Engineering Safety- and Security-Related Requirements for Software-Intensive Systems

Presented at SEPG'2007

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Donald Firesmith
13 March 2007

© 2007 Carnegie Mellon University

Contents

© 2007 Carnegie Mellon University

Three Disciplines:

Requirements, Safety, and Security Engineering

© 2007 Carnegie Mellon University

Three Related Disciplines

Safety Engineering

> the engineering discipline within systems engineering concerned with lowering the risk of *unintentional unauthorized* harm to valuable assets to a level that is acceptable to the system's stakeholders by preventing, detecting, and reacting to such harm, mishaps (i.e., accidents and incidents), hazards, and safety risks

Security Engineering

> the engineering discipline within systems engineering concerned with lowering the risk of *intentional unauthorized* harm to valuable assets to a level that is acceptable to the system's stakeholders by preventing, detecting, and reacting to such harm, misuses (i.e., attacks and incidents), threats, and security risks

Requirements Engineering

> the engineering discipline within systems/software engineering concerned with identifying, analyzing, reusing, specifying, managing, verifying, and validating goals and requirements (including safety- and security-related requirements)

© 2007 Carnegie Mellon University

Challenges:

Combining Requirements, Safety, and Security Engineering

© 2007 Carnegie Mellon University

Challenges$_1$

Requirements Engineering, Safety Engineering, and Security Engineering:

- Different *Communities*
- Different *Disciplines* with different Training, Books, Journals, and Conferences
- Different *Professions* with different Job Titles
- Different fundamental underlying *Concepts* and *Terminologies*
- Different *Tasks, Techniques,* and *Tools*

Safety and Security Engineering are:

- Typically treated as secondary Specialty Engineering Disciplines
- Performed separately from, largely independently of, and lagging behind the primary Engineering Workflow
  (Requirements, Architecture, Design, Implementation, Integration, Testing)

© 2007 Carnegie Mellon University

Challenges$_2$

Current separate Processes for Requirements, Safety, and Security are Inefficient and Ineffective.

Separation of Requirements Engineering, Safety Engineering, and Security Engineering:

- Causes *poor* Safety- and Security-related Requirements.
  - Goals rather than Requirements
  - Vague, unverifiable, unfeasible, architectural and design constraints
- Inadequate and too late to drive architecture and testing
- Difficult to achieve Certification and Accreditation

© 2007 Carnegie Mellon University

Challenges$_3$

Poor requirements are a primary cause of more than half of all project failures (defined in terms of):

- Major Cost Overruns
- Major Schedule Overruns
- Major Functionality not delivered
- Cancelled Projects
- Delivered Systems that are never used

Poor Requirements are a major Root Cause of many (or most) Accidents involving Software-Intensive Systems.

Security 'Requirements' often mandated:

- Industry Best Practices
- Security Functions

© 2007 Carnegie Mellon University

Challenges$_4$

How Safe and Secure is Safe and Secure *enough*?

Situation Cries out for Process Improvement:

- Better Consistency between Safety and Security Engineering
  - More consistent Concepts and Terminology
  - Reuse of Techniques across Disciplines
  - Less Unnecessary Overlap and Avoidance of Redundant Work
- Better Collaboration:
  - Between Safety and Security Engineering
  - With Requirements Engineering
- Better Safety- and Security-related Requirements

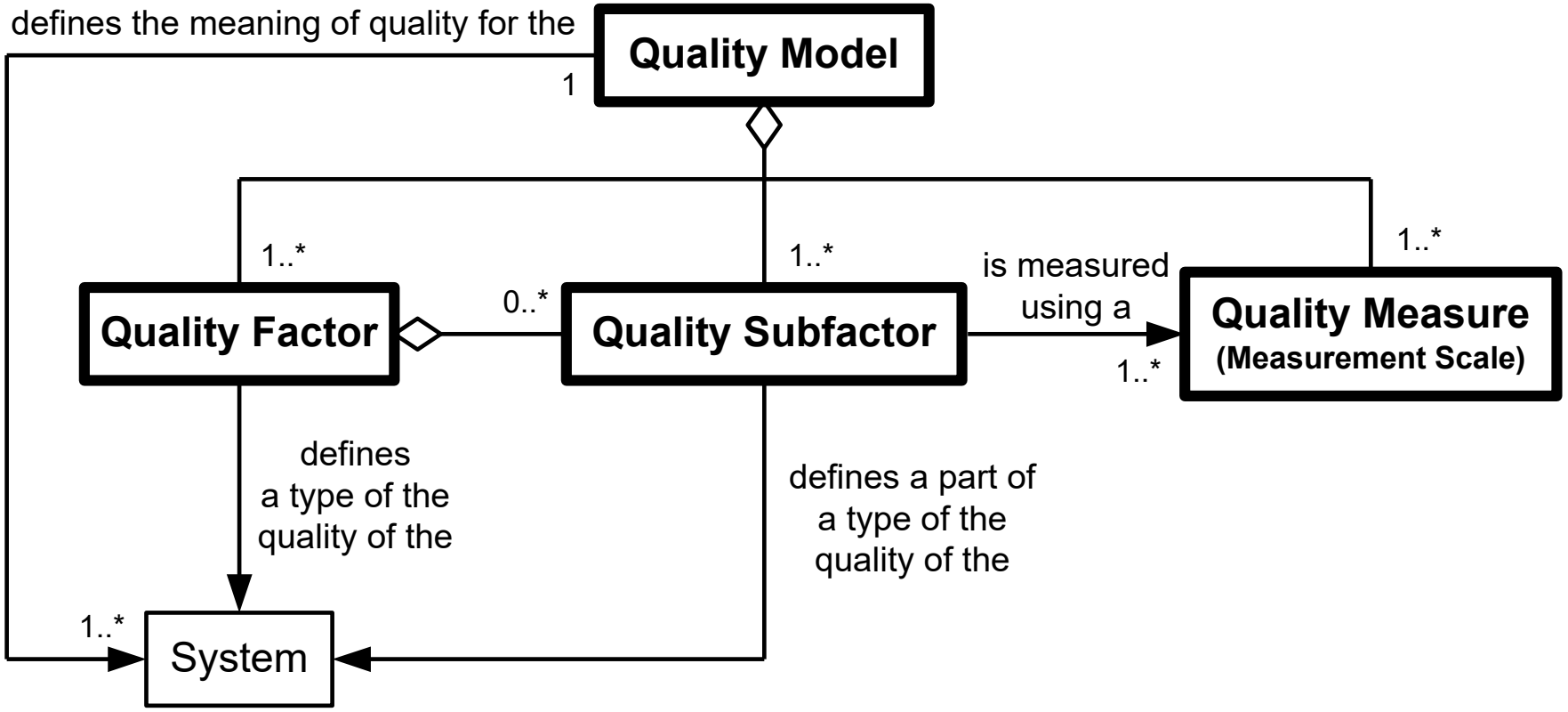
© 2007 Carnegie Mellon University

Fundamental Concepts:

A Foundation for Understanding

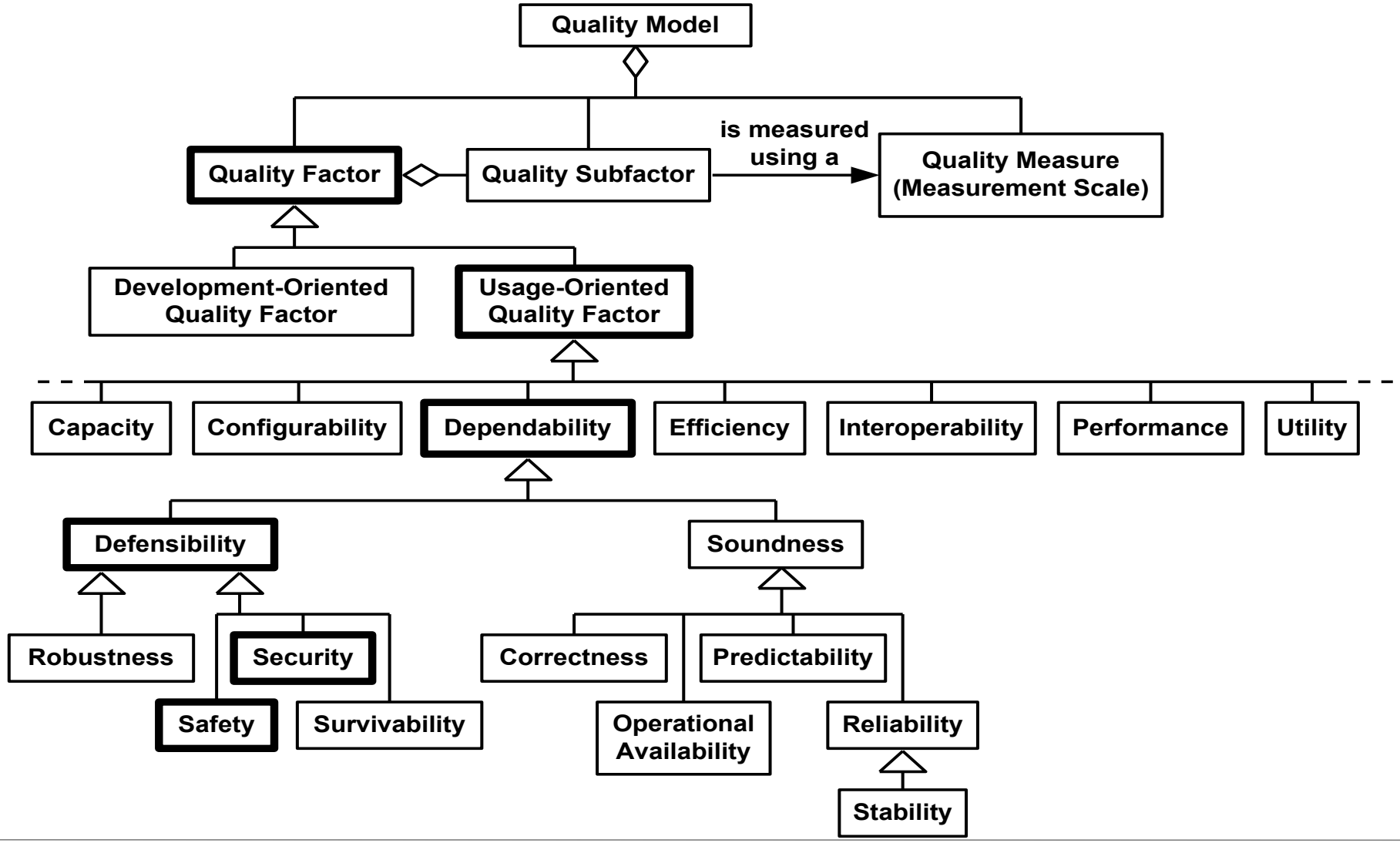
© 2007 Carnegie Mellon University

Quality Model



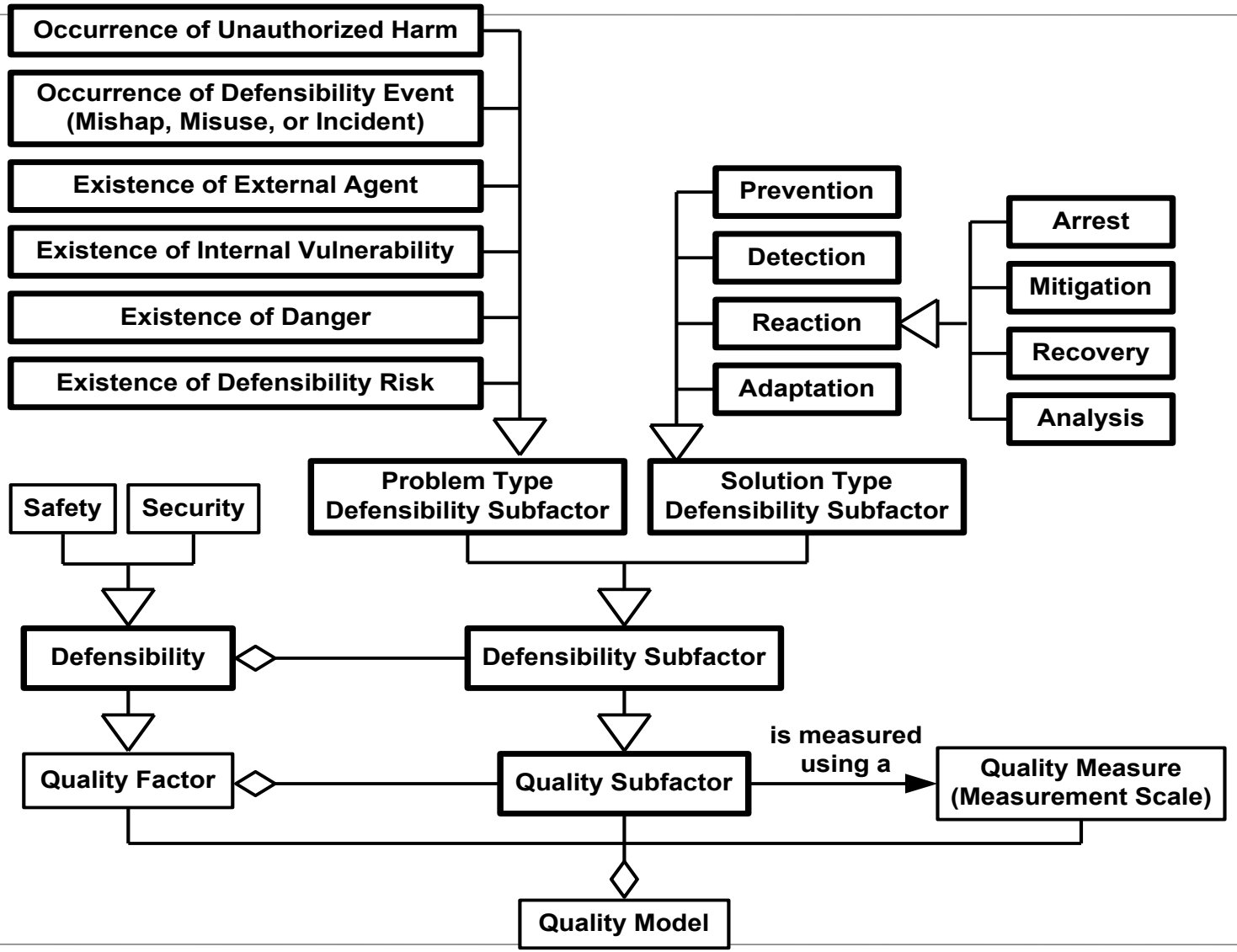
© 2007 Carnegie Mellon University

Quality Factors



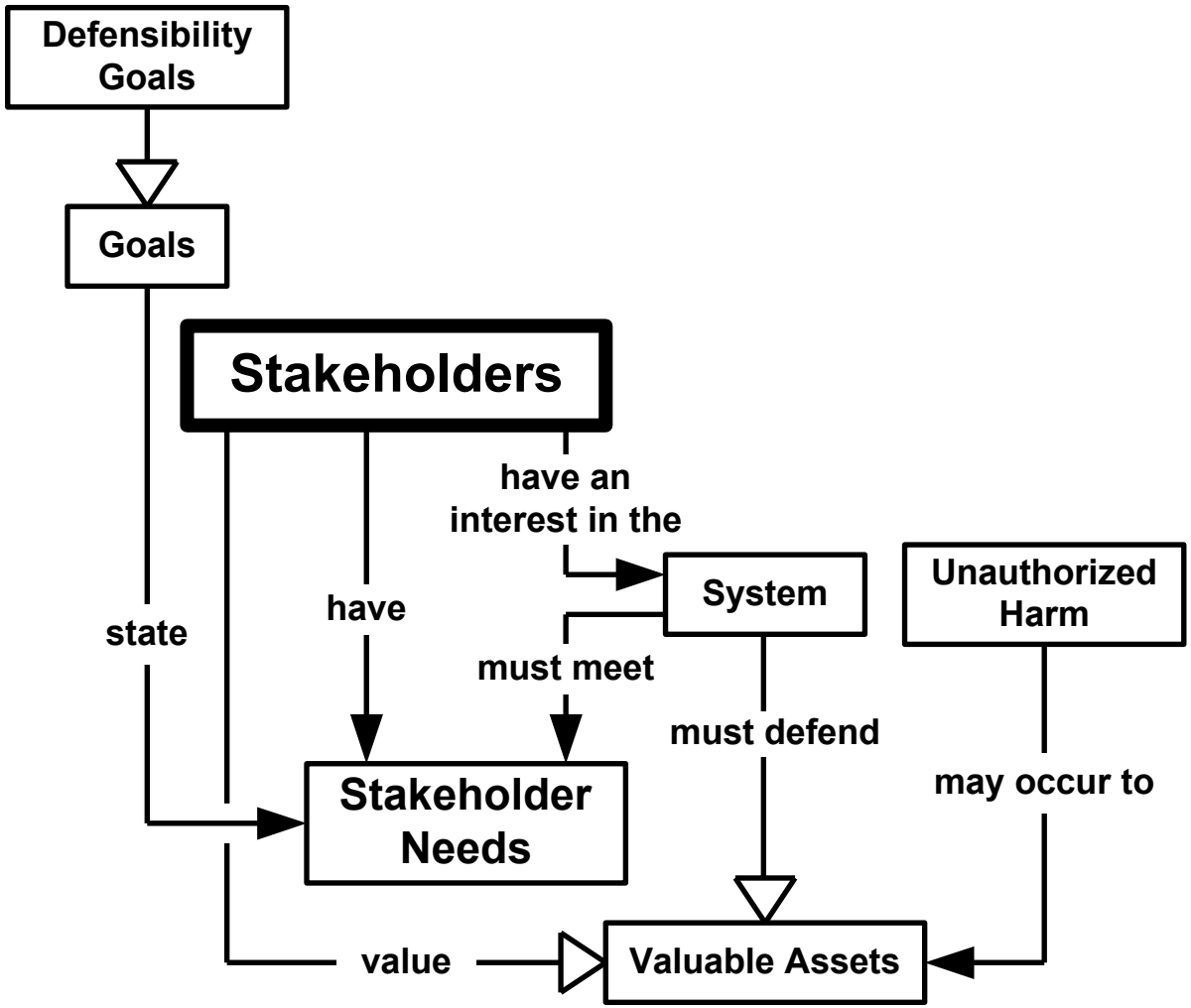
© 2007 Carnegie Mellon University

Defensibility Quality Subfactors



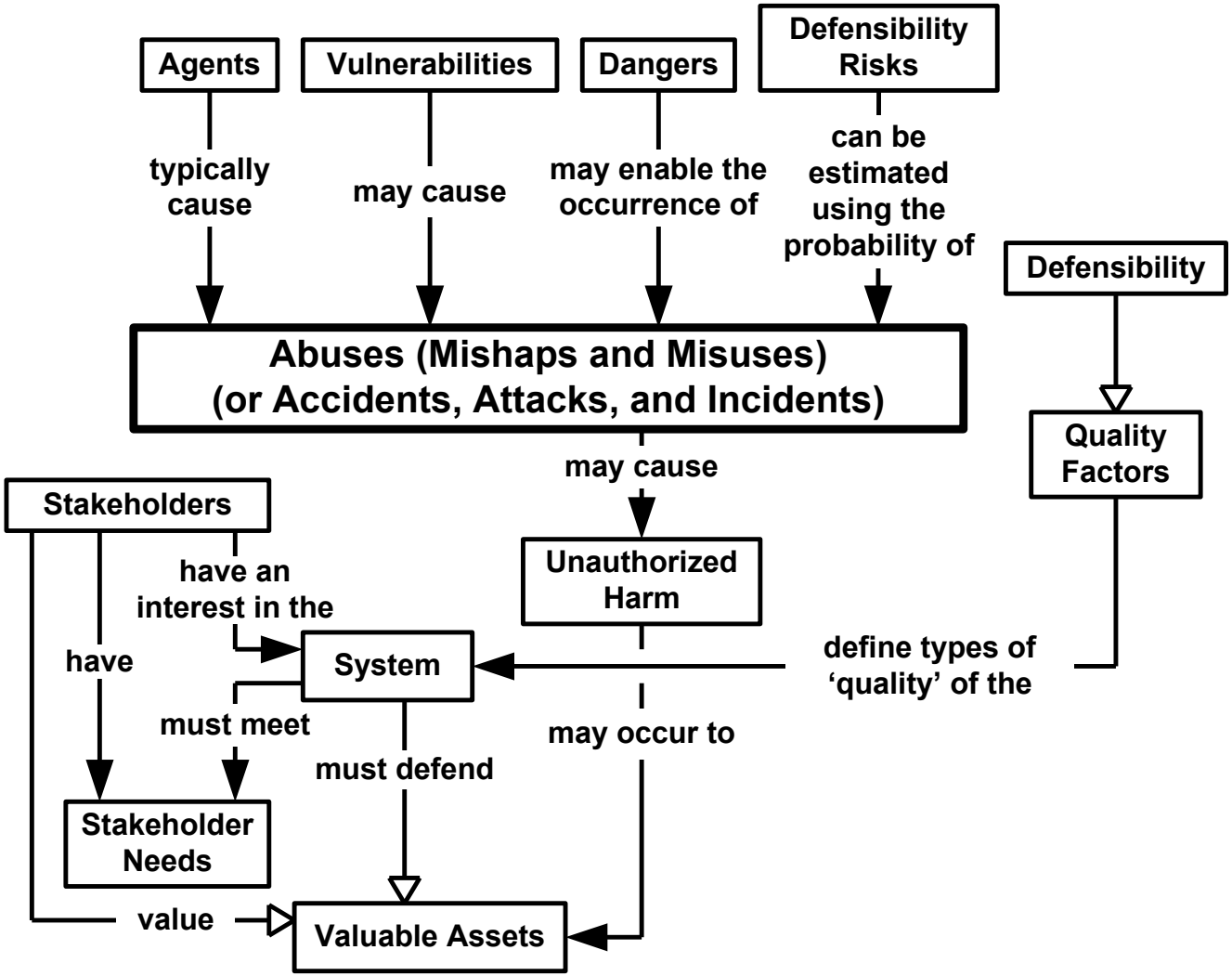
© 2007 Carnegie Mellon University

Low-Level Fundamental Concepts



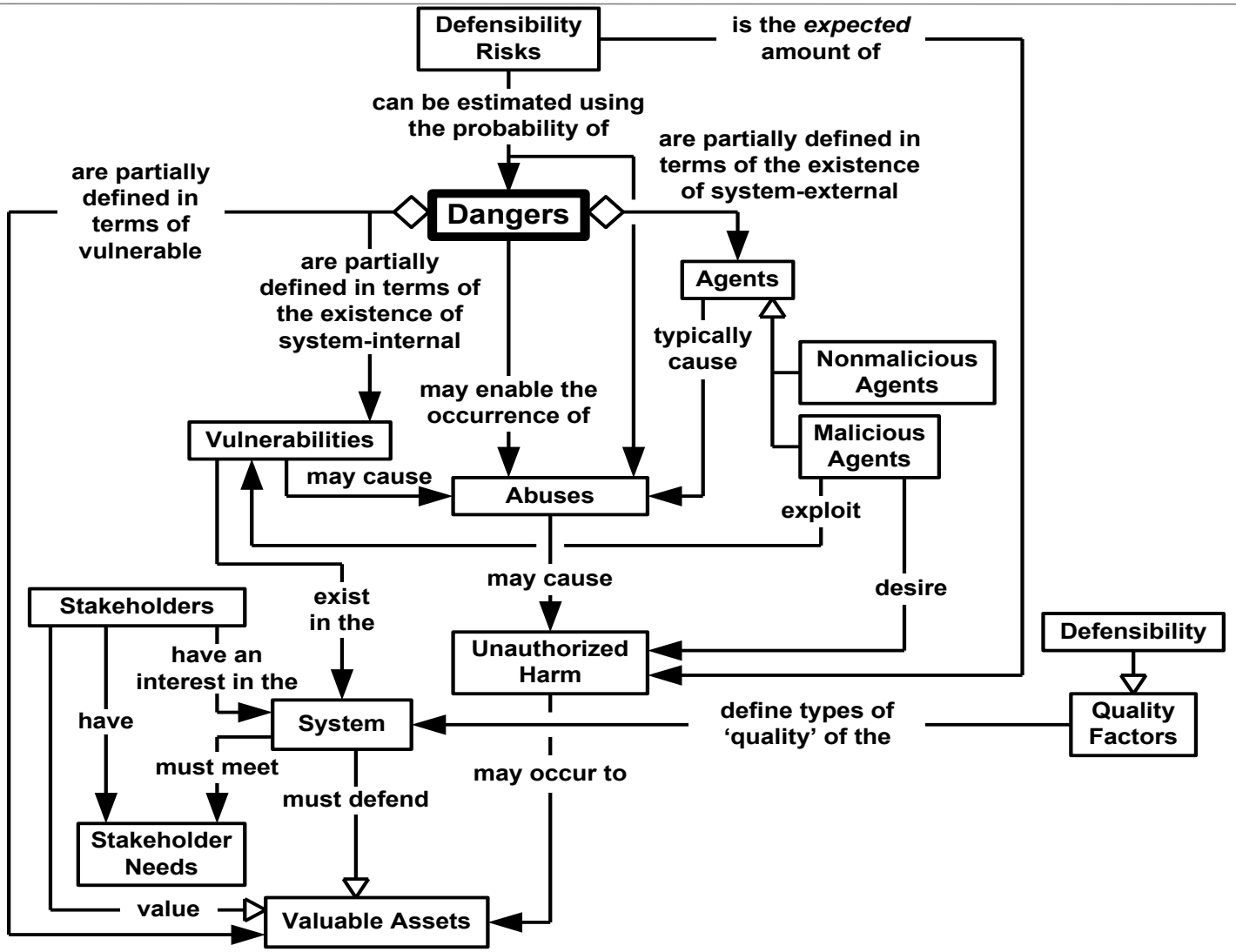
© 2007 Carnegie Mellon University

Abuses (Accidents, Attacks, and Incidents)



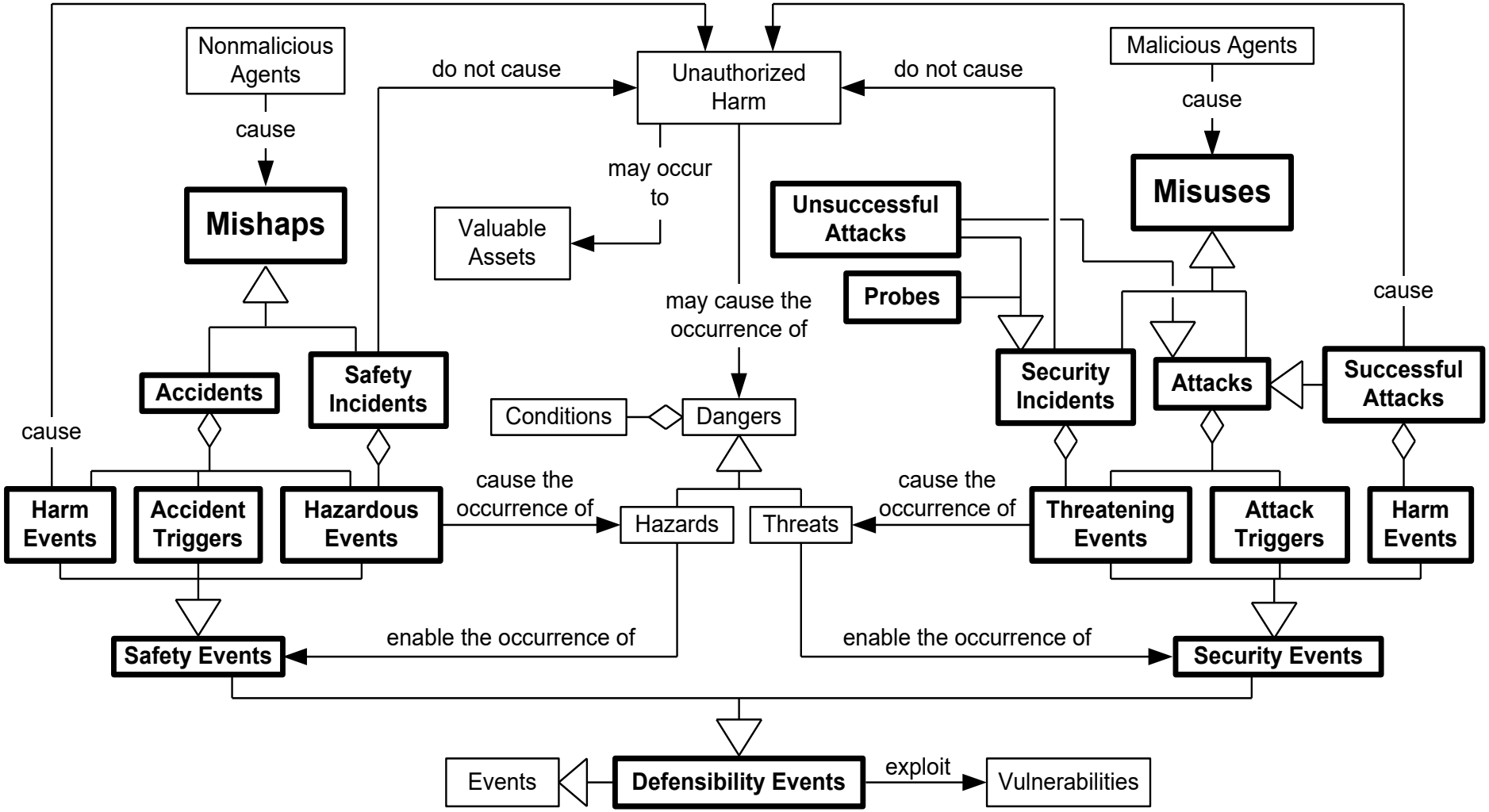
© 2007 Carnegie Mellon University

Dangers and Related Concepts



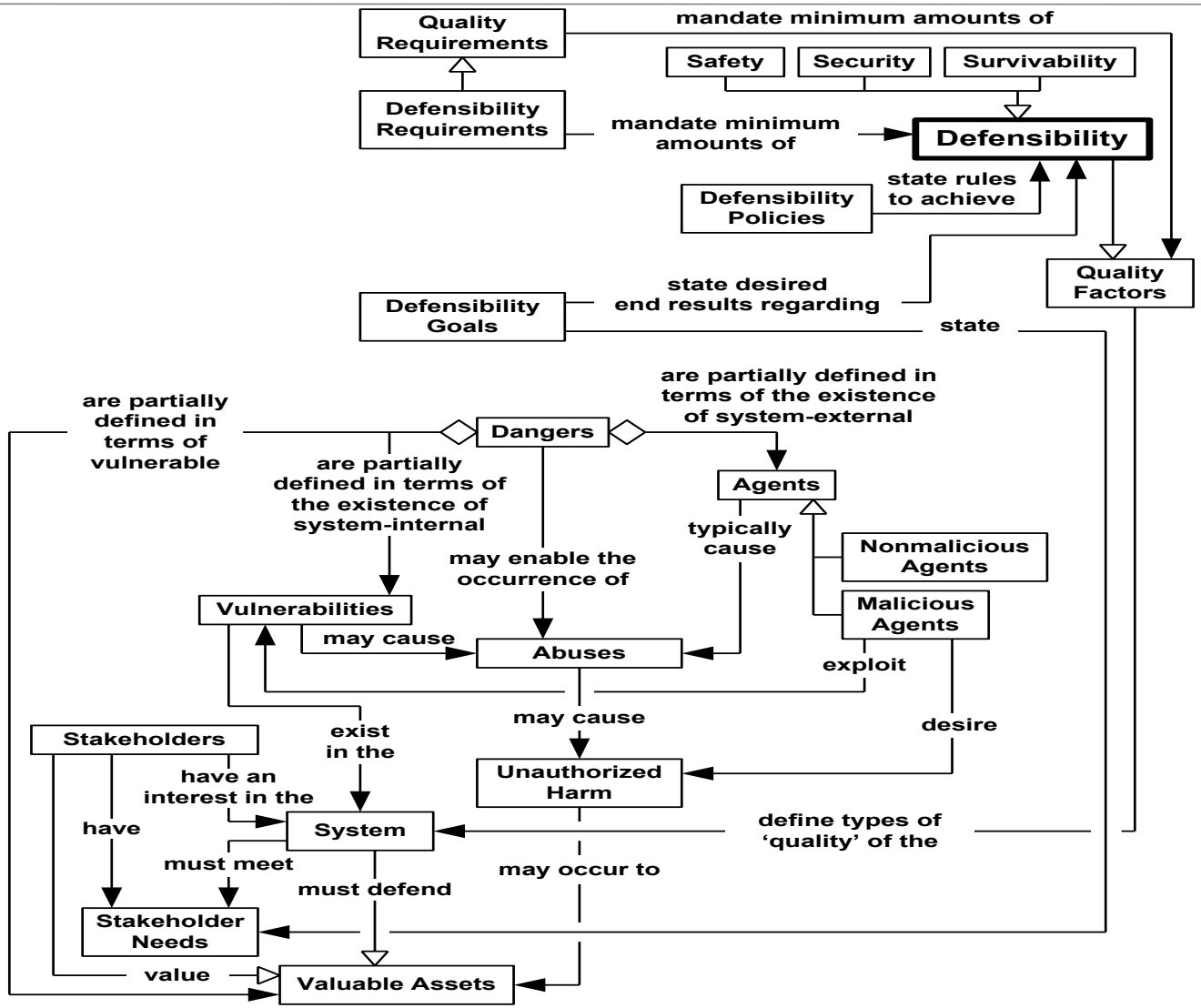
© 2007 Carnegie Mellon University

Mishaps and Misuses vs. Hazards and Threats



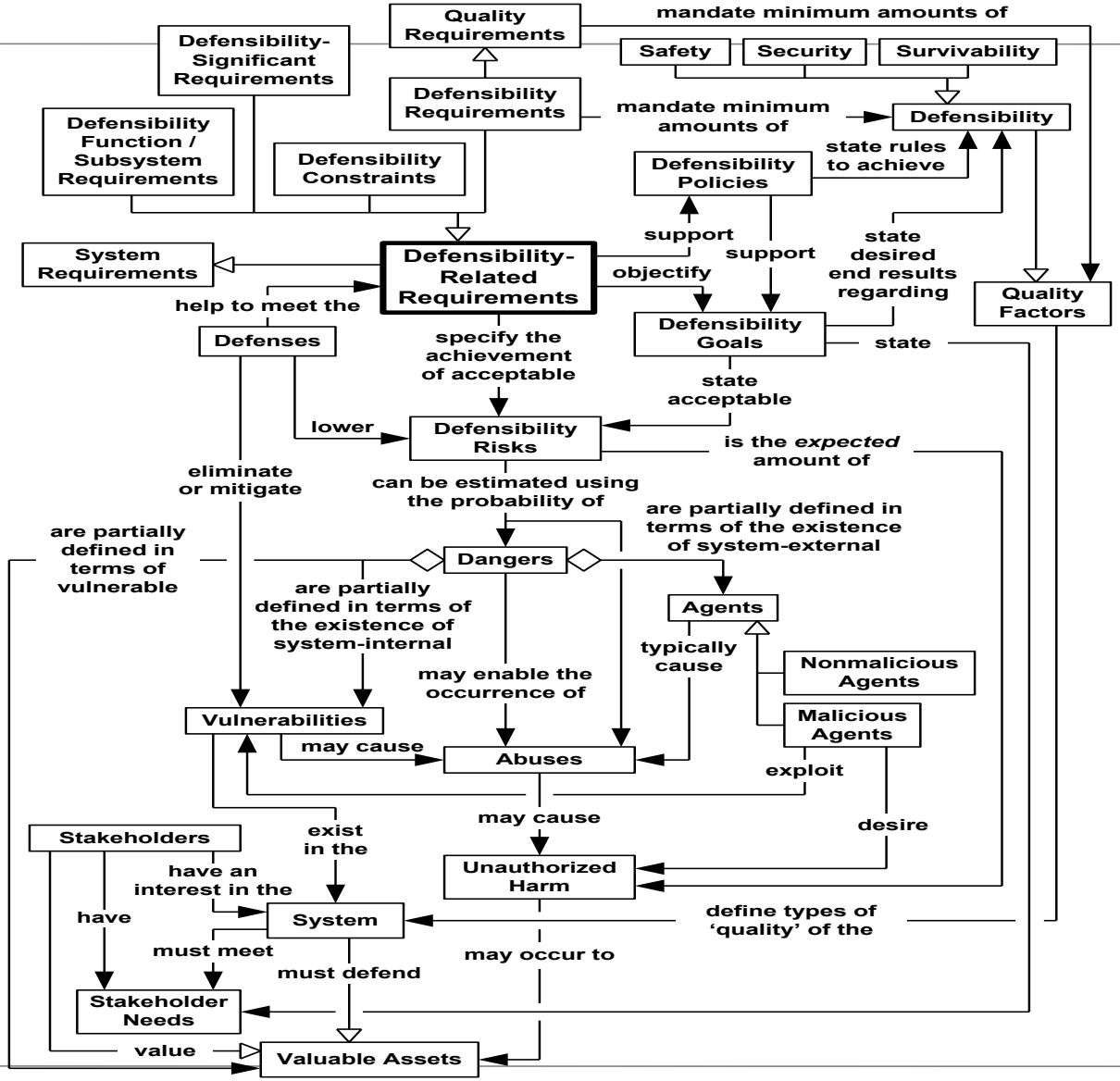
© 2007 Carnegie Mellon University

Defensibility



© 2007 Carnegie Mellon University

Concepts with Defensibility-Related Requirements



© 2007 Carnegie Mellon University

Safety- and Security-Related Requirements

© 2007 Carnegie Mellon University

Types of Safety- and Security-Related Requirements

Too often only a Single Type of Requirements is considered.

Not just:

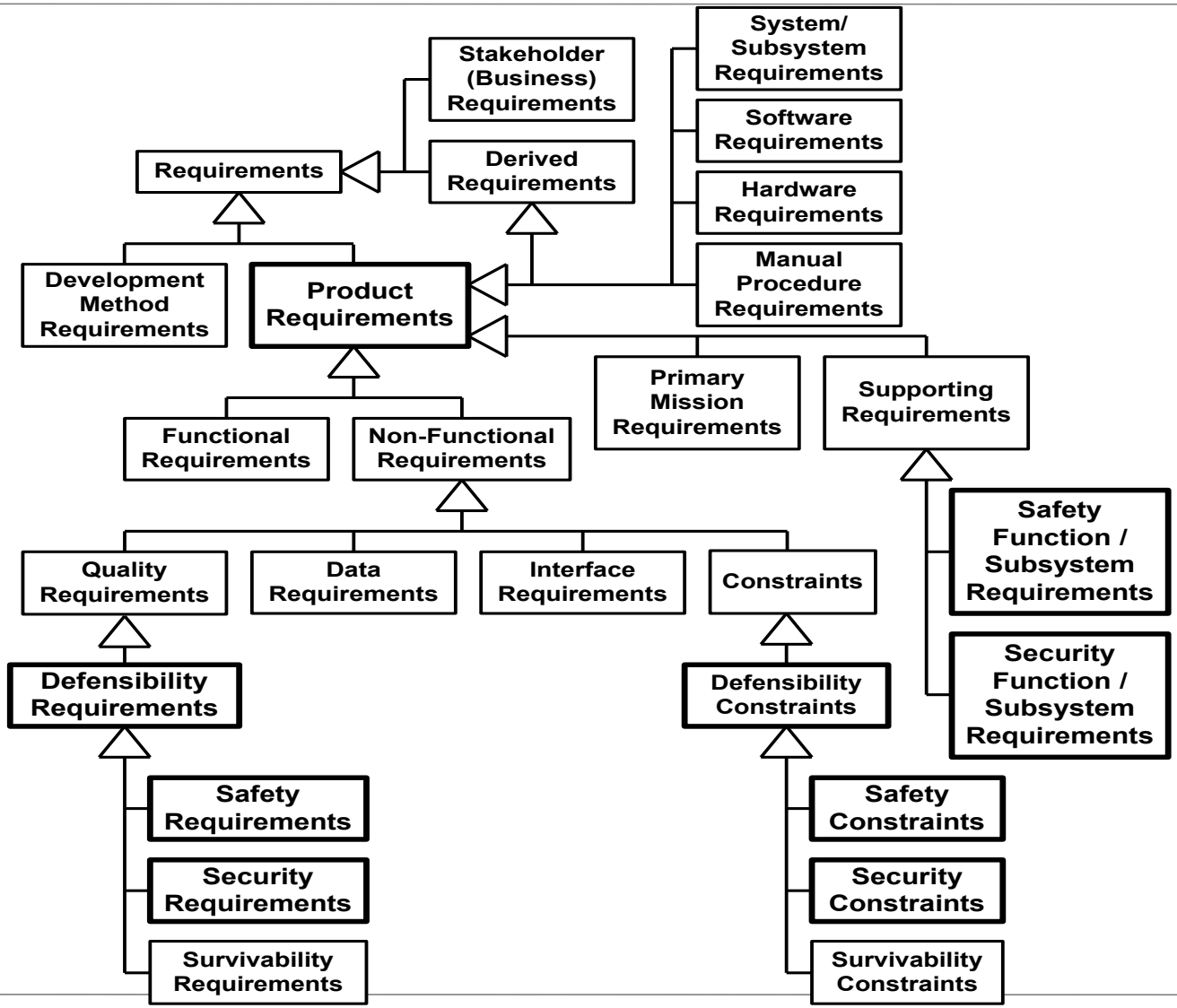
- Specific Types of Non-Functional Requirements (NFRs):
  - Safety and Security Requirements are Quality Requirements are NFRs
- Safety- and Security-Significant Functional, Data, and Interface Requirements
- Architecture and Design Constraints
- Safety and Security Functions/Subsystems
- Software Requirements
- Constraints on Functional Requirements

Reason for Presentation Title

Safety- and Security-Related Requirements for Software-Intensive Systems

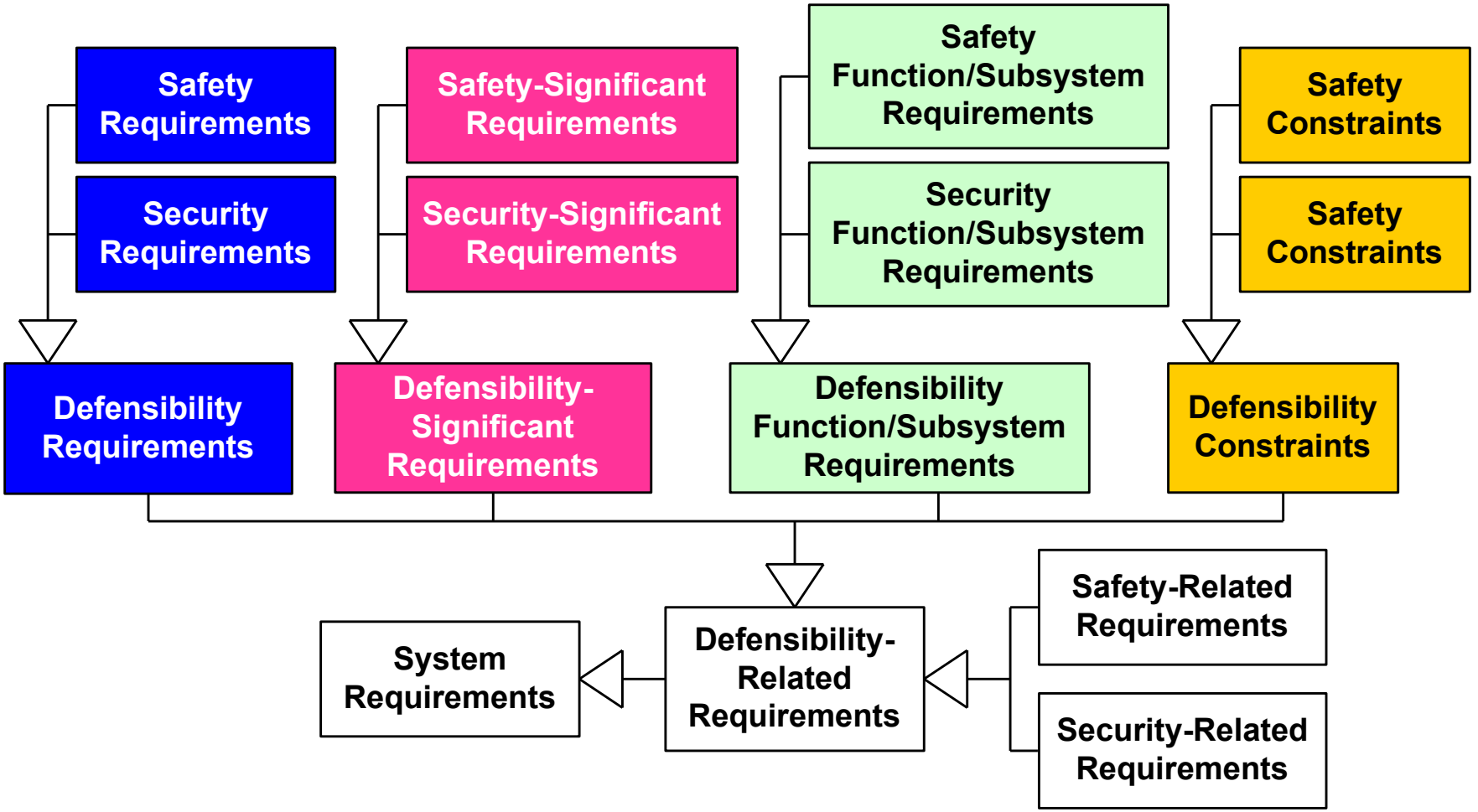
© 2007 Carnegie Mellon University

Types of Safety- and Security-Related Requirements



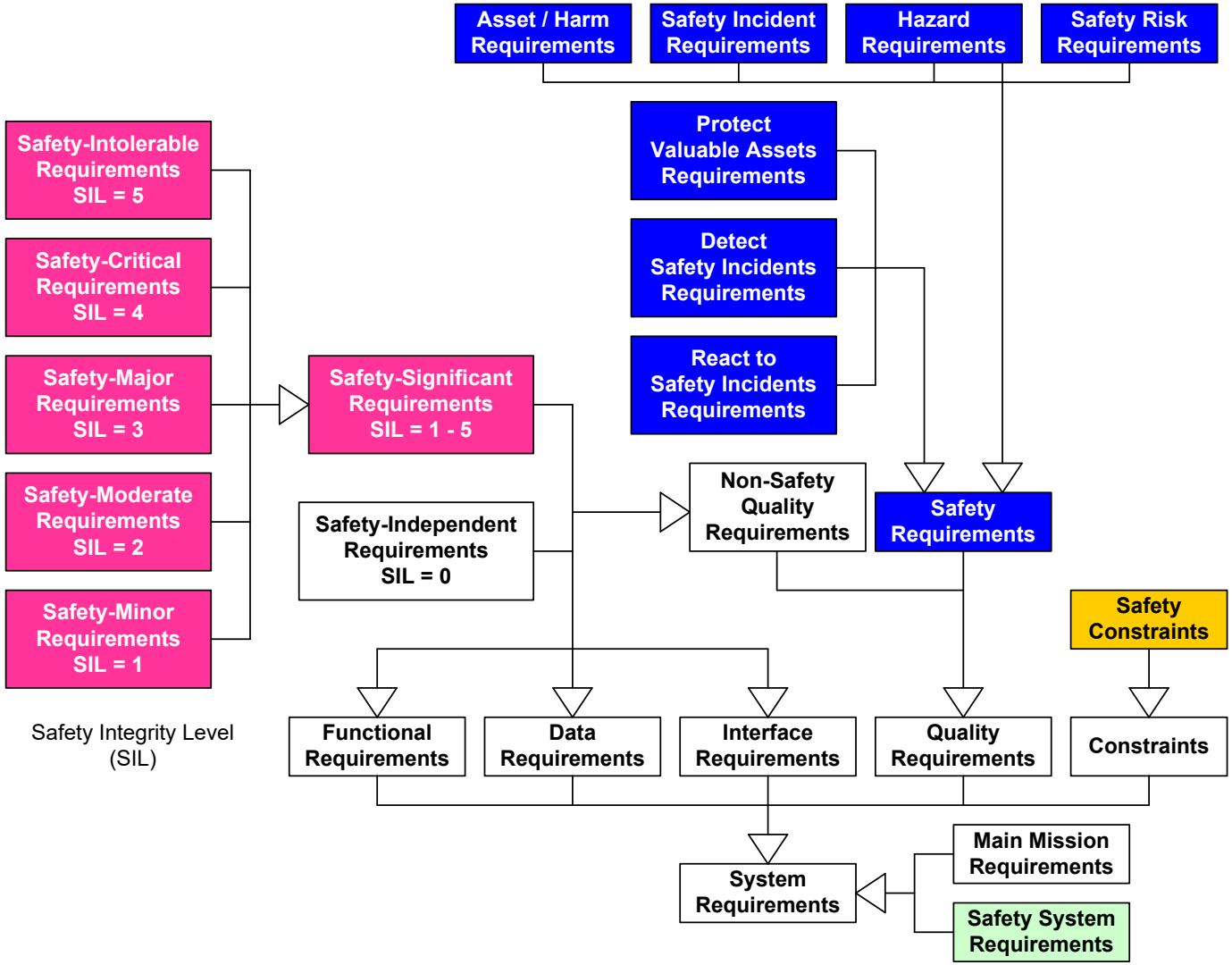
© 2007 Carnegie Mellon University

Types of Defensibility-Related Requirements



© 2007 Carnegie Mellon University

Types of Safety-Related Requirements



© 2007 Carnegie Mellon University

Example Safety- and Security-Related Requirements (Actually Goals – Requirements are more specific)

Safety / Security Requirement (6*4 or 6*7 types each):

- The system shall not cause an average of more than X accidents of harm per year.
- The system shall detect and remove Y% of viruses in input files.

Safety / Security Significant Requirement

- The system shall automatically transport passengers between stations.
- The system shall enable users to update their personal information.

Safety / Security Function / Subsystem Requirement

- The system shall include a fire detection and suppression subsystem.
- The system shall support the encryption/decryption of sensitive data.

Safety / Security Constraint

- The system shall not be composed of any hazardous materials.
- The system shall use passwords for user authentication.

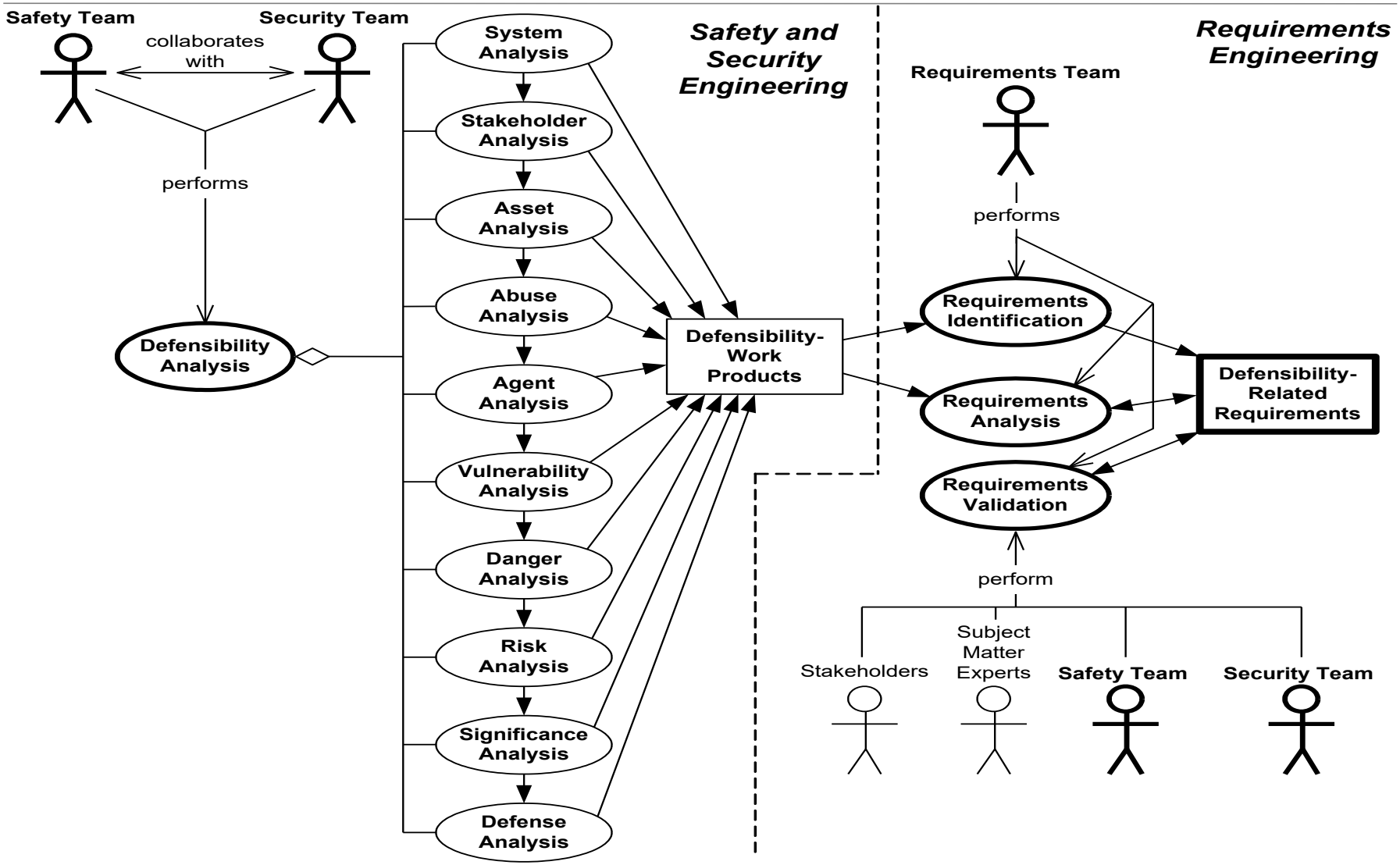
© 2007 Carnegie Mellon University

Common Process:

A Basis for Effective Collaboration

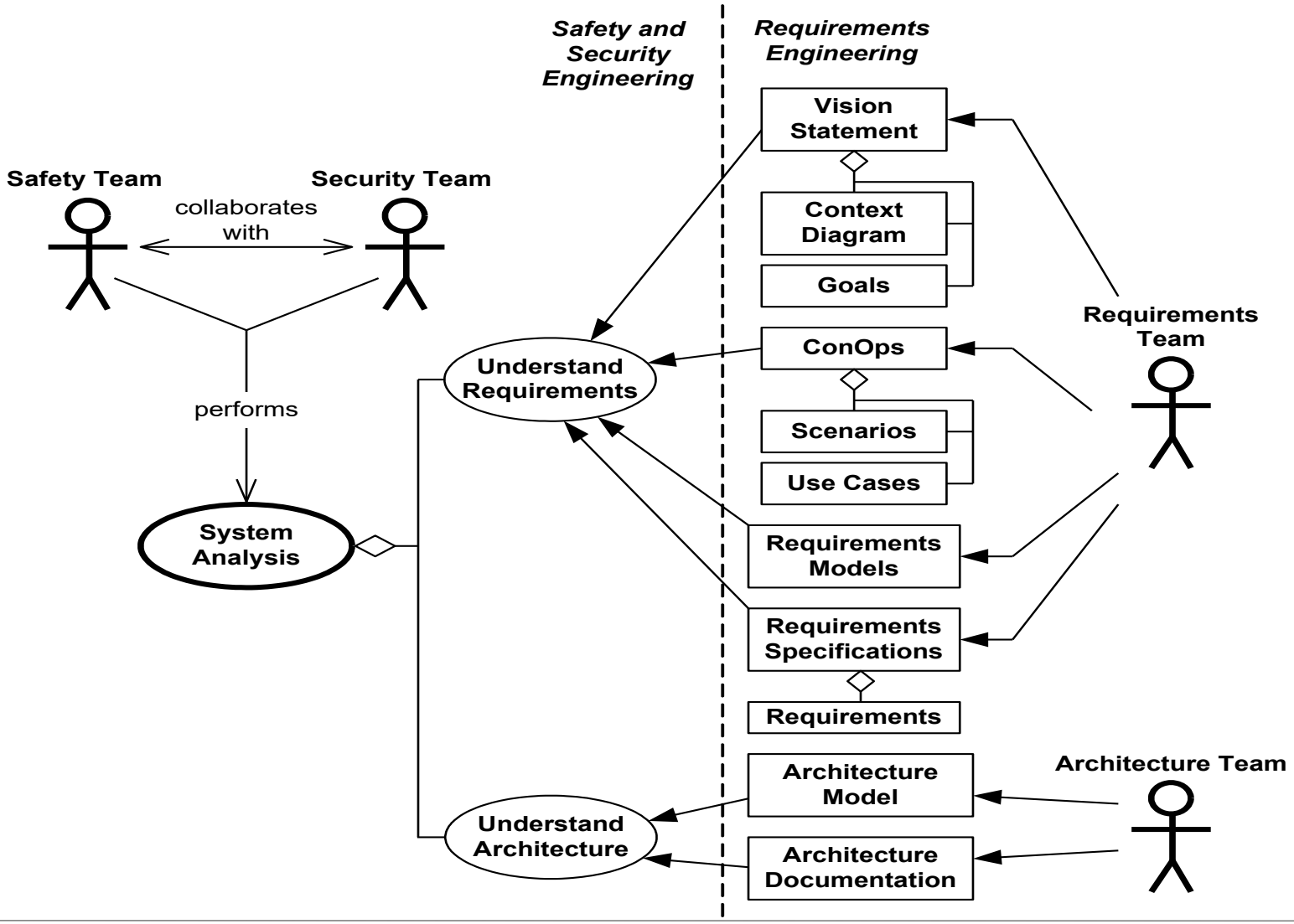
© 2007 Carnegie Mellon University

Defensibility & Requirements Engineering



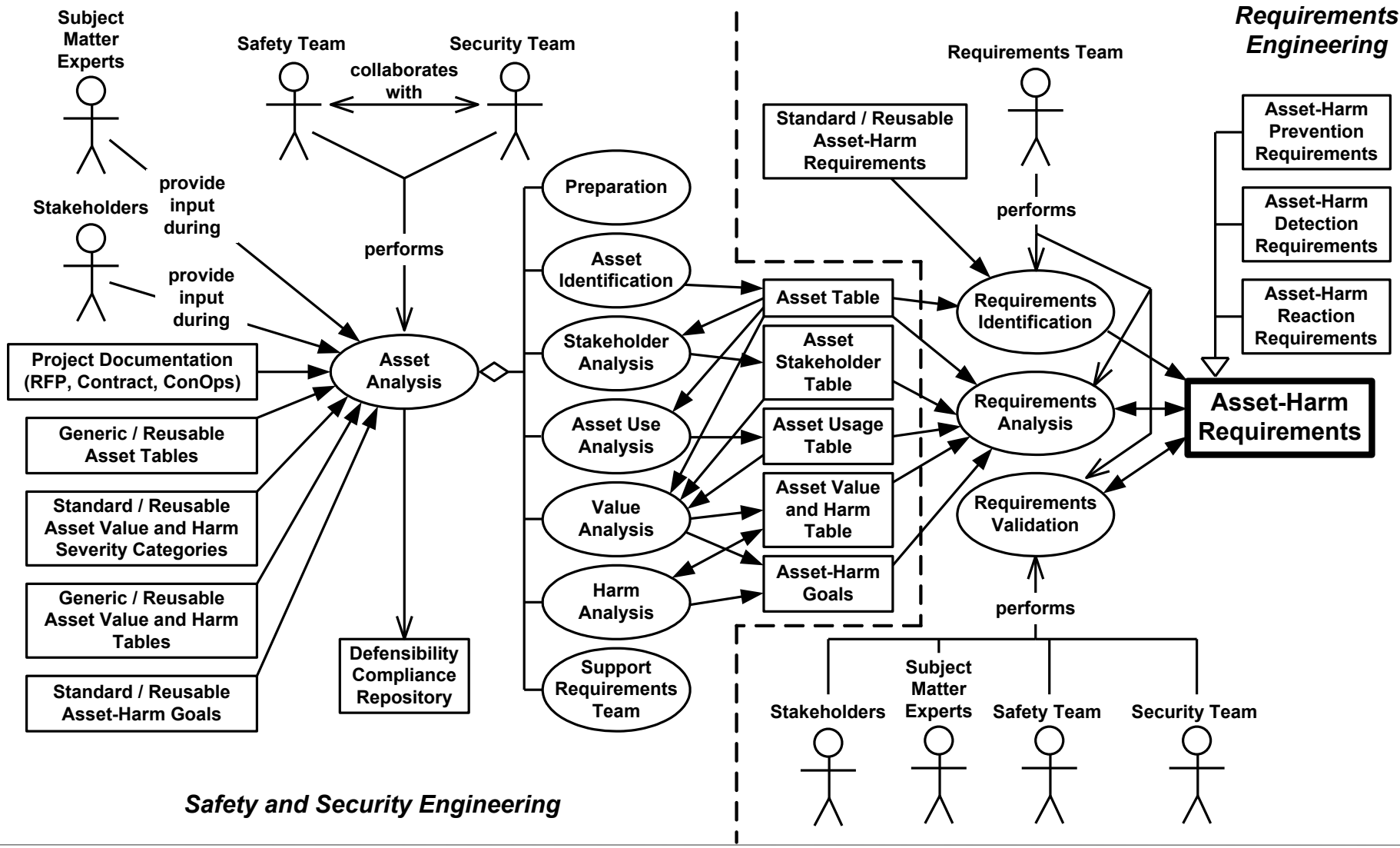
© 2007 Carnegie Mellon University

Systems Analysis



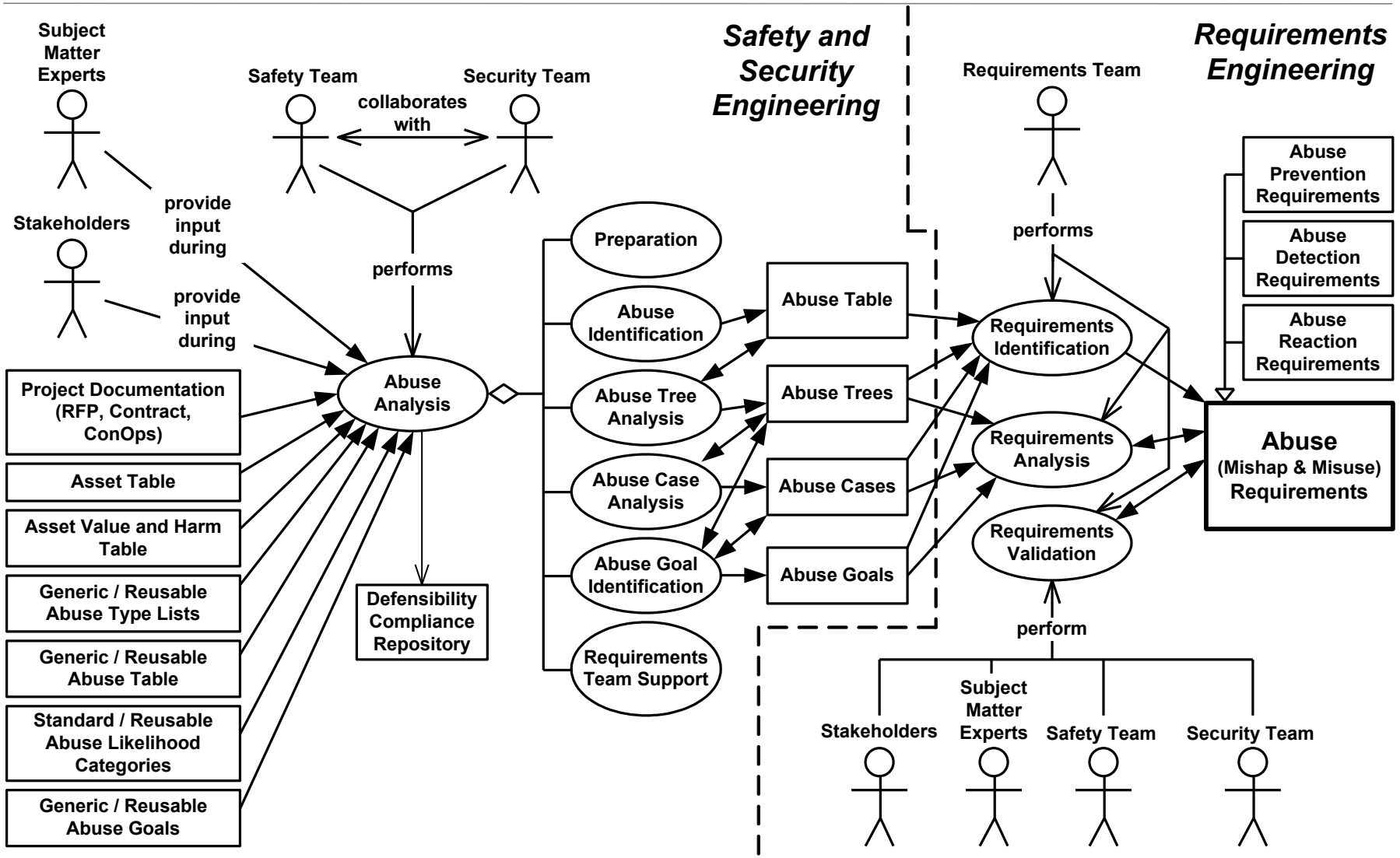
© 2007 Carnegie Mellon University

Asset Analysis



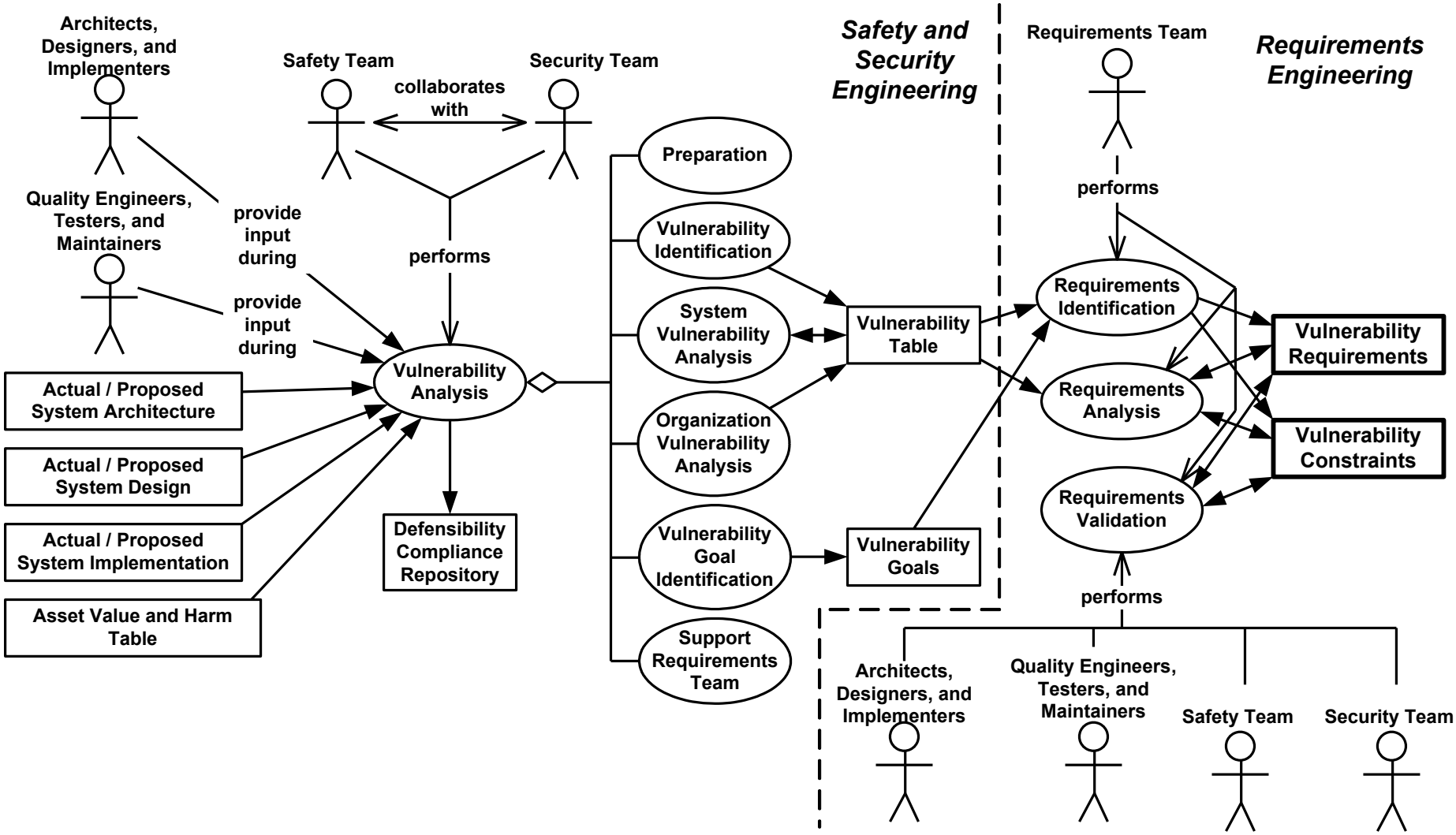
© 2007 Carnegie Mellon University

Abuse (Misuse and Mishap) Analysis



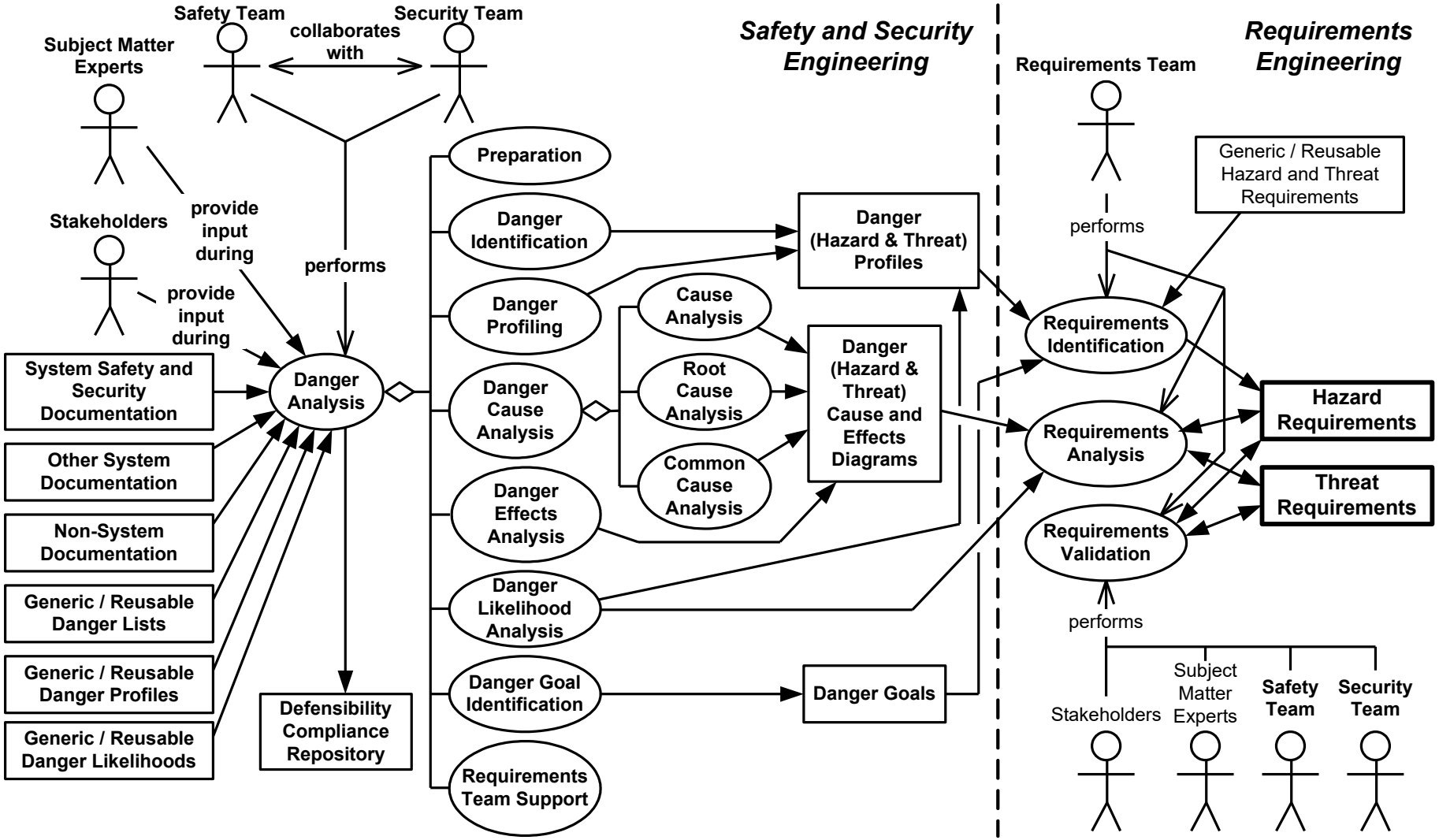
© 2007 Carnegie Mellon University

Vulnerability Analysis



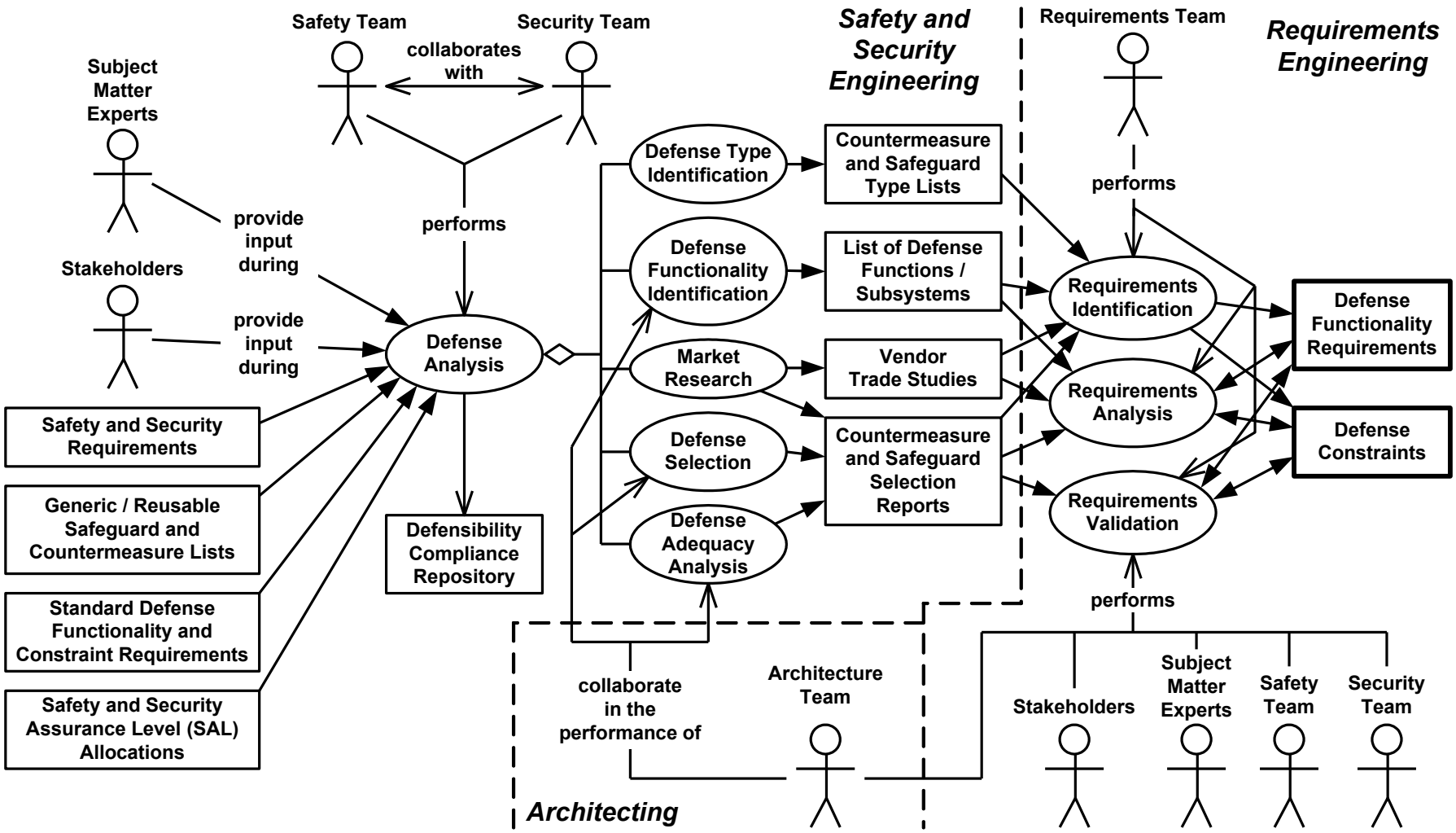
© 2007 Carnegie Mellon University

Danger Analysis



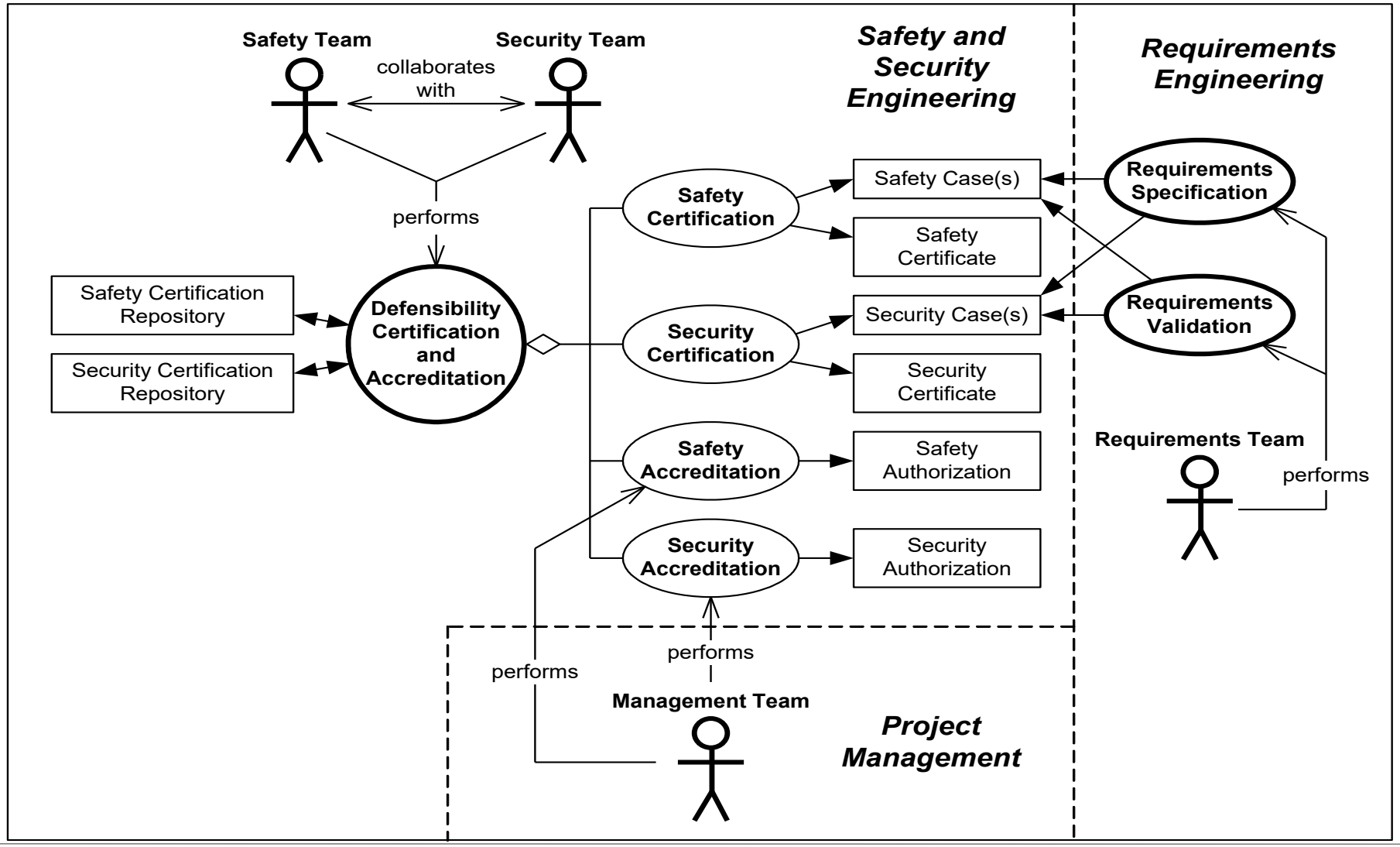
© 2007 Carnegie Mellon University

Defense Analysis



© 2007 Carnegie Mellon University

Defense Certification and Accreditation



© 2007 Carnegie Mellon University

Conclusion:

Process Improvement Recommendations

© 2007 Carnegie Mellon University

Process Improvement Recommendations$_1$

Ensure close Collaboration among Safety, Security, and Requirements Teams.

Better Integrate Safety and Security Processes:

- Concepts and Terminology
- Techniques and Work Products
- Provide Cross Training

Better Integrate Safety and Security Processes with Requirements Process:

- Early during Development Cycle
- Clearly define Team Responsibilities
- Provide Cross Training

Develop all types of Safety- and Security-related Requirements.

Ensure that these Requirements have proper Properties.

© 2007 Carnegie Mellon University

Software Engineering Institute | Carnegie Mellon