SQUARE Process

Nancy Mead

January 2006

ABSTRACT: System Quality Requirements Engineering (SQUARE) is a process model that was developed at Carnegie Mellon University, with Nancy Mead as Principal Investigator [Mead 05a].The SQUARE work was supported by the Army Research Office through grant number DAAD19-02-1-0389 ("Perpetually Available and Secure Information Systems") to Carnegie Mellon University's CyLab. It provides a means for eliciting, categorizing, and prioritizing security requirements for information technology systems and applications. The focus of the model is to build security concepts into the early stages of the development life cycle. The model can also be used for documenting and analyzing the security aspects of fielded systems and for steering future improvements and modifications to those systems.

Workshop, tutorial, and academic educational materials on SQUARE are available for download on the CERT web site.

OVERVIEW

Security Quality Requirements Engineering (SQUARE) provides a means for eliciting, categorizing, and prioritizing security requirements for information technology systems and applications. The focus of this methodology is to build security concepts into the early stages of the development life cycle. The model can also be used for documenting and analyzing the security aspects of fielded systems and for steering future improvements and modifications to those systems.

After its initial development, SQUARE was applied in a series of client case studies. Carnegie Mellon graduate students worked on this project during the summer and fall of 2004 and the summer of 2005. The case study results were published [Chen 04, Gordon 05, Xie 04]. Prototype tools were also developed to support the process. The draft process was revised and baselined after the case studies were completed; the baselined process is shown in Table 1. In principle, Steps 1-4 are actually activities that precede security requirements engineering but are necessary to ensure that it is successful. Brief descriptions of each step follow; a detailed discussion of the method can be found in [Mead 05a].

Table 1. SQUARE Process

Software Engineering Institute
Carnegie Mellon University
4500 Fifth Avenue
Pittsburgh, PA 15213-2612

Phone: 412-268-5800
Toll-free: 1-888-201-4479

www.sei.cmu.edu

Number	Step	Input	Techniques	Participants	Output
1	Agree on definitions	Candidate definitions from IEEE and other standards	Structured interviews, focus group	Stakeholders, requirements engineer	Agreed-to definitions
2	Identify assets and security goals	Definitions, candidate goals, business drivers, policies and procedures, examples	Facilitated work session, surveys, interviews	Stakeholders, requirements engineer	Assets and goals
3	Develop artifacts to support security requirements definition	Potential artifacts (e.g., scenarios, misuse cases, templates, forms)	Work session	Requirements engineer	Needed artifacts: scenarios, misuse cases, models, templates, forms
4	Perform risk assessment	Misuse cases, scenarios, security goals	Risk assessment method, analysis of anticipated risk against organizational risk tolerance, including threat analysis	Requirements engineer, risk expert, stakeholders	Risk assessment results
5	Select elicitation techniques	Goals, definitions, candidate techniques, expertise of stakeholders, organizational style, culture, level of security needed, cost/benefit analysis, etc.	Work session	Requirements engineer	Selected elicitation techniques
6	Elicit security requirements	Artifacts, risk assessment results, selected techniques	Joint Application Development (JAD), interviews, surveys, model-based analysis, checklists, lists of reusable requirements types, document reviews	Stakeholders facilitated by requirements engineer	Initial cut at security requirements

7	Categorize requirements as to level (system, software, etc.) and whether they are requirements or other kinds of constraints	Initial requirements, architecture	Work session using a standard set of categories	Requirements engineer, other specialists as needed	Categorized requirements
8	Prioritize requirements	Categorized requirements and risk assessment results	Prioritization methods such as Analytical Hierarchy Process (AHP), Triage, Win-Win	Stakeholders facilitated by requirements engineer	Prioritized requirements
9	Inspect requirements	Prioritized requirements, candidate formal inspection technique	Inspection method such as Fagan, peer reviews	Inspection team	Initial selected requirements, documentation of decision-making process and rationale

A Brief Description of SQUARE

The SQUARE process is best applied by the project's requirements engineers and security experts in the context of supportive executive management and stakeholders. We've observed that the process works best when elicitation occurs after risk assessment (Step 4) has been done and when security requirements are specified before critical architecture and design decisions. Thus critical security risks to the business will be considered in the development of the security requirements.

Step 1, Agree on Definitions, is needed as a prerequisite to security requirements engineering. On a given project, team members tend to have definitions in mind, based on their prior experience, but those definitions often differ [Woody 05]. For example, for some government organizations, security has to do with access based on security clearance levels, whereas for others security may have to do with physical security or cyber security. It is not necessary to invent definitions. Sources such as the Institute for Electrical and Electronics Engineers (IEEE) and the Software Engineering Body of Knowledge (SWEBOK) provide a range of definitions to select from or tailor. A focus group meeting with the interested parties most likely enables the selection of a consistent set of definitions for the security requirements activity.

Step 2, Identify Assets and Security Goals, should be done at the organizational level and is needed to support software development in the project at hand. This provides a consistency check with the organization's policies and operational security environment. Different stakeholders usually have different goals. For example, a stakeholder in human resources may be concerned about maintaining the confidentiality of personnel records, so from their point of view the personnel records are an asset, whereas a stakeholder in a financial area may be concerned with ensuring that financial data is not accessed or modified without authorization, so they will feel that financial data are an asset. It is important to have a representative set of stakeholders, including those with operational expertise. Once the assets and goals of the various stakeholders have been identified, they need to be prioritized. In the absence of consensus, an executive decision may be needed to prioritize these assets and goals.

Step 3, Develop Artifacts, is necessary to support all subsequent security requirements engineering activities. It is often the case that organizations do not have a documented concept of operations for a project, succinctly stated project goals, documented normal usage and threat scenarios, misuse or abuse cases, and other documents needed to support requirements definition. This means that either the entire requirements process is built on unstated assumptions or a lot of time is spent backtracking to try to obtain such documentation.

Step 4, Perform Risk Assessment, requires an expert in risk assessment methods, the support of the stakeholders, and the support of a security requirements engineer. There are a number of risk assessment methods to select from. The risk assessment expert can recommend a specific method based on the needs of the organization. The artifacts from Step 3 provide the input to the risk assessment process. The outcomes of the risk assessment can help in identifying the high-priority security exposures. Organizations that do not perform risk assessment typically do not have a logical approach to considering organizational risks when identifying security requirements but tend to select specific solutions or technologies, such as encryption, without really understanding the problem that is being solved.

Step 5, Select Elicitation Technique, becomes important when there are diverse stakeholders. A more formal elicitation technique, such as the Accelerated Requirements Method [Hubbard 99], Joint Application Design [Wood 89], or structured interviews can be effective in overcoming communication issues when there are stakeholders with different cultural backgrounds. In other cases, elicitation may simply consist of sitting down with a primary stakeholder to try to understand that stakeholder's security requirements needs.

Step 6, Elicit Security Requirements, is the actual elicitation process using the selected technique. Most elicitation techniques provide detailed guidance on how to perform elicitation. This builds on the artifacts that were developed in earlier steps, such as misuse and abuse cases, attack trees, threats, and scenarios.

Step 7, Categorize Requirements, allows the security requirements engineer to distinguish among essential requirements, goals (desired requirements), and architectural constraints that may be present. Requirements that are actually constraints typically occur when a specific system architecture has been chosen prior to the requirements process. This is good, as it allows assessment of the risks associated with these constraints. This categorization also helps in the prioritization activity that follows.

Step 8, Prioritize Requirements, depends not only on the prior step but may also involve performing a cost/benefit analysis to determine which security requirements have a high payoff relative to their cost. Of course prioritization may also depend on other consequences of security breaches, such as loss of life, loss of reputation, and loss of consumer confidence.

Step 9, Inspect Requirements, can be done at varying levels of formality, from Fagan inspections (a highly structured and proven technique for requirements inspection) [Fagan 99] to peer reviews. Once inspection is complete, the project team should have an initial set of prioritized security requirements. It should also understand which areas are incomplete and must be revisited at a later time. Finally, the project team should understand which areas are dependent on specific architectures and implementations and should plan to revisit those as well.

How to Measure and Manage

Although quantitative measures do not exist, the clients for the case studies mentioned earlier recognized the value of the new security requirements and have started to take steps to incorporate them into the system. Important considerations for management are the amount of resources to be invested in this activity and in the implementation of the resultant requirements [Xie 04]. Management also needs to provide insights into the business environment and drivers and the mission of the system under development, as well as input as to the essential services and assets of the system.

Additional Resources for SQUARE

The definitive technical report on SQUARE is [Mead 05a].

The following presentations and activities were intended to initiate further discussion on the management issues addressed by the SQUARE process:

- "Considering Operational Security Risks During Systems Development," SEPG 2004, Orlando, Florida, March 9, 2004 [Alberts 04]
- "Can Secure Systems be Built Using Today's Development Processes?" European SEPG, London, England, June 17, 2004 [Woody 04]

A workshop on Requirements for High Assurance Systems (RHAS '04) was held in conjunction with the International Conference on Requirements Engineering on September 6, 2004. The workshop proceedings for this and prior RHAS workshops were published by the SEI [SEI 04]. SQUARE was presented at an International Conference on Software Engineering (ICSE) workshop as well [Mead 05b].

TOOLS

A prototype tool has been developed to support SQUARE. It primarily provides an organizational framework for the artifact documents, and it also provides default content for some of the steps. It provides limited support for sophisticated functions such as traceability and prioritization. This prototype tool is undergoing redevelopment in 2008-2009 so that it will be more robust, provide better support to the SQUARE process, and be more attractive to users. The current status of the SQUARE process and tool, as well as contact information, can be found on the CERT website.

EXPECTED RESULTS

When you apply SQUARE, you can expect to have relevant security requirements identified and documented for the system or software that is being developed. SQUARE is more suited to a system under development than one that has already been fielded, although it has been used for both. Although quantitative measures do not exist, case study clients recognized the value of the new security requirements and have taken steps to incorporate them into their system specifications. You'll need to consider the resources required for this activity and for the implementation of the resulting requirements [Xie 04].

Our experience with SQUARE suggests that the system and its elements have to be considered within the context or environment in which it operates. For example, a system that operates on a single isolated workstation will have very different security requirements from a similar system that is Web based. In a similar fashion, a medical information system will have different security requirements for workstations that are isolated in a physician's office than for those that are in

a public area in a hospital. These differences should be accounted for in the artifacts developed in Step 3, for example in usage scenarios and misuse or abuse cases. When the context changes on a project, you should revisit the security requirements and reapply the SQUARE process. It may be that a subset of the SQUARE steps will be sufficient for this purpose, but we do not yet have enough experience with subsequent applications of SQUARE to the same system to make that determination.

SQUARE SAMPLE OUTPUTS

Several case studies have been conducted using the SQUARE process model [Chen 04, Gordon 05]. The goals of the case studies were to experiment with each step of the SQUARE process, make recommendations, and determine the feasibility of integrating SQUARE into standard software development practices. The case studies involved real-world clients that were developing large-scale IT projects. The clients included an IT firm in Pittsburgh, Pennsylvania, a federal government research institute, and a department of the federal government.

The IT Firm Acme Corporation

Acme Corporation,[1] a private company headquartered in Pittsburgh, provides technical and management services to various public sectors and a number of diversified private sectors. Its product, the Asset Management System (AMS) version 2,[2] provides a tool for companies to make strategic allocations and plans for their critical IT assets. It provides specialized decision support capabilities via customized views. AMS provides a graphical interface to track and analyze the state of important assets. The security requirements surrounding the AMS are the subject of one of our case studies and the source of the sample outputs that follow.

It is important to note here that the AMS is a fielded system, undergoing major upgrades, so the results from these case studies may not be a perfect fit for determining SQUARE's usefulness in a pre-production environment. However, the willingness of the client to participate was an important factor in its selection.

---

1 Acme Corporation (Acme) is an alias used to protect the identity of the client under study.

2 Asset Management System (AMS) is an alias used to protect the identity of the client under study.

Further, the results of these case studies are important in beginning to understand the effectiveness of the nine steps of the SQUARE process.

Output from SQUARE Steps

In each case study, the student teams focused part of their efforts on researching various methods to conduct each step. In some cases, redundant work was completed to determine which methods might lend themselves better to SQUARE. In order to provide concrete examples of the nine SQUARE steps, we present here a sample of the output from each individual step (all taken from the case studies) to demonstrate how SQUARE looks in action.

Step 1: Agree on Definitions

We worked with the client to agree on a common set of security definitions to create a common base of understanding. The following is a small subset of the definitions that were agreed on:

- access control: Ensures that resources are granted only to those users who are entitled to them.
- access control list: A table that tells a computer operating system which access rights or explicit denials each user has to a particular system object, such as a file directory or individual file.
- antivirus software: A class of program that searches hard drives and memory for any known or potential viruses.

The full set of definitions was drawn from resources such as IEEE, Carnegie Mellon University, industry, and dictionaries.

Step 2: Identify Safety and Security Goals

We worked with the client to flesh out security goals that mapped to the company's overall business goals. This is one example set of goals:

- Business Goal of AMS: To provide an application that supports asset management and planning.
- Safety and Security Goals: Three high-level safety and security goals were derived for the system:
    1. Management shall exercise effective control over the system's configuration and use.
    2. The confidentiality, accuracy, and integrity of the AMS shall be maintained.
    3. The AMS shall be available for use when needed.

Step 3: Develop Artifacts

Architectural diagrams, use cases, misuse cases, abuse case diagrams, attack trees, and essential assets and services were documented in this portion of SQUARE. For instance, an attack scenario was documented in the following way:

System administrator accesses confidential information

1. by being recruited OR
    1. by being bribed OR
    2. by being threatened OR
    3. through social engineering OR
2. by purposefully abusing rights

An example abuse case diagram is shown in Figure 1.

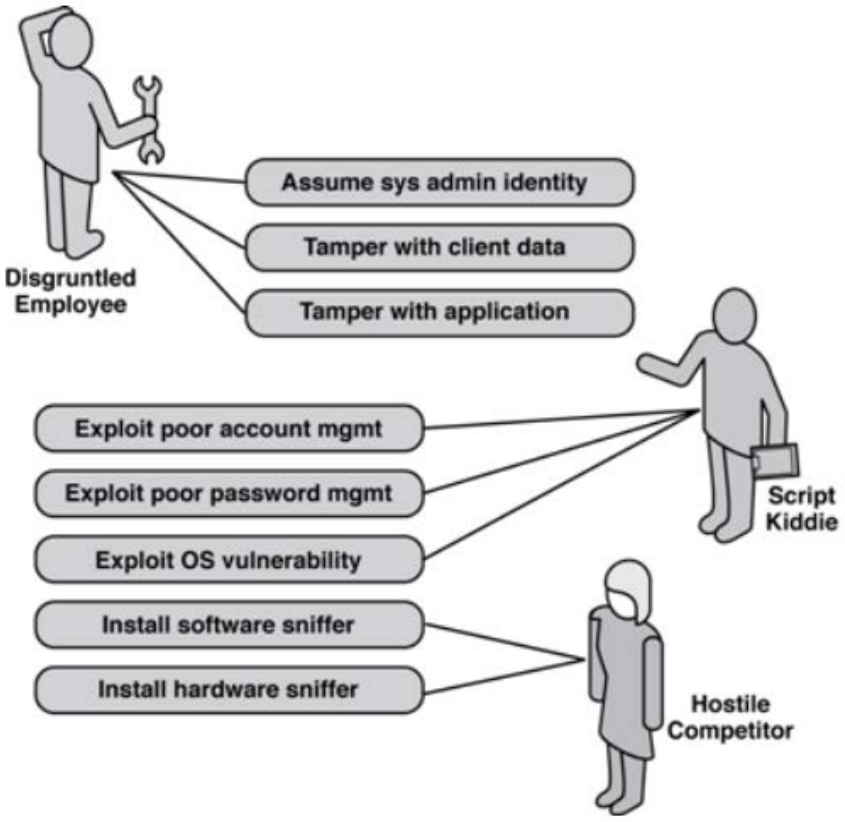

Figure 1. Abuse case example

This step creates needed documentation that serves as input into the following steps.

Step 4: Perform Risk Assessment

The risk assessment techniques that were field tested were selected after completing a literature review. This review examined the usefulness and applicability of eight risk assessment techniques:

1. General Accounting Office Model [GAO 99]
2. National Institute of Standards and Technology (NIST) Model [Stoneburner 02]
3. NSA's INFOSEC Assessment Methodology [NSA 04]
4. Shawn Butler's Security Attribute Evaluation Method [Butler 02]
5. Carnegie Mellon's Vendor Risk Assessment and Threat Evaluation [Lipson 01]
6. Yacov Haimes's Risk Filtering, Ranking, and Management Model [Haimes 04]
7. Carnegie Mellon's Survivable Systems Analysis Method [Mead 02]
8. Martin Feather's Defect Detection and Prevention Model [Cornford 04]

Each technique was ranked in four categories:

1. suitability for small companies
2. feasibility of completion in the time allotted
3. lack of dependence on historical threat data
4. suitability in addressing requirements

The results of the ranking are shown in Table 2.

Table 2. Ranking of assessment techniques

Techniques		Suitable for small companies	Feasible to complete within time frame	Does not require additional data collection	Suitable for requirements	Average score
	GAO	2	4	2	2	2.50
	NIST	2	2	1	1	1.50
	NSA/IAM	3	3	2	2	2.50
	SAEM	4	4	4	4	4.00
	V-Rate	3	4	4	4	3.75
	Haimes	2	2	2	2	2.00

	SSA	2	2	2	4	2.50
	DDP/Feather	3	4	2	4	3.25

After averaging scores from the four categories, NIST's and Haimes's models were selected as useful techniques for the risk assessment step. Brainstorming, attack tree, and misuse case documentation were used to identify potential threat scenarios. The two independent risk assessment analyses produced a useful risk profile for the company's system. The two most meaningful findings were

1. Insider threat poses the highest impact risk to the AMS.
2. Because of weak controls, it is easy for an insider or unauthorized user to defeat authentication.

In this particular case study, we also identified a set of essential services and assets as part of the artifact generation. This is not part of the standard SQUARE process but nevertheless can be a beneficial exercise if enough architectural information already exists to support it. All findings from the risk assessment, along with the findings from the essential services and asset identification process, were used to determine the priority level associated with each of the nine requirements.

We analyzed the importance of each of the major system services, outlined in the 11 use cases shown in Table 3, and made a determination as to which were essential:

Table 3. Classification of use cases

Use Case	Service	Status
UC-1	View Floor Plans	*Essential*
UC-2	Enter Damage Assessment	*Essential*
UC-3	Add/Delete/Edit Post-it Notes	Non-Essential
UC-4	Find Specialized Employees	*Important*
UC-5	Create Journal Entry	Non-Essential
UC-6	Install the Asset Management System	Non-Essential
UC-7	Create Links to Documents	Non-Essential
UC-8	Archibus Admin- Add User and Assign Privileges	Non-Essential
UC-9	View Contact Information for Maintenance Tasks	*Important*
UC-10	Create Open Space Report	*Essential*
UC-11	View Incident Command	*Essential*

There are two major assets in this system. The first is the Windows Server Computer, which houses the majority of the production system's intellectual assets (that is, the code that runs the system). This computer acts as a server that allows remote users to access the Asset Management System. The second major asset is the information inside the Windows Server Computer, specifically the files stored in the Microsoft IIS server and the information stored in the Sybase Database and the MapGuide Database, is critical toward making informed decisions. If this information is lost or compromised, the ability to make accurate decisions is lost.

Step 5: Select Elicitation Techniques

For this step, teams tested various elicitation techniques and models. It is often the case that multiple techniques will work for the same project. The difficulty is in choosing a technique that can adapt to the number and expertise of stakeholders, the size and scope of the client project, and the expertise of the requirements engineering team. It is extremely unlikely that any single technique will work for all projects under all circumstances, though our experience has shown that the Accelerated Requirements Method [Hubbard 00] has been successful in eliciting security requirements.

The following is a sample of elicitation techniques that may be appropriate:

- structured/unstructured interviews
- use/misuse cases [Jacobson 92]
- facilitated meeting sessions, such as Joint Application Development and the Accelerated Requirements Method [Wood 89, Hubbard 99]
- Soft Systems Methodology [Checkland 89]
- Issue-Based Information Systems [Kunz 70]
- Quality Function Deployment [QFD 05]
- Feature-Oriented Domain Analysis [Kang 90]
- Controlled Requirements Expression [Mullery 79]
- Critical Discourse Analysis [Schiffrin 94]

Steps 6 and 7: Elicit and Categorize Safety and Security Requirements

Nine security requirements were identified and then organized to map to the three high-level security goals (see Step 2). Some of the requirements were

- Req 1: The system is required to have strong authentication measures in place at all system gateways and entrance points (maps to Goals 1 and 2).
- Req 2: The system is required to have sufficient process-centric and logical means to govern which system elements (data, functionality, etc.) users can view, modify, and/or interact with (maps to Goals 1 and 2).Req 3: A conti-

nuity of operations plan (COOP) is required to assure system availability (maps to Goal 3).

- Req 6: It is required that the system's network communications be protected from unauthorized information gathering and/or eavesdropping (maps to Goals 1 and 2).

the nine security requirements were central to the security requirements document that was ultimately delivered to the client.

Step 8: Prioritize Requirements

In the first case study, the nine security requirements were prioritized based on the following qualitative rankings:

Essential: Product will be unacceptable if this requirement is absent.

- Conditional: Requirement enhances security, but the product is acceptable if this requirement is absent.
- Optional: Requirement is clearly of lower priority than Essential and Conditional requirements.

Req 1 from Steps 6 and 7, which dealt with authentication at borders and gateways, was deemed essential because of its importance in protecting against the high-impact, authentication-related risks identified in the risk assessment. Req 3, dealing with continuity of operations planning, was still seen as an important element and worth considering, but it was found to be an optional requirement relative to the other eight requirements. That is, though COOP plans are valuable, the risk assessment phase found that greater threats to the system resulted from unauthorized disclosure of information than from availability attacks.

We also used the Analytical Hierarchy Process (AHP) methodology to prioritize requirements and found it to be successful both in client acceptance and in its ability to handle security requirements [Karlsson 97, Saaty 80]. AHP is a technique for decision making in situations in which multiple objectives are present. The method calculates the relative value and cost among security requirements. By using AHP, the requirements engineer can also confirm the consistency of the stakeholders' results, which can prevent subjective judgment errors and increase the likelihood that the results are more reliable. The stakeholders found AHP valuable not only for its ability to quickly prioritize the security requirements but also because of the internal discussion that is stimulated.

Step 9: Inspect Requirements

We experimented with different inspection techniques and had varying levels of success with each. None of the inspection techniques were sufficiently effective

in identifying defects in the security requirements. Instead, we recommend experimenting with the Fagan inspection technique.

In one case study instance, each team member played a role in inspecting the quality of the team's work and deliverables. A peer review log was created to document what had been reviewed and was used to maintain a log of all problems, defects, and concerns. Each entry in the log was numbered and dated, addressing the date, origin, defect type, description, severity, owner, reviewer, and status. Each entry was assigned to an owner, who was responsible for making sure that defects were fixed. This step was used as a sanity check to ensure that the system met quality goals and expectations.

SQUARE Final Results

The final output to the client was a security requirements document. The client could then use this document in the early stages of the development life cycle to make sure that security requirements were built into project plans. Once a system has been deployed, the organization can look back to its requirements documentation to analyze whether it met its requirements and thus satisfied its security goals. As change occurs—be it a configuration concern in the system, the organization's risk profile, or a business goal—the SQUARE process can be reused to determine how the change might affect the system's security requirements. SQUARE can thus be reapplied to a system as needed.

Because the key players include a dedicated task force with knowledge of security who team with a group of knowledgeable client personnel, conducting a SQUARE assessment only requires that a firm have the time and human resources available to assist a group of outside analysts. Further, a firm knowledgeable in security could be in a position to conduct SQUARE analysis without outside help. The first graduate team spent a significant amount of time with the client in helping the client develop documentation. Many firms may complete this step before the SQUARE analysis begins. The second phase team made use of this documentation and was able to complete its assessment with very little client/analyst interaction. The SQUARE analysis was very lightweight and unobtrusive to the client in this regard. The third team worked with the initial client and two other clients, focusing on Steps 5 through 9.

GLOSSARY

Fagan Inspection	A formal inspection method developed by Michael Fagan. http://www.mfagan.com/process.html
JAD	Joint Application Development—a focused workshop. http://www.credata.com/research/jad.html

SWEBOK	Software Engineering Body of Knowledge. http://www.swebok.org

REFERENCES

[Alberts 04] Alberts, C.; Dorofee, A.; & Woody, C. "Considering Operational Security Risks During Systems Development." SEPG 2004 (CD-ROM). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004.

[Butler 02] Butler, Shawn. "Security Attribute Evaluation Method: A Cost-Benefit Approach," 232-240. Proceedings of the 24th International Conference on Software Engineering. Orlando, FL, May 19-25, 2002. New York, NY: ACM Press, 2002.

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

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

[Cornford 04] Cornford, Steven L.; Feather, Martin S.; & Hicks, Kenneth A. DDP – A Tool for Life-Cycle Risk Management. http://ddptool.jpl.nasa.gov/docs/f344d-slc.pdf (2004).

[Fagan 99] Fagan, Michael E. "Design and Code Inspections to Reduce Errors in Program Development." IBM Systems Journal 38, 2 & 3 (1999): 258–287. http://www.research.ibm.com/journal/sj/382/fagan.pdf

[GAO 99] U.S. General Accounting Office. "Information Security Risk Assessment: Practices of Leading Organizations, A Supplement to GAO's May 1998 Executive Guide on Information Security Management." Washington, D.C.: U.S. General Accounting Office, 1999.

[Gordon 05] Gordon, D.; Mead, N. R.; Stehney, T.; Wattas, N.; & Yu, E. System Quality Requirements Engineering (SQUARE): Case Study on Asset Management System, Phase II (CMU/SEI-2005-SR-005). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2005. http://www.sei.cmu.edu/publications/documents/05.reports/05sr005.html.

[Haimes 04] Haimes, Yacov Y. Risk Modeling, Assessment, and Management, 2nd ed. Hoboken, NJ: John Wiley and Sons, Inc., 2004.

[Hubbard 99] Hubbard, R. "Design, Implementation, and Evaluation of a Process to Structure the Collection of Software Project Requirements." PhD diss., Colorado Technical University, 1999.

[Hubbard 00] Hubbard, R.; Mead, N.; & Schroeder, C. “An Assessment of the Relative Efficiency of a Facilitator-Driven Requirements Collection Process With Respect to the Conventional Interview Method.” Proceedings of the International Conference on Requirements Engineering. June 2000. Los Alamitos, CA: IEEE Computer Society Press, 2000.

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

[Kang 90] Kang, K.; Cohen, S.; Hess, J.; Novak, W.; & Peterson, A. Feature-Oriented Domain Analysis (FODA) 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 97] Karlsson, J. & Ryan K. “A Cost-Value Approach for Prioritizing Requirements.” IEEE Software 14, 5 (Sept./Oct. 1997): 67-74.

[Kunz 70] Kunz, Werner & Rittel, Horst. “Issues as Elements of Information Systems.” http://www-iurd.ced.berkeley.edu/pub/WP-131.pdf (1970).

[Lipson 01] Lipson, Howard F.; Mead, Nancy R.; & Moore, Andrew P. A Risk-Management Approach to the Design of Survivable COTS-Based Systems. http://www.cert.org/research/isw/isw2001/papers/Lipson-29-08-a.pdf (2001).

[Mead 02] Mead, N. R. Survivable Systems Analysis Method. http://www.cert.org/archive/html/analysis-method.html (2002).

[Mead 04] Mead, N. R. “Requirements Elicitation and Analysis Processes for Safety & Security Requirements.” Proceedings of the Third International Workshop on Requirements for High Assurance Systems (RHAS 2004). Kyoto, Japan, Sept. 6, 2004. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004. http://www.sei.cmu.edu/community/rhas-workshop/rhas04-proceedings.pdf.

[Mead 05a] Mead, N.R., Hough, E. & Stehney, T. Security Quality Requirements Engineering (SQUARE) Methodology (CMU/SEI-2005-TR-009). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2005. http://www.sei.cmu.edu/publications/documents/05.reports/05tr009.html.

[Mead 05b] Mead, N. R. & Stehney, T. “Security Quality Requirements Engineering (SQUARE) Methodology.” Software Engineering for Secure Systems (SESS05), ICSE 2005 International Workshop on Requirements for High Assurance Systems. St. Louis, MO, May 15-16, 2005.

[Mullery 79] Mullery, G. P. “CORE: A Method for Controlled Requirements Specification.” Proceedings of the 4th International Conference on Software Engineering. Los Alamitos, CA: IEEE Computer Society Press, 1979.

[NSA 04] National Security Agency. INFOSEC Assessment Methodology. http://www.iatrp.com/iam.cfm (2004).

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

[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, 1994.

[SEI 04] Software Engineering Institute. International Workshop on Requirements for High Assurance Systems, Kyoto, Japan, Sept. 6, 2004. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004. http://www.sei.cmu.edu/community/rhas-workshop/rhas04-proceedings.pdf.

[Stoneburner 02] Stoneburner, Gary; Goguen, Alice; & Feringa, Alexis. Risk Management Guide for Information Technology Systems (Special Publication 800-30). Gaithersburg, MD: National Institute of Standards and Technology, 2002. http://csrc.nist.gov/publications/nistpubs/800-30/sp800-30.pdf.

[Wood 89] Wood, Jane & Silver, Denise. Joint Application Design: How to Design Quality Systems in 40% Less Time. New York, NY: John Wiley & Sons, 1989.

[Woody 04] Woody, Carol; Hall, Anthony; & Clark, John. "Can Secure Systems be Built Using Today's Development Processes?" Panel presentation, European SEPG, London, England, June 17, 2004. http://www.cert.org/archive/pdf/eursepg04.pdf.

[Woody 05] Woody, C. Eliciting and Analyzing Quality Requirements: Management Influences on Software Quality Requirements (CMU/SEI-2005-TN-010). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004. http://www.sei.cmu.edu/publications/documents/05.reports/05tn010.html.

[Xie 04] Xie, Nick & Mead, Nancy R. SQUARE Project: Cost/Benefit Analysis Framework for Information Security Improvement Projects in Small Companies (CMU/SEI-2004-TN-045). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004. http://www.sei.cmu.edu/publications/documents/04.reports/04tn045.html.

This material is based upon work funded and supported by Department of Homeland Security under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center sponsored by the United States Department of Defense.

DM-0001120