
Related Security Requirement(s)	SR-2, SR-3, SR-5, SR-6

Table 30: IBIS Map Descriptor for Map IM-08

Number	**IMD-08**
Issue	What incident recovery mechanisms should the system utilize, if any?
Positions	1. Cold Site 2. Hot Site 3. Mobile 4. Warm Site
Security Attributes Affected	_ Confidentiality x Integrity x Availability
Survivability Attributes Affected	_ Resist _ Evolve _ Recognize x Recover
Related IBIS Map Descriptors	IMD-17
Related Security Requirement(s)	SR-7, SR-8, SR-9

Table 31: IBIS Map Descriptor for Map IM-09

Number	**IMD-09**
Issue	What level of individual configuration for each client does the system require?
Positions	1. Standard Installation 2. Custom 3. None
Security Attributes Affected	_ Confidentiality _ Integrity x Availability
Survivability Attributes Affected	_ Resist x Evolve _ Recognize _ Recover
Related IBIS Map Descriptors	IMD-01, IMD-02, IMD-03, IMD-04, IMD-06
Related Security Requirement(s)	SR-1, SR-2, SR-3

Table 32: IBIS Map Descriptor for Map IM-10

Number	**IMD-10**
Issue	What measures should be in place to protect against configuration errors, if any?
Positions	1. Acme Group will do initial installation 2. Manual 3. Training Course 4. Video 5. Help Desk
Security Attributes Affected	x Confidentiality x Integrity _ Availability
Survivability Attributes Affected	x Resist _ Evolve _ Recognize _ Recover
Related IBIS Map Descriptors	IMD-1, IMD-10
Related Security Requirement(s)	SR-4

Table 33: IBIS Map Descriptor for Map IM-11

Number	**IMD-11**
Issue	What platforms should the system be designed for?
Positions	1. Windows 2. Linux/Unix/Mac OS X
Security Attributes Affected	x Confidentiality _ Integrity _ Availability
Survivability Attributes Affected	x Resist _ Evolve _ Recognize _ Recover
Related IBIS Map Descriptors	IMD-03, IMD-17
Related Security Requirement(s)	SR-1, SR-2, SR-4

Table 34: IBIS Map Descriptor for Map IM-12

Number:	**IMD-12**
Issue	What authentication mechanisms, if any, should the system utilize?
Positions	1. Windows logon 2. SSL
Security Attributes Affected	x Confidentiality _ Integrity _ Availability
Survivability Attributes Affected	x Resist _ Evolve _ Recognize _ Recover
Related IBIS Map Descriptors	IMD-03, IMD-14, IMD-15, IMD-16
Related Security Requirement(s)	SR-1, SR-2, SR-3, SR-5, SR-6

Table 35: IBIS Map Descriptor for Map IM-13

Number:	**IMD-13**
Issue	What data should be classified?
Positions	1. Floor plan 2. Alert notification email
Security Attributes Affected	x Confidentiality _ Integrity _ Availability
Survivability Attributes Affected	x Resist _ Evolve _ Recognize _ Recover
Related IBIS Map Descriptors	None
Related Security Requirement(s)	SR-3, SR-5

Table 36: IBIS Map Descriptor for Map IM-14

Number	**IMD-14**
Issue	Should the user be allowed to log in more than once simultaneously?
Positions	1. Yes 2. No
Security Attributes Affected	x Confidentiality _ Integrity _ Availability
Survivability Attributes Affected	x Resist _ Evolve x Recognize _ Recover
Related IBIS Map Descriptors	IMD-03, IMD-12, IMD-15
Related Security Requirement(s)	SR-1, SR-2, SR-5, SR-6

Table 37: IBIS Map Descriptor for Map IM-15

Number	**IMD-15**
Issue	What kind of logs should you keep in the asset management system?
Positions	1. Access Log 2. Application Log
Security Attributes Affected	_ Confidentiality x Integrity _ Availability
Survivability Attributes Affected	_ Resist _ Evolve x Recognize _ Recover
Related IBIS Map Descriptors	IMD-01, IMD-12, IMD-14
Related Security Requirement(s)	SR-5, SR-6

Table 38: IBIS Map Descriptor for Map IM-16

Number	**IMD-16**
Issue	What mechanisms should be in place to provide non-repudiation service?
Positions	1. Encryption 2. Digital Signature
Security Attributes Affected	x Confidentiality x Integrity _ Availability
Survivability Attributes Affected	x Resist _ Evolve _ Recognize _ Recover
Related IBIS Map Descriptors	IMD-01, IMD-06, IMD-12
Related Security Requirement(s)	SR-2, SR-5

Table 39: IBIS Map Descriptor for Map IM-17

Number	**IMD-17**
Issue	What backup method should the system utilize, if any?
Positions	1. Full Backup Method 2. Incremental Backup Method 3. Differential Backup Method
Security Attributes Affected	_ Confidentiality _ Integrity x Availability
Survivability Attributes Affected	_ Resist _ Evolve _ Recognize x Recover
Related IBIS Map Descriptors	IMD-02, IMD-08, IMD-11
Related Security Requirement(s)	SR-7, SR-8, SR-9

Table 40: IBIS Map Descriptor for Map IM-18

Number	**IMD-18**
Issue	What measure should be in place for ongoing maintenance to prevent configuration error?
Positions	1. Acme 2. Manual 3. Training Course 4. Help Desk
Security Attributes Affected	_ Confidentiality x Integrity x Availability
Survivability Attributes Affected	x Resist x Evolve _ Recognize _ Recover
Related IBIS Map Descriptors	IMD-09
Related Security Requirement(s)	SR-4

Appendix F Acme Group Security Requirements

Security Requirement: SR-1

The system shall implement access control via a secure login screen.

We believe that the decision to implement Windows logon as the primary source of authentication exposes Acme Group to significant vulnerabilities. It is imperative that the login screen be "hardened." The username/password combination is the primary means of defense for the system; if it is exploited, very little can be done to prevent malicious activity.

Security Requirement SR-1 Properties

Category	Resistance
Priority	Medium
Related Security Goal[10]	SG-01
Related IBIS Map(s)	IM-04, IM-05, IM-06, IM-09, IM-11, IM-12, IM-14
Architectural Recommendation(s)	Use SSL/TLS to encrypt a login Web site for the clients. The OpenSSL toolkit, available free from http://www.openssl.org, can be used to generate the necessary public/private key pairs. Add the login site as a "Trusted" site in Internet Explorer (IE). Block all non-SSL communication between IE browsers and IIS servers.
Policy Recommendation(s)	None

[10] Security goals are developed in Step 2 of the SQUARE process. The focus of the project this report describes was the analysis of requirements-elicitation techniques for determining security requirements.

Security Requirement SR-2

The system shall identify and authenticate all the users who attempt to access it.

Strong authentication is a primary defense of the asset management system and best practices are strongly recommended when requiring users to authenticate to the system. Strong requirements should be in place for verifying that the user has selected a password that adheres to the password policy set forth by administrators. Periodic testing should be conducted to ensure that passwords cannot guessed easily by cracking tools.

Security Requirement SR-2 Properties

Category	Resistance
Priority	High
Related Security Goal	SG-01
Related IBIS Map(s)	IM-01, IM-04, IM-05, IM-06, IM-07, IM-09, IM-11, IM-12, IM-14, IM-16
Architectural Recommendation(s)	Store the MD5 hash of user passwords. Never store plaintext passwords.
Policy Recommendation(s)	Require passwords with at least eight characters. When setting up initial password, provide user with tips on choosing a strong password (see http://www.us-cert.gov/cas/tips/ST04-002.html) Force password changes periodically.

Security Requirement SR-3

The server-side components and files contained therein shall have their access restricted to authorized personnel.

Proper management of access control for legitimate users and malicious users is of the utmost importance for the security of the asset management system. The threat is not limited to outside malicious users but also legitimate users engaged in illegitimate activity. Managers of the system must not limit their defensive strategies to the "fortress" model of security.

Security Requirement SR-3 Properties

Category	Resistance
Priority	High
Related Security Goal(s)	SG-01
Related IBIS Map(s)	IM-01, IM-04, IM-05, IM-06, IM-07, IM-09, IM-12, IM-13
Architectural Recommendation(s)	Install Host Intrusion Detection System to identify and isolate attacks by watching audit data. Install Network Intrusion Detection System to identify and isolate attacks by watching network traffic logs.
Policy Recommendation(s)	Develop security policies, procedures, and guidelines. The policies must reflect the organizational culture regarding the protection of the system resources, including all data processed and stored. Procedures should include how to purchase, process, store, and dispose of media and equipment and should apply to all internal and external personnel whether they are employees, partners, contractors, or suppliers. Implement physical access controls such as badges, guards, and locks. Hold access-control testing to ensure that all access controls and mechanisms are working properly and meet the security policy, goals, and objectives.

Security Requirement SR-4

Fault tolerance shall be provided for the asset management system's essential services (IIS server, GIS server, and network lines).

Security Requirement SR-4 Properties

Category	Resistance
Priority	Low
Related Security Goal(s)	SG-03
Related IBIS Map(s)	IM-03, IM-06, IM-10, IM-11, IM-18
Architectural Recommendation(s)	Use Windows fault-tolerance options—Redundant Array of Independent Disks (RAID).
Policy Recommendation(s)	Prepare an emergency plan. Enforce periodical emergency plan drill.

Security Requirement SR-5

The system shall maintain data integrity via logged modifications and user access control.

Data integrity is crucial to the mission of the asset management system, an Internet-based application utilizing username/password for authentication. The system's data integrity can be maintained well through monitoring activity for suspicious actions and strong controls on user activity.

Security Requirement SR-5 Properties

Category	Recognition
Priority	High
Related Security Goal(s)	SG-01, SG-02
Related IBIS Map(s)	IM-02, IM-04, IM-05, IM-07, IM-12, IM-13, IM-14, IM-15, IM-16
Architectural Recommendation(s)	Implement security audit mechanisms. Define the user privileges based on their tasks or roles in the organization. Define adequate clipping level, which is a baseline of normal user mistake activity or expected fat finger behavior on the system.
Policy Recommendation(s)	Monitor the audit logs periodically. Assign a specific person to be in charge of the audit logs. Report any major or outstanding modification of the system. Provide security awareness and training to inform users about their roles in protecting the information system and their responsibility regarding violating the security policies.

Security Requirement SR-6

An access control system shall be configured for optimal information gathering for auditing purposes (access log and application log).

The security of the system cannot be maintained without accurate data on the transactions executed on the system. The asset management system must be deployed in a manner that allows security personnel to accurately monitor the activity of the system and ensures security personnel can take the appropriate defensive measures.

Security Requirement SR-6 Properties

Category	Recognition
Priority	High
Related Security Goal	SG-01, SG-02
Related IBIS Map(s)	IM-04, IM-05, IM-07, IM-12, IM-14, IM-15
Architectural Recommendation(s)	The access-control system shall record the following events: • Internet connection event data • User name • Host name • Source and destination Internet Protocol addresses • Source and destination port numbers • Timestamp • System-level event data • Logon attempts (successful and unsuccessful) • Logon ID • Data and time of each logon and logoff • Devices used • Functions performed • Application-level event data • Data files opened/closed • Specific actions (read, edit, delete, and print) • Record modification • Denied access • User-level event data • User-initiated commands • Identification/authentication attempts • Files and resources accessed
Policy Recommendation(s)	Implement awareness training to help end users understand their role in protecting the information system resources, and let the users know that the system is being monitored. Put in place control mechanisms that allow only authorized system administrators or security officers to read, print, or delete the audit log files. Protect the storage location of audit logs.

Security Requirement SR-7

The system shall recover from attacks, failures, and accidents in less than one minute.

Given the critical service provided by the application, availability is of the utmost importance. In the event of an actual emergency, downtime is not acceptable. Resilience and redundancy must be guaranteed given the results of a system failure.

Security Requirement SR-7 Properties

Category	Recovery
Priority	Low
Related Security Goal	SG-03
Related IBIS Map(s)	IM-03, IM-08, IM-17
Architectural Recommendation(s)	Hot site redundancy Portability to varied production servers
Policy Recommendation(s)	Synchronize with hot site every two to five minutes

Security Requirement SR-8

A backup shall consist of a complete reproduction of every file on the server.

The asset management system must be able to fully recover should a system failure occur. We recommend that every file be copied when creating a backup of the system. When the recovery process begins from the backup files it will be important to have every file available for forensic investigation after system failures and to provide full functionality after system recovery.

Security Requirement SR-8 Properties

Category	Recovery
Priority	Medium
Related Security Goal(s)	SG-03
Related IBIS Map(s)	IM-03, IM-08, IM-17
Architectural Recommendation(s)	Windows Server 2003 Backup Utility Third-party backup utility
Policy Recommendation(s)	Perform daily backups. Maintain recent copies of backups on-site and at an off-site location. Store older backups off-site. Maintain backups in an area with adequate protection against natural disasters (e.g., flood or fire).

Security Requirement SR-9

The system shall be able to provide full functionality from backup.

In the event of a failure, the asset management system must recover and be able to fulfill its mission. The essential services that it provides to the community must be quickly and fully restored in a very short time frame. Given the serious consequences of failure during a disaster, we strongly suggest that Acme Group ensure that full service will be provided from stored backups.

Security Requirement SR-9 Properties

Category	Recovery
Priority	Medium
Related Security Goal(s)	SG-03
Related IBIS Map(s)	IM-03, IM-08, IM-17
Architectural Recommendation(s)	Maintain cold/warm/hot site
Policy Recommendation(s)	Maintain backups in an area with adequate protection against natural disasters (e.g., flood or fire). Store older backups off-site. Maintain copies of installation software with backups.

Appendix G ARM Preparation Memorandum[11]

The Goals

- The primary goal of this project is to build a new business system for the Beta customer team designed to facilitate the work to capture, retain, analyze, and extract structured information.
- A secondary goal for the project is to take advantage of more formal engineering processes to build the information systems necessary to support this change in a core business function.

The Objectives

- Design the framework needed to support detailed analysis of the attributes that the Beta team needs to support existing and future work.
- Capture the requirements needed and inherent risks to engineer a change to both the business processes and tools the team needs to support this work.
- Map the process requirements and risks to existing information systems.
- Use the unmapped requirements and risks to form the basis of requirements and risks for the new systems.
- Survey new and existing technologies that may be used to support the new business processes by modifying existing technologies or utilizing new technologies.
- Architect, design, and build the new business processes.
- Architect, design, and build the new business systems.
- Integrate the new processes and systems into core work.

The Project Success Criteria

We will evaluate the success of this session using the project success criteria (PSC). This will allow you to define exactly how the project will be a business success in terms of the functional capabilities of the system. The following is the list of PSC that you have provided.

- framework established and well understood
- requirements and risks enumerated

[11] In this appendix, the content is presented as it was used with the Beta customer, expect for some adjustment for the layout of this report.

- new business processes
- new business systems
- integrated business processes and tools
- processed and tools established to sustain team growth and adaptation

The Scope

There are two kinds of scope: "in" and "out." "In" scope items are the subjects that will be discussed during the Session Phase. On the other hand, "out" of scope items are the subjects that will not be discussed during the Session Phase.

Table 41: In and Out Scope Items for the ARM Session Phase

Preliminary Session Scope Statements	
In Scope	**Out of Scope**
Universal Scope – Applies to All Applications	
• Business goals • Stakeholders • Training • New business processes • Basic functionality for new business systems	• Human resources/recruiting • Current sponsor relationships • Specific technology solutions • Existing work requirements (e.g., alerts) • Beta customer sensitive information
Stakeholders	
• Beta Team • Other Beta teams • Beta's parent program • Beta's parent organization • Vendors • Public	• U.S. Government • International teams
Inputs and Outputs	
• Sources of information • Products • Interfaces	• Things that "compete" with existing sponsor products • Classified information • Intelligence

The Partitions

Partitioning is a means to break a large problem into several small parts.

- framework development for data analysis
- inventory of existing business processes and systems
- development of processes and tools for data collection
- development of processes and tools for data analysis
- development of processes and tools for data and analysis exchange

- training and mentoring
- deployment
- project management

The Participants

Please write down the participants of this project here:

- Beta team leader
- Beta team analysts
- Beta team developers
- Beta architecture group
- Beta parent program leadership
- Beta team

The Environmental Aspects

Environmental and logistic aspects are the essential elements to support the project. Examples of these aspects include room selection and arrangement, enabling technology arrangement, refreshments, and the like.

- meeting rooms
- projectors
- flip charts
- speaker phone

Appendix H ARM Instructions[12]

Session Phase

Preparation

Frank: Set up the projector.
Don and Lydia: Send out the document package.

Focus Question

Frank: Introduce the focus question.

Candidate Security Requirements

Frank: Provide instruction to the candidate security requirements.
Participant: Write down candidate security requirements.
Don: Keep time.

Top Three Security Requirements

Frank: Provide instruction regarding the top three security requirements.
Participant: Choose the three most important security requirements.
Don: Keep time.

Cardstorming

Frank: Provide instruction regarding the cardstorm.
Lydia: Collect 5 x 8 inch cards.
Don: Keep time.
Don: Read the cards aloud after gathering all of them.
Lydia: Place the cards on the wall.
Eric: Write down the result using ARM_Session_Phase_Requirement_Form.doc.

[12] In this appendix, the content is provided as it was prepared for use during the project, except for some adjustment for the layout of this report.

Reflecting

Frank: Provide instruction regarding the reflecting activity.
Participants: Reflect on what they thought about the cards.

Organizing

Frank: Provide instruction regarding the organizing activity.
Participants: Group the cards.
Eric: Write down the result using ARM_Session_Phase_Grouping_Form.doc.

Naming

Frank: Provide instruction regarding the naming activity.
Participant: Create a unique name for each group of cards
Eric: Write down the result using ARM_Session_Phase_Grouping_Form.doc

Detail

Frank: Provide instruction regarding the details.
Participants: Answer those questions.
Eric: Write down the result using ARM_Session_Phase_Detail_Form.doc.

Prioritization

Frank: Provide instruction regarding the prioritization.
Eric: Show the security requirements with number using the ARM_Prioritization_Form.doc.
Don and Lydia: Collect the prioritization forms.

Feedback

Frank: Provide instruction to the feedback forms.
Don and Lydia: Collect the feedback forms.

Security Requirements Form

No.	Security Requirement
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	

Grouping Form

Group 1:	**Group 2:**
Group 3:	**Group 4:**
Group 5:	**Group 6:**

Detail Form

No.	
Requirement	
Benefit Points	
Proof Points	
Special Constraints	
Assumptions	
Issues	
Action Items	
Notes	
Comments	

Prioritization Form

Requirement	Vote (A, B, C)
Requirement 1	
Requirement 2	
Requirement 3	
Requirement 4	
Requirement 5	
Requirement 6	
Requirement 7	
Requirement 8	
Requirement 9	
Requirement 10	

Appendix I Feedback from the JAD Project Definition Phase and Research Phase[13]

Purpose

The purpose is usually expressed through sentences describing the entire intent of this project. It tells why the system is being designed.

Table 42: Example Purpose Statement

The company faces increased demands for catering services at company functions. Senior management has determined that a roving robot system can fulfill these culinary needs in a cost-effective manner. The Robot will prepare and serve meals for breakfast and luncheon meetings as well as cater larger social functions, usually held in the evening [Wood 89].

Please write down your purpose of this project here:

Good software engineering depends on all aspects of the software engineering life cycle. A framework that supports a long-term integrated, holistic software engineering perspective based on life-cycle roles, activities, and common problems is essential to decision making for the direction and content of the Delta Web site.

The initial content development effort for the Web site is focused on helping our audience to understand and reduce software vulnerabilities. Any framework should help identify content needed to substantially improve individual components and to leverage those improved components into increased assurance for the system as a whole. Ellison and colleagues describe this framework as being composed of two complementary approaches:

1. increasing assurance through better engineering
2. increasing assurance by measurement [Ellison 04]

Even when techniques for improving software are well-known academically, putting that knowledge into practice in the community remains difficult. Reasons include

- failure to make the practicing community aware of new techniques
- poor usability or documentation of the new techniques

[13] In this appendix, the content is provided as it was developed during the project, except for some adjustment for the layout of this report.

- inappropriate or missing motivational justifications for using new techniques
- insufficient risk-related historical data to make cost-benefit tradeoffs regarding new techniques

In part, this Web site is intended to counter these forces.

Scope

The scope tells who will use the system and how often. Table 43 shows an example of a scope statement.

Table 43: Example Scope Statements

Scope
Ninety percent of the robot's services will be used by the following departments: Communications, Sales and marketing.
The expected volume of requests for catering can be estimated based on last year's volume, which totaled:
55 breakfast meetings averaging 15 people per meeting
80 luncheon meetings averaging 20 people per meeting [Wood 89]

Please write down the scope of this project here:

- **requirements of appearance**
  There are certain requirements that are practically derived from the content of the Web site itself.
- **quality requirements**
  Given the nature of the content of the Web site, a specific list of quality requirements is essential. These requirements are expected to cover security, performance, scalability, maintainability, and other software engineering "-ilities."
- **functional (end user) requirements**
  This set of requirements meets the specific business objectives of the Delta project.
- **constraints**
  These "requirements" are largely constraints on the project that are beyond control.

Management Objectives

These objectives tell what management expects to gain from the system. Table 44 shows an example of management objectives.

Table 44: Example of Management Objectives

There are four objectives:
Reduce by 20% the cost of having meetings catered by outside food services.
Increase reliability. The company has had a long history of bad luck with local caterers. With our own robot, we will have more control over the kind of service provided.
Decrease by 50% the corporate services work load of having to process meal requests, contact caterers, and call out for pizza when food does not arrive.
Increase the quality of food served at company functions. Satisfied, well-fed clients translate into more business for the company [Wood 89].

Please write down the management objectives of this project here:

- N/A

Security Objectives

Please write down the security objectives of this project here:

- integrity
- privacy
- availability

Functions

Functions tell what the system will do while objectives tell what management will gain from these functions. Table 45 shows some examples of functions.

Table 45: Example Functions

The functions of a roving robot system will be listed as:
Plan meals
Print reports
Prepare meals
Serve meals
Clean up [Wood 89]

Functional Requirements

This section presents the functional requirements for the Web site. These requirements are divided into two subsets: (1) document life-cycle requirements and (2) actor-based requirements.

Document Life-Cycle Requirements

Figure 18 presents a Unified Modeling language (UML) state diagram for the complete life cycle for "documents." The various life-cycle phases for "documents" are as follows:

- **null**
  This is the state of nonexistence, either because the document has not yet been created or because it has already been deleted.
- **editable**
  In this state, authors may make changes to the document. This is the only state in which its content may be changed.
- **frozen**
  This state represents a historical record of the document contents. Every time a document enters this state, its content is archived in a version control system and may be recalled (e.g., checked out) at any later date.
- **reviewable**
  In this state, *reviewers* and *advisors* may comment on the contents. Comments are "attached" to the document but do not become part of its content. Documents with comments can only be *rejected* back to an *author*. Documents without comments may be *approved* for publication.
- **publishable**
  This is a transitory state. It is expected that a document in this state will be *published* by an automatic process, probably with some degree of configurability (with respect to scheduling).
- **published**
  This is the only state in which the *public* can see the document.

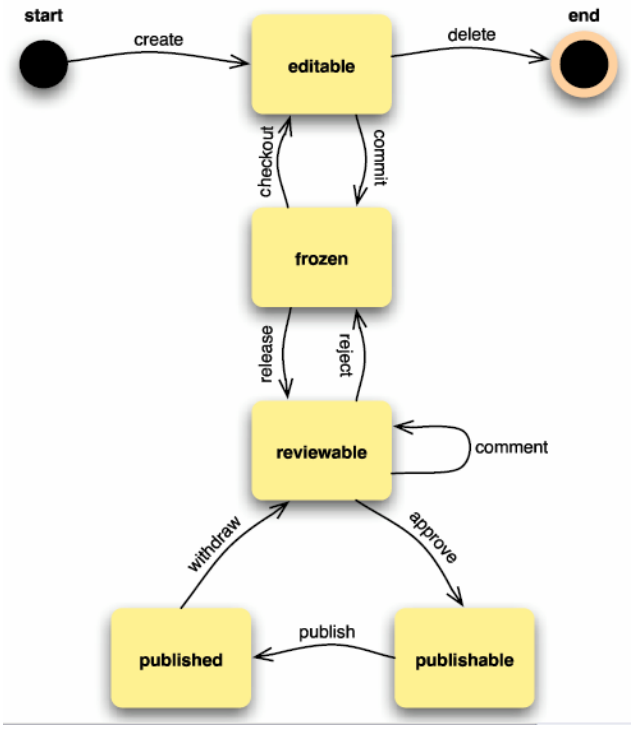


Figure 18: UML State Diagram of "Documents"

In addition, a number of nonworkflow requirements relate to the document life cycle.

- All previous revisions of documents that enter the **frozen** state must be retrievable.
- Documents must be able to be accessed through multiple names (i.e., URLs).
- Documents can be cloned (i.e., checked out from the **frozen** state into a new name).
- Hyperlinks to external documents or other internal documents must be supported.
- Documents in any arbitrary format must be supported, at least for all nonediting operations.
- Each document must support an arbitrary number of properties (i.e., attribute-value pairs).
- Documents must be able to be grouped for the purposes of state transitions (i.e., a group of documents **frozen**, **reviewed**, and **published** *en masse*).
- Documents must be able to be edited by external tools via some acceptable protocol.
- Documents must be able to be exported *en masse* into some acceptable portable external format.
- Document form and content must be separated to the extent that authorization to modify either must be independent.
- Document activity or tracking reports must be available to track the state documents and document groups.
- Documents must be checked out exclusively to one principal at a time.

- Document state changes must be notified through a publish/subscribe mechanism.
- All documents must be accurately attributed, including the ability to delegate attribution.

Actor-Based Requirements (Use Cases)

Figure 19 provides a simple representation of the general categories of actors involved with the system. We will present requirements from the viewpoint of each class of actor, beginning with **Public** (at the extreme right of Figure 19) because it requires all functions; we will then describe the roles as they appear from left to right in the figure. With the exception of work-flow manager, which is not involved in this particular system, we briefly describe each of the roles.

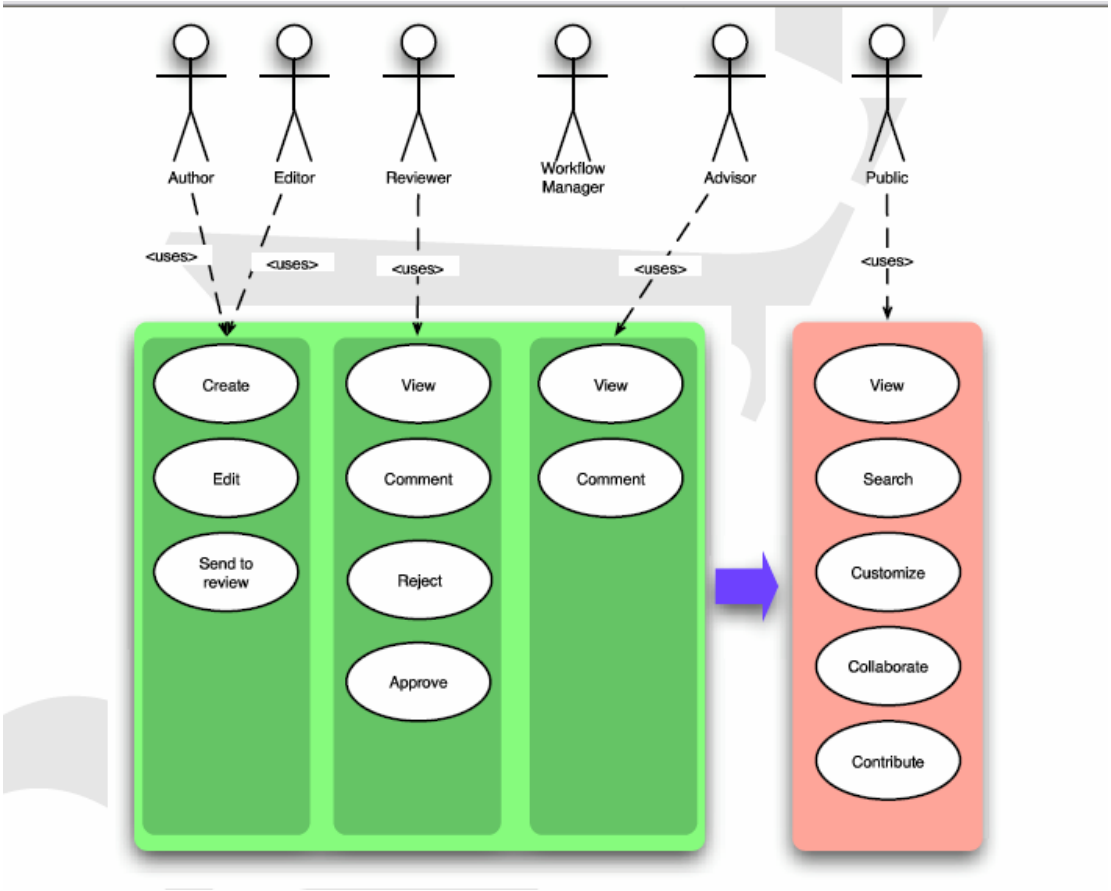


Figure 19: Actors Involved in System

Public

Collaboration Support

Because we expect increasing dependence on collaboration over time, we will need

- a user-to-user communication mechanism (e.g., forums)
- a knowledge acquisition tool (e.g., wiki)

- a mechanism to allow the users to comment (i.e., send feedback) on the content
- a mechanism to allow the users to contribute their experiences with respect to the content

Portal Capabilities
Each user should be able to register and maintain a profile on the site. User profiles should support users customizing their view of the content. This is not about "skins." The basic idea is that the site will likely have a lot of content that may or may not be relevant to any particular user. Users should therefore be able to choose which parts are relevant to maximize the utility of the site. This capability implies that content will have sufficient metadata (e.g., properties). JSR-168 compliance is required for portlets; JSR-170 compliance is required (when finalized) for the content repository.[14]

Label-Based Dynamic Navigation
Dynamic navigation should be based on a meta-data "labeling" mechanism similar to that described by Seacord and Householder [Seacord 05].

Author and Editor

The distinction between **author** and **editor** is subtle. In large part, both interact with the system in the same ways. The principal difference is technical content authority versus document organization and technical writing authority. The **author** defines the technical content of documents; the **editor** organizes content for optimal presentation in a Web environment.

Reviewer and Advisor

The distinction between **reviewer** and **advisor** is also subtle. They both are responsible for reviewing and commenting on content. Comments applied by either class of actor prevent the document from progressing further in the workflow and require it to be returned to the author/editor to address the comments.

Content Management System

This section lists specific requirements of the content management system, without regard to any particular actor. The requirements are as follows:

- content deployment onto multiple sites
- content technicalities
  Multiple rich site summary (RSS) feeds (RSS 1.0, preferably) should be supported for syndication of content. User profiles should be able to support the content seen by visitors to the site. Email notification of new content should be a user-configurable option. Content should be presented to browsers as valid XHTML. Content should be available as XML.

[14] For more information, go to http://www.jcp.org/en/jsr/detail?id=168 and http://www.jcp.org/en/jsr/detail?id=170, respectively.

Constraints

This section describes security-related limitations to consider when designing the system. Table 46 shows some examples of constraints.

Table 46: Example of Constraints

Constraints are
The robot must be ready for use by Dec 10th of next year to serve the annual Christmas party.
The robot must perform all food preparation in the 800 square foot space now occupied by the copy center [Wood 89].

Schedule

The first revision of the Web site must be in place on June 15, 2005. New revisions with additional functionality must be released every three months until the requirements are met (or it is determined they cannot be met). Determination of what goes into each release will be the collaborative result of the project team and its contractors.

Technology

Environmental Compatibility

The underlying technology will be owned and operated by the Delta Team. As such, it must be *compatible* with its surrounding environment. Specific requirements in that regard will be supplied by that team. Examples of compatibility specifications include:

- server hardware and software platforms
- client hardware and software platforms
- browsers
- Java software development kit (SDK) Version

Persistent Storage

All persistent internal representation of content must be stored either in a database or in File-system in extensible markup language (XML) specified with published XML schema definition (XSD) schema or document type definition (DTD). Any database must be a “standard” relational database. Either of the following is acceptable:

- If an open source database is used, it must be PostgreSQL.
- If a commercial database is used, it must be Oracle.

There appears to be no compelling reason to prefer one database over the other. The size of the content is not likely to require an Oracle-class database anytime soon, so PostgreSQL is a

reasonable alternative. In any case, the system must not depend on features of either database that are not in the other's repertoire.

Architectural Framework

The Web site software must be implemented in strict Java 2 Enterprise Edition (J2EE) framework. It must comply with the published framework in all cases where an exception has not been explicitly granted. Use of technologies not within the J2EE specification is allowed—with permission—where they meet all of these technological requirements, especially licensing and documentation.[15]

Licensing and Ownership

The entire system must be open source. Exceptions will be made for "interchangeable" technologies where an equivalent open source product exists. This means the following:

- Existing software assimilated into this application should already be open source.
- Software written for this application should be capable of being open sourced.

Proprietary code or code incapable of being open source licensed may be incorporated with permission and sufficient justification, but that code must be clearly identified and isolated to the extent possible.

Development

Process

No particular process is mandated, but one must be selected and used. Undisciplined engineering is oxymoronic and will not be allowed.

Documentation

The following artifacts are required:

- requirements specification
- architectural description
- security considerations
- design and implementation documentation (This will largely be covered by Javadoc API documents for all public and protected features of every class and interface in every package.)
- deployment documentation
- operational documentation, especially backup and restore procedures

[15] For more information, go to http://java.sun.com/javaee/index.jsp.

Documents must be updated concurrently with new releases. Documentation for the initial June 15, 2005 release may lag behind the release.

Development Tools

All tooling must be specified. Proprietary tools are acceptable. Typical tools include

- Eclipse and a particular array of plugins
- Netbeans

Geography

Development may be done at any geographic location—with permission. A single version control repository must be used. That repository must be on the Delta network and it will use Subversion.[16] Version-control workflow procedures must be written.

Assumptions

Assumptions are basic business decisions about the security of the system that you have agreed upon and must be kept in mind during the design process.

Table 47: Examples of Assumptions

Assumptions about the roving robot system are
The corporate services department will manage all services relating to the robot.
Robot services will be charged back to the departments using them [Wood 89].

Please write down the assumptions of this project here:

This section presents some basic assumptions regarding the establishment of the Delta Web site that are likely to impact the requirements for that site.

- Modification and augmentation will be required. The success of this project depends on our ability to modify and augment the software that implements the site throughout its life. Unaugmented off-the-shelf software will be insufficient to meet our needs.
- High return on investment is necessary. Our teams are small. We cannot afford unjustified experimentation. We cannot afford reimplementation. We must profit as much as possible from others' experience and investments.
- Technology is independent of programmatic concerns. Programmatically, we need to be knowledgeable in multiple platforms, environments, frameworks, architectures, languages, etc. However, this need cannot be allowed to interfere with the design and implementation of this business system.

[16] For more information, go to http://www.tigris.org/.

Implementation Responsibilities

Foundational Technology

Under all currently envisioned scenarios, the installation and ongoing operation of the hardware and software (i.e., platform) on which the Web site will operate shall be the responsibility of the Delta Team. The precise composition and configuration of the software shall be specified by some subset of the following potential participants:

- designated members of the development team
- designated members of the Delta customer team
- contractor(s) designated by the development team

Content Style and Navigation Criteria

The precise style and content-independent navigation within the site shall be specified by some subset of the following potential participants:

- designated members of the development team
- designated members of the Delta team
- contractor(s) designated by the development team

Content Authoring

The content to be published through the site shall be specified by some subset of the following potential participants:

- designated members of the development team
- collaborators designated by the development team

Content Management

The content that actually appears in the Web site will be a transformation of the output from the activities involved in the requirements elicitation. Content management will be the responsibility of

- designated members of the development team
- contractor(s) designated by the development team

Open Issues

Open issues are unresolved questions about the system's security design.

Table 48: Examples of Open Issues

Open issues are
How will we handle the conflicting food and beverage preferences among departments? For example, there are several opposing views on what should be included on the wine list.
How will we accommodate special requests, such as for vegetarian and kosher foods [Wood 89]?

Please write down the open issues of this project here:

The following are items largely relate to the selection of technology and a vendor to help.

- Please provide a detailed description of your proposed software and system architecture.
- Please provide a detailed description of your proposed high-availability solution in the Web-tier.
- Please provide a detailed description of your proposed high-availability solution in the database-tier.
- Please discuss the security model and security configuration/implementation details of the proposed software architecture.
- Please discuss the software development life-cycle methodology you would use in the proposed development project.
- Please discuss your approach to change control in the proposed development project.
- Please discuss your approach to functional and nonfunctional requirements analysis for the proposed development project.
- What is your approach to version control?
- What is your approach to configuration management?
- What is your approach to defect and issue management?
- What is the testing methodology to be used in the proposed project?
- Please provide details with respect to unit testing, integration testing, subsystem testing, system testing, regression testing, load testing, and static and dynamic security testing. Do you maintain a separate environment for testing, or is testing performed on development servers?
- What is your approach to release management? Do you maintain a separate environment for staging, or is staging performed from development or quality assurance (QA) servers?
- Please specify what artifacts and additional documentation you provide to your clients during the development of your projects and as final deliverables.
- Please discuss your approach to the in-line documentation of source code for the proposed project.

- Please discuss how you perform patch management during the development process. Do you specifically evaluate patches to operating systems and applications prior to applying them?
- Please discuss your approach to project management for the proposed effort.

Participants

This section lists all the participants who will either be attending the session or be on call during the session.

Please write down the participants in each of the following areas here:

- acquisition
- mission definition
- concept of operations definition
- project planning
- requirements engineering
- architecture
- design
- implementation
- testing and quality assurance
- integration
- deployment
- evolution
- operations
- software process improvement
- project management
- configuration management
- systems administration

Workflow

A data flow diagram (DFD) is a graphic representation of the "flow" of data through business functions or processes. A data flow diagram contains four kinds of information:

1. **data flow**
   Data flow is the information that moves through the system. For example, "order," "acknowledgement," and "order information" are data flows.

2. **process**
   Process is what causes data to change. For example, “E-Commerce” and “CyberCheck” are processes.
3. **data store**
   Data store is a repository for data. For example, “Customer Database” and “Inventory” are data stores.
4. **external entities**
   An external entity is anything the system interacts with that is not actually part of that system. For example, “Customer” and “Credit Card Company” are external entities.

Please draw the workflow here:

The workflow of the project is still under development.

Appendix J AHP Instruction and Prioritization Matrix[17]

Summary

We are in Step 8 of the SQUARE process—Prioritizing Requirements. Our goal is to prioritize the security requirements collected from Step 6 and categorized in Step 7.

In 2004, a student team applied the numeral assignment technique to prioritize the security requirements. However, our team found that the numeral assignment technique is too subjective and lacks any scientific verification. For these reasons, we are implementing another well-known technique, AHP, for this year's project.

Analytic Hierarchy Process (AHP)

AHP is a method for decision making in situations where multiple objectives are present. This method uses a pair-wise comparison matrix to determine the relative value and cost between security requirements. The entry in row i and column j of the matrix, labeled a_{ij}, indicates how much higher (or lower) the value/cost is for Requirement i is than Requirement j. The value/cost is measured on an integer-valued scale from 1 to 9, with each number having the interpretation shown in Table 49 and Table 50.

Table 49: Interpretation of Values in Matrix

Intensity of Value	Interpretation
1	Requirements i and j are of equal value.
3	Requirement i has a slightly higher value than j.
5	Requirement i has a strongly higher value than j.
7	Requirement i has a very strongly higher value than j.
9	Requirement i has an absolutely higher value than j.
2, 4, 6, 8	These are Intermediate scales between two adjacent judgments.
Reciprocals	If Requirement i has a lower value than j

[17] In this appendix, the content is provided as it was prepared for use during the project, except for some adjustment for the layout of this report.

Table 50: Interpretation of Costs in Matrix

Intensity of Value	Interpretation
1	Requirements i and j are of equal cost.
3	Requirement i has a slightly higher cost than j.
5	Requirement i has a strongly higher cost than j.
7	Requirement i has a very strongly higher cost than j.
9	Requirement i has an absolutely higher cost than j.
2, 4, 6, 8	These are intermediate scales between two adjacent judgments.
Reciprocals	If Requirement i has a lower cost than j

Instruction

Table 51 is an example of a prioritization matrix.

Table 51: Example Prioritization Matrix

	Requirement 1	Requirement 2	Requirement 3
Requirement 1	1	7	1/3
Requirement 2	1/7	1	1/4
Requirement 3	3	4	1

If Requirement 1 compares to itself, the requirements are always of equal value and thus a_{11} is 1. If Requirement 1 has a very strongly higher value than Requirement 2, a_{12} is 7. If Requirement 3 has somewhere between a slightly and a strongly higher value than Requirement 2, a_{32} is 4. If Requirement 1 has a slightly lower value than Requirement 3, a_{13} is 1/3.

You will see your prioritization matrix in a separate Excel file. There are two worksheets in the file, one is called “Value” and the other is called “Cost.” Please fill out both of them. You only need to write down the yellow cell value because the other half of the matrix is just the reciprocal of the yellow one. (Figure 20 is a sample blank AHP prioritization matrix; Table 52 includes the list of Acme Group security requirements.)

Now, we can start to build your prioritization matrix. Have fun!

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9
SR-1	1								
SR-2		1							
SR-3			1						
SR-4				1					
SR-5					1				
SR-6						1			
SR-7							1		
SR-8								1	
SR-9									1

Figure 20: AHP Prioritization Matrix

Table 52: Acme Group Security Requirements

No.	Security Requirement
SR-1	The system shall implement access control via a secure login screen.
SR-2	The system shall identify and authenticate all the users who attempt to access it.
SR-3	The server-side components and files contained therein shall have their access restricted to authorized personnel.
SR-4	Fault tolerance shall be provided for the asset management system's essential services (IIS server, GIS server, and network lines).
SR-5	The system shall maintain data integrity via logged modifications and user access control.
SR-6	An access-control system shall be configured for optimal information gathering for auditing purposes (access log and application log).
SR-7	The system shall recover from attacks, failures, and accidents in less than one minute.
SR-8	A backup shall consist of a complete reproduction of every file on the server.
SR-9	The system shall be able to provide full functionality from backup.

Appendix K Acme Group Original AHP Feedback, Prioritization, and Consistency Check

Acme Group Original AHP Feedback

Table 53: Acme Group Participant 1, Original Feedback—Value

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9
SR-1	1	1/7	1/7	1/5	1/3	1/3	1/5	1/3	1/3
SR-2	7	1	1	3	1	1	5	7	7
SR-3	7	1	1	5	1	1	5	3	3
SR-4	5	1/3	1/5	1	1/3	1/3	1	3	3
SR-5	3	1	1	3	1	1	5	5	5
SR-6	3	1	1	3	1	1	5	7	7
SR-7	5	1/5	1/5	1	1/5	1/5	1	3	3
SR-8	3	1/7	1/3	1/3	1/5	1/7	1/3	1	1
SR-9	3	1/7	1/3	1/3	1/5	1/7	1/3	1	1

Table 54: Acme Group Participant 2, Original Feedback—Value

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9
SR-1	1	8	1/5	3	1	2	2	3	1
SR-2	1/8	1	1/5	1/7	1/7	1/7	1/7	1/9	1/9
SR-3	5	5	1	1	2	1	3	1	1
SR-4	1/3	7	1	1	1/2	1/2	3	1/2	1
SR-5	1	7	1/2	2	1	3	3	1	1/3
SR-6	1/2	7	1	2	1/3	1	1/3	1	1
SR-7	1/2	7	1/3	1/3	1/3	3	1	3	2
SR-8	1/3	9	1	2	1	1	1/3	1	1/6
SR-9	1	9	1	1	3	1	1/2	6	1

Table 55: Acme Group Participant 3, Original Feedback—Value

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9
SR-1	1	7	1/7	3	3	3	3	3	1/7
SR-2	1/7	1	1/9	1/7	1/5	1/7	1/7	1/5	1/9
SR-3	7	9	1	3	5	5	5	5	3
SR-4	1/3	7	1/3	1	1	3	5	5	1/5
SR-5	1/3	5	1/5	1	1	3	1/3	1/3	1/9
SR-6	1/3	7	1/5	1/3	1/3	1	1/5	1/7	1/9
SR-7	1/3	7	1/5	1/5	3	5	1	5	3
SR-8	1/3	5	1/5	1/5	3	7	1/5	1	1/7
SR-9	7	9	1/3	5	9	9	1/3	7	1

Table 56: Acme Group Participant 1, Original Feedback—Cost

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9
SR-1	1	1/5	1/3	1/9	1/3	1/3	1/9	1/7	1/7
SR-2	5	1	1/7	1/9	1/5	1/5	1/9	1/9	1/9
SR-3	3	7	1	1/7	1	1	1/7	1/7	1/7
SR-4	9	9	7	1	9	9	1	9	5
SR-5	3	5	1	1/9	1	1	1/9	1/7	1/7
SR-6	3	5	1	1/9	1	1	1/9	1/7	1/7
SR-7	9	9	7	1	9	9	1	7	9
SR-8	7	9	7	1/9	7	7	1/7	1	1
SR-9	7	9	7	1/5	7	7	1/9	1	1

Table 57: Acme Group Participant 2, Original Feedback—Cost

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9
SR-1	1	1	1	1/9	1/3	1/5	1/9	1/5	1/9
SR-2	1	1	1	1/9	1/3	1/5	1/9	1/5	1/9
SR-3	1	1	1	1/9	1/3	1/3	1/9	1/5	1/9
SR-4	9	9	9	1	9	8	1	8	8
SR-5	3	3	3	1/9	1	1/2	1/9	1/5	1/7
SR-6	5	5	3	1/8	2	1	1/9	1/5	1/7
SR-7	9	9	9	1	9	9	1	8	8
SR-8	5	5	5	1/8	5	5	1/8	1	1/7
SR-9	9	9	9	1/8	7	7	1/8	7	1

Table 58: Acme Group Participant 3, Original Feedback—Cost

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9
SR-1	1	3	1/3	1/5	1/3	1/7	1/7	1/3	1/3
SR-2	1/3	1	1/3	1/7	1/5	1/7	1/9	1/5	1/7
SR-3	3	3	1	1/5	1	1	1/5	1/3	1/5
SR-4	5	7	5	1	5	5	1/3	7	1
SR-5	3	5	1	1/5	1	1	1/7	1/7	1/7
SR-6	7	7	1	1/5	1	1	1/7	1/7	1/7
SR-7	7	9	5	3	7	7	1	7	7
SR-8	3	5	3	1/7	7	7	1/7	1	1/5
SR-9	3	7	5	1	7	7	1/7	5	1

Acme Group Original AHP Prioritization

Table 59: Acme Group Participant 1, Original Prioritization—Value

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9	Normalized Matrix									Scores
SR-1	1	1/7	1/7	1/5	1/3	1/3	1/5	1/3	1/3	0.0270	0.0288	0.0274	0.0119	0.0633	0.0647	0.0087	0.0110	0.0110	0.0282
SR-2	7	1	1	3	1	1	5	7	7	0.1892	0.2015	0.1920	0.1779	0.1899	0.1941	0.2187	0.2308	0.2308	0.2027
SR-3	7	1	1	5	1	1	5	3	3	0.1892	0.2015	0.1920	0.2964	0.1899	0.1941	0.2187	0.0989	0.0989	0.1866
SR-4	5	1/3	1/5	1	1/3	1/3	1	3	3	0.1351	0.0672	0.0384	0.0593	0.0633	0.0647	0.0437	0.0989	0.0989	0.0744
SR-5	3	1	1	3	1	1	5	5	5	0.0811	0.2015	0.1920	0.1779	0.1899	0.1941	0.2187	0.1648	0.1648	0.1761
SR-6	3	1	1	3	1	1	5	7	7	0.0811	0.2015	0.1920	0.1779	0.1899	0.1941	0.2187	0.2308	0.2308	0.1907
SR-7	5	1/5	1/5	1	1/5	1/5	1	3	3	0.1351	0.0403	0.0384	0.0593	0.0380	0.0388	0.0437	0.0989	0.0989	0.0657
SR-8	3	1/7	1/3	1/3	1/5	1/7	1/3	1	1	0.0811	0.0288	0.0640	0.0198	0.0380	0.0277	0.0146	0.0330	0.0330	0.0378
SR-9	3	1/7	1/3	1/3	1/5	1/7	1/3	1	1	0.0811	0.0288	0.0640	0.0198	0.0380	0.0277	0.0146	0.0330	0.0330	0.0378

Table 60: Acme Group Participant 2, Original Prioritization—Value

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9	Normalized Matrix									Scores
SR-1	1	8	1/5	3	1	2	2	3	1	0.1021	0.1333	0.0321	0.2405	0.1074	0.1582	0.1503	0.1806	0.1314	0.1373
SR-2	1/8	1	1/5	1/7	1/7	1/7	1/7	1/9	1/9	0.0128	0.0167	0.0321	0.0115	0.0153	0.0113	0.0107	0.0067	0.0146	0.0146
SR-3	5	5	1	1	2	1	3	1	1	0.5106	0.0833	0.1604	0.0802	0.2148	0.0791	0.2254	0.0602	0.1314	0.1717
SR-4	1/3	7	1	1	1/2	1/2	3	1/2	1	0.0340	0.1167	0.1604	0.0802	0.0537	0.0395	0.2254	0.0301	0.1314	0.0968
SR-5	1	7	1/2	2	1	3	3	1	1/3	0.1021	0.1167	0.0802	0.1603	0.1074	0.2373	0.2254	0.0602	0.0438	0.1259
SR-6	1/2	7	1	2	1/3	1	1/3	1	1	0.0511	0.1167	0.1604	0.1603	0.0358	0.0791	0.0250	0.0602	0.1314	0.0911
SR-7	1/2	7	1/3	1/3	1/3	3	1	3	2	0.0511	0.1167	0.0535	0.0267	0.0358	0.2373	0.0751	0.1806	0.2628	0.1155
SR-8	1	8	1/5	3	1	2	2	3	1	0.0340	0.1500	0.1604	0.1603	0.1074	0.0791	0.0250	0.0602	0.0219	0.0887
SR-9	1/8	1	1/5	1/7	1/7	1/7	1/7	1/9	1/9	0.1021	0.1500	0.1604	0.0802	0.3223	0.0791	0.0376	0.3612	0.1314	0.1582

Table 61: Acme Group Participant 3, Original Prioritization—Value

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9	Normalized Matrix									Scores
SR-1	1	7	1/7	3	3	3	3	3	1/7	0.0595	0.1228	0.0525	0.2162	0.1175	0.0830	0.1972	0.1125	0.0183	0.1088
SR-2	1/7	1	1/9	1/7	1/5	1/7	1/7	1/5	1/9	0.0085	0.0175	0.0408	0.0103	0.0078	0.0040	0.0094	0.0075	0.0142	0.0133
SR-3	7	9	1	3	5	5	5	5	3	0.4164	0.1579	0.3676	0.2162	0.1958	0.1383	0.3287	0.1874	0.3837	0.2658
SR-4	1/3	7	1/3	1	1	3	5	5	1/5	0.0198	0.1228	0.1225	0.0721	0.0392	0.0830	0.3287	0.1874	0.0256	0.1112
SR-5	1/3	5	1/5	1	1	3	1/3	1/3	1/9	0.0198	0.0877	0.0735	0.0721	0.0392	0.0830	0.0219	0.0125	0.0142	0.0471
SR-6	1/3	7	1/5	1/3	1/3	1	1/5	1/7	1/9	0.0198	0.1228	0.0735	0.0240	0.0131	0.0277	0.0131	0.0054	0.0142	0.0348
SR-7	1/3	7	1/5	1/5	3	5	1	5	3	0.0198	0.1228	0.0735	0.0144	0.1175	0.1383	0.0657	0.1874	0.3837	0.1248
SR-8	1/3	5	1/5	1/5	3	7	1/5	1	1/7	0.0198	0.0877	0.0735	0.0144	0.1175	0.1937	0.0131	0.0375	0.0183	0.0640
SR-9	7	9	1/3	5	9	9	1/3	7	1	0.4164	0.1579	0.1225	0.3603	0.3525	0.2490	0.0219	0.2624	0.1279	0.2301

Table 62: Acme Group Participant 1, Original Prioritization—Cost

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9	Normalized Matrix									Scores
SR-1	1	1/5	1/3	1/9	1/3	1/3	1/9	1/7	1/7	0.0213	0.0037	0.0106	0.0383	0.0094	0.0094	0.0391	0.0076	0.0086	0.0164
SR-2	5	1	1/7	1/9	1/5	1/5	1/9	1/9	1/9	0.1064	0.0185	0.0045	0.0383	0.0056	0.0056	0.0391	0.0059	0.0067	0.0256
SR-3	3	7	1	1/7	1	1	1/7	1/7	1/7	0.0638	0.1292	0.0318	0.0493	0.0281	0.0281	0.0503	0.0076	0.0086	0.0441
SR-4	9	9	7	1	9	9	1	9	5	0.1915	0.1661	0.2224	0.3450	0.2533	0.2533	0.3520	0.4817	0.2997	0.2850
SR-5	3	5	1	1/9	1	1	1/9	1/7	1/7	0.0638	0.0923	0.0318	0.0383	0.0281	0.0281	0.0391	0.0076	0.0086	0.0375
SR-6	3	5	1	1/9	1	1	1/9	1/7	1/7	0.0638	0.0923	0.0318	0.0383	0.0281	0.0281	0.0391	0.0076	0.0086	0.0375
SR-7	9	9	7	1	9	9	1	7	9	0.1915	0.1661	0.2224	0.3450	0.2533	0.2533	0.3520	0.3747	0.5395	0.2997
SR-8	7	9	7	1/9	7	7	1/7	1	1	0.1489	0.1661	0.2224	0.0383	0.1970	0.1970	0.0503	0.0535	0.0599	0.1259
SR-9	7	9	7	1/5	7	7	1/9	1	1	0.1489	0.1661	0.2224	0.0690	0.1970	0.1970	0.0391	0.0535	0.0599	0.1281

Table 63: Acme Group Participant 2, Original Prioritization—Cost

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9	Normalized Matrix									Scores
SR-1	1	1/5	1/3	1/9	1/3	1/3	1/9	1/7	1/7	0.0233	0.0233	0.0244	0.0394	0.0098	0.0064	0.0396	0.0080	0.0063	0.0200
SR-2	5	1	1/7	1/9	1/5	1/5	1/9	1/9	1/9	0.0233	0.0233	0.0244	0.0394	0.0098	0.0064	0.0396	0.0080	0.0063	0.0200
SR-3	3	7	1	1/7	1	1	1/7	1/7	1/7	0.0233	0.0233	0.0244	0.0394	0.0098	0.0107	0.0396	0.0080	0.0063	0.0205
SR-4	9	9	7	1	9	9	1	9	5	0.2093	0.2093	0.2195	0.3547	0.2647	0.2561	0.3564	0.3200	0.4504	0.2934
SR-5	3	5	1	1/9	1	1	1/9	1/7	1/7	0.0698	0.0698	0.0732	0.0394	0.0294	0.0160	0.0396	0.0080	0.0080	0.0392
SR-6	3	5	1	1/9	1	1	1/9	1/7	1/7	0.1163	0.1163	0.0732	0.0443	0.0588	0.0320	0.0396	0.0080	0.0080	0.0552
SR-7	9	9	7	1	9	9	1	7	9	0.2093	0.2093	0.2195	0.3547	0.2647	0.2882	0.3564	0.3200	0.4504	0.2969
SR-8	7	9	7	1/9	7	7	1/7	1	1	0.1163	0.1163	0.1220	0.0443	0.1471	0.1601	0.0446	0.0400	0.0080	0.0887
SR-9	7	9	7	1/5	7	7	1/9	1	1	0.2093	0.2093	0.2195	0.0443	0.2059	0.2241	0.0446	0.2800	0.0563	0.1659

Table 64: Acme Group Participant 3, Original Prioritization—Cost

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9	Normalized Matrix									Scores
SR-1	1	3	1/3	1/5	1/3	1/7	1/7	1/3	1/3	0.0309	0.0638	0.0154	0.0329	0.0113	0.0049	0.0606	0.0158	0.0328	0.0298
SR-2	1/3	1	1/3	1/7	1/5	1/7	1/9	1/5	1/7	0.0103	0.0213	0.0154	0.0235	0.0068	0.0049	0.0471	0.0095	0.0141	0.0170
SR-3	3	3	1	1/5	1	1	1/5	1/3	1/5	0.0928	0.0638	0.0462	0.0329	0.0339	0.0341	0.0848	0.0158	0.0197	0.0471
SR-4	5	7	5	1	5	5	1/3	7	1	0.1546	0.1489	0.2308	0.1643	0.1693	0.1707	0.1413	0.3309	0.0984	0.1788
SR-5	3	5	1	1/5	1	1	1/7	1/7	1/7	0.0928	0.1064	0.0462	0.0329	0.0339	0.0341	0.0606	0.0068	0.0141	0.0475
SR-6	7	7	1	1/5	1	1	1/7	1/7	1/7	0.2165	0.1489	0.0462	0.0329	0.0339	0.0341	0.0606	0.0068	0.0141	0.0660
SR-7	7	9	5	3	7	7	1	7	7	0.2165	0.1915	0.2308	0.4930	0.2370	0.2390	0.4240	0.3309	0.6888	0.3391
SR-8	3	5	3	1/7	7	7	1/7	1	1/5	0.0928	0.1064	0.1385	0.0235	0.2370	0.2390	0.0606	0.0473	0.0197	0.1072
SR-9	3	7	5	1	7	7	1/7	5	1	0.0928	0.1489	0.2308	0.1643	0.2370	0.2390	0.0606	0.2364	0.0984	0.1676

Acme Group Original AHP Consistency Check

Table 65: Acme Group Participant 1, Original Consistency Check—Value

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9	Product	Ratio
SR-1	1	1/7	1/7	1/5	1/3	1/3	1/5	1/3	1/3	0.2593	9.1943
SR-2	7	1	1	3	1	1	5	7	7	2.0340	10.0321
SR-3	7	1	1	5	1	1	5	3	3	1.8807	10.0778
SR-4	5	1/3	1/5	1	1/3	1/3	1	3	3	0.7348	9.8782
SR-5	3	1	1	3	1	1	5	5	5	1.7701	10.0529
SR-6	3	1	1	3	1	1	5	7	7	1.9212	10.0725
SR-7	5	1/5	1/5	1	1/5	1/5	1	3	3	0.6589	10.0264
SR-8	3	1/7	1/3	1/3	1/5	1/7	1/3	1	1	0.3605	9.5462
SR-9	3	1/7	1/3	1/3	1/5	1/7	1/3	1	1	0.3605	9.5462
										CI	0.1031
										CI/RI	0.0711

Table 66: Acme Group Participant 2, Original Consistency Check—Value

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9	Product	Ratio
SR-1	1	8	1/5	3	1	2	2	3	1	1.5427	11.2344
SR-2	1/8	1	1/5	1/7	1/7	1/7	1/7	1/9	1/9	0.1549	10.5917
SR-3	5	5	1	1	2	1	3	1	1	1.9647	11.4415
SR-4	1/3	7	1	1	1/2	1/2	3	1/2	1	1.0743	11.0955
SR-5	1	7	1/2	2	1	3	3	1	1/3	1.4065	11.1681
SR-6	1/2	7	1	2	1/3	1	1/3	1	1	0.9550	10.4813
SR-7	1/2	7	1/3	1/3	1/3	3	1	3	2	1.2740	11.0301
SR-8	1/3	9	1	2	1	1	1/3	1	1/6	0.9134	10.2961
SR-9	1	9	1	1	3	1	1/2	6	1	1.7547	11.0884
										CI	0.2420
										CI/RI	0.1669

Table 67: Acme Group Participant 3, Original Consistency Check—Value

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9	Product	Ratio
SR-1	1	7	1/7	3	3	3	3	3	1/7	1.4189	13.0375
SR-2	1/7	1	1/9	1/7	1/5	1/7	1/7	1/5	1/9	0.1449	10.8614
SR-3	7	9	1	3	5	5	5	5	3	3.5252	13.2631
SR-4	1/3	7	1/3	1	1	3	5	5	1/5	1.4709	13.2233
SR-5	1/3	5	1/5	1	1	3	1/3	1/3	1/9	0.5075	10.7745
SR-6	1/3	7	1/5	1/3	1/3	1	1/5	1/7	1/9	0.3301	9.4734
SR-7	1/3	7	1/5	1/5	3	5	1	5	3	1.6554	13.2642
SR-8	1/3	5	1/5	1/5	3	7	1/5	1	1/7	0.6854	10.7176
SR-9	7	9	1/3	5	9	9	1/3	7	1	2.9835	12.9664
										CI	0.3692
										CI/RI	0.2546

Table 68: Acme Group Participant 1, Original Consistency Check—Cost

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9	Product	Ratio
SR-1	1	1/5	1/3	1/9	1/3	1/3	1/9	1/7	1/7	0.1625	9.8867
SR-2	5	1	1/7	1/9	1/5	1/5	1/9	1/9	1/9	0.2223	8.6749
SR-3	3	7	1	1/7	1	1	1/7	1/7	1/7	0.4677	10.6082
SR-4	9	9	7	1	9	9	1	9	5	3.7216	13.0586
SR-5	3	5	1	1/9	1	1	1/9	1/7	1/7	0.3979	10.6015
SR-6	3	5	1	1/9	1	1	1/9	1/7	1/7	0.3979	10.6015
SR-7	9	9	7	1	9	9	1	7	9	3.9821	13.2853
SR-8	7	9	7	1/9	7	7	1/7	1	1	1.5084	11.9770
SR-9	7	9	7	1/5	7	7	1/9	1	1	1.5242	11.8979
										CI	0.2721
										CI/RI	0.1877

Table 69: Acme Group Participant 2, Original Consistency Check—Cost

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9	Product	Ratio
SR-1	1	1	1	1/9	1/3	1/5	1/9	1/5	1/9	0.1865	9.3050
SR-2	1	1	1	1/9	1/3	1/5	1/9	1/5	1/9	0.1865	9.3050
SR-3	1	1	1	1/9	1/3	1/3	1/9	1/5	1/9	0.1938	9.4484
SR-4	9	9	9	1	9	8	1	8	8	3.9675	13.5232
SR-5	3	3	3	1/9	1	1/2	1/9	1/5	1/7	0.3557	9.0634
SR-6	5	5	3	1/8	2	1	1/9	1/5	1/7	0.5067	9.1847
SR-7	9	9	9	1	9	9	1	8	8	4.0227	13.5470
SR-8	5	5	5	1/8	5	5	1/8	1	1/7	0.9613	10.8338
SR-9	9	9	9	1/8	7	7	1/8	7	1	2.0671	12.4584
										CI	0.2176
										CI/RI	0.1501

Table 70: Acme Group Participant 3, Original Consistency Check—Cost

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9	Product	Ratio
SR-1	1	3	1/3	1/5	1/3	1/7	1/7	1/3	1/3	0.2975	9.9784
SR-2	1/3	1	1/3	1/7	1/5	1/7	1/9	1/5	1/7	0.1701	10.0261
SR-3	3	3	1	1/5	1	1	1/5	1/3	1/5	0.4737	10.0589
SR-4	5	7	5	1	5	5	1/3	7	1	2.2805	12.7531
SR-5	3	5	1	1/5	1	1	1/7	1/7	1/7	0.4583	9.6471
SR-6	7	7	1	1/5	1	1	1/7	1/7	1/7	0.6115	9.2676
SR-7	7	9	5	3	7	7	1	7	7	4.1901	12.3583
SR-8	3	5	3	1/7	7	7	1/7	1	1/5	1.3247	12.3587
SR-9	3	7	5	1	7	7	1/7	5	1	2.1689	12.9425
										CI	0.2554
										CI/RI	0.1762

Appendix L Acme Group Revised AHP Feedback, Prioritization, and Consistency Check

Acme Group Revised AHP Feedback

Table 71: Acme Group Revised Feedback—Value

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9
SR-1	1	1/6	1/7	1/2	1/4	1/4	1/2	1/3	1/3
SR-2	6	1	1	7	1	1	7	7	7
SR-3	7	1	1	7	1	1	7	7	7
SR-4	2	1/7	1/7	1	1/4	1/4	1	1	1
SR-5	4	1	1	4	1	1	5	1	1
SR-6	4	1	1	4	1	1	5	1	1
SR-7	2	1/7	1/7	1	1/5	1/5	1	5	5
SR-8	3	1/7	1/7	1	1	1	1/5	1	1
SR-9	3	1/7	1/7	1	1	1	1/5	1	1

Table 72: Acme Group Revised Feedback—Cost

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9
SR-1	1	5	1/3	1/9	1/5	1/3	1/9	1/7	1/7
SR-2	1/5	1	1/7	1/9	1/5	1/5	1/9	1/9	1/9
SR-3	3	7	1	1/7	1	1	1/9	1/7	1/7
SR-4	9	9	7	1	9	9	1/5	9	5
SR-5	5	5	1	1/9	1	1	1/9	1/7	1/7
SR-6	3	5	1	1/9	1	1	1/9	1/7	1/7
SR-7	9	9	9	5	9	9	1	7	7
SR-8	7	9	7	1/9	7	7	1/7	1	1
SR-9	7	9	7	1/5	7	7	1/7	1	1

Acme Group Revised AHP Prioritization

Table 73: Acme Group Revised Prioritization—Value

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9	Normalized Matrix									Scores
SR-1	1	1/6	1/7	1/2	1/4	1/4	1/2	1/3	1/3	0.0313	0.0352	0.0303	0.0189	0.0373	0.0373	0.0186	0.0137	0.0137	0.0262
SR-2	6	1	1	7	1	1	7	7	7	0.1875	0.2111	0.2121	0.2642	0.1493	0.1493	0.2602	0.2877	0.2877	0.2232
SR-3	7	1	1	7	1	1	7	7	7	0.2188	0.2111	0.2121	0.2642	0.1493	0.1493	0.2602	0.2877	0.2877	0.2267
SR-4	2	1/7	1/7	1	1/4	1/4	1	1	1	0.0625	0.0302	0.0303	0.0377	0.0373	0.0373	0.0372	0.0411	0.0411	0.0394
SR-5	4	1	1	4	1	1	5	1	1	0.1250	0.2111	0.2121	0.1509	0.1493	0.1493	0.1859	0.0411	0.0411	0.1406
SR-6	4	1	1	4	1	1	5	1	1	0.1250	0.2111	0.2121	0.1509	0.1493	0.1493	0.1859	0.0411	0.0411	0.1406
SR-7	2	1/7	1/7	1	1/5	1/5	1	5	5	0.0625	0.0302	0.0303	0.0377	0.0299	0.0299	0.0372	0.2055	0.2055	0.0743
SR-8	3	1/7	1/7	1	1	1	1/5	1	1	0.0938	0.0302	0.0303	0.0377	0.1493	0.1493	0.0074	0.0411	0.0411	0.0645
SR-9	3	1/7	1/7	1	1	1	1/5	1	1	0.0938	0.0302	0.0303	0.0377	0.1493	0.1493	0.0074	0.0411	0.0411	0.0645

Table 74: Acme Group Revised Prioritization—Cost

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9	Normalized Matrix									Scores
SR-1	1	5	1/3	1/9	1/5	1/3	1/9	1/7	1/7	0.0226	0.0847	0.0100	0.0161	0.0056	0.0094	0.0544	0.0076	0.0097	0.0245
SR-2	1/5	1	1/7	1/9	1/5	1/5	1/9	1/9	1/9	0.0045	0.0169	0.0043	0.0161	0.0056	0.0056	0.0544	0.0059	0.0076	0.0135
SR-3	3	7	1	1/7	1	1	1/9	1/7	1/7	0.0679	0.1186	0.0299	0.0207	0.0282	0.0281	0.0544	0.0076	0.0097	0.0406
SR-4	9	9	7	1	9	9	1/5	9	5	0.2036	0.1525	0.2091	0.1450	0.2542	0.2533	0.0980	0.4817	0.3405	0.2376
SR-5	5	5	1	1/9	1	1	1/9	1/7	1/7	0.1131	0.0847	0.0299	0.0161	0.0282	0.0281	0.0544	0.0076	0.0097	0.0413
SR-6	3	5	1	1/9	1	1	1/9	1/7	1/7	0.0679	0.0847	0.0299	0.0161	0.0282	0.0281	0.0544	0.0076	0.0097	0.0363
SR-7	9	9	9	5	9	9	1	7	7	0.2036	0.1525	0.2688	0.7248	0.2542	0.2533	0.4899	0.3747	0.4768	0.3554
SR-8	7	9	7	1/9	7	7	1/7	1	1	0.1584	0.1525	0.2091	0.0161	0.1977	0.1970	0.0700	0.0535	0.0681	0.1247
SR-9	7	9	7	1/5	7	7	1/7	1	1	0.1584	0.1525	0.2091	0.0290	0.1977	0.1970	0.0700	0.0535	0.0681	0.1262

Acme Group Revised AHP Consistency Check

Table 75: Acme Group Revised Consistency Check—Value

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9	Product	Ratio
SR-1	1	1/6	1/7	1/2	1/4	1/4	1/2	1/3	1/3	0.2659	10.1336
SR-2	6	1	1	7	1	1	7	7	7	2.5867	11.5890
SR-3	7	1	1	7	1	1	7	7	7	2.6130	11.5272
SR-4	2	1/7	1/7	1	1/4	1/4	1	1	1	0.4296	10.9028
SR-5	4	1	1	4	1	1	5	1	1	1.4940	10.6240
SR-6	4	1	1	4	1	1	5	1	1	1.4940	10.6240
SR-7	2	1/7	1/7	1	1/5	1/5	1	5	5	0.9312	12.5366
SR-8	3	1/7	1/7	1	1	1	1/5	1	1	0.6074	9.42463
SR-9	3	1/7	1/7	1	1	1	1/5	1	1	0.6074	9.42463
										CI	0.21926
										CI/RI	0.1512

Table 76: Acme Group Revised Consistency Check —Cost

	SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9	Product	Ratio
SR-1	1	5	1/3	1/9	1/5	1/3	1/9	1/7	1/7	0.2273	9.2896
SR-2	1/5	1	1/7	1/9	1/5	1/5	1/9	1/9	1/9	0.1334	9.9188
SR-3	3	7	1	1/7	1	1	1/9	1/7	1/7	0.3950	9.7341
SR-4	9	9	7	1	9	9	1/5	9	5	3.3861	14.2543
SR-5	5	5	1	1/9	1	1	1/9	1/7	1/7	0.4095	9.9084
SR-6	3	5	1	1/9	1	1	1/9	1/7	1/7	0.3606	9.9322
SR-7	9	9	9	5	9	9	1	7	7	4.7047	13.2377
SR-8	7	9	7	1/9	7	7	1/7	1	1	1.4481	11.6108
SR-9	7	9	7	1/5	7	7	1/7	1	1	1.4692	11.6464
										CI	0.2573
										CI/RI	0.1775

Appendix M Delta AHP Feedback, Prioritization, and Consistency Check

Delta AHP Feedback

Table 77: Delta Feedback—Value

	SR1	SR2	SR3	SR4	SR5	SR6	SR7
SR1	1	1/5	1/5	1/2	1/8	1/8	1/9
SR2	5	1	2	1/2	1/8	1/8	1/9
SR3	5	1/2	1	2	1/8	1/8	1/9
SR4	2	2	1/2	1	1/8	1/8	1/9
SR5	8	8	8	8	1	1/5	1/6
SR6	8	8	8	8	5	1	1/2
SR7	9	9	9	9	6	2	1

Table 78: Delta Feedback—Cost

	SR1	SR2	SR3	SR4	SR5	SR6	SR7
SR1	1	1/5	9	1/4	6	5	7
SR2	5	1	9	1	8	6	5
SR3	1/9	1/9	1	1/8	1/9	2	2
SR4	4	1	8	1	5	6	8
SR5	1/6	1/8	9	1/5	1	2	1
SR6	1/5	1/6	1/2	1/6	1/2	1	2
SR7	1/7	1/5	1/2	1/8	1	1/2	1

Delta AHP Prioritization

Table 79: Delta Prioritization—Value

	SR1	SR2	SR3	SR4	SR5	SR6	SR7	Normalized Matrix							Scores
SR1	1	1/5	1/5	1/2	1/8	1/8	1/9	0.0263	0.0070	0.0070	0.0172	0.0100	0.0338	0.0526	0.0220
SR2	5	1	2	1/2	1/8	1/8	1/9	0.1316	0.0348	0.0697	0.0172	0.0100	0.0338	0.0526	0.0500
SR3	5	1/2	1	2	1/8	1/8	1/9	0.1316	0.0174	0.0348	0.0690	0.0100	0.0338	0.0526	0.0499
SR4	2	2	1/2	1	1/8	1/8	1/9	0.0526	0.0697	0.0174	0.0345	0.0100	0.0338	0.0526	0.0387
SR5	8	8	8	8	1	1/5	1/6	0.2105	0.2787	0.2787	0.2759	0.0800	0.0541	0.0789	0.1796
SR6	8	8	8	8	5	1	1/2	0.2105	0.2787	0.2787	0.2759	0.4000	0.2703	0.2368	0.2787
SR7	9	9	9	9	6	2	1	0.2368	0.3136	0.3136	0.3103	0.4800	0.5405	0.4737	0.3812

Table 80: Delta Prioritization—Cost

	SR1	SR2	SR3	SR4	SR5	SR6	SR7	Normalized Matrix							Scores
SR1	1	1/5	9	1/4	6	5	7	0.0942	0.0714	0.2432	0.0872	0.2776	0.2222	0.2692	0.1807
SR2	5	1	9	1	8	6	5	0.4708	0.3568	0.2432	0.3488	0.3702	0.2667	0.1923	0.3213
SR3	1/9	1/9	1	1/8	1/9	2	2	0.0105	0.0396	0.0270	0.0436	0.0051	0.0889	0.0769	0.0417
SR4	4	1	8	1	5	6	8	0.3766	0.3568	0.2162	0.3488	0.2314	0.2667	0.3077	0.3006
SR5	1/6	1/8	9	1/5	1	2	1	0.0157	0.0446	0.2432	0.0698	0.0463	0.0889	0.0385	0.0781
SR6	1/5	1/6	1/2	1/6	1/2	1	2	0.0188	0.0595	0.0135	0.0581	0.0231	0.0444	0.0769	0.0421
SR7	1/7	1/5	1/2	1/8	1	1/2	1	0.0135	0.0714	0.0135	0.0436	0.0463	0.0222	0.0385	0.0356

Delta AHP Consistency Check

Table 81: Delta Consistency Check—Value

	SR1	SR2	SR3	SR4	SR5	SR6	SR7	Scores	Product	Ratio
SR1	1	1/5	1/5	1/2	1/8	1/8	1/9	0.0220	0.1609	7.3193
SR2	5	1	2	1/2	1/8	1/8	1/9	0.0500	0.3786	7.5781
SR3	5	1/2	1	2	1/8	1/8	1/9	0.0499	0.3617	7.2515
SR4	2	2	1/2	1	1/8	1/8	1/9	0.0387	0.3071	7.9445
SR5	8	8	8	8	1	1/5	1/6	0.1796	1.5828	8.8155
SR6	8	8	8	8	5	1	1/2	0.2787	2.6511	9.5120
SR7	9	9	9	9	6	2	1	0.3812	3.4605	9.0773
									CI	0.2023
									CI/RI	0.1532

Table 82: Delta Consistency Check—Cost

	SR1	SR2	SR3	SR4	SR5	SR6	SR7	Scores	Product	Ratio
SR1	1	1/5	9	1/4	6	5	7	0.1807	1.6232	8.9815
SR2	5	1	9	1	8	6	5	0.3213	2.9557	9.2004
SR3	1/9	1/9	1	1/8	1/9	2	2	0.0417	0.2989	7.1740
SR4	4	1	8	1	5	6	8	0.3006	2.6056	8.6680
SR5	1/6	1/8	9	1/5	1	2	1	0.0781	0.7032	9.0007
SR6	1/5	1/6	1/2	1/6	1/2	1	2	0.0421	0.3129	7.4376
SR7	1/7	1/5	1/2	1/8	1	1/2	1	0.0356	0.2832	7.9651
									CI	0.2245
									CI/RI	0.1700

References

URLs are valid as of the publication date of this document.

[Alvarez 02] Alvarez, R. "Discourse Analysis of Requirements and Knowledge Elicitation Interviews." *Proceedings of the 35th Hawaii International Conference on System Sciences (HICSS-35).* Big Island, HI, January 7–10, 2002. http://csdl.computer.org/comp/proceedings/hicss/2002/1435/08/14350255abs.htm (2002).

[Answers 06] Answers Corporation. *Fagan inspection.* http://www.answers.com/topic/fagan-inspection (2006).

[Anton 01] Anton, A. I.; Dempster, J. H.; & Siege, D. F. "Deriving Goals from a Use Case Based Requirements Specification for an Electronic Commerce System," 10–19. *Proceedings of the Sixth International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ 2000)*. Stockholm, Sweden, June 5–6, 2000. London, England: Springer-London, 2001.

[Boehm 89] Boehm, B. & Ross, R. "Theory-W Software Project Management: Principles and Examples." *IEEE Transactions on Software Engineering 15,* 4 (July 1989): 902–916.

[Brackett 90] Brackett, J. W. *Software Requirements* (SEI-CM-19-1.2, ADA235642). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1990. http://www.sei.cmu.edu/publications/documents/cms/cm.019.html.

[Checkland 89] Checkland, P. *Soft Systems Methodology: Rational Analysis for a Problematic World.* New York, NY: John Wiley & Sons, 1989.

[Checkland 90] Checkland, P. *Soft System Methodology in Action*. Toronto, Ontario, Canada: John Wiley & Sons, 1990.

[Christel 92] Christel, M. & Kang, K. *Issues in Requirements Elicitation* (CMU/SEI-92-TR-012, ADA258932). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1992. http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.012.html.

[Doolan 92] Doolan, E. P. "Experience with Fagan's Inspection Method." *Software – Practice and Experience 22*, 2 (February 1992): 173–182.

[Eickelmann 03] Eickelmann, N. S.; Ruffolo, F.; Baik, J.; & Anant, A. "An Empirical Study of Modifying the Fagan Inspection Process and the Resulting Main Effects and Interaction Effects Among Defects Found, Effort Required, Rate of Preparation and Inspection, Number of Team Members and Product 1st Pass Quality," 58–64. *Proceedings of the 27th Annual NASA Goddard/IEEE Software Engineering Workshop (SEW-27'02).* Greenbelt, Maryland, December 5–6, 2002. Washington, D.C.: IEEE Computer Society, 2003.

[Ellison 04] Ellison, R. J.; Moore, A. P.; Bass, L.; Klein, M.; & Bachmann, F. Security and Survivability Reasoning Frameworks and Architectural Design Tactics (CMU/SEI-2004-TN-022, ADA443487). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004. http://www.sei.cmu.edu/publications/documents/04.reports/04tn022.html.

[Fagan 86] Fagan, M. E. "Advances in Software Inspections." *IEEE Transactions on Software Engineering*, *SE-12* 7 (July 1986): 744–751.

[Fagan 01] Fagan, M. E. "Reviews and Inspections," 562–573. *Pioneers and Their Contributions to Software Engineering: sd&m Conference on Software Pioneers.* Bonn, Germany, June 28–29, 2001. Berlin, Germany: Springer Verlag, 2001.

[Finkelstein 92] Finkelstein, A. "TARA: Tool Assisted Requirements Analysis," 413–432. *Conceptual Modeling, Databases and CASE: An Integrated View of Information Systems Development*. New York, NY: John Wiley & Sons, 1992.

[Jacobson 92] Jacobson, I., et al., *Object-Oriented Software Engineering: A Use Case Driven Approach*. Boston, MA: Addison-Wesley, 1992.

[Kang 90] Kang, K. C.; Cohen, S. G.; Hess, J. A.; Novack, W. E.; & Peterson, A. S. *Feature-Oriented Domain Analysis Feasibility Study* (CMU/SEI-90-TR-021, ADA235785). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1990. http://www.sei.cmu.edu/publications/documents/90.reports/90.tr.021.html.

[Karlsson 95] Karlsson, J. "Towards a Strategy for Software Requirements Selection. Licentiate." Thesis 513, Linköping University, October 1995.

[Karlsson 96] Karlsson, J. "Software Requirements Prioritizing," 110–116. *Proceedings of the Second International Conference on Requirements Engineering (ICRE'96).* Colorado Springs, CO, April 15–18, 1996. Los Alamitos, CA: IEEE Computer Society, 1996.

[Karlsson 97] Karlsson, J. & Ryan, K. "A Cost-Value Approach for Prioritizing Requirements." *IEEE Software 14,* 5 (September/October 1997): 67–74.

[Kean 97] Kean, L. "Feature-Oriented Domain Analysis." *Software Technology Roadmap*. http://www.sei.cmu.edu/str/descriptions/foda_body.html (1997).

[Kuloor 02] Kuloor, C. & Eberlein, A. *Requirements Engineering for Software Product Lines*. http://www2.enel.ucalgary.ca/People/eberlein /publications/ProdLine_ICSSEA2002.pdf (2002).

[Kunz 70] Kunz, W. & Rittel, H. W. J. *Issues as Elements of Information Systems, Working Paper 131*. Institute of Urban & Regional Development, University of California, Berkeley: Berkeley, CA, 1970. http://www-iurd.ced.berkeley.edu/pub/WP-131.pdf.

[Macaulay 96] Macaulay, L. A. *Requirements Engineering*. Berlin, Germany: Springer-Verlag, 1996.

[Mead 04] Mead, N. R.; Chen, P.; Dean, M.; Ojoko-Adams, D.; Osman, H.; Lopez, L.; & Xie, N. *System Quality Requirements Engineering (SQUARE) Methodology: Case Study on Asset Management System* (CMU/SEI-2004-SR-015, ADA431068). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004. http://www.sei.cmu.edu/publications/documents/04.reports /04sr015.html.

[Mullery 79] Mullery, G. P. "CORE: A Method for Controlled Requirements Specification," 126–135. *Proceedings of the 4th International Conference on Software Engineering (ICSE-4)*. Munich, Germany, September 17–19, 1979. Los Alamitos, CA: IEEE Computer Society Press, 1979.

[Park 95a] Park, R. E. *A Manager's Checklist for Validating Software Cost and Schedule Estimates* (CMU/SEI-95-SR-004, ADA293298). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1995. http://www.sei.cmu.edu/publications/documents /95.reports/95.sr.004.html.

[Park 95b] Park, R. E. *Checklists and Criteria for Evaluating the Cost and Schedule Estimating Capabilities of Software Organization* (CMU/SEI-95-SR-005, ADA293299) Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1995. http://www.sei.cmu.edu/publications/documents/95.reports/95.sr.005.html

[Park 99] Park, J.; Port, D.; & Boehm B. "Supporting Distributed Collaborative Prioritization for Win-Win Requirements Capture and Negotiation," 578–584. *Proceedings of the International Third World Multiconference on Systemics, Cybernetics and Informatics (SCI'99) Vol. 2*. Orlando, FL, July 31-August 4, 1999. Orlando, FL: International Institute of Informatics and Systemics (IIIS), 1999.

[Pohl 93] Pohl, K. "The Three Dimensions of Requirements Engineering," 175–292. *Proceedings of the Fifth International Conference on Advanced Information Systems Engineering (CAISE'93)*. Paris, France, June 8–11, 1993. Berlin, Germany: Springer-Verlag, 1993.

[Powell 03] Powell, D. *Software Review and Inspection*. http://www.int.gu.edu.au/courses/7014int/insp-notes.html (2003).

[QFD 05] QFD Institute. *Frequently Asked Questions About QFD*. http://www.qfdi.org/what_is_qfd/faqs_about_qfd.htm (2005).

[Rumbaugh 94] Rumbaugh, J. "Getting Started: Using Use Cases to Capture Requirements." *Journal of Object-Oriented Programming 7*, 5 (September 1994): 8–23.

[Saaty 80] Saaty, T. L. *The Analytic Hierarchy Process*, New York, NY: McGraw-Hill, 1980.

[Schiffrin 94] Schiffrin, D. *Approaches to Discourse*. Oxford, England: Blackwell Publishers Ltd, 1994.

[SDS 86] Systems Designers Scientific. *CORE—The Method: User Manual*. London, England: SD-Scicon, 1986.

[Seacord 05] Seacord, R. & Householder, A. *A Structured Approach to Manual Code Review* (CMU/SEI-2005-TN-003, ADA430968). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2005. http://www.sei.cmu.edu/publications/documents/05.reports/05tn003.html

[SEI 04] Software Engineering Institute. *What is a CASE Environment?* http://www.sei.cmu.edu/legacy/case/case_whatis.html (2004).

[Sindre 00] Sindre, G. & Opdahl, A. L. ”Eliciting Security Requirements by Misuse Cases,” 120–131. *Proceedings of the 37th International Conference on Technology of Object-Oriented Languages (Tools 37–Pacific 2000)*. Sydney, Australia, November 20-23, 2000. Los Alamitos, CA: IEEE Computer Society, 2000.

[Wiegers 05a] Wiegers, K. E. *Seven Truths About Peer Reviews: Process Impact*. http://www.processimpact.com/articles/seven_truths.html (2005).

[Wiegers 05b] Wiegers, K. E. *Improving Quality Through Software Inspections: Process Impact*. http://www.processimpact.com/articles /inspects.html (2005).

[Wilson 90] Wilson, B. *System: Concept, Methodologies and Applications*. New York, NY: John Wiley & Sons, 1990.

[Wood 89] Wood, J. & Silver, D. *Joint Application Design: How to Design Quality Systems in 40% Less Time.* New York, NY: John Wiley & Sons, Inc., 1989 (ISBN 0471504629).

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
(Leave Blank)	May 2006	Final

4. TITLE AND SUBTITLE	5. FUNDING NUMBERS
Security Quality Requirements Engineering (SQUARE): Case Study Phase III	FA8721-05-C-0003

6. AUTHOR(S)

Authors: Lydia Chung; Frank Hung; Eric Hough; & Don Ojoko-Adams
Advisor: Nancy R. Mead

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213	CMU/SEI-2006-SR-003

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER
HQ ESC/XPK 5 Eglin Street Hanscom AFB, MA 01731-2116	

11. SUPPLEMENTARY NOTES

12A DISTRIBUTION/AVAILABILITY STATEMENT	12B DISTRIBUTION CODE
Unclassified/Unlimited, DTIC, NTIS	

13. ABSTRACT (MAXIMUM 200 WORDS)

This special report is the third in a series by the Software Engineering Institute focusing on the practical application of the Security Quality Requirements Engineering (SQUARE) process. In this report, a student team presents their results of working with three clients over the course of a semester. Each client was developing a large-scale software application and worked with the students to generate security requirements. The students' main contribution to the SQUARE process was to determine how existing software requirements-elicitation techniques could be applied to software security requirements (as opposed to end-user requirements).

With each client, the students implemented a different structured requirements-elicitation technique: Issue-Based Information Systems with an information technology firm, Joint Application Development (JAD) with the Delta client, and the Accelerated Requirements Method (ARM) with the Beta client. The ARM technique, which is a variant of JAD, held the most promise for inclusion in future applications of SQUARE. In addition to an analysis of the three elicitation techniques, the student team also generated feedback and recommendations on different steps of the SQUARE process, such as requirements prioritization and inspection. They found the Analytic Hierarchy Process to be highly useful for prioritizing requirements quickly; however, they did not find a requirements inspection technique that was well suited for any of the clients.

14. SUBJECT TERMS	15. NUMBER OF PAGES
requirements engineering, requirements elicitation, security requirements, system security, SQUARE	198

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL