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

One-Day Tutorial
32nd International Conference on Software Engineering
4 May 2010

Donald G. Firesmith
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Tutorial Goals

Familiarize requirements, safety, and security engineers with:

- *Common concepts and terminology underlying each other's disciplines*
- *Useful reusable techniques from each other's disciplines*
- Different types of *safety- and security-related requirements*
- A common consistent collaborative *method* for engineering these requirements

Enable requirements, safety, and security teams to better collaborate together to engineer better safety- and security-related requirements

Decrease the incidence of accidents and successful attacks due to poor safety- and security-related requirements

We Are Here

Three Disciplines

Challenges

Common Example

Requirements Engineering Overview

Safety and Security Engineering Overview

Types of Safety- and Security-related Requirements

Collaborative Defensibility Engineering Method

Conclusion

Safety Engineering – Traditional Definition

Safety

freedom from accidental harm to people, property, and the environment

Safety Engineering

the process of ensuring that a system is sufficiently safe to operate

Major Limitations:

- No nontrivial system is free from hazards that can cause accidental harm.
- Not just harm, but also hazards and accidents
- In spite of best efforts, accidents can (and sometimes do) happen.
- It is always a matter of degree and risk management.
- It is important to address not just the prevention of accidental harm (and hazards, accidents, and risks), but also detecting their existence/occurrence, and reacting properly

Security Engineering – Traditional Definitions

Security (of information and services) is often defined in terms of specific properties, whereby a system is secure when it exhibits these properties *in spite of attack*:

- Access Control (including identification, authentication, and authorization)
- Accountability (e.g., non-repudiation of transactions)
- Availability (in spite of attack) – not standard quality characteristic
- Confidentiality (including both privacy and anonymity)
- Integrity (including data and software)

Security Engineering

the process of ensuring that a system is sufficiently secure to operate

Major limitations:

- No nontrivial system is free from threats that can cause malicious harm.
- Not just harm, but also threats and attacks.
- In spite of best efforts, attacks can (and typically will) happen.
- It is always a matter of degree and risk management.
- It is important to address not just the prevention of malicious harm (and threats, attacks, and risks), but also detecting their existence/occurrence, and reacting properly.

Safety and Security Engineering – New Definitions

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 accidental 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 malicious harm, misuses (i.e., attacks and incidents), threats, and security risks

Major Differences (between safety and security):

- Unintentional (accidental) vs. intentional (malicious) harm
- Mishaps vs. misuses
- Hazards vs. threats

Note *system* as opposed to just *software* engineering.

Requirements Engineering

Requirements Engineering

> the engineering discipline within systems/software engineering consisting of the cohesive collection of all *tasks* that are primarily performed to produce the *requirements* and *other related requirements work products* for an *endeavor*

This includes the safety- and security-related requirements.

We Are Here

Three Disciplines

Challenges

Common Example

Requirements Engineering Overview

Safety and Security Engineering Overview

Types of Safety- and Security-related Requirements

Collaborative Defensibility Engineering Method

Conclusion

Challenges$_1$

Requirements engineering, safety engineering, and security engineering have different:

- *Communities*
- *Disciplines* with different education, training, books, journals, and conferences
- *Professions* with different *job titles*
- Fundamental underlying *concepts* and *terminologies*
- *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, deployment, sustainment)

Challenges$_2$

Separation of requirements engineering, safety engineering, and security engineering:

- Causes redundant and uncoordinated work to be performed
- Causes *poor* safety- and security-related requirements that are often:
  - Vague, unverifiable, unfeasible, architectural and design constraints
  - Capabilities or goals rather than requirements
  - Inadequate and too late to drive architecture and testing
- Makes it unnecessarily harder to achieve certification and accreditation

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
- Large number of defects 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.

Poor requirements result in:

- Vulnerabilities within the system
- Dangers (hazards and threats)
- Abuses (mishaps and misuses)

Challenges$_4$

Most mandated security "requirements" are actually constraints such as:

- Security functions or subsystems
- Industry "best practices"

How much safe and secure is *sufficient*?

Traditional hazard analysis techniques are inadequate:

- Based on component *reliability*:
  - Assume that accidents are caused by component failures
  - Based on fault trees, event trees, and FMECA tables
  - Inadequate for software and human error (not just "user error")
- Many accidents caused by:
  - Incorrect interactions among "correct" and "reliable" components

Need new models and techniques that better address software and human issues

We Are Here

Three Disciplines

Challenges

Common Example

Requirements Engineering Overview

Safety and Security Engineering Overview

Types of Safety- and Security-related Requirements

Collaborative Defensibility Engineering Method

Conclusion

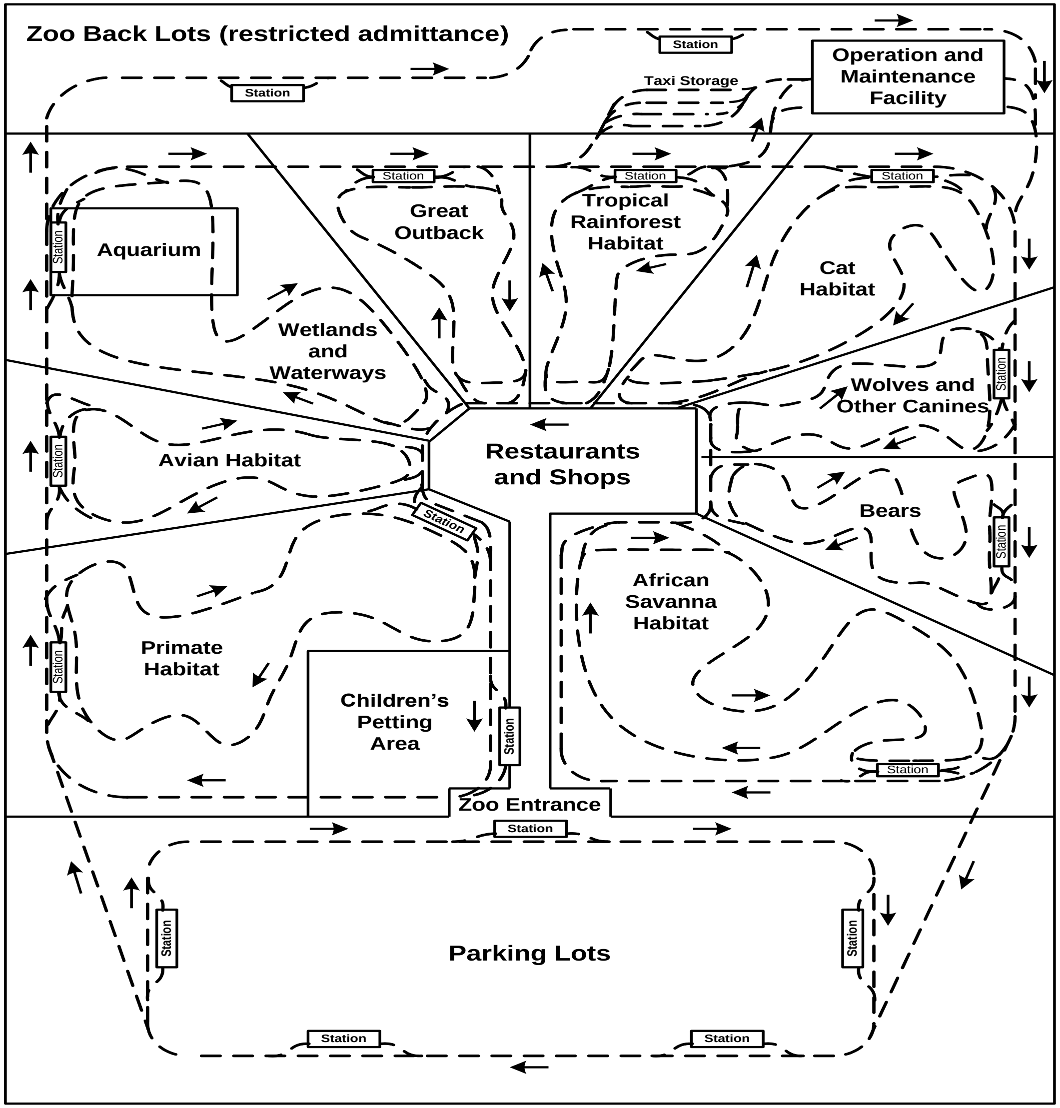
Common Example use throughout Tutorial

Problem: How to enable large numbers of zoo patrons to quickly and conveniently tour the habitats of a huge new zoo?

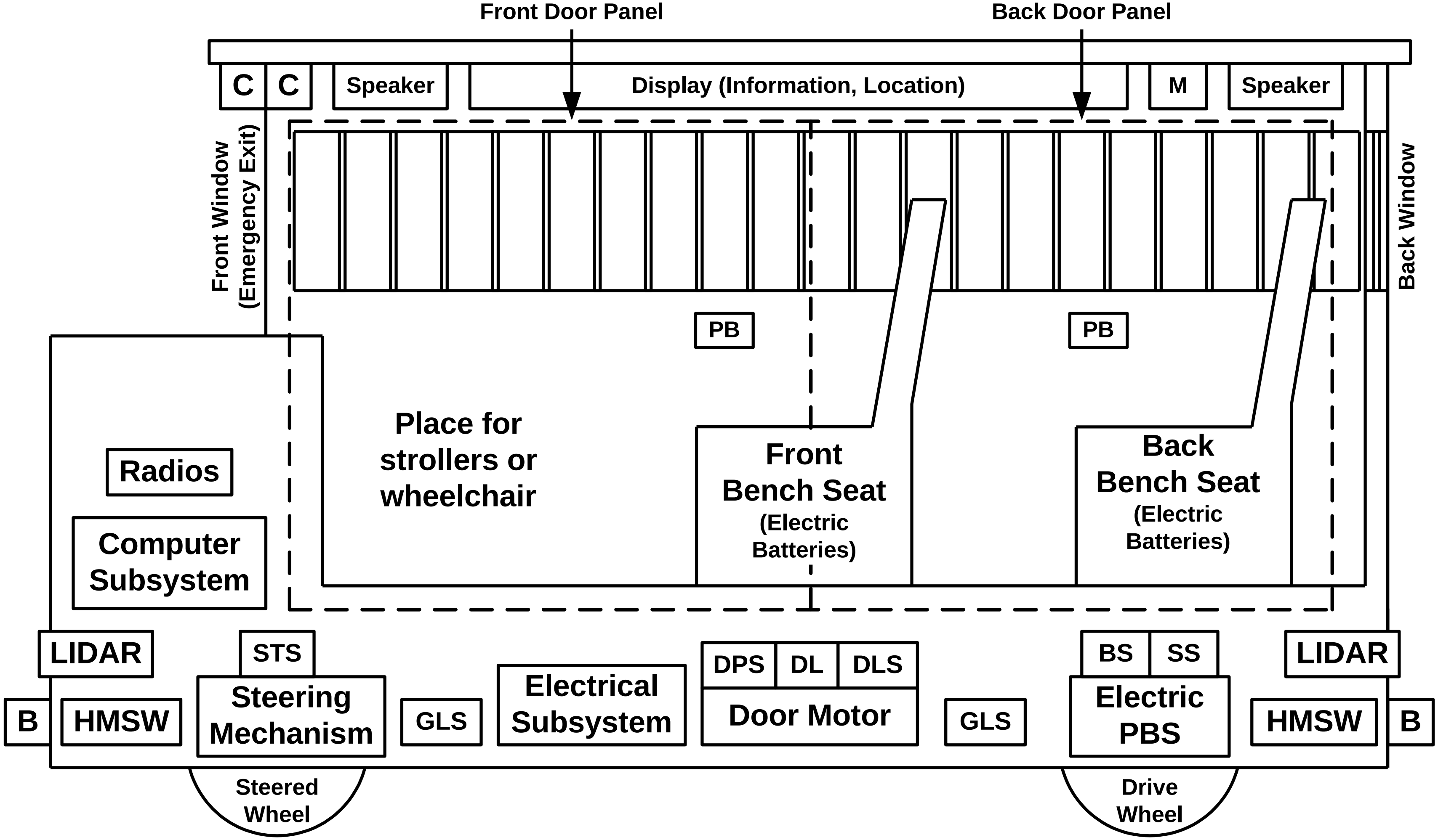
Proposed solution: the Zoo Automated Taxi System (ZATS)

- Numerous small family-sized automated **taxis**:
  - Leisurely tours of habitats and fast transport between habitats
  - Inexpensive to operate (no driver salaries and benefits)
  - Reliable, easy to use, safe, and secure
  - Green (electric to minimize air and noise pollution)
- Elevated concrete **guideways**:
  - Separate passengers from animals
  - Provide good views
  - Avoid collisions of taxis with pedestrians and vehicles in parking lot
- Conveniently located **taxi stations**
- **Operations and maintenance facility** in zoo back lots

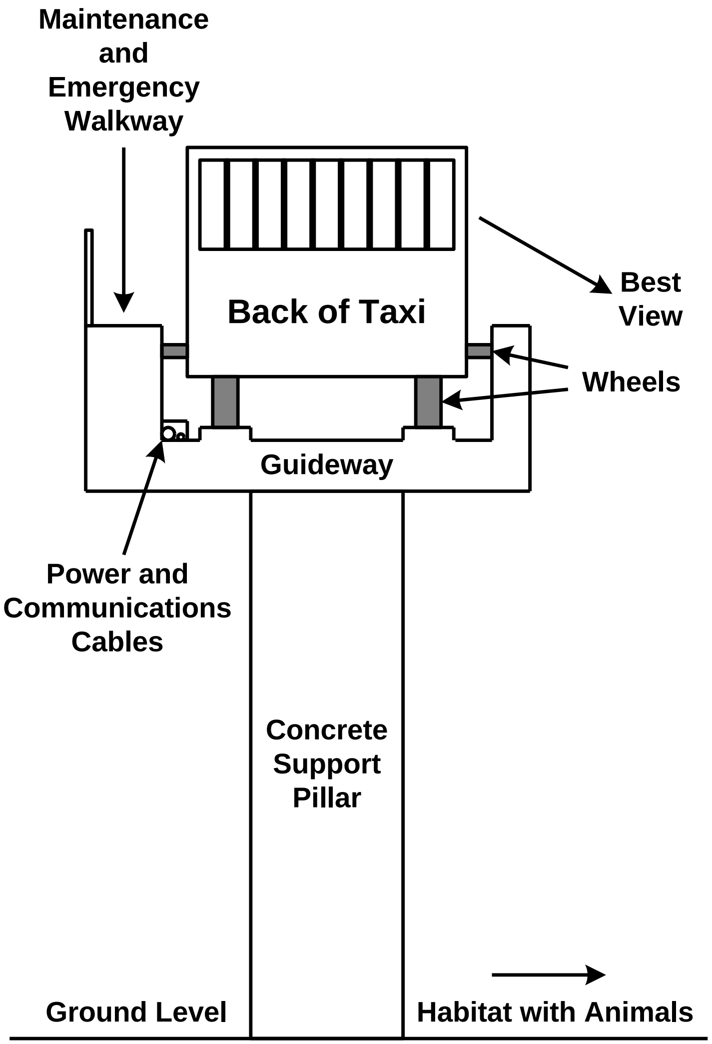
Proposed Guideway Layout



Proposed Taxi



Automated Taxis On Elevated Guideways



We Are Here

Three Disciplines

Challenges

Common Example

Requirements Engineering Overview

Safety and Security Engineering Overview

Types of Safety- and Security-related Requirements

Collaborative Defensibility Engineering Method

Conclusion

Requirements Engineering Tasks

Business Analysis (i.e., Customer, Competitor, Market, Technology, and User Analysis as well as Stakeholder Identification and Profiling)

Visioning

Requirements Identification (a.k.a., Elicitation)

Requirements Reuse

Requirements Prototyping

Requirements Analysis

Requirements Specification

Requirements Management

Requirements Validation

Scope Management (Management)

Change Control (Configuration Management)

Quality Control (Quality Engineering)

Requirements Engineering Work Products

Business analyses

Stakeholder profiles

Vision statement

- **Capabilities and Goals**

Concept of Operations (ConOps) or operational concept document (OCD)

- **Use cases and usage scenarios (some requirements models)**

Requirements repository and published specifications

- **Actual requirements**

Requirements prototypes

Domain model

Glossary

Difficulty of Requirements Engineering

"The hardest single part of building a software system is deciding precisely what to build. No other part of the conceptual work is as difficult as establishing the detailed technical requirements, including all the interfaces to people, to machines, and to other software systems. No other part of the work so cripples the resulting system if done wrong. No other part is more difficult to rectify later."

F. Brooks, *No Silver Bullet*, IEEE Computer, 1987

Goals

Goal

an *informally documented perceived need* of a *legitimate stakeholder*

Goals are:

- *Not* requirements.
- Drive the identification and analysis of the requirements.
- Typically ambiguous and/or unrealistic (i.e. impossible to guarantee).

Major problems with safety and security goals:

- Ambiguous
- Stated as absolutes
- Not 100% feasible
- Not verifiable

Typically documented in a vision statement.

Representative ZATS Goals

ZATS will take passengers on leisurely tours that provide excellent viewing of the zoo habitats.

ZATS will take passengers rapidly to their desired taxi stations in the zoo and its parking lot.

ZATS will prevent animals from reaching passengers in taxis and taxi stations.

ZATS will never injure or kill a passenger.

ZATS will not cause air and noise pollution.

ZATS will be easy and intuitive for all of its passengers to use.

ZATS will allow passengers to use securely use bank cards to pay for trips.

Use Case, Use Case Path, and Usage Scenario

Usage Scenario

a *specific* functionally cohesive sequence of interactions between user(s), the system, and potentially other actors that provides value to a stakeholder

Use Case

a general way to perform a function

a functionally cohesive class of usage scenarios

Use Case Path (a.k.a., flow and course)

an equivalence set of usage scenarios that follow the same course through a use case

Use Case, Use Case Path, and Usage Scenario

Use case paths:

- Can be either:
  - Normal (Sunny Day or Happy Path)
  - Exceptional (Rainy Day)
- Should have their own preconditions, triggers, and postconditions
- Are often documented with text, sequence diagrams, or activity diagrams

Use cases, use case paths, and usage scenarios:

- Typically documented in a ConOps or operational concept document (OCD)
- Drive the identification and analysis of the [primarily functional] requirements
- Often include potential architectural and design information

Use cases are far more than merely use case diagrams.

Characteristics of Good Requirements

All Types of Requirements

Correct

Cohesive (individual)

Complete

Concise

Feasible

Mandatory (necessary)

Normal and Exceptional

Relevant

Unique

Unambiguous

Validatable

Verifiable

What or how well, not how

Active Voice

Configuration Controlled

Consistent (internally and with other requirements)

Differentiated from Non-requirements

Externally observable

Grammatically Correct and no Typos

Managed (in requirements repository)

Prioritized (for scheduling implementation)

Properly Specified

Rationalized

Scheduled

Stakeholder-centric

Situation-specific (to mode, state, and/or event)

Traced (to source and to architecture)

Uniquely identified

Usable by stakeholders

http://www.jot.fm/issues/issue_2003_07/column7

Poor Requirements Cause Accidents$_1$

"For the 34 (safety) incidents analyzed, 44% had inadequate specification as their primary cause."

> Health and Safety Executive (HSE), *Out of Control: Why Control Systems Go Wrong and How to Prevent Failure* (2nd Edition), 1995

"Almost all accidents related to software components in the past 20 years can be traced to flaws in the requirements specifications, such as unhandled cases."

> Grady Lee, Jeffry Howard, and Patrick Anderson, "Safety-Critical Requirements Specification and Analysis using SpecTRM", *Safeware Engineering*, 2002

Poor Requirements Cause Accidents$_2$

"Erroneous specification is a major source of defects and subsequent failure of safety-critical systems. Many failures occur in systems using software that is perfect, it is just not the software that is needed because the specification is defective."

> John C. McKnight, "Software Challenges in Aviation Systems," *Proceedings of the 21st International Conference on Computer Safety, Reliability and Security*, 2002

"Software-related accidents almost always are due to misunderstandings about what the software should do."

> Kathryn Anne Weiss, "An Analysis of Causation in Aerospace Accidents," *Proceedings of the 2001 Digital Avionics Systems Conference*, updated 7 September 2004

Poor Requirements Cause Accidents$_3$

"Software-related accidents are **usually caused** by flawed requirements. Incomplete or wrong assumptions about the operation of the controlled system can cause software related accidents, as can incomplete or wrong assumptions about the required operation of the computer. Frequently, omitted requirements leave unhandled controlled-system states and environmental conditions."

Nancy G. Leveson, *Safeware: System Safety and Computers*, 2003

"Software-related accidents are **almost all caused** by flawed requirements:

- Incomplete or wrong assumptions about the operation of the controlled system or required operation of the computer
- Unhandled controlled-system states and environmental conditions."

Nancy G. Leveson, *A new Approach to Ensuring Safety in Software and Human Intensive Systems*, July 2009

On the Other Hand

Most accidents and successful attacks have *multiple*, sometimes-independent causes (so that there may be no single "root" cause):

- Hardware causes
- Human (e.g., management and developer) causes (not just operator error)
- Manufacturing and maintenance cause
- Software causes

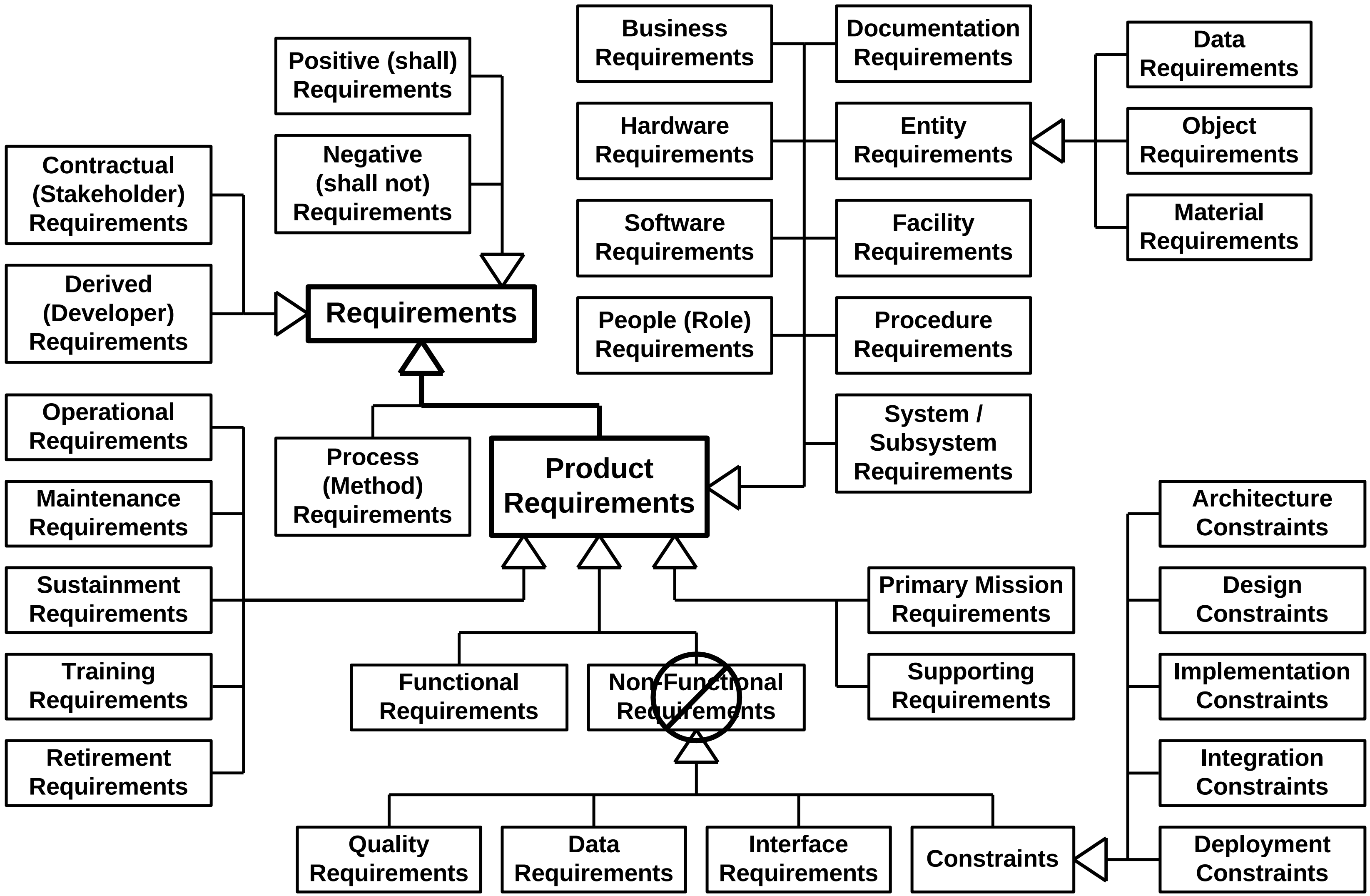
Some requirements defects are due to:

- Poor management decisions including inadequate:
  - Schedule and funding
  - Quality control
  - Training
- Inadequate requirements techniques

Even better requirements does not guarantee that these requirements will be properly implemented.

Fault tolerance may mitigate many requirements defects, but may also mask them unless adequate fault logging, reporting, and analysis is performed.

Types of Requirements

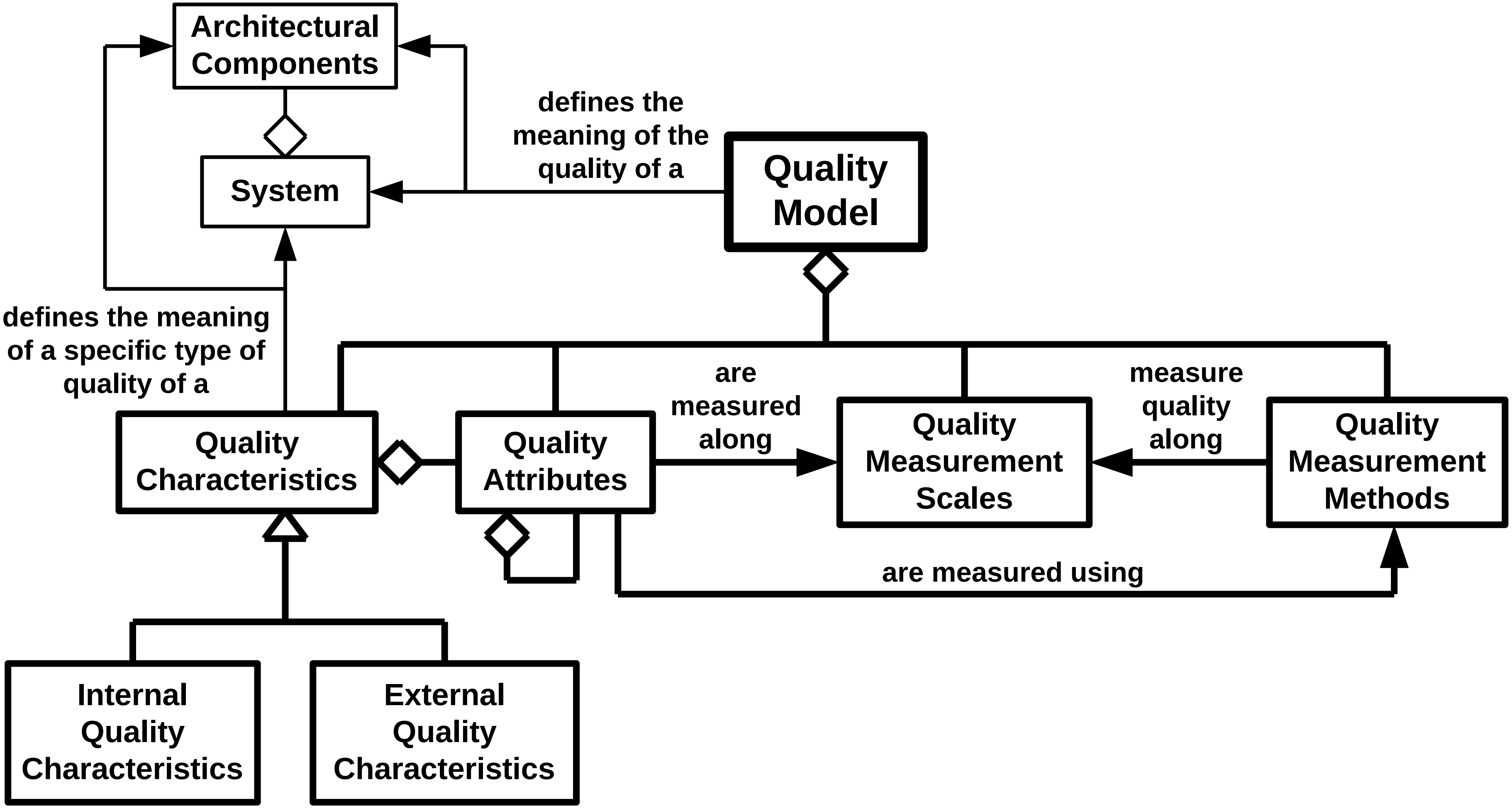


Product Requirements

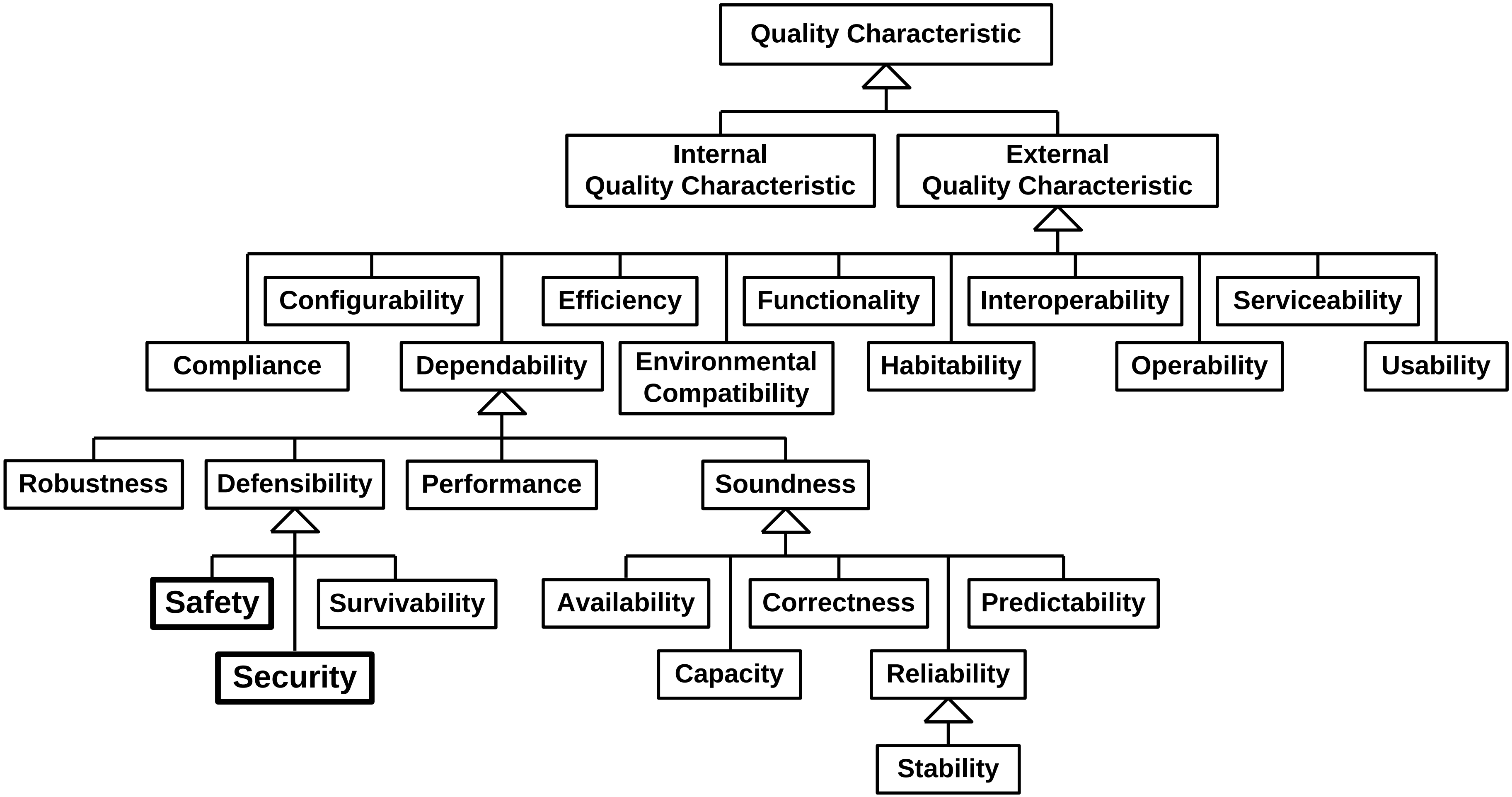
A **product requirement** is a requirement for a *product* (e.g., system, subsystem, software application, or component).

- A **functional requirement** is a product requirement than specifies a mandatory function (i.e., behavior) of the product.
- A **data requirement** is a product requirement that specifies mandatory [types of] data that must be manipulated by the product.
- An **interface requirement** is a product requirement that specifies a mandatory interface with (or within) the product.
- A **quality requirement** is a product requirement that specifies a mandatory amount of a type of product quality (characteristic or attribute).
- A **constraint** is a property of the product (e.g., architecture or design decision) that would ordinarily not be a requirement but which is being mandated as if it were a normal requirement

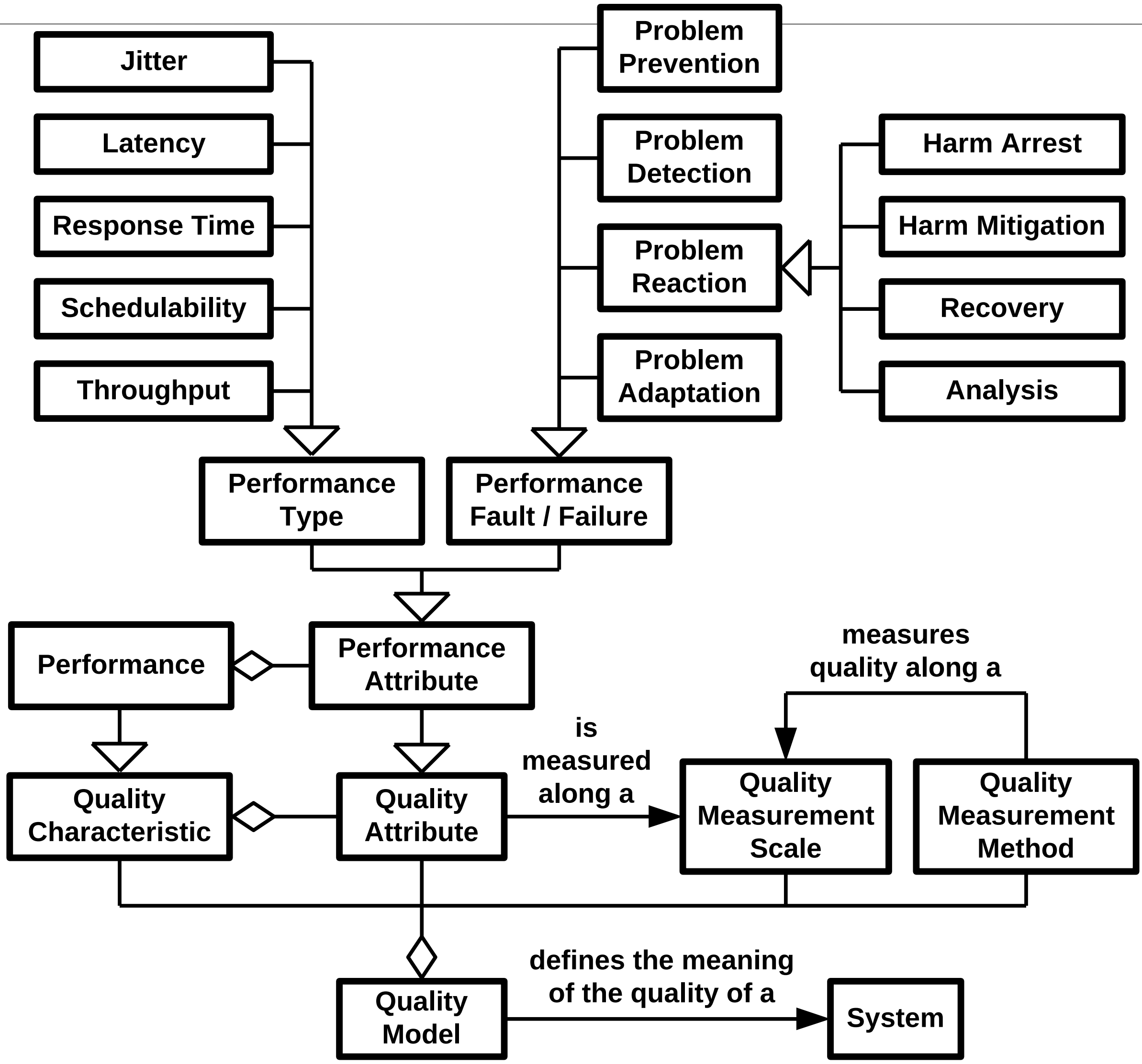
Quality Model



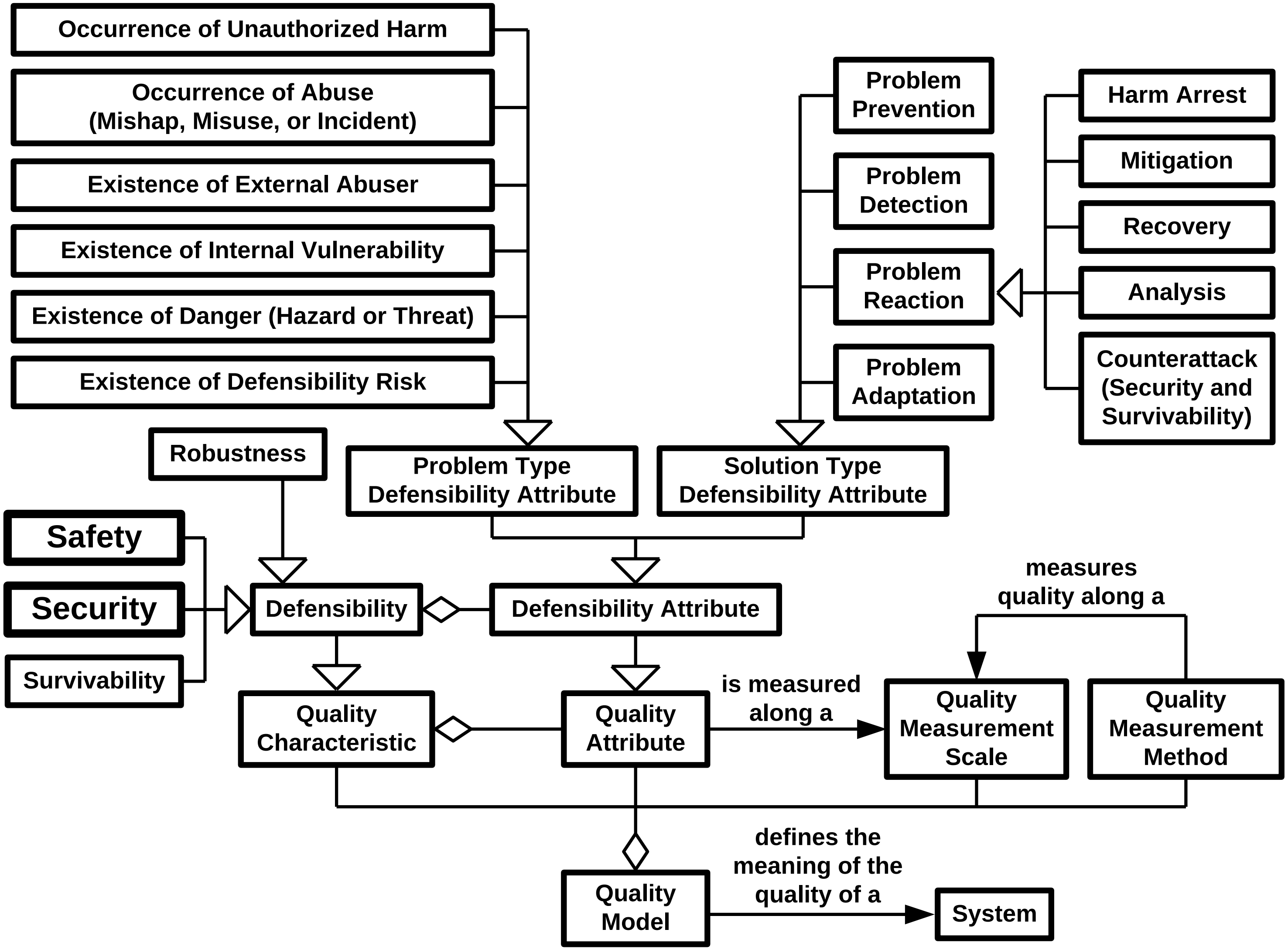
Quality Characteristics (External)



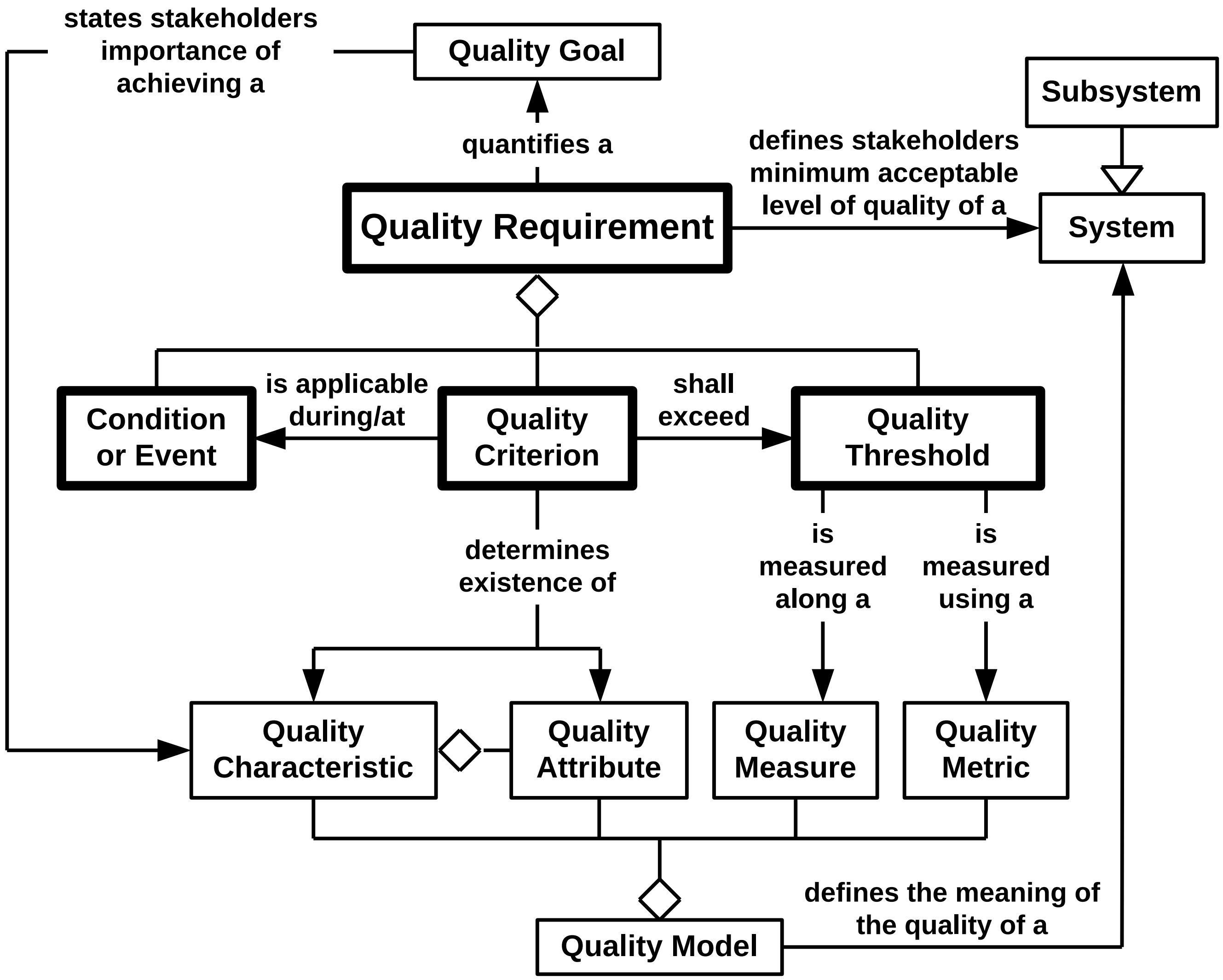
Performance Attributes



Defensibility Quality Attributes



Components of a Quality Requirement



Example Quality Requirement

Hazard prevention safety requirement:
"Under normal operating conditions, ZATS shall not move when it's doors are open at an average rate higher than once every 10,000 trips."

Component parts:

- **Condition:**
  "Under normal operating conditions"
- **Mandatory system-specific quality criterion:**
  "ZATS shall not *move* when it's doors are *open*"
- **Measurement threshold:**
  "at an average rate higher than once every 10,000 *trips*."

Definitions needed to avoid ambiguity:

- Moving – traveling faster than 0.1 cm per second
- Open – open more than 1 cm between doors
- Normal operating conditions – neither during maintenance nor a fire
- Trip – travel with passengers from a starting taxi station to the associated destination taxi station

Importance of Measurement Threshold

Measurement threshold is:

- Critical
- Difficult (but not impossible) to determine
- Often left out of quality requirements
- Needed to avoid ambiguity

States *how much* quality is necessary (sufficient)

Enables architects and architecture evaluators to:

- Determine if architecture is adequate
- Make engineering tradeoffs between competing quality characteristics and attributes

Enables tester to determine the:

- Test completion criteria
- Number and types of test cases

We Are Here

Three Disciplines

Challenges

Common Example

Requirements Engineering Overview

Safety and Security Engineering Overview

Types of Safety- and Security-related Requirements

Collaborative Defensibility Engineering Method

Conclusion

Fundamental Safety and Security Concepts

Safety and security as quality characteristics with associated quality attributes

Stakeholders

Valuable assets

Unauthorized harm to valuable assets

Abuses (accidents, attacks, and incidents)

Vulnerabilities (system-internal weaknesses or defects)

Abusers (external and internal, malicious and non-malicious)

Dangers (hazards and threats)

Defensibility risks (safety and security)

Goals, policies, and requirements

Defenses (safeguards and counter measures)

Safety as a Quality Characteristic

Safety is the subclass of defensibility capturing the *degree* to which:

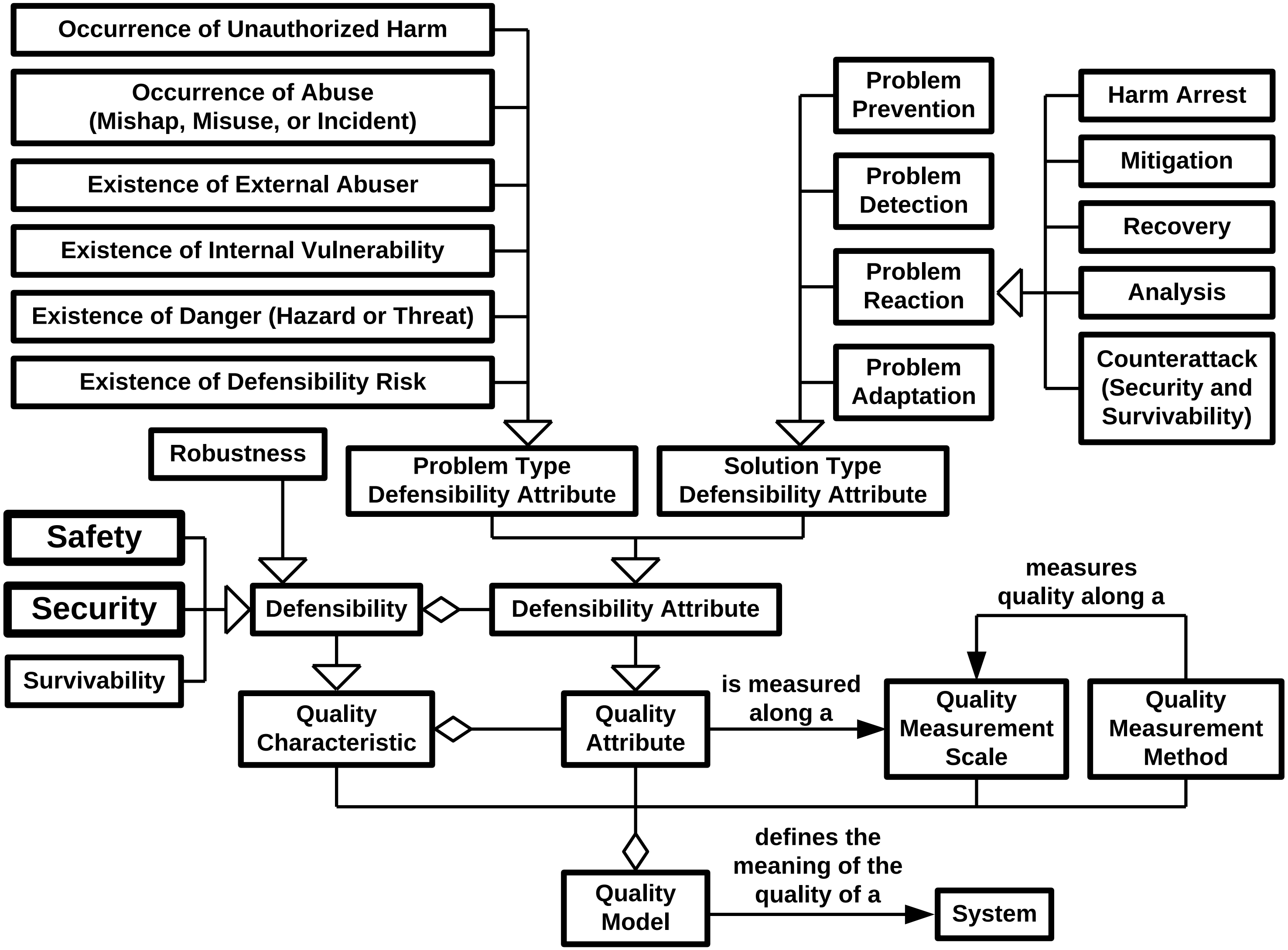
- *The following safety problems:*
  - *Accidental harm* to valuable assets
  - *Safety abuses (mishaps such as accidents and safety incidents)*
  - *Safety abusers (people, systems, and the environment)*
  - *Safety vulnerabilities*
  - *Safety dangers (hazards*) including the existence (conditions) of non-malicious abusers who unintentionally exploit system vulnerabilities to accidentally harm vulnerable valuable assets
  - *Safety risks*
- Have safety solutions:
  - *Prevented (eliminated, mitigated, keep acceptably low)*
  - *Detected*
  - *Reacted to*
  - *Adapted to*

Security as a Quality Characteristic

Security is the subclass of defensibility capturing the *degree* to which:

- *The following security problems:*
  - *Malicious harm* to valuable assets
  - *Security abuses (misuses such as attacks and security incidents)*
  - *Security abusers (attackers and malware – systems, software, and hardware)*
  - *Security vulnerabilities*
  - *Security dangers (threats*) including the existence (conditions) of malicious abusers who can exploit system vulnerabilities to harm vulnerable valuable assets
  - *Security risks*
- Are security solutions:
  - *Prevented (eliminated, mitigated, keep acceptably low)*
  - *Detected*
  - *Reacted to*
  - *Adapted to*

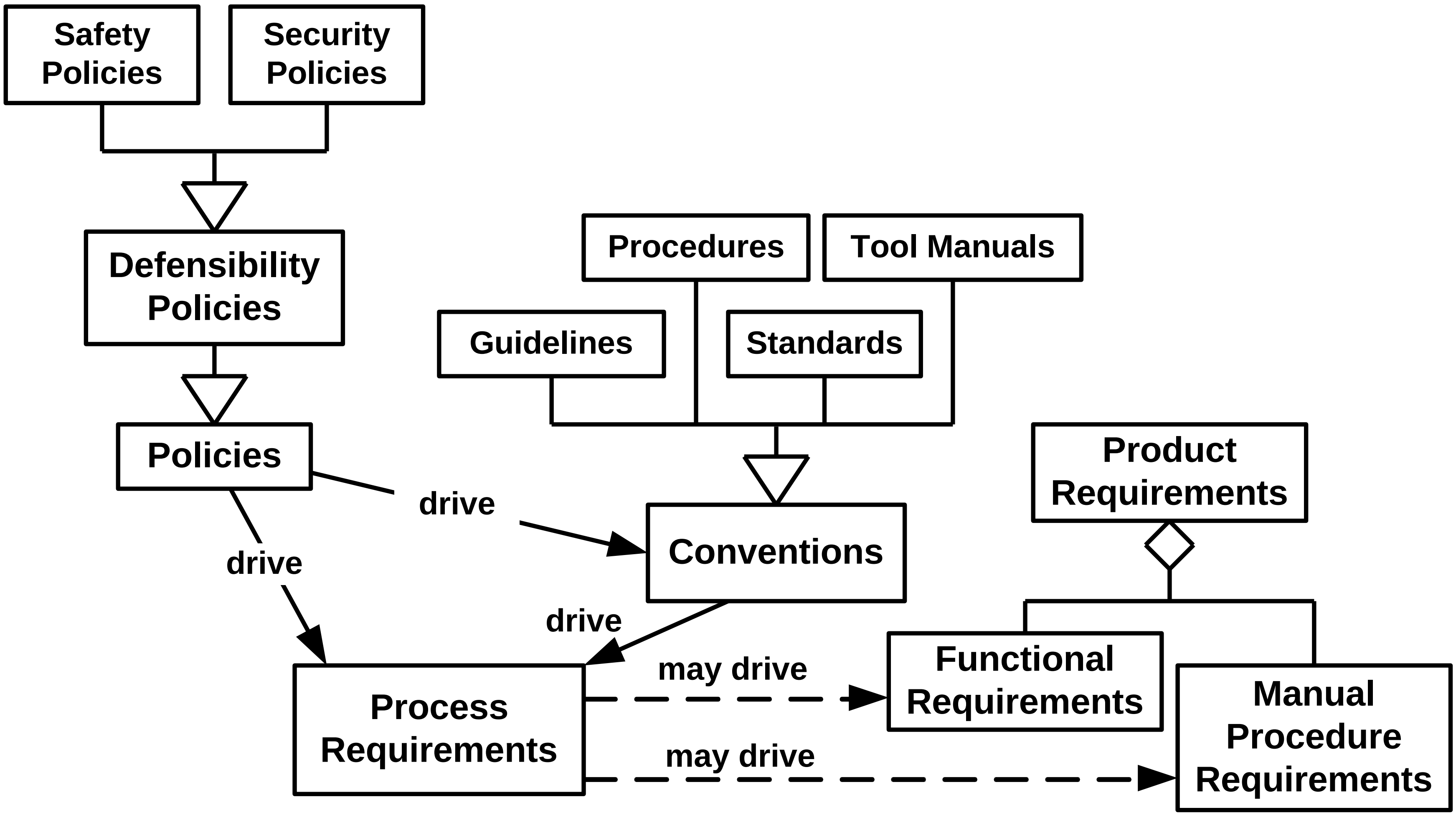
Defensibility Quality Attributes



Different Types of Defensibility Requirements

	Unauthorized Harm	Abuse	Abuser	Vulnerability	Danger	Defensibility Risk
Prevention (current)	Prevent Occurrence of Unauthorized Harm	Prevent Occurrence of Abuse	Prevent Abuser Means or Opportunity	Prevent Existence of Vulnerability	Prevent Existence of Danger	Prevent Existence of Defensibility Risk
Detection (current)	Detect Occurrence of Unauthorized Harm	Detect Occurrence of Abuse	Detect Existence of Abuser	Detect Existence of Vulnerability	Detect Existence of Danger	Detect Existence of Defensibility Risk
Reaction (current)	React to Occurrence of Unauthorized Harm	React to Occurrence of Abuse	React to Existence of Abuser	React to Existence of Vulnerability	React to Existence of Danger	React to Existence of Defensibility Risk
Adaptation (future)	Adapt due to Unauthorized Harm	Adapt to Future Occurrence of Abuse	Adapt to Future Existence of Abusers	Adapt to Future Existence of Vulnerability	Adapt to Future Existence of Danger	Adapt due to Existence of Defensibility Risk

Safety and Security Policies and Conventions



Safety and Security Policies

Policy

a strategic *process* mandate that establishes a desired goal

Defensibility Policy

a policy that enables the achievement of one or more *safety* or *security goals*

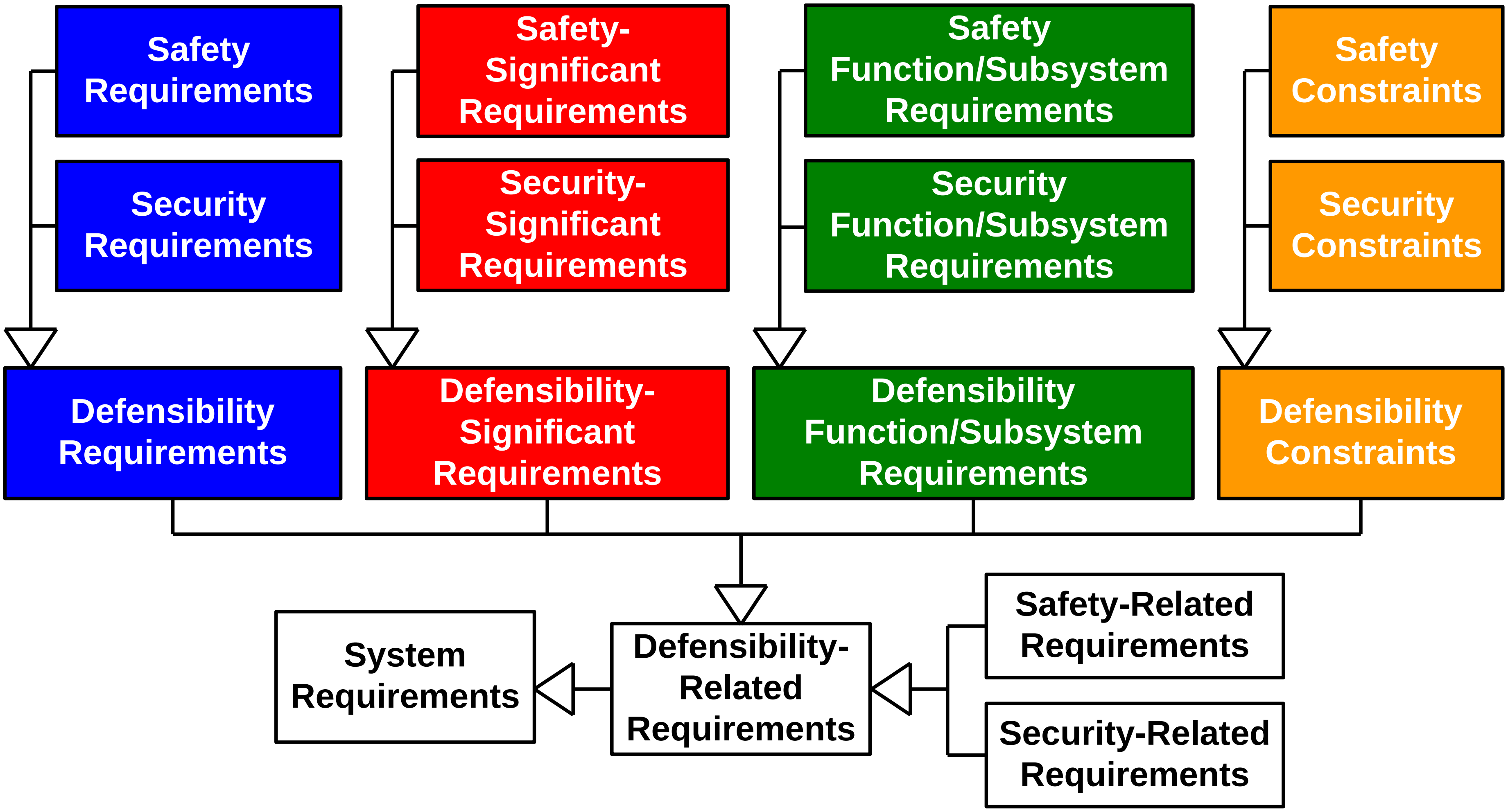
Examples

- “The overall responsibility for safety must be identified and communicated to all stakeholders.”
- “A preliminary hazard analysis shall be performed during early in the project.”
- “All users will have security training.”

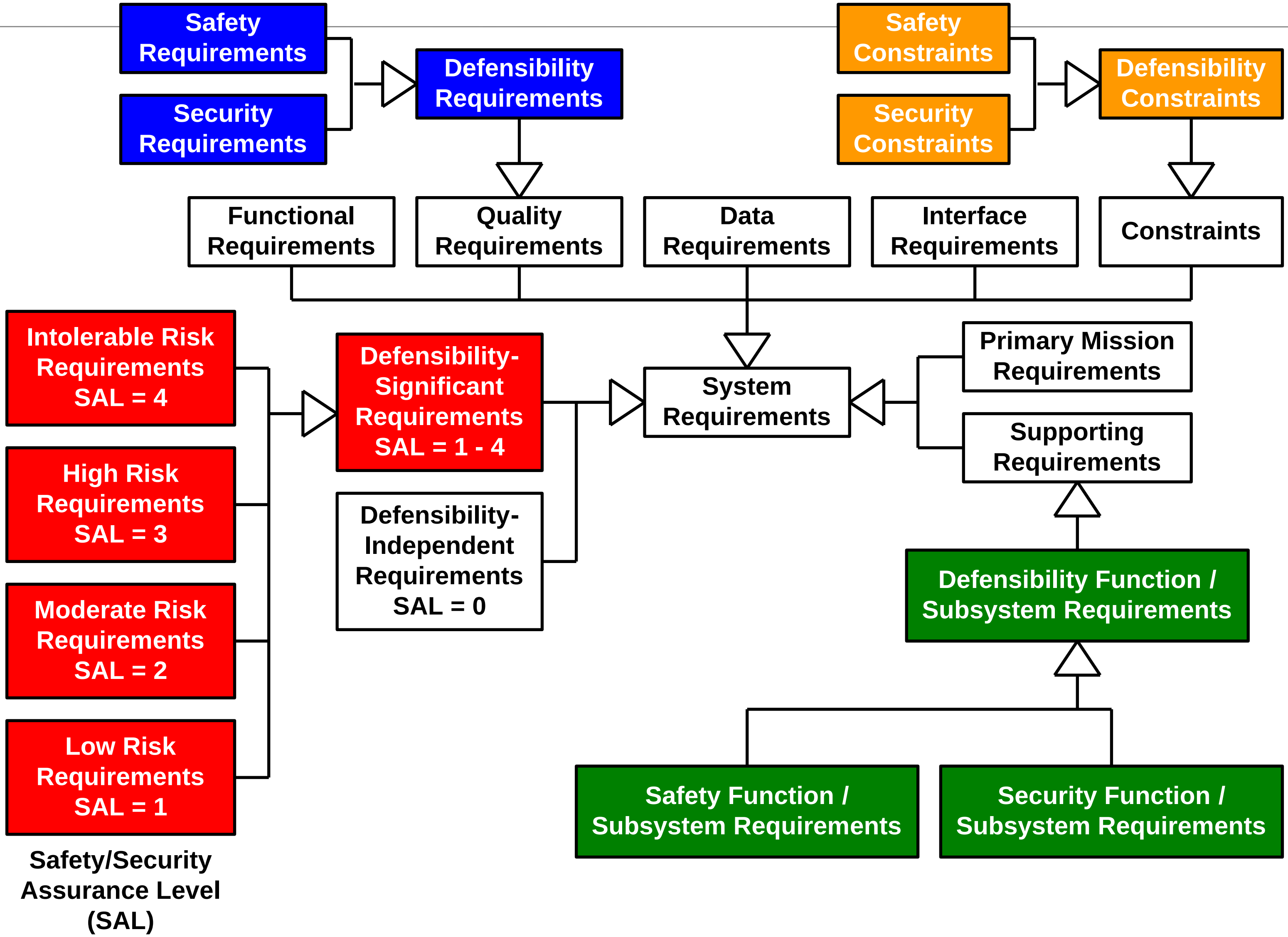
Although policies are not product requirements, they may necessitate the engineering of derived requirements

We Are Here

Types of Defensibility-Related Requirements



Four Types of Defensibility-Related Requirements



1) Safety and Security Requirements

Safety and security requirements are *quality* requirements.

Quality requirements are product requirements that specify a mandatory minimum amount of a type of product quality:

- Quality characteristic (generally)
- Quality attributes (specifically)

Safety and security requirements:

- Are typically **negative** requirements
- Specify what the system ***shall not*** cause, enable, or allow to:
  - Occur
    (e.g., unauthorized harm to valuable assets, accidents, attacks)
  - Exist
    (e.g., hazards, threats, vulnerabilities, risks)

Safety and Security Requirements

Quality requirements should be:

- Scalar (how well or how much)
- Based on a quality model defining the specific types of quality and how their measurement scales
- Stored in requirements repositories and specified in requirements specifications, NOT just in:
  - Secondary specifications
  - Safety/security documents

Quality requirements are critically important drivers of the architecture and testing.

Example *Safety* Requirement Templates

When in mode V, the system shall limit the occurrence of *accidental harm* of type W to valuable assets of type X to an average rate of no more than Y asset value per Z time duration.

When in mode W, the system shall not cause *mishaps* of type X with an average rate of more than Y mishaps per Z trips.

When in mode X, the system shall not cause *hazard* Y to exist for more than an average of Z percent of the time.

When in mode X, the system shall not have a residual *safety risk level* of Y or above.

When in mode X, the system shall *detect accidents* of type Y an average of at least Z percent of the time.

Upon detecting an accident of type W when in mode X, the system shall *react* by performing functions Y an average of at least Z percent of the time.

Example *Security* Requirement Templates

When in mode V, the system shall limit the occurrence of *malicious harm* of type W to valuable assets of type X to an average rate of less than Y asset value per Z time duration.

When in mode W, the system shall prevent the first *successful attacks* of type X for a minimum of Z time duration.

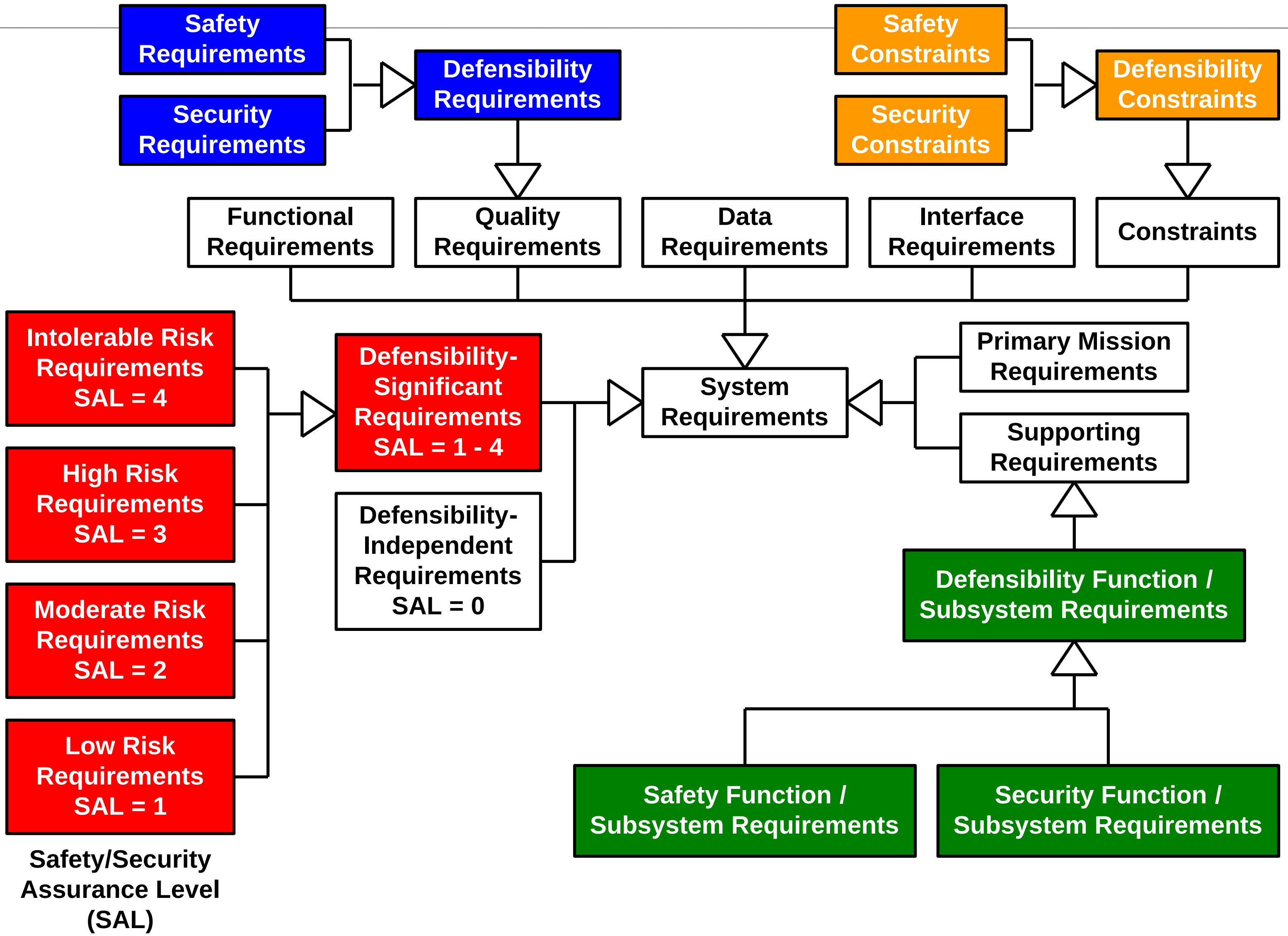
When in mode X, the system shall not have *security vulnerability* Y for more than an average of Z percent of the time.

When in mode X, the system shall not have a *security risk level* of Y.

When in mode X, the system shall *detect misuses* of type Y an average of at least Z percent of the time.

Upon detecting a misuse of type W when in mode X, the system shall *react* by performing functions Y an average of at least Z percent of the time.

2) Safety- and Security-Significant Requirements



Safety- and Security-Significant Requirements

Defensibility-significant requirement

a requirement with significant safety or security ramifications

Are identified based on safety or security (e.g., hazard or threat) analysis

Can be any kind of product requirements, but most interesting if **not**:

- Pure safety and security requirements
- Safety and security function/subsystem requirements
- Safety and security constraints

Example Safety-Significant Requirements

Firing missiles from military aircraft requirements:

- When to arm missiles
- When not to arm missiles (e.g., detecting weight-on-wheels)
- Controlling weapons bay doors before and after firing missiles

Chemical plant requirements:

- Mixing and heating toxic chemicals
- Controlling exothermic reactions
- Detecting and controlling temperature, pressure, and flow-rate

Example Security-Significant Requirements

Access control requirements:

- Identification, authentication, and authorization

Accountability (e.g., non-repudiation requirements):

- Creation, storage, and transmission of financial transactions

Availability (under attack) requirements:

- Services subject to denial-of-services attacks

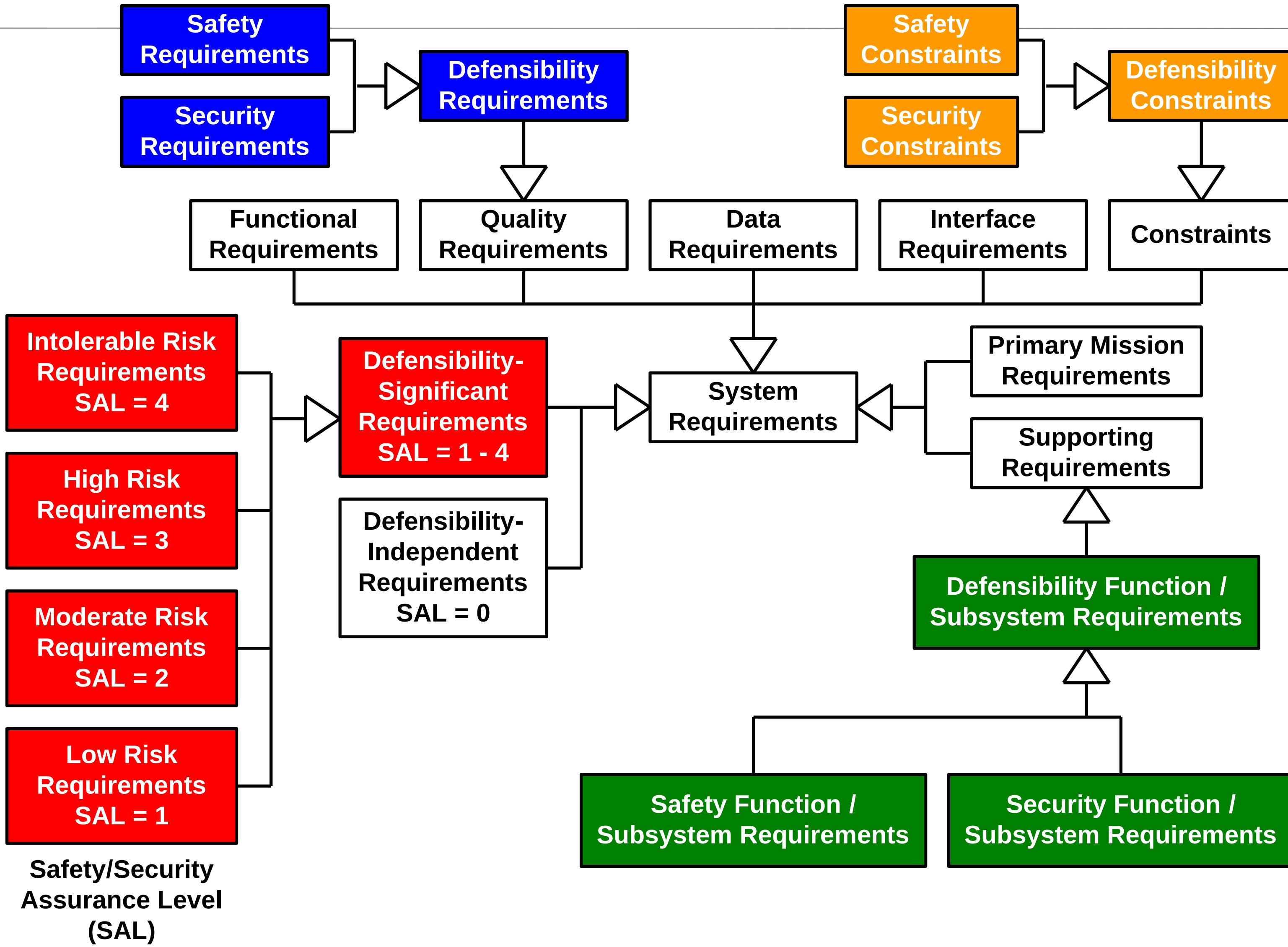
Confidentiality requirements:

- Storage and transmission of sensitive information
- Confidential intellectual property in the software or its documentation

Integrity requirements:

- Storage and transmission of sensitive data
- Software that might get Infected by malware

3) Safety and Security Function/Subsystem Rqmts



Safety and Security Function/Subsystem Rqmts

Defensibility function/subsystem requirements are requirements for functions or subfunctions that exist strictly to improve defensibility (as opposed to support the primary mission requirements).

- **Safety function/subsystem requirements** are requirements for safety functions or subsystems.
- **Security function/subsystem requirements** are requirements for security functions or subsystems.

Example Safety Function/Subsystem Rqmts

Requirements for functions or subsystems added strictly for safety:

- Aircraft safety subsystems:
  - Airborne wind shear detection and alert system
  - Ejection seat and parachute
  - Engine fire detection and suppression
  - Ground proximity warning system (GPWS)
  - Minimum Safe Altitude Warning (MSAW)
  - Traffic alert and Collision Avoidance System (TCAS)
- Automobiles:
  - Adaptive Cruise Control
  - Adaptive Headlights and Highbeam Assist
  - Airbags
  - Anti-Lock Breaking System (ABS)
  - Backup Camera
  - Backup Sensors
  - Electronic Stability Control (ESC)
  - Seatbelts
  - Traction Control System (TCS)

Example Safety Function/Subsystem Rqmts

Except when the weapons bay doors are open or have been open within the previous 90 seconds, the weapons bay cooling subsystem shall maintain the temperature of the air in the weapons bay at or below X C.

The Fire Detection and Suppression Subsystem (FDSS) shall detect smoke above X ppm in the weapons bay within 2 seconds at least 99.9% of the time.

The FDSS shall detect temperatures above X C in the weapons bay within 2 seconds at least 99% of the time.

Upon detection of smoke or excess temperature, the FDSS shall begin fire suppression within 1 second at least 99.9% of the time.

Example Security Function/Subsystem Rqmts

Functions or subsystems strictly added for security:

- Access control
- Antivirus / antispyware / antispam / antiphishing subsystems
- Encryption/decryption subsystem
- Firewalls
- Intrusion detection subsystem (IDS) / intrusion prevention subsystem (IPS)

All requirements for such functions/subsystems are security-related.

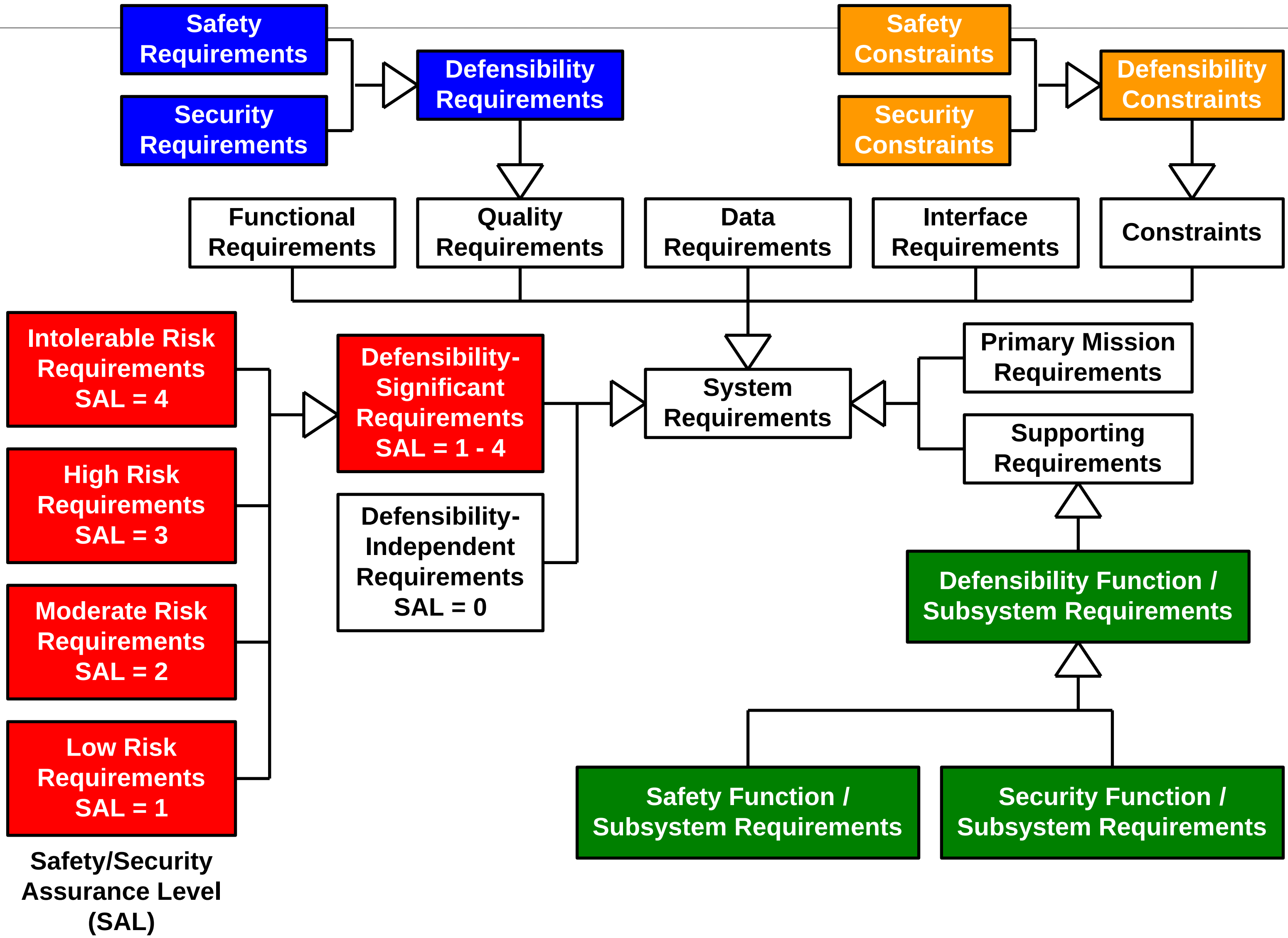
Look in the *Common Criteria* (ISO/IEC 15408) for many generic reusable security function requirements.

Example Security Function/Subsystem Rqmts

Access Control Function:

- The Access Control Function shall require at least 99.99% of users to identify themselves before enabling them to perform the following actions: …
- The Access Control Function shall require at least 99.99% of users to successfully authenticate their claimed identity before enabling them to perform the following actions: …
- The Access Control Function shall authorize the system administrators to configure the maximum number of unsuccessful authentication attempts between the range of 1 and X.
- The Access Control Function shall perform the following actions when the maximum number of unsuccessful authentication attempts has been exceeded: …

4) Safety and Security Constraints



Safety and Security Constraints

A **constraint** is any engineering decision that has been chosen to be mandated as a requirement. For example:

- Architecture constraints
- Design constraints
- Implementation constraints
(e.g., coding standards, safe language subset, and nonhazardous materials)
- Integration constraints
- Deployment/configuration constraints

A **safety constraint** is any constraint primarily intended to ensure a minimum level of safety (e.g., a mandated safeguard).

A **security constraint** is any constraint primarily intended to ensure a minimum level of security (e.g., a mandated countermeasure).

Safety and security standards often mandate industry best practices as constraints.

Example Safety Constraints

The system shall use hardware interlocks to prevent users from physically coming into contact with moving parts.

The system shall not have a single point of failure that can cause an accident unless the associated risk is acceptable to authoritative stakeholders.

The system shall use a safe subset of C++.

The system shall not contain any of the hazardous materials in Table X in concentrations in excess of those listed in the table:

- **Biologically** Hazardous Materials (e.g., infectious agents or biotoxins)
- **Chemically** Hazardous Materials (e.g., carcinogens, corrosives, heptatoxins, irritants, mutagens, nephratoxins, neurotoxins, poisons, or teratogens)
- **Physically** Hazardous Materials (e.g., combustible chemicals, compressed gases, explosives, flammable chemicals, oxidizers, or pyrophorics)

Example Security Constraints

The system shall user use IDs and passwords for identification and authentication.

The system shall incorporate COTS firewalls to protect servers.

The system shall incorporate a COTS virus detection and removal product.

The system shall use public key encryption to protect confidential information.

The system shall use digital signatures to provide nonrepudiation.

We Are Here

Three Disciplines

Challenges

Common Example

Requirements Engineering Overview

Safety and Security Engineering Overview

Types of Safety- and Security-related Requirements

Collaborative Defensibility Engineering Method

Conclusion

Desired Method Properties

Help meet challenges listed at start of tutorial

Promote close collaboration among safety, security, and requirements teams

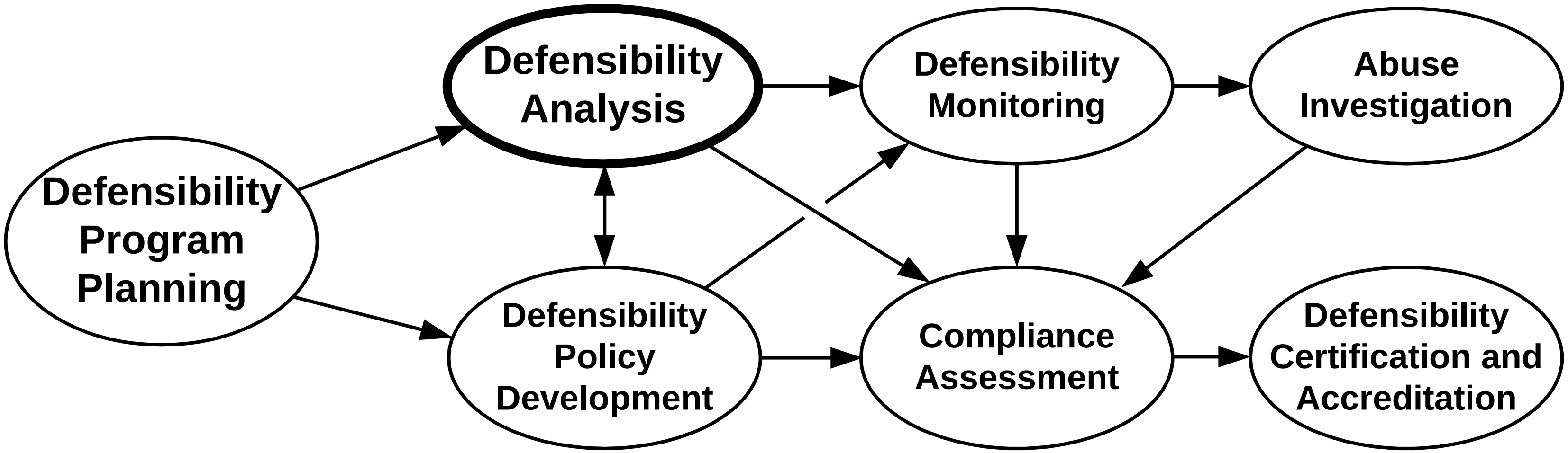
Better integrate safety and security methods:

- Based on common foundational concepts and terminology
- Reuse of techniques and work products
- Based on defensibility (safety and security) analysis

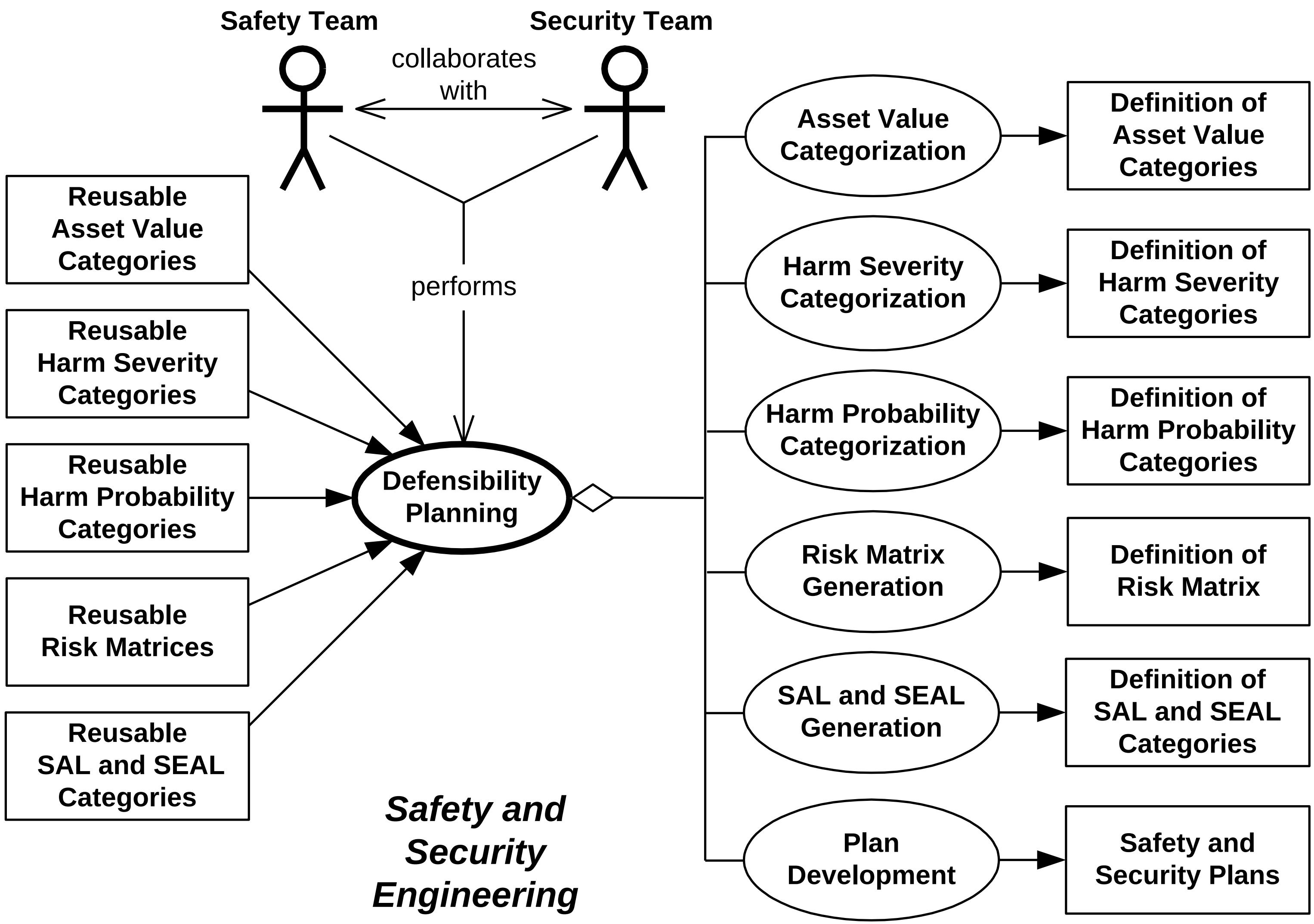
Better integrate safety and security engineering with requirements engineering:

- Clearly defined role and team responsibilities
- Early input to requirements engineering
- Develop all types of safety- and security-related requirements
- Ensure these requirements have the proper characteristics

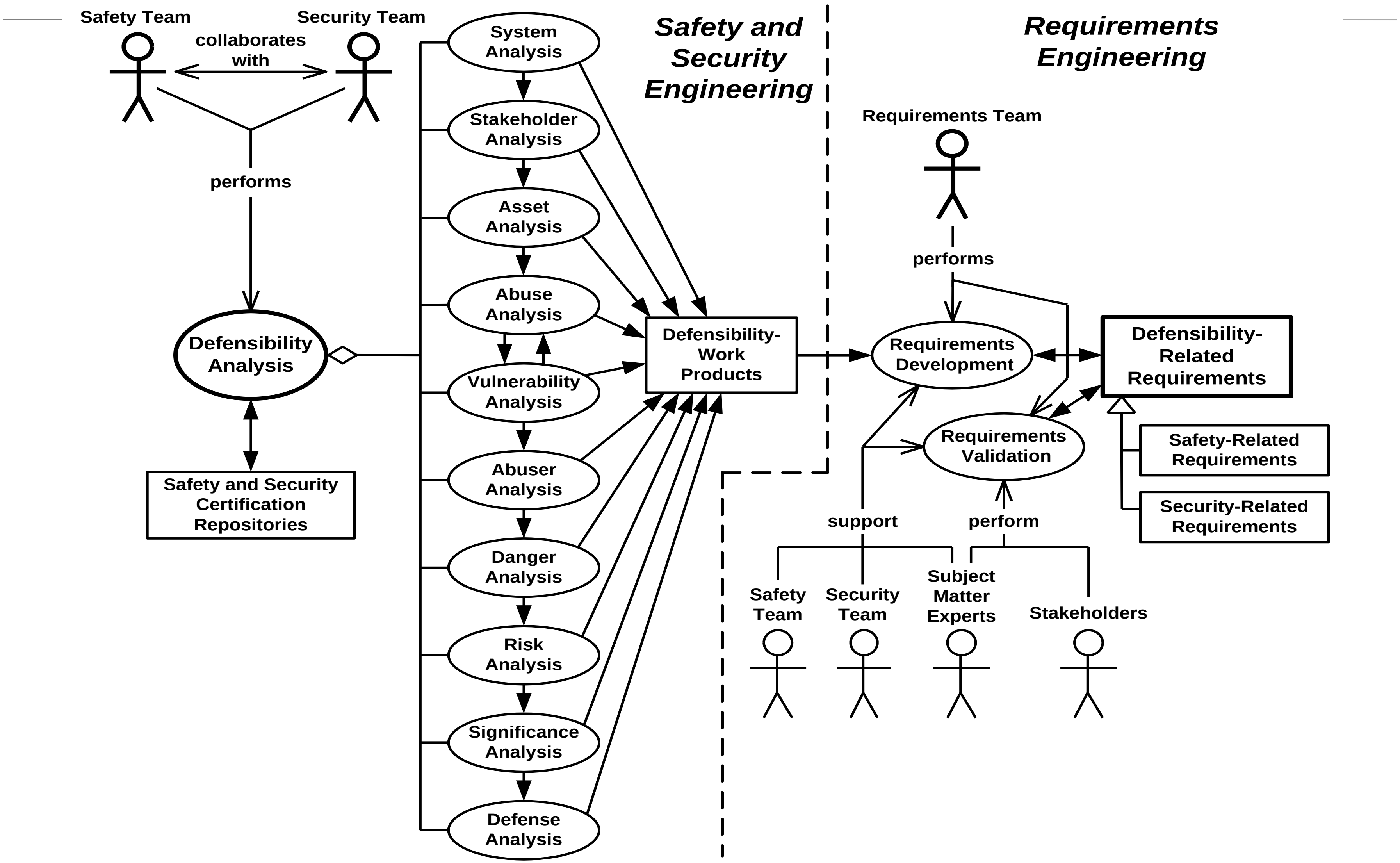
Overall Defensibility Engineering Method



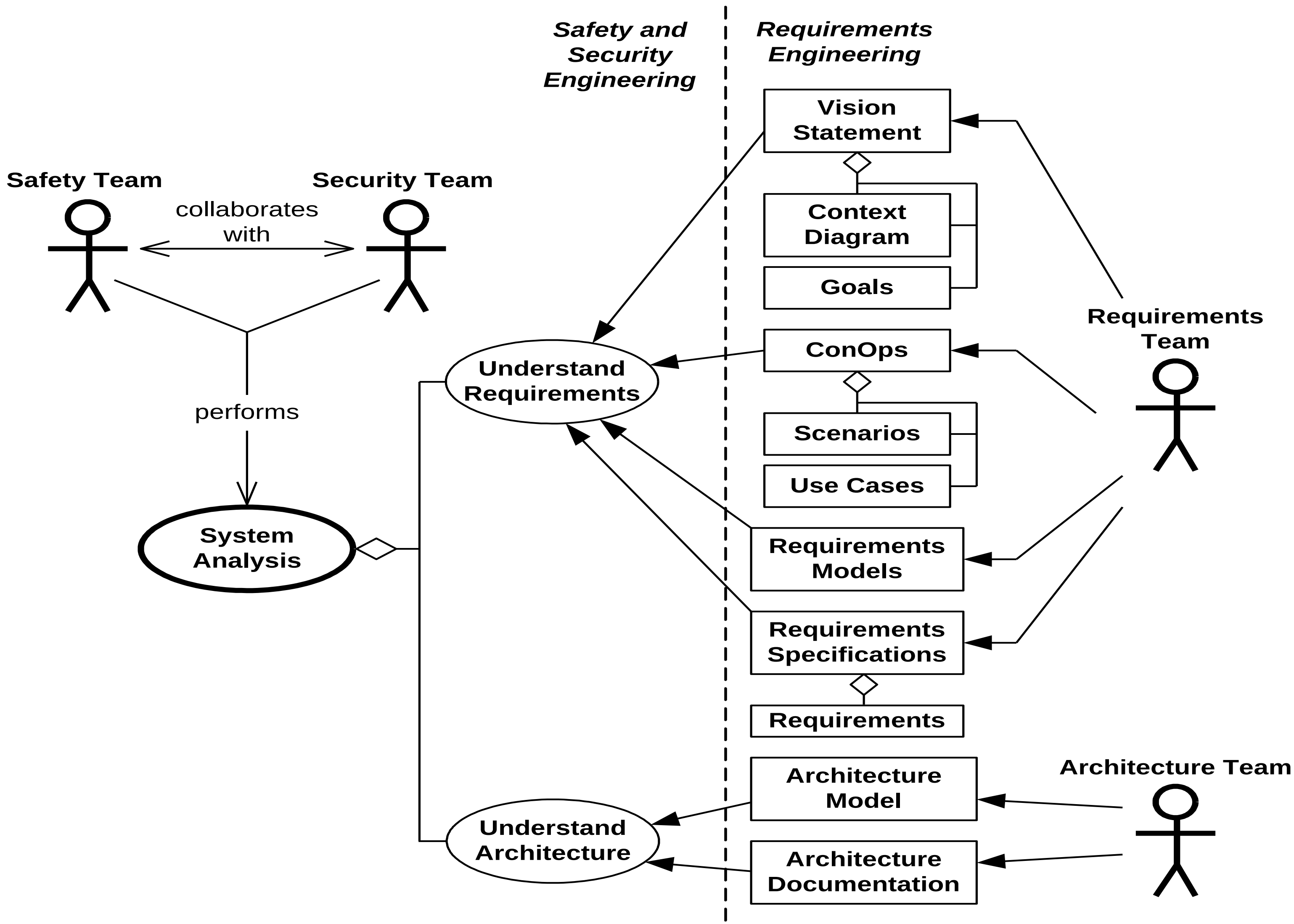
Defensibility Program Planning



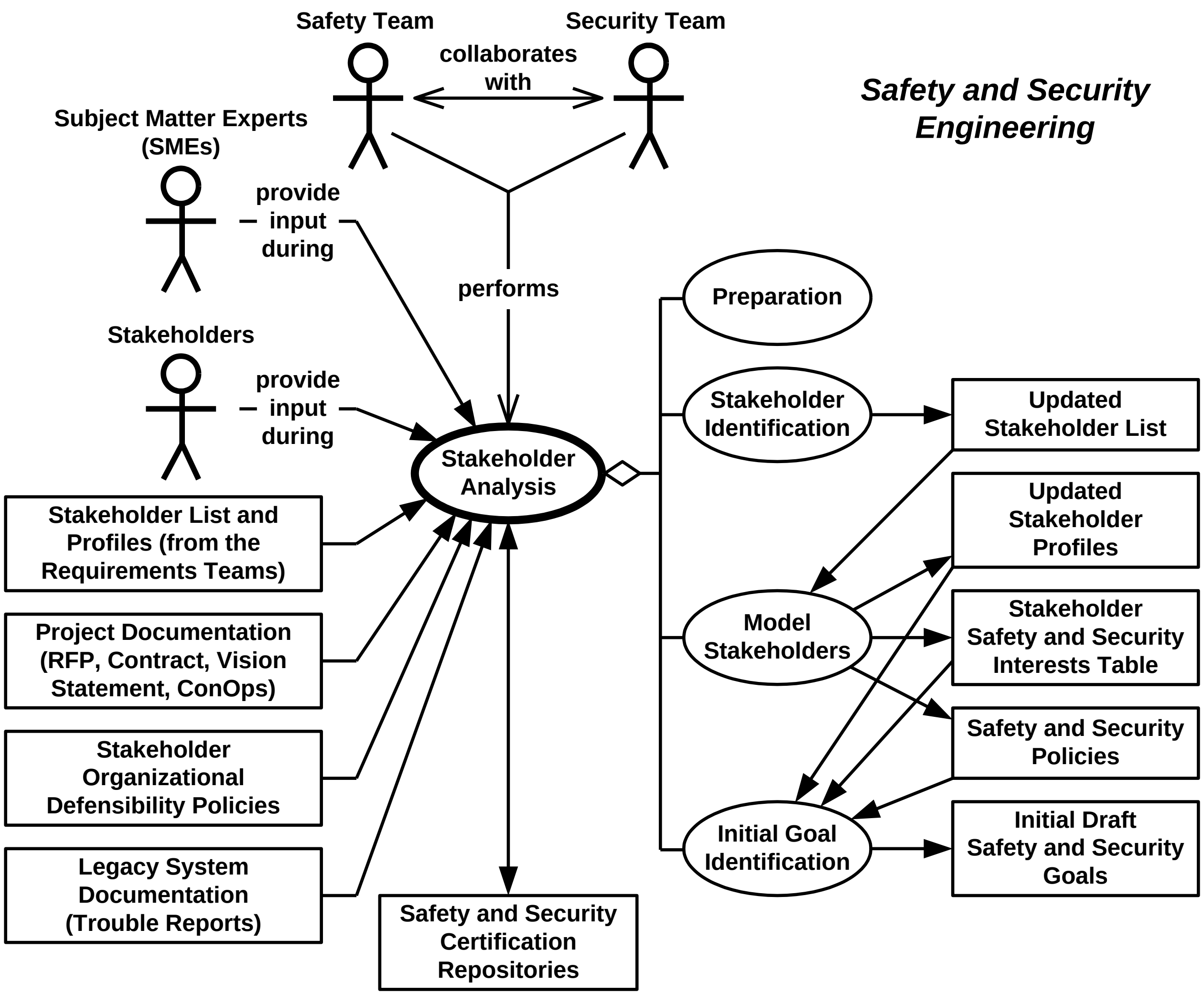
Defensibility Analysis



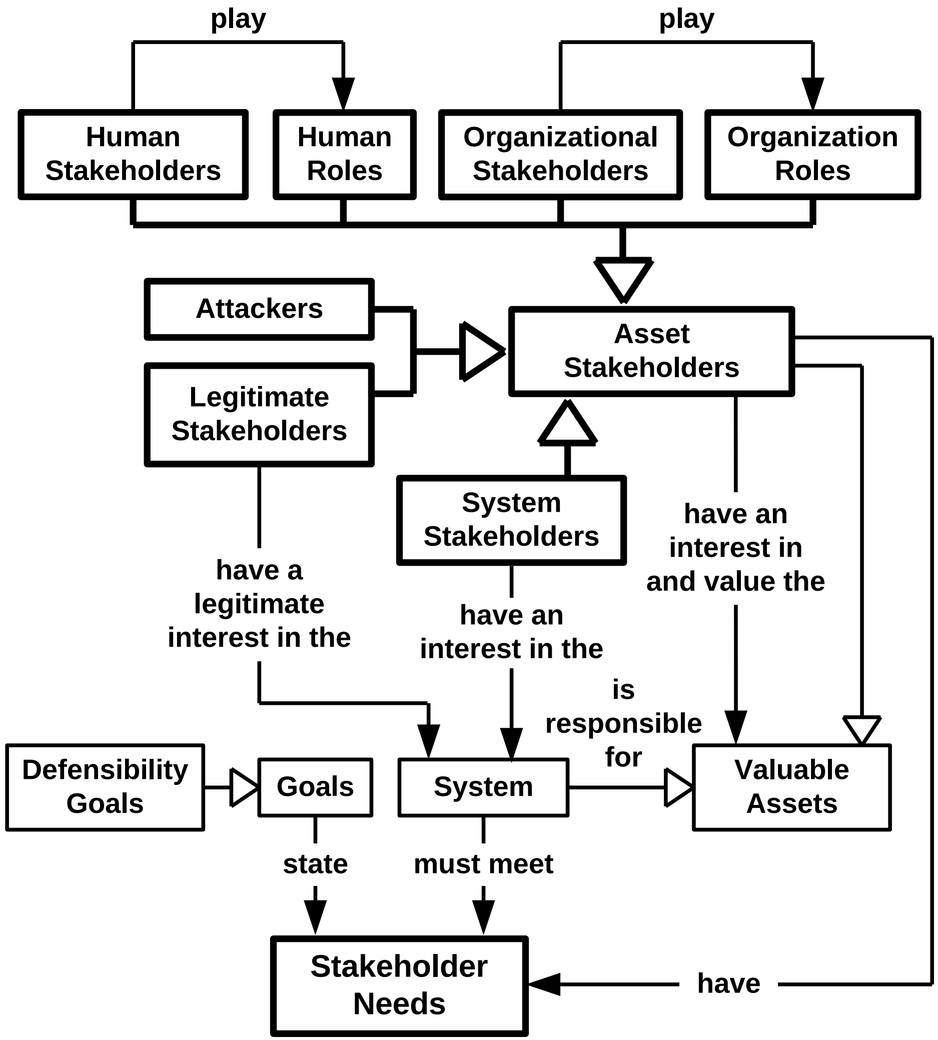
Systems Analysis



Stakeholder Analysis



Types of Stakeholders



Representative ZATS Stakeholders

Human Roles:

- Passenger
  the role played by any person while riding ZATS taxis
- Maintainer
  the role played by employees of PMI while they maintain ZATS and its components

Organizations:

- Metropolitan Zoo Authority (MZA)
  the organization that operates the Metropolitan Zoo and acquires ZATS
- People Mover Incorporated (PMI)
  the company that is building and will operate ZATS

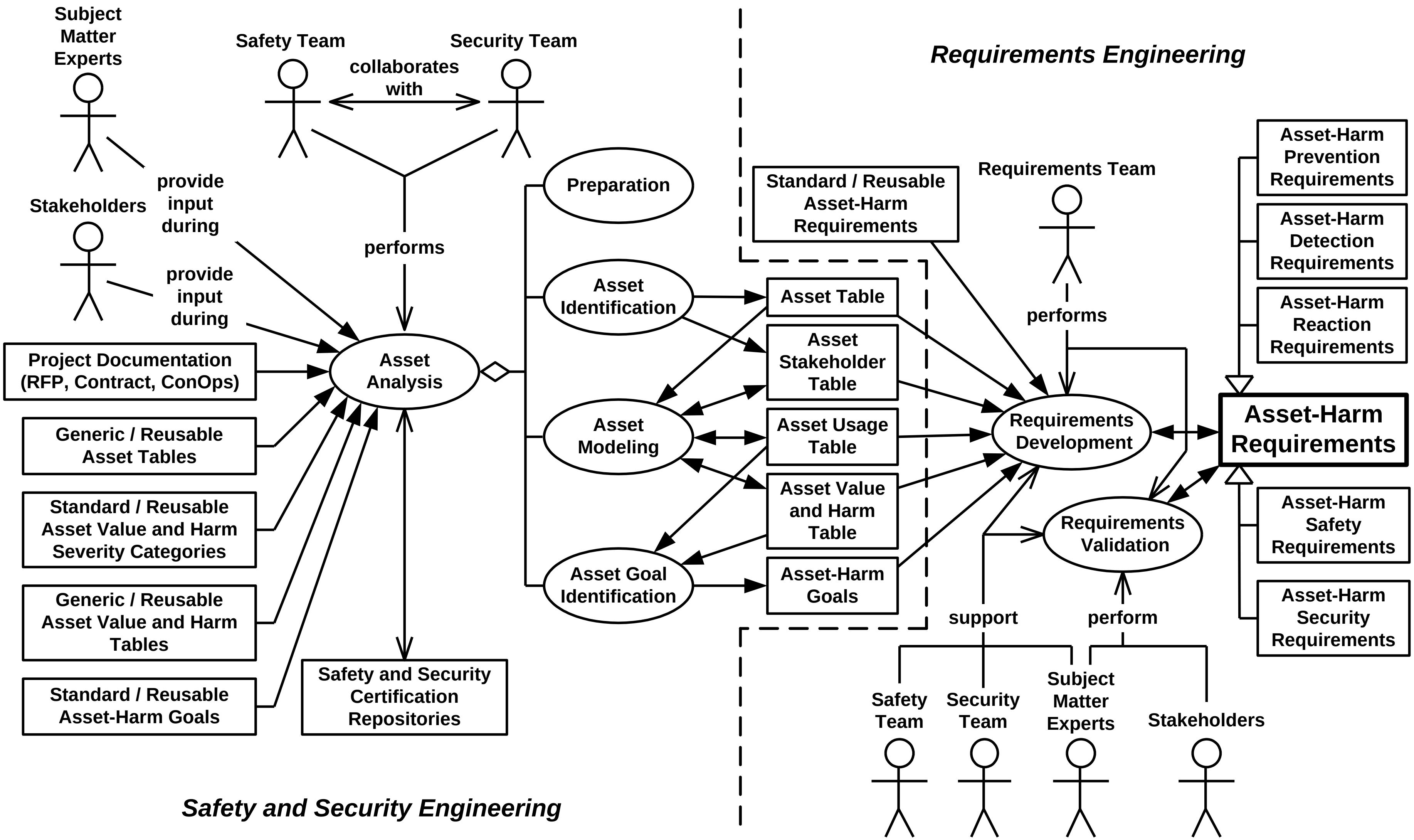
Partial ZATS Stakeholder Defensibility Interest Table

ZATS Stakeholder	Safety Interests	Security Interests
Passenger any person who rides on a ZATS taxi	Not be killed or injured in a taxi accident (i.e., collision)	Not be killed or injured in a physical attack (e.g., mugging or rape)
	Not be stranded in a taxi, especially in very hot or cold weather	Not have confidential bank card information disclosed
	Not be trapped in a taxi in case of an emergency (e.g., taxi fire, tornado, or earthquake)	Not be maliciously overcharged or lose money when using the travel card vending machines
	Not be trapped in a taxi station elevator or by taxi station doors	
	Not be accidentally overcharged or lose money when using the travel card vending machines	
	Not be frightened and put at risk by a taxi speeding (i.e., overspeed) or tailgating (i.e., inadequate headway)	
Police Officer any warrented law enforcment officer of a police force who is responsible for: (1) crime pre-vention, detection, and reaction, (2) ...		To provide people with reasonable protection against physical attack
		To apprehend ZATS-related criminals
		To capture sufficient evidence to permit prosecution of ZATS-related

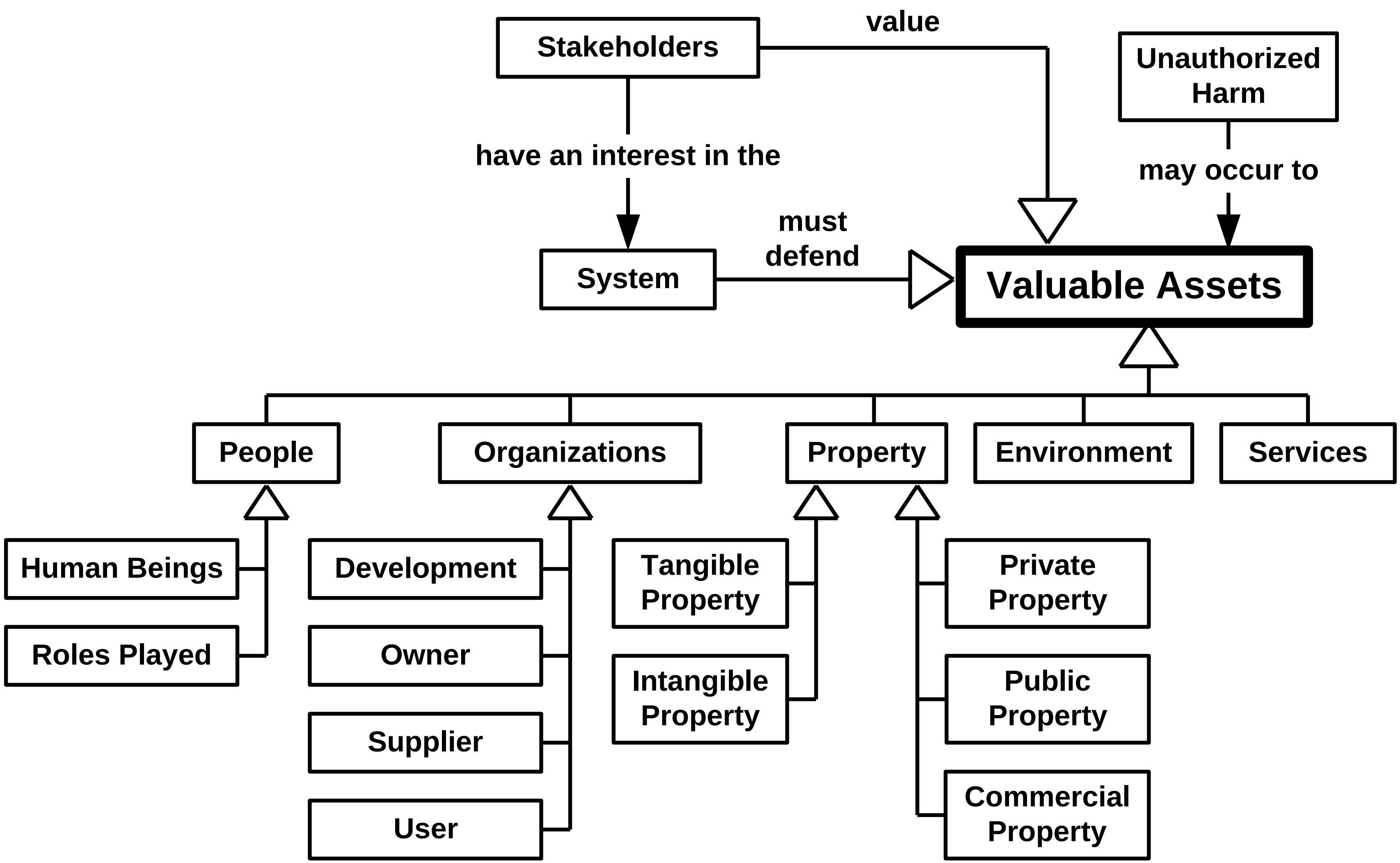
Example Stakeholder Safety/Security Analysis

Safety/Security Analysis of PMI Corporate Management
General Responsibilities: – Ensure the short- and long-term profitability of PMI. – Ensure the marketability of PMI automated people movers (APM). – Ensure customer and user satisfaction with PMI APMs. – Ensure the quality of PMI APMs (e.g., capacity, maintainability, reliability, usability).
Safety/Security Responsibilities: – Provide leadership in safety and security matters. – Ensure that PMI APM vehicles will operate safely. – Ensure that PMI APM systems will be secure against attack (e.g., cybercrime and theft). – Ensure that PMI proprietary information will be secure from corporate espionage. – Ensure that effective safety and security policies will exist and be enforced. – Provide oversight of PMI projects with regard to safety and security.
Context: – PMI is subject to very stiff competition from both domestic and foreign competitors. – PMI's ability to create APMs that enable large numbers of small vehicles to safely share guideways with minimal headway is a cutting-edge technology that provides a competitive edge over some of PMI's rivals. – Corporate management is primarily trained in business management and does not understand the technology, especially the importance of software to achieving technical requirements. – PMI's previous smaller and simpler APMs have not suffered serious accidents or security attacks. – ZATS is PMI's current flagship APM.
Actual or Potential Process Model Defects: – Has underestimated the maturity and risk of the new technology. – Has assumed that ZATS development productivity will be the same or better than previous projects. – Believes that the large number of failures found and fixed during initial prototyping indicate that the major problems are all solved and are not indicative of future failures once full scale development starts and ZATS has been placed into operation.
Actual or Potential Dangerous Decisions and Actions: – Has pushed ZATS project management to lower their estimates of the cost of ZATS development and the time required for ZATS development in order to win the contract. – Is counting on the project future income from ZATS, even though ZATS is likely to suffer from both cost and schedule overruns. – May let safety and security policies and requirements be violated in order to meet project budget, schedule, and functionality goals. – May provide inadequate oversight of the ZATS project with regard to safety and security.

Asset Analysis



Valuable Assets



Representative ZATS Assets$_1$

People:

- Passengers - people who are riding ZATS taxis

Organizations:

- Metropolitan Zoo Authority (MZA) - the organization that operates the Metropolitan Zoo and acquires ZATS

Property:

- ZATS line replaceable units (LRUs) - ZATS architectural components (data, hardware, software, subsystem) that can be repaired or replaced by the maintainer after ZATS has been placed in operation

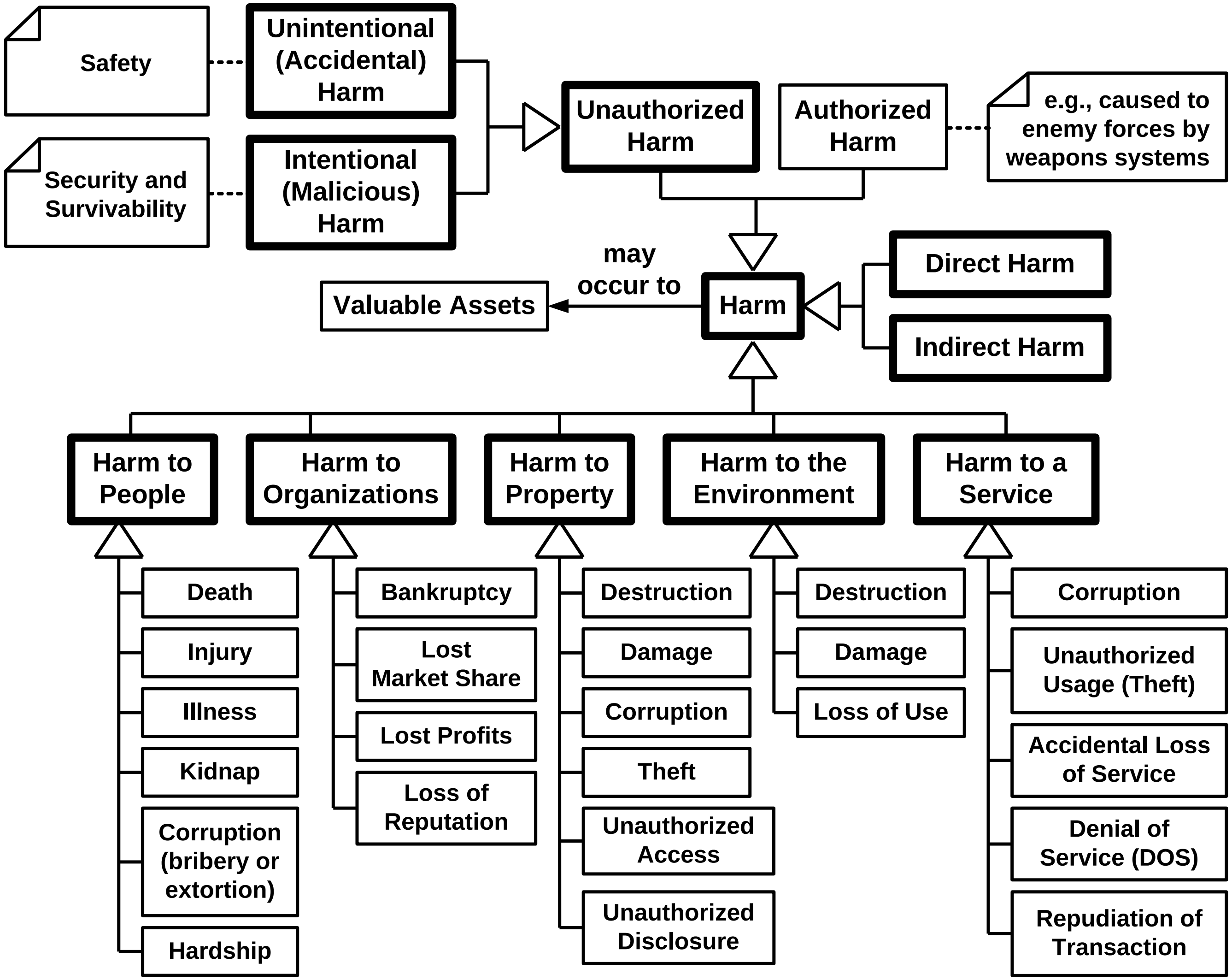
Environment:

- Atmosphere - the layer of gas surrounding the earth that is retained by gravity and subject to both air and noise pollution

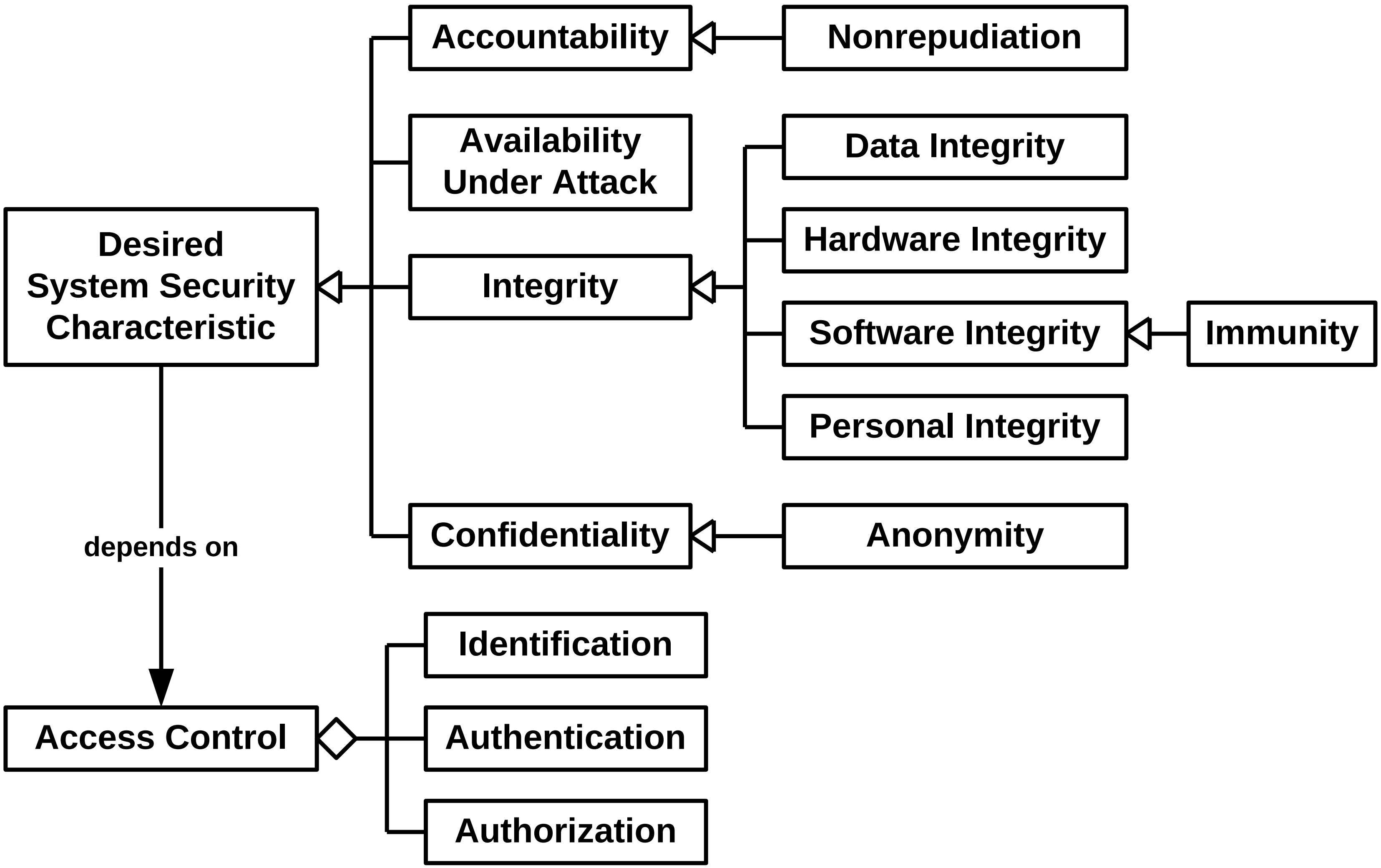
Service:

- Taxi service - transportation by ZATS taxis of passengers and their personal property around the zoo

Types of Harm to Valuable Assets



Security Characteristics as Types of Harm



Harm Severity

Harm severity is an appropriate categorization of the amount of harm:

- Potential harm
- Actual harm
- Maximum credible harm

Harm severity categories can be standardized (ISO, military, industry-wide) or endeavor-specific.

Harm severity categories need to be:

- Clearly identified.
- Appropriately and unambiguously defined.

Note that some standards confuse harm severity with hazard "severity" (i.e., categorization of hazard based on the severity of harm that its accidents can cause)

Representative Harm to a ZATS Valuable Asset

Person:

- Death
- Injury – minor, major, and disability (physical and psychological)
- Occupational illness (maintainer)
- Loss of money – various amounts
- Lost of confidentiality of sensitive information (passenger identity theft)
- Loss of reputation (managers, developers)

The ZATS Harm Severity Categories

Catastrophic:

Potential ZATS lifespan harm that is unacceptable to authoritative stakeholders (loss of priceless valuable asset)

Major:

Potential five-year harm that is only acceptable to authoritative stakeholders after major actions have been taken to lower its risk (loss of extremely valuable asset)

Minor:

Potential yearly harm that is acceptable to authoritative stakeholders after minor action has been taken to lower its risk (loss of moderate or low value asset)

Negligible:

Potential yearly harm that is acceptable to authoritative stakeholders but that does not justify any action to lower its risk (loss of inconsequential asset)

The ZATS Harm Likelihood Categories

Frequent:

An average of once or more times every month

Probable:

Less than an average of once every month (but more than occasional)

Occasional:

Less than an average of once every year (but more than remote)

Remote:

Less than an average of once every 5 years (but more than implausible)

Implausible:

Less than a 10% chance of happening during the entire ZATS 30 year planned lifespan

Representative ZATS Asset-Harm Safety and Security Goals

Asset-Harm Prevention Safety Goal:

- ZATS will not accidentally kill or injure its passengers.

Asset-Harm Detection Security Goal:

- ZATS will detect infection of its data and software files by malware.

Asset-Harm Reaction Security Goal:

- On detecting malware infection, ZATS will quarantine the infected file and notify the operator and maintainer.

Representative ZATS
Asset-Harm Safety and Security Requirements

Asset-Harm Prevention Safety Requirement:

- **Passenger death prevention:**
  ZATS shall ensure that the expected frequency with which it accidentally kills a passenger does not exceed 0.1 passenger during the projected 30 year system lifespan.

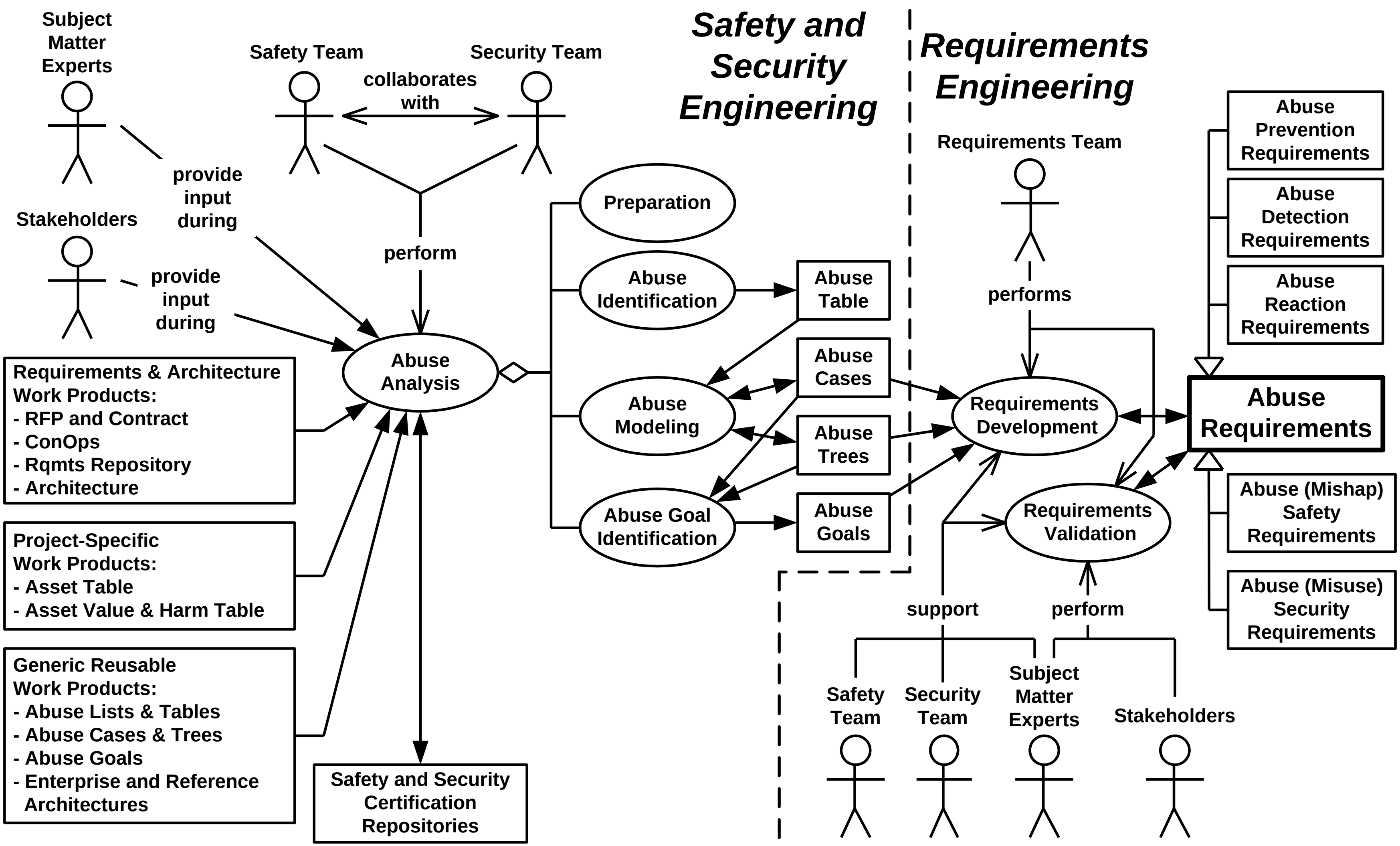
Asset-Harm Detection Security Requirement:

- **Malware detection:**
  ZATS shall detect infection of its data and software files by at least 99% of known malware.

Asset-Harm Reaction Security Requirement:

- **Malware reaction:**
  On detecting malware infection, ZATS shall quarantine the infected file and notify the operator and maintainer at least 99.9% of the time.

Abuse (Misuse and Mishap) Analysis



Abuses (Mishaps and Misuses)

Abuse (defensibility)

a series (or network) of one or more unwanted events that cause (or can cause) unauthorized harm to one or more valuable assets

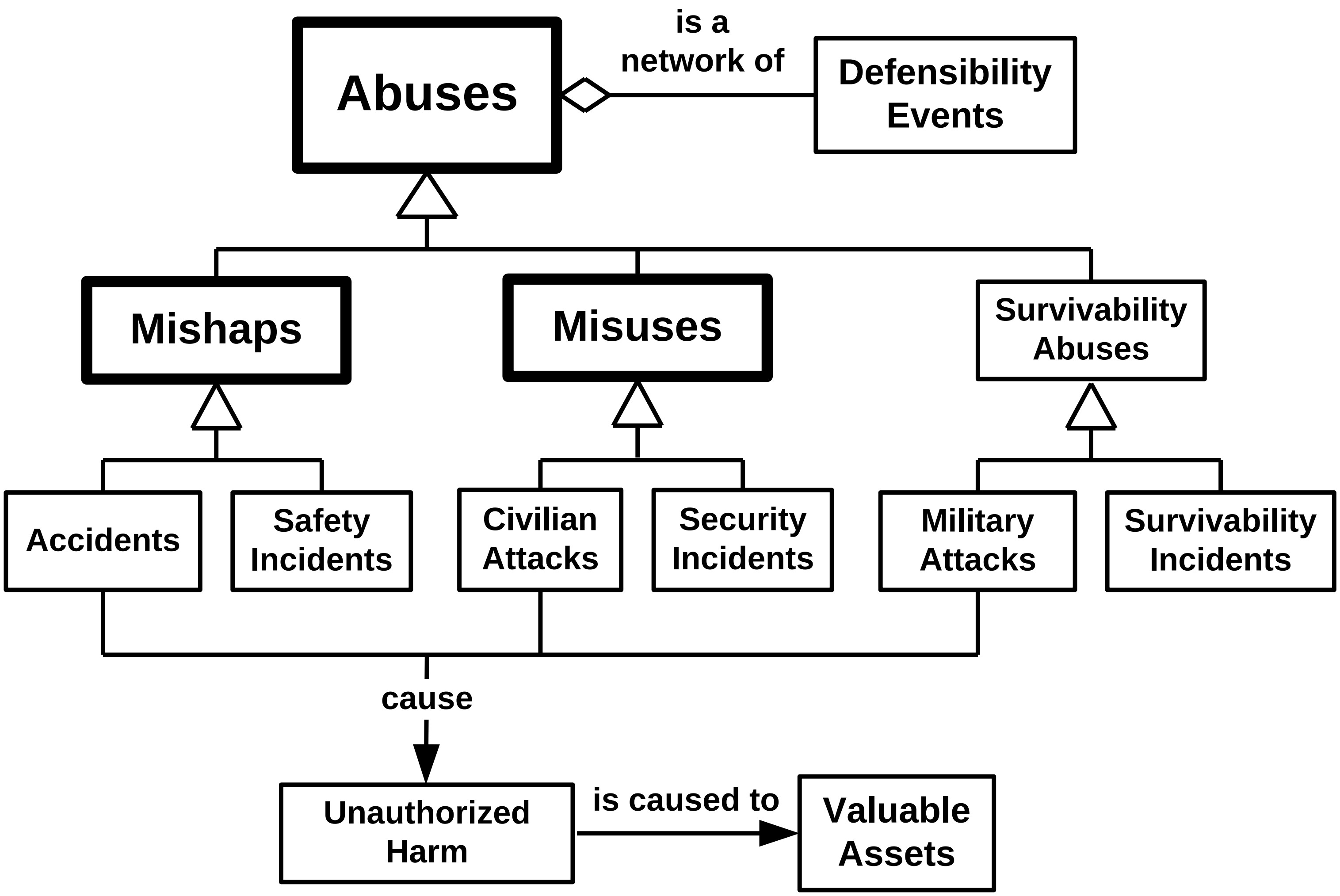
Mishap (safety)

an accidental abuse

Misuse (security or survivability)

a intentional (malicious) abuse

Types of Abuses

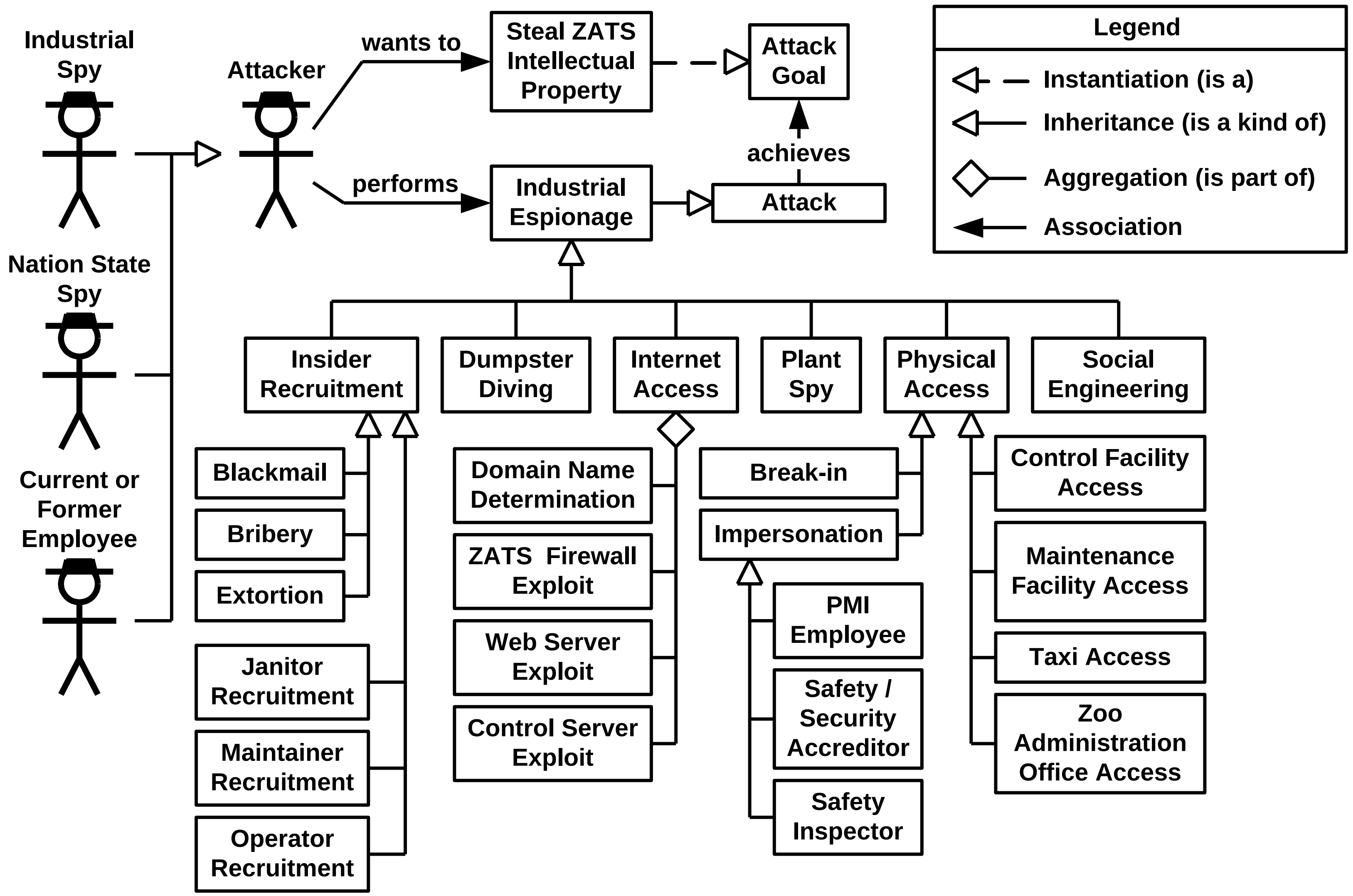


Importance of Accidents

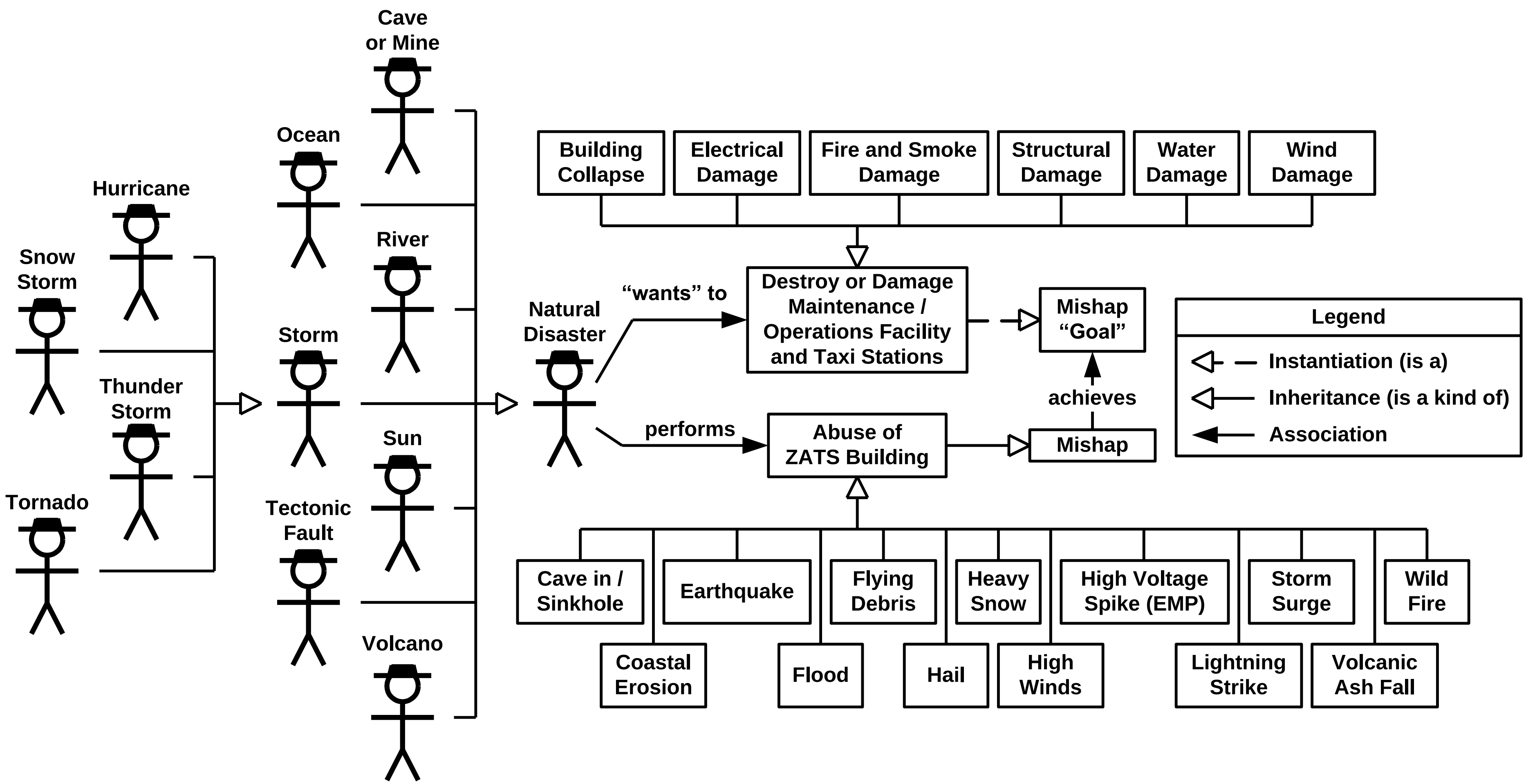
Accidents can have expensive and potentially fatal repercussions:

- Ariane 5 maiden launch
  - Reuse of Ariane 4 software not matching Ariane 5 specification
- Mars Climate Orbiter ($125 million)
  - English vs. Metric units mismatch
- Mars Polar Lander
  - Missing requirement concerning touchdown sensor behavior
- Therac–25 Radiation Therapy Machine
  - Timing of unusual input sequence results in extreme radiation doses
- Patriot Missile Battery Misses SCUD missile
  - Missing availability (uptime) requirement

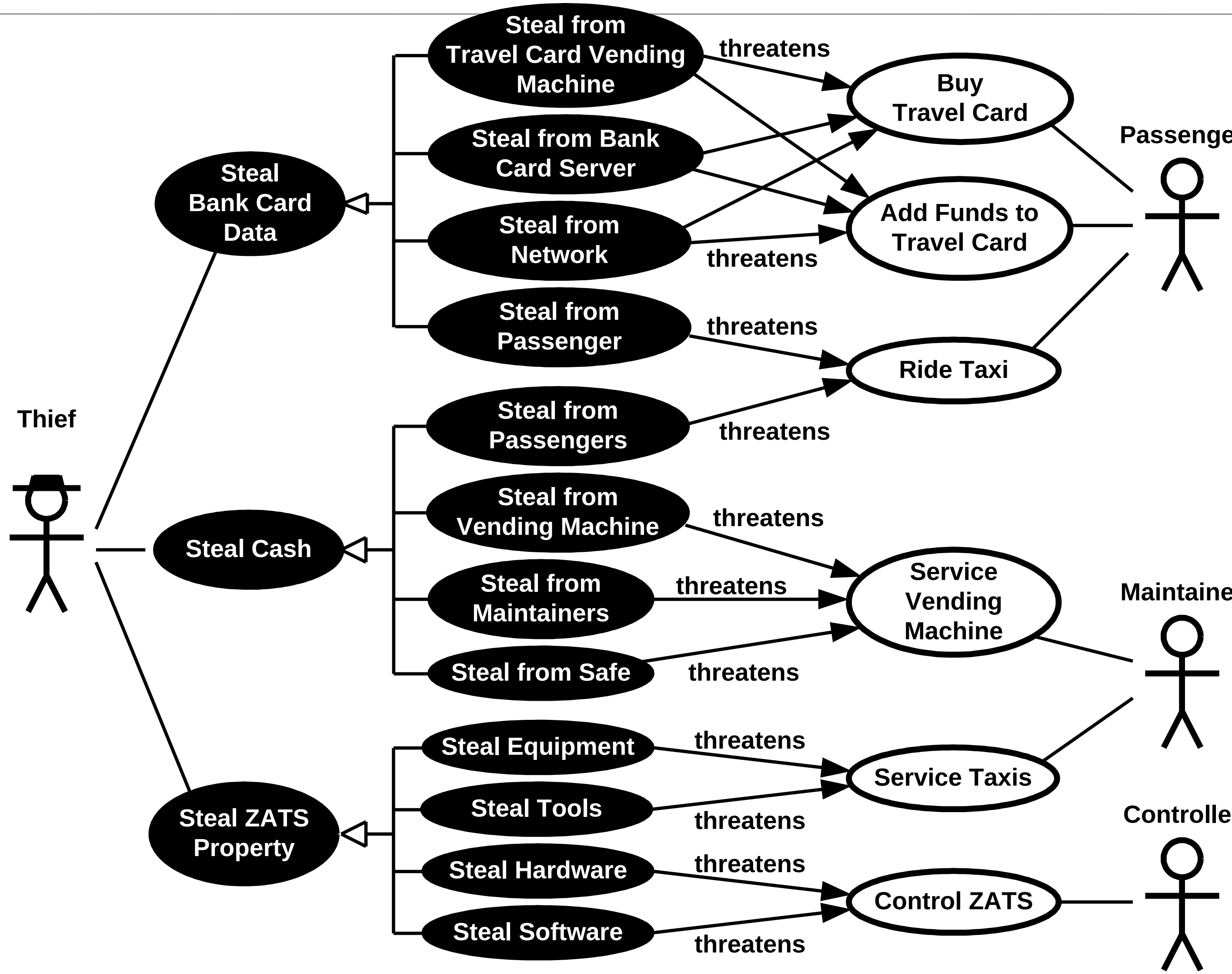
Example ZATS Security Abuse (Attack) Tree



Representative ZATS Safety Abuse (Mishap) Tree



Example ZATS Abuse (Misuse) Cases



Representative Potential Types of ZATS Safety Abuses (Mishaps)

Accident:

- **Rear-end Collision**
  One taxi rear-ends another taxi causing damage to one or both of the taxis.

Safety Incidents:

- **Bumping Bumpers**
  One taxi lightly bumps into another taxi without causing damage to one or both taxis.
- **Inadequate Headway (tailgating)**
  The distance between two adjacent taxis *becomes* less than the minimum safe braking distance.

  Starting to tailgate (event) is an safety incident (near miss), whereas Tailgating (condition) is a hazard.

Representative Potential Types of ZATS Security Abuses (Misuses)

Successful attacks:

- **Arson of Taxi Station**
  An arsonists starts a fire in a taxi station.
- **Denial of Service (DoS)**
  A cybercriminal uses signal jamming to mount a successful DoS attack against the ZATS taxis.

Security Incidents:

- **Unsuccessful Malware Infection**
  Software malware (e.g., worms and viruses) fails to infect a ZATS computer (e.g., because of the existence of properly installed antivirus software with current virus definitions).
- **Undetected Probe**
  A cybercriminal’s attempt to identify computer vulnerabilities (e.g., open ports) goes undetected.

Representative ZATS Abuse (Mishap and Misuse) Requirements

Mishap prevention requirement:

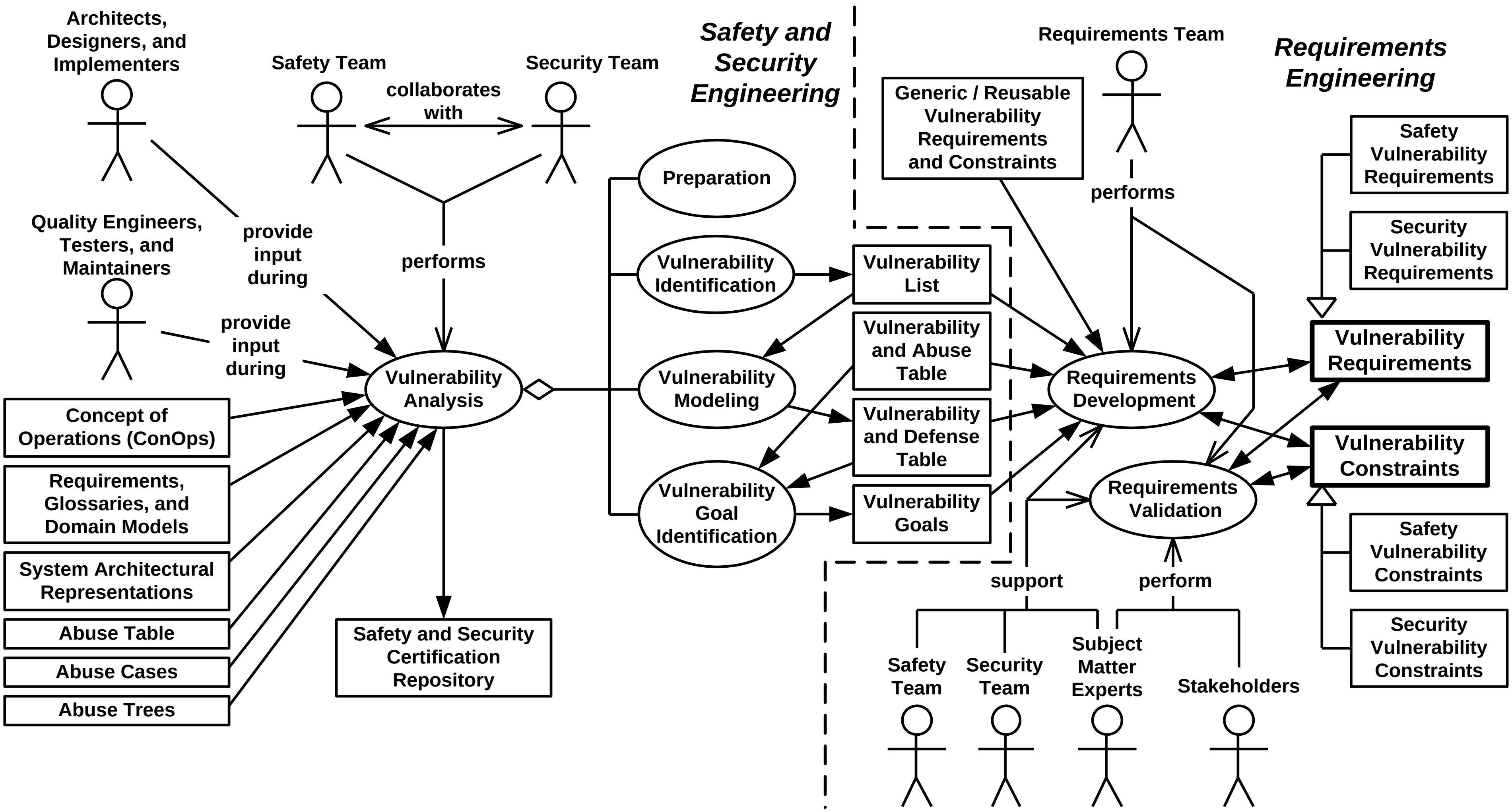
Taxi collisions: Under normal operating conditions, ZATS shall ensure that the rate of major collisions between taxis* is less than X per [Y trips | time unit].

* Terms must be properly defined. For example, a major taxi collision could be defined as one with a relative speed of at least 10 mph. It could also be defined in terms of cost but…

Misuse prevention requirement:

Denial of service: ZATS shall detect Internet launched denial of service attacks within 2 minutes at least 99% of the time.

Vulnerability Analysis



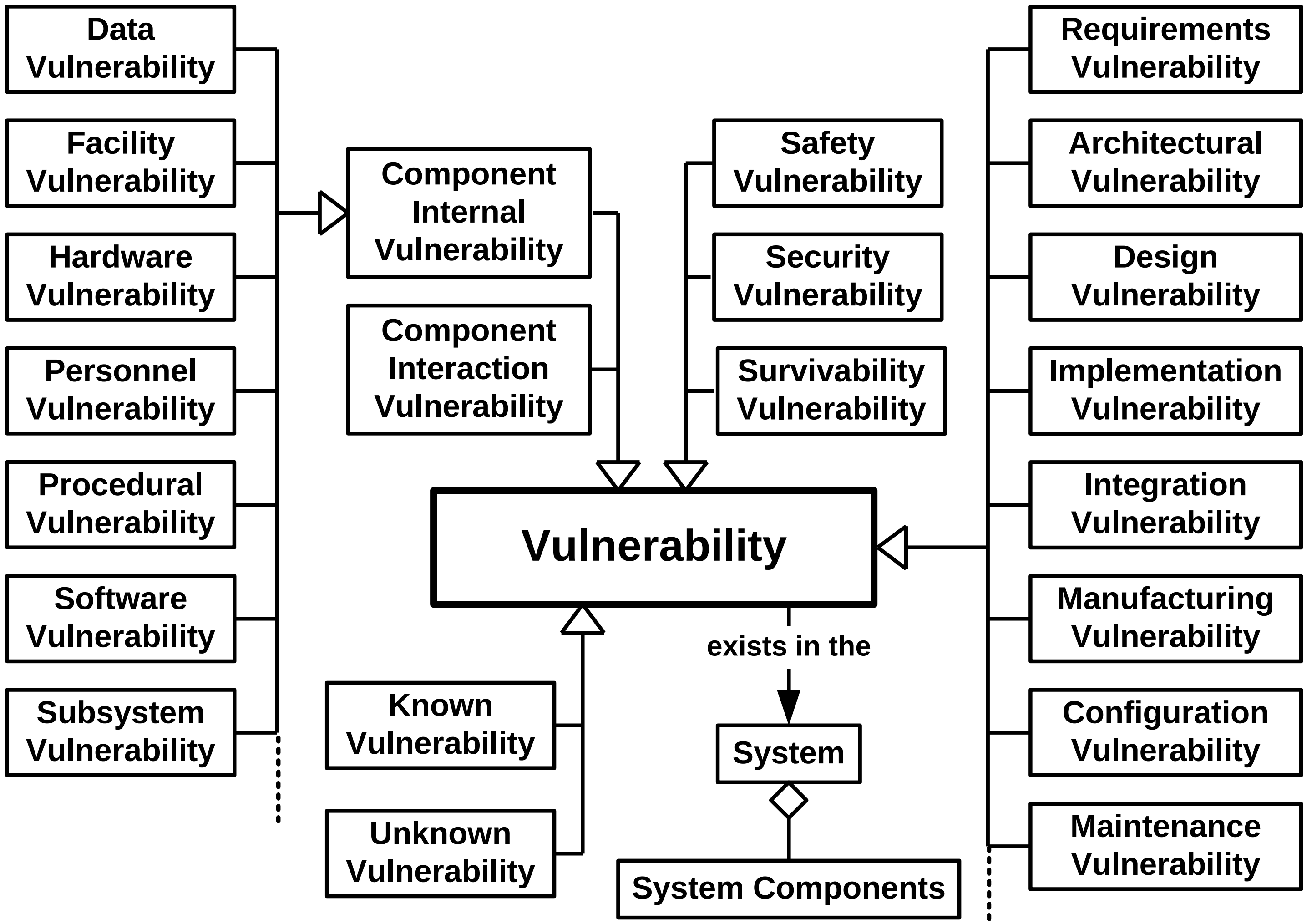
Vulnerabilities

Vulnerability

a potential or actual *system-internal weakness* or *defect* **<u>in</u>** the system that enables or causes:

- A danger (hazard or threat) to exist
- An abuse (mishap or misuse) to occur

Types of Vulnerabilities



Representative ZATS Vulnerabilities (Potential or Actual)

Defensibility vulnerability:

- Lack of or defective fire detection and suppression system in the ZATS buildings

Safety vulnerability:

- Defect (accidentally incorporated) in the software of the LIDAR proximity detection subsystem

Security vulnerability:

- Lack of or defect in the encryption/decryption software
- Incorporation of a backdoor, logic bomb, time bomb, or Trojan horse

Ways to Identify Vulnerabilities

Analyze historical data:

- Published lists of commonly occurring vulnerabilities and vulnerability types

Brainstorm plausible “defects”:

- Requirements defects
(missing, ambiguous, incomplete, or incorrect requirements)
- Component internal defects (component fails to meet its requirements)
- Component interaction defects (components meet individual requirements)

Consider human issues beyond “user error”:

- Human limitations, financial and schedule pressures, psychology, etc.

Representative ZATS Safety Vulnerability Requirements

Vulnerability:

Taxi Doors Cannot Lock: A ZATS taxi cannot lock its closed doors when moving.

Vulnerability prevention requirement:

Lock Taxi Doors: When its doors are closed, each ZATS taxi shall be able to lock and unlock its doors with a success rate of at least 99.99%.

Vulnerability detection requirement:

Detection of Unlocked Taxi Doors: When it is moving and its doors are closed, each ZATS taxi shall be able to determine if its doors are unlocked at least 99.99% of the time.

Vulnerability reaction requirement:

Reaction to Unlocked Taxi Doors: If any ZATS taxi is moving and detects that its doors are not locked, then the taxi shall :

- Notify the ZATS operator within 5 seconds at least 99.99% of the time
- Warn the taxi passengers to stay away from the doors until the taxi has berthed at the next taxi station within 1 second at least 99.99% of the time

Representative ZATS Security Vulnerability Requirements

Vulnerability:

Server Infected with Malware: A ZATS server is infected with malware.

Vulnerability prevention requirement:

Malware Infection Prevention: ZATS servers shall not be infected with known malware at least 99.99% of the time.

Vulnerability detection requirement:

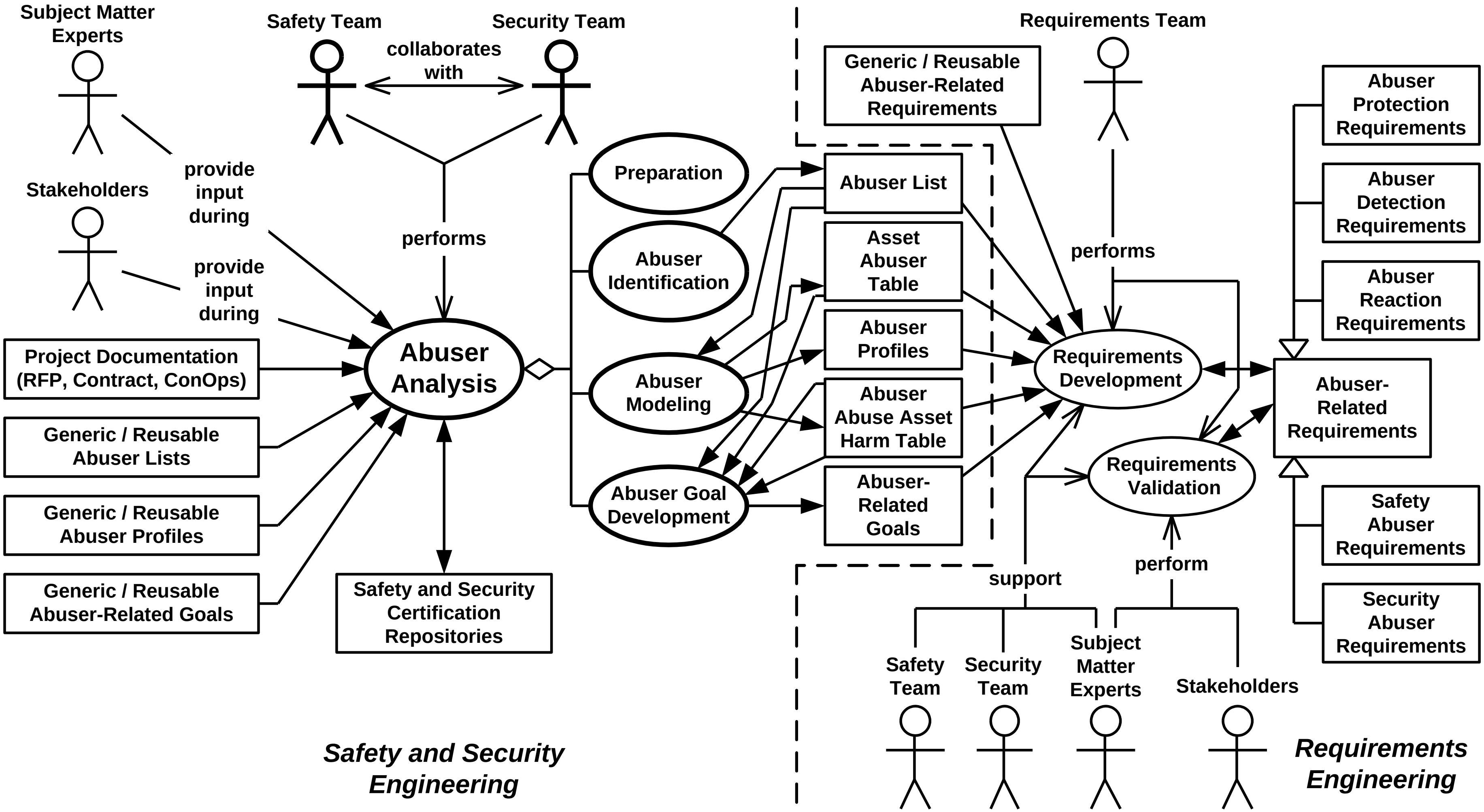
Malware Infection Detection: Each ZATS shall detect when it is infected with known malware at least 99.99% of the time.

Security vulnerability reaction requirement:

Malware Infection Reaction : When a ZATS server detects that it is infected by malware, it shall:

- Delete the malware at least 95% of the time
- Quarantine any malware it could not remove at least 99% of the time
- Notify the ZATS operator of the infection and the status (i.e., malware deleted, malware quarantined, or malware unaffected) at least 99% of the time

Abuser Analysis



Abuser

Abuser

any person or thing external to the system, the actions of whom or which either has or could have caused harm to a valuable asset (including the system)

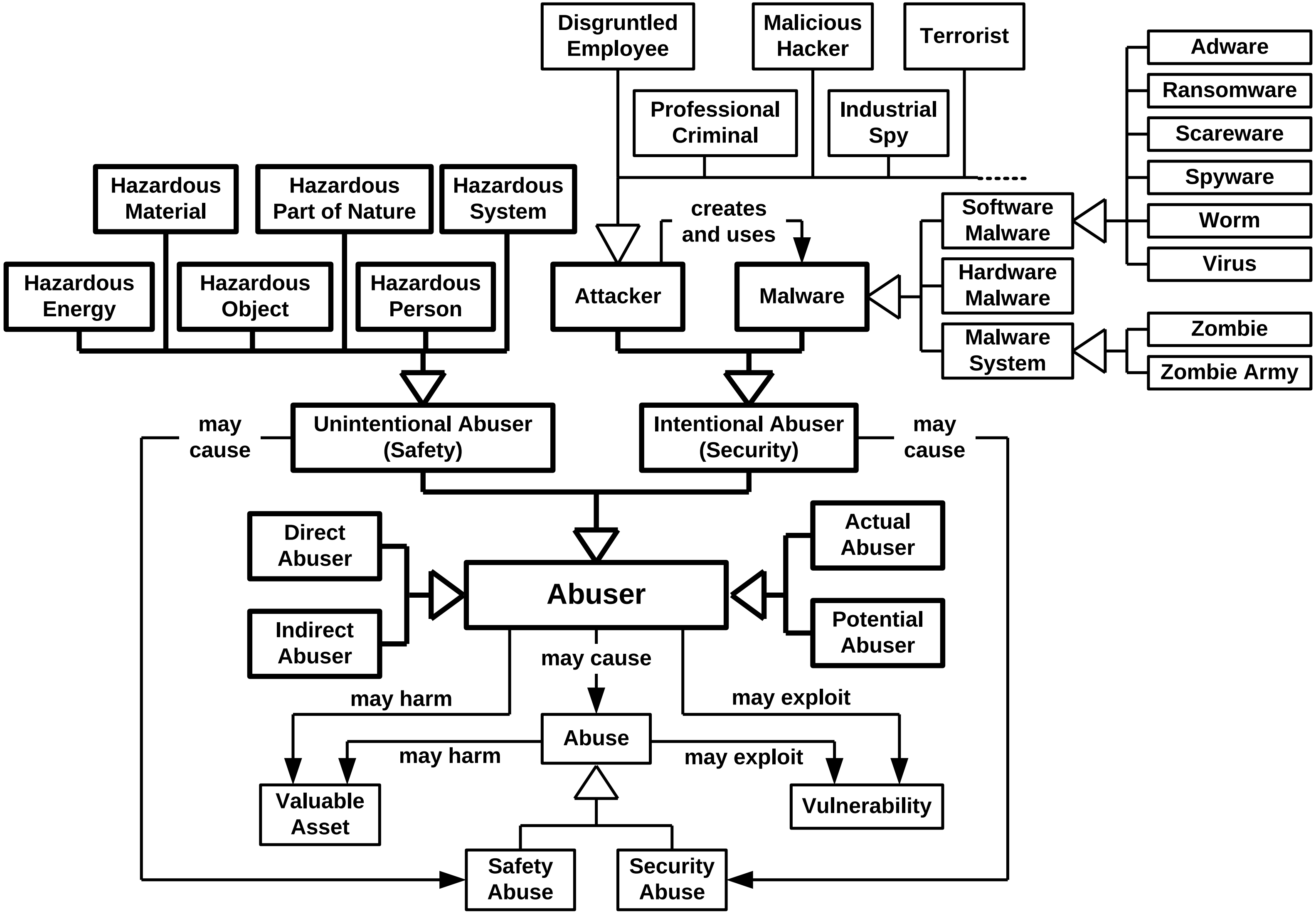
Unintentional Abuser (safety)

any abuser, the *unintentional* actions of which either has or could have caused a mishap (accident or safety incident)

Intentional Abuser (security)

any abuser, the *intentional* actions of whom or which either has or could have caused a misuse (attack or security incident)

Types of Abusers



Abuser Profiles

Abuser Profile

a description of a specific type of abuser, typically including:

- *Means* including tools, training, expertise, support, and time
- *Motive* including both desired harm and risk aversion (for attackers)
- *Opportunity* including access to system and valuable assets
- *External influences* such as project cost and schedule pressures, personal financial pressures
- *Abuser weaknesses* such as human mental and physical limitations, untreated addictions and mental illness,

Profiles are highly reusable.

Profiles more commonly used for attackers than non-malicious abusers, but are useful for both.

Representative Potential ZATS Abusers

Accidental human abuser:

- Maintainers (who make mistakes if inattentive, careless, or fatigued)

Accidental external system abuser:

- Electrical power grid (which may provides current or voltage spikes and surges, sustained overvoltage or undervoltage, complete power loss, random noise, or electromagnetic interference)

Accidental environmental abuser:

- Storm (which may provide high winds, hail, heavy snow, ice, etc.)

Attacker (malicious human abuser):

- Industrial spy (who may steal proprietary data)

Malware (malicious nonhuman abuser):

- Adware, ransomware, scareware, spyware, virus, or worm

Representative ZATS Abuser Requirements

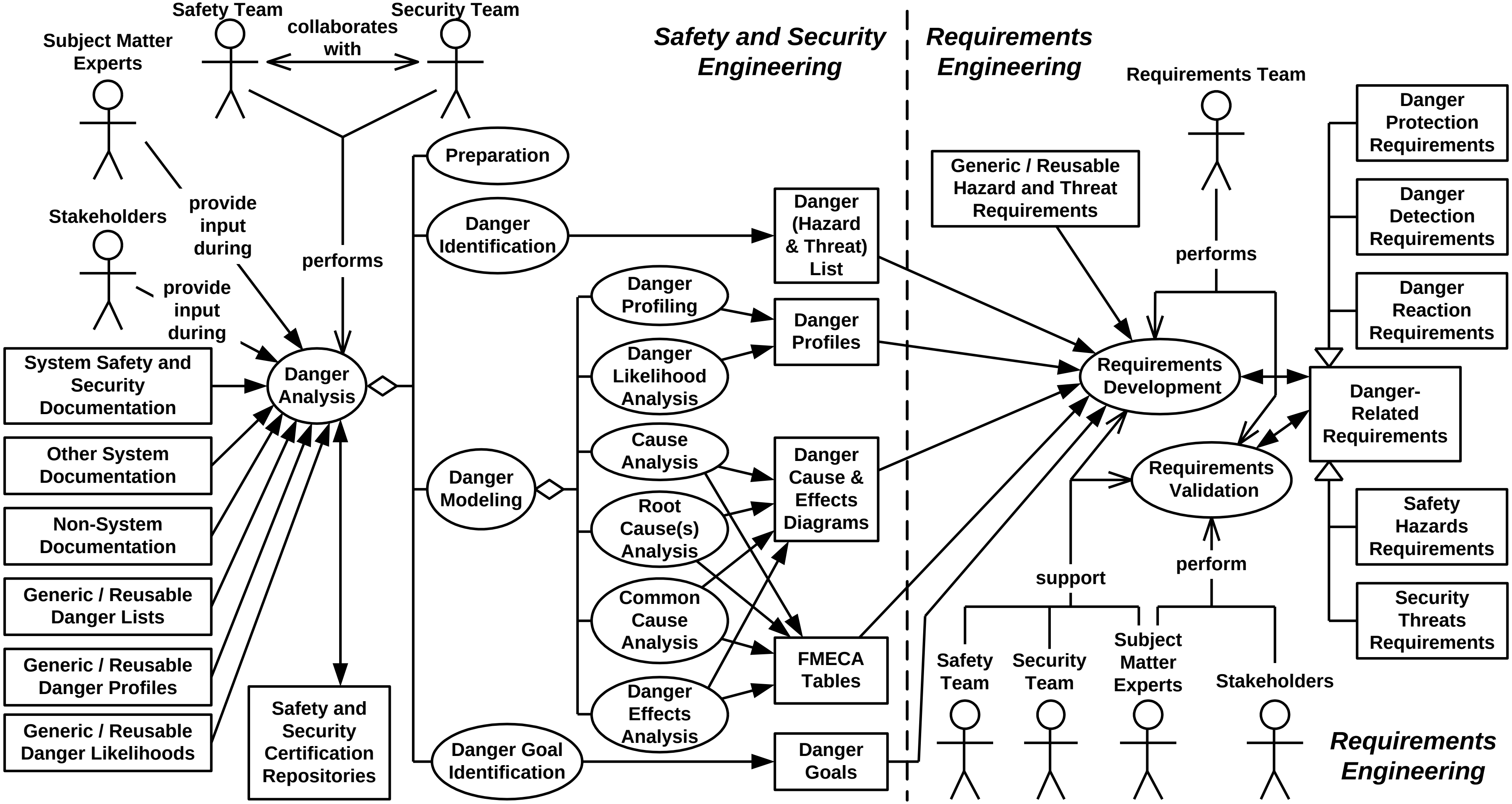
Safety abuser prevention requirement:

ZATS shall ensure that its operations and maintenance facility is beyond reach of the nearby river, even in the event of 100 year floods.

Security abuser prevention requirement:

ZATS shall prevent attackers of type X with profile Y from successfully causing the loss of confidentiality and integrity of sensitive information Z by preventing attacks lasting no more than 8 total hours at least 99% of the time.

Danger Analysis



Dangers

Danger

a potential or actual situation that increases the likelihood of one or more related abuses

Hazard

a danger that increases the likelihood of mishaps (safety)

Threat

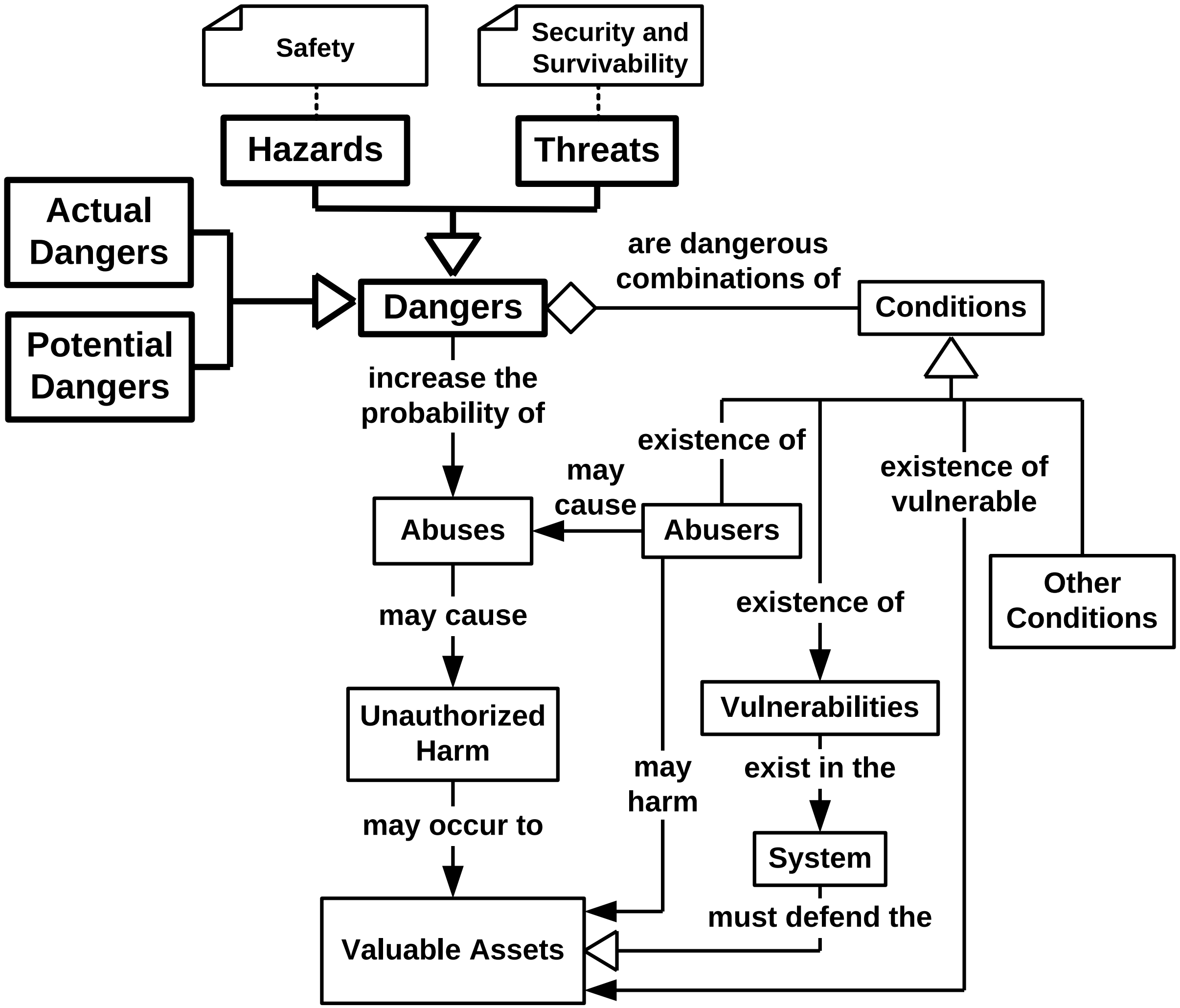
a danger that increases the likelihood of misuses (security)

A danger consists of one or more *conditions* concerning the *existence* of:

- Vulnerable valuable assets that can be harmed by abuses
- System-*internal* vulnerabilities or other system-*internal* conditions, states, or modes
- System-*external* abusers or other system-*external* conditions, states, or modes

Dangers are sets of concurrent conditions, whereas abuses are networks of events.

Types of Dangers



Identifying Dangers (Hazards and Threats)

Several approaches exist that can be used to identify dangers:

- **Abuse-Based Identification** – Consider each identified abuse to identify the associated dangers that increase the probability of the abuse occurring.
- **Abuser-Based Identification** – Consider each identified abuser to identify the associated dangers that include the existence of the abuser as a component condition. This is especially useful for identifying security threats based on generic types of attackers and malware.
- **Asset-Based Identification** – Consider each valuable asset to identify the dangerous conditions that can lead to that asset being harmed.
- **Task-Based Identification** – Consider each task performed by a “user” of the system to identify the relevant dangers that can exist. Observe the users while performing their tasks.
- **Use-Case-Based Identification** – Consider each use case and use case flow to identify the associated dangers.
- **Vulnerability-Based Identification** – Consider each vulnerability to identify the associated dangers.

Traditional Hazard Analysis (Safety)$_1$

As used in the safety community, hazard analysis usually implies the analysis of assets, harm, incidents, hazards, and risks.

Hazard analysis often occurs multiple times before various milestones:

- Preliminary Hazard Analysis (PHA)
- System Hazard Analysis (SHA)

Hazard analysis should probably be performed continuously.

Traditional Hazard Analysis (Safety)$_2$

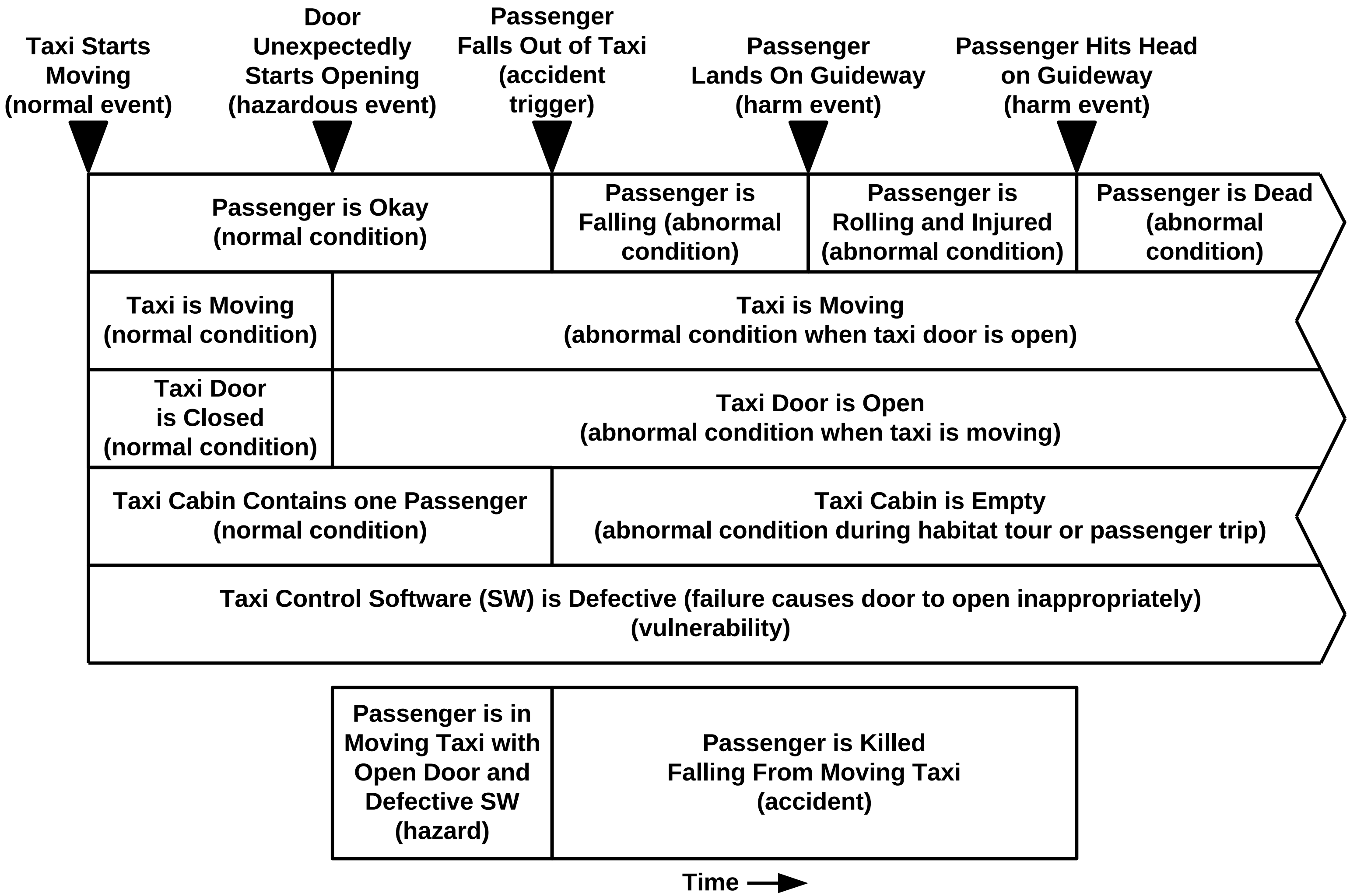
Traditional hazard analysis techniques:

- Come from reliability analysis
- Concentrate on individual component failures
- Do not address all (or even most) safety concerns
- Are inadequate for software and "human error"

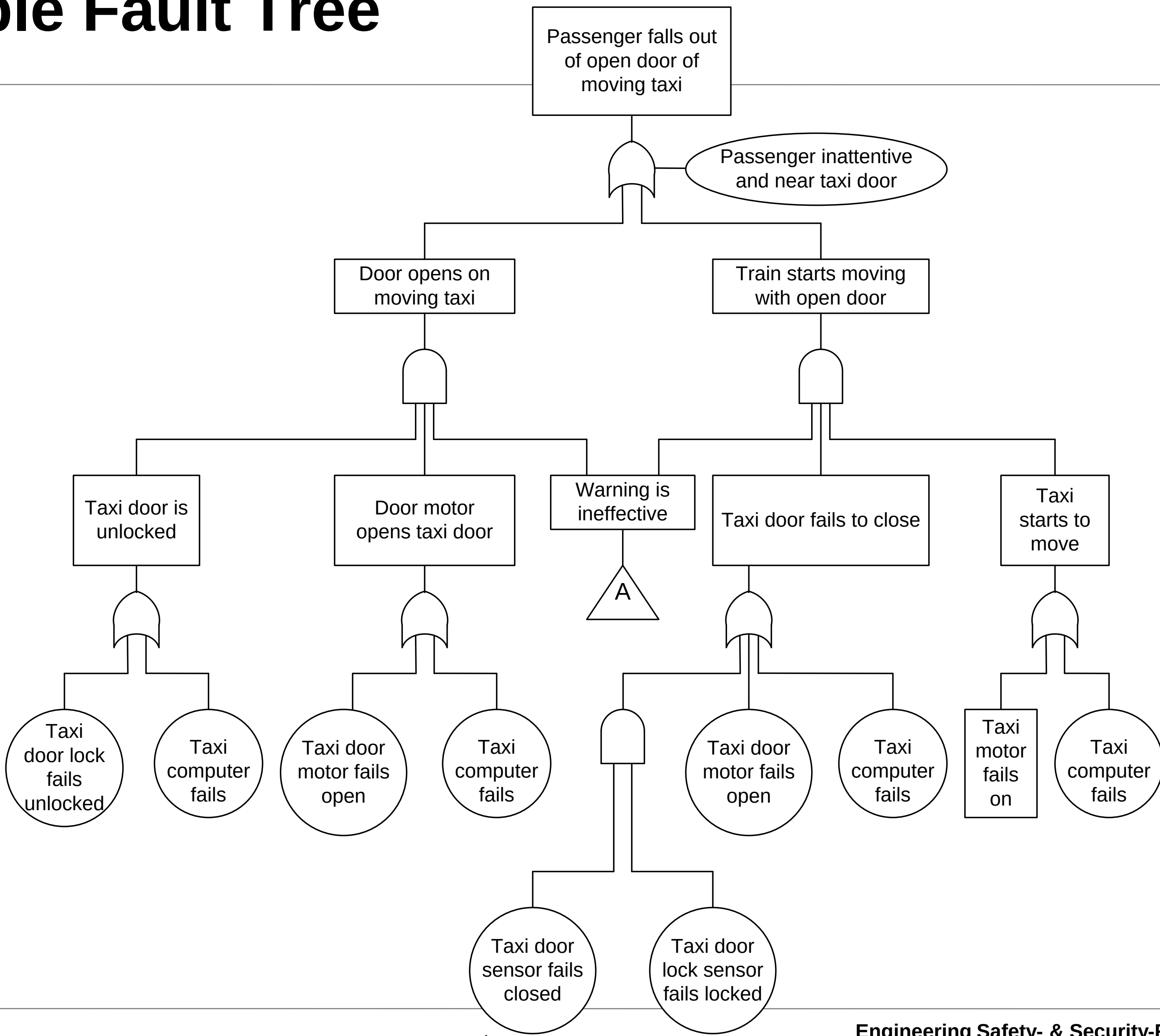
Traditional techniques borrowed from reliability include:

- Event Tree Analysis (ETA)
- Fault Tree Analysis (FTA)
- Hazard Cause and Effect Analysis (HCEA)
- Failure Mode Effects Criticality Analysis (FMECA)

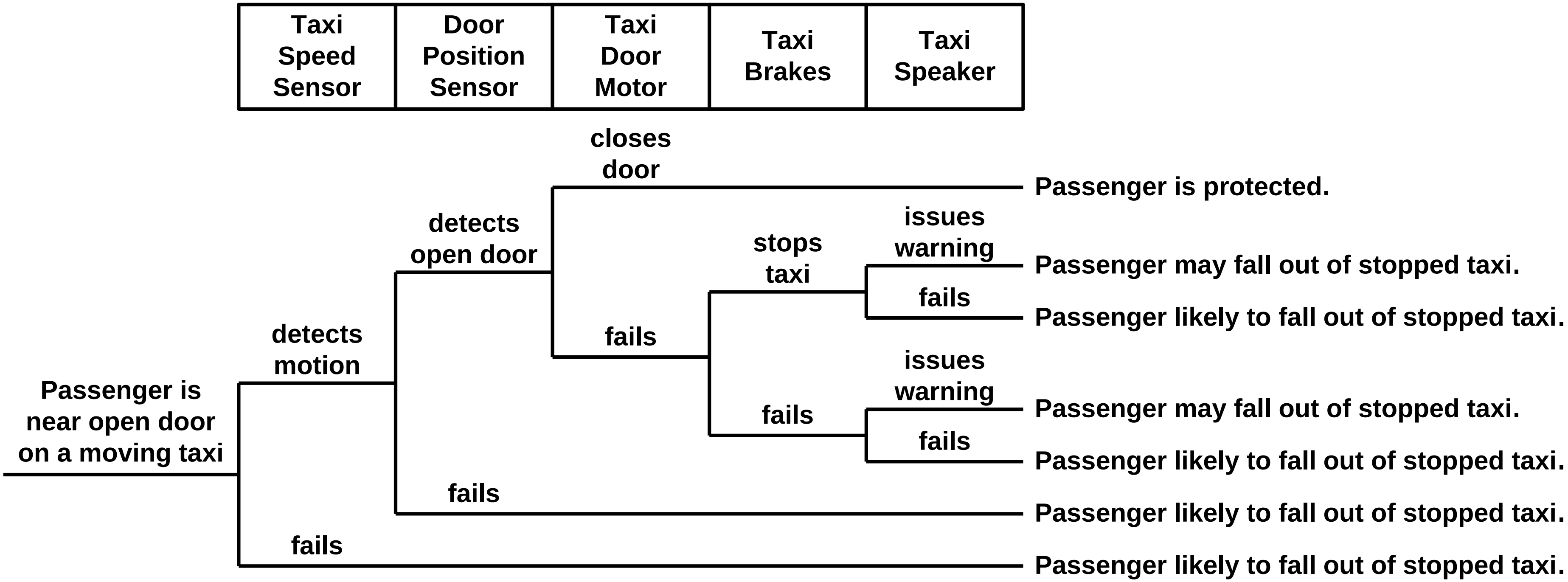
Example ZATS Hazard, Events, Harm, and Assets



Example Fault Tree



Example Event Tree



Example ZATS FMECA Table

Component that fail	Failure Mode	Failure Cause	Failure Effect	Failure Severity	Failure Likelihood	Criticality (Risk)	Safety Controls
Accelerometer (Taxi sensor)	No data, Bad data (0, last value, maximum value)	Hardware failure, loss of electrical power, wiring fails	Excessive acceleration or deceleration causing passenger injury	Minor	Low	Moderate	Hardware redundancy, High-reliability COTS component, SW fault tolerance
Computer Hardware (Taxi)	Loss of function (complete or intermittent), bad data	CPU, electrical power (loss or spike), hard drive, motherboard, or RAM failure, high temperature	Taxi not controllable (e.g., braking, power, steering), collision between taxis, collision with guideway, unexpected or emergency braking	Severe	Moderate	Critical	Hardware redundancy, High-reliability COTS components, temperature sensor, SW fault tolerance
Computer Software (Taxi)	Loss of function (complete or intermittent), incorrect function, bad timing of function	CPU, electrical power (loss or spike), hard drive, motherboard, or RAM failure, high temperature	Taxi not controllable (e.g., braking, power, steering), collision between taxis, collision with guideway, unexpected or emergency braking	Severe	High	Critical	SEAL 1 applied (e.g., SW fault tolerance, real-time operating system, safe language subset, formal specification of core functions, etc.)

Hazard Analysis (Safety) – A Relook$_1$

Safety and reliability are not the same:

- Unreliable systems can be safe (failsafe systems – failures do not cause harm)
- Reliable systems can be unsafe (poor requirements lead to accidents)

Reliability-based safety analysis assumes:

- Accidents are due to the independent failures of individual components
- Defective components fail to meet their requirements

However, most accidents are due to:

- Poor (missing, ambiguous, incomplete, or incorrect) requirements
- Component interaction defects, whereby the individual components meet their requirements and are thus reliable
- Software, that glues complex components together
- Human causes of accidents beyond “user” errors:
  - Management and developer rather than operator (e.g., pilot) errors
  - Unavoidable human limitations, financial and schedule pressures, psychology, sociology, etc.

Hazard Analysis (Safety) – A Relook$_2$

Modern Hazard Analysis:

- Add requirements defects to component defects.
- Add component interaction failures to individual component failures.
- View safety and security as emergent system characteristics arising from the interaction among system components and the system's environment rather than the characteristics of individual system components in isolation.
- Emphasize software and humans over hardware.

Danger Analysis:

- Emphasize safe and secure interfaces.
- View safety and security not as a reliability problem but rather as a problem of controlling dangers by enforcing safety- and security-related requirements.
- Address indirect and systemic causes of accidents and successful attacks.

Representative ZATS Hazard - Overspeed

Primary Condition:

- Taxi exceeds the speed limit.

Existence of one or more **Vulnerable Assets:**

- Guideways
- One or more passengers
- Passenger property
- Taxis
- Taxi stations

Existence of one or more system **Vulnerabilities:**

- Defective speed sensor
- Defective power braking system
- Defective taxi processor
- Defective associated software

Existence of one or more **Non-malicious Abusers:**

- None (taxi speed is automatically controlled by the taxi)

Representative ZATS Hazard Requirements - Overspeed

Overspeed Requirements:

- ZATS taxis shall not exceed the speed limit by more than one mile an hour more than 0.001 % of the time.
- ZATS taxis shall not exceed the speed limit by more than five miles an hour more than 0.0001 % of the time.
- ZATS taxis shall not exceed the speed limit by more than one mile an hour for durations longer than 10 seconds more than 0.0001 % of the time.

Representative ZATS Threat (Bank Card Confidentiality)

Primary Condition:

- Travel Card Vending Machine computer connected to Internet via Operations Facility Server

Existence of one or more **Vulnerable Assets:**

- Passengers' bank card information

Existence of one or more system **Vulnerabilities:**

- Lack of encryption during transmission
- Lack of encryption during storage
- Weak encryption
- Lack of firewalls or firewalls improperly configured

Existence of one or more **Malicious Abusers:**

- Cyber-thief:
  - *Means*: Expertise plus relevant hacking tools
  - *Motive*: Greed
  - *Opportunity*: Computer access to the Internet

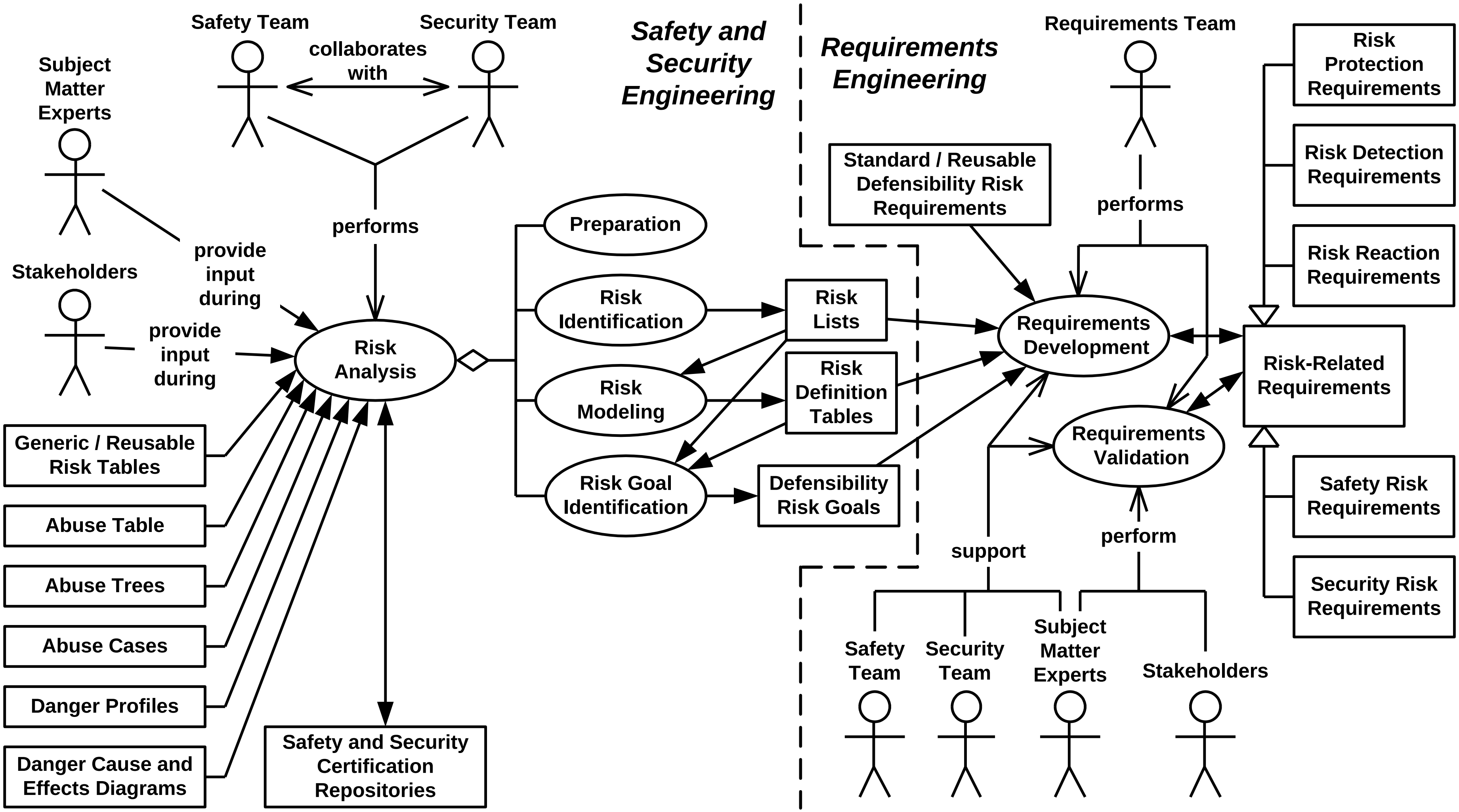
Representative ZATS Threat Requirements

Threat Requirements:

- ZATS shall encrypt all passenger bank card information while stored within ZATS.
- ZATS shall not store passenger bank card information in the travel card vending machines.
- ZATS shall encrypt all passenger’s bank card information sent to the bank card processing gateway.

  Rationale: to make the bank card information unusable by cyberthieves if accessed.

Defensibility Risk Analysis



Defensibility Risks

Risk

the *expected* or maximum credible amount of unauthorized harm

Traditionally calculated as the product of the:

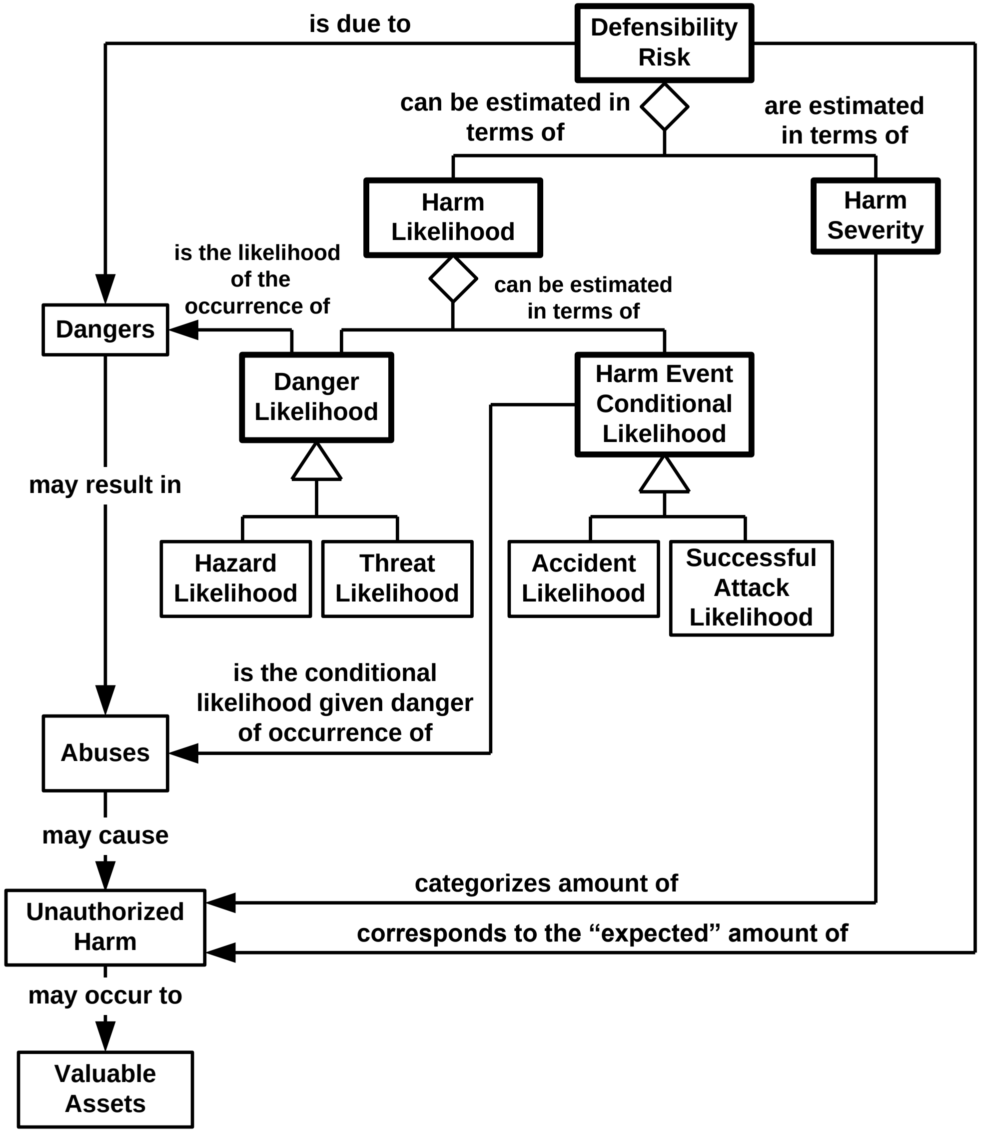
- probability that harm will occur
- the amount or severity of the harm

Defensibility Risks

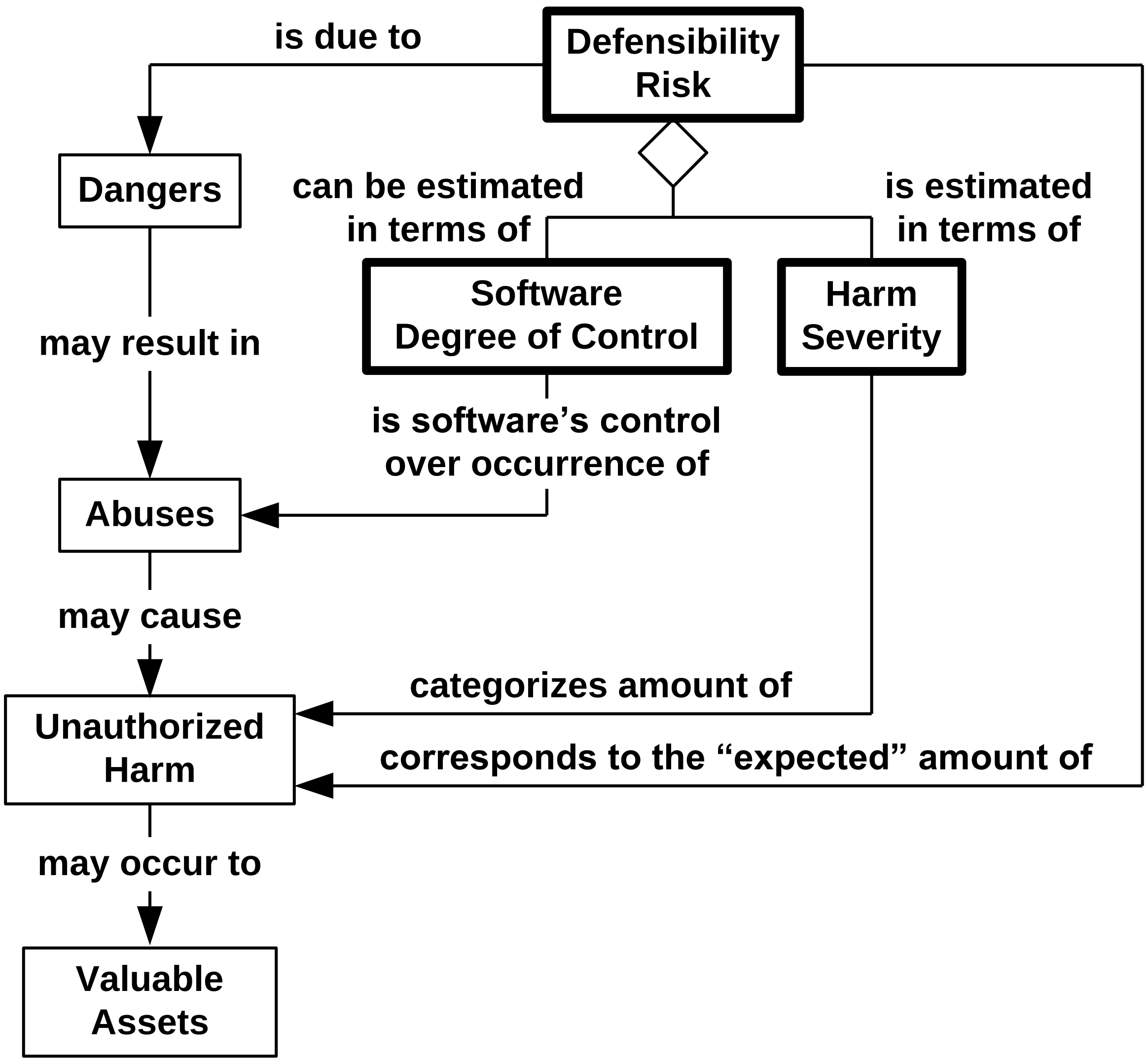
the expected or maximum credible amount of unauthorized harm that can occur

- To [a type of] valuable assets
- Due to a specific [type] of abuse
- Due to the existence of [a type of] vulnerability, abuser, or danger

Defensibility Risks



Risk in terms of Software Degree of Control



The ZATS Safety Risk Matrix

Safety risk matrix defines safety risk as a function of:

- Harm severity
- Frequency of *harm occurrence*, *abuse* occurrence or *danger* existence

Safety Risks / Safety Assurance Levels (SALs)					
	Frequency of Abuse / Danger Occurrence				
Harm Severity	**Frequent**	**Probable**	**Occasional**	**Remote**	**Implausible**
Catastrophic	Intolerable	Intolerable	Intolerable	High	Medium
Critical	Intolerable	Intolerable	High	Medium	Medium
Major	High	High	Medium	Medium	Low
Minor	High	Medium	Low	Negligible	Negligible
Negligible	Medium	Low	Negligible	Negligible	Negligible

Representative ZATS Defensibility Risk Goals and Requirements

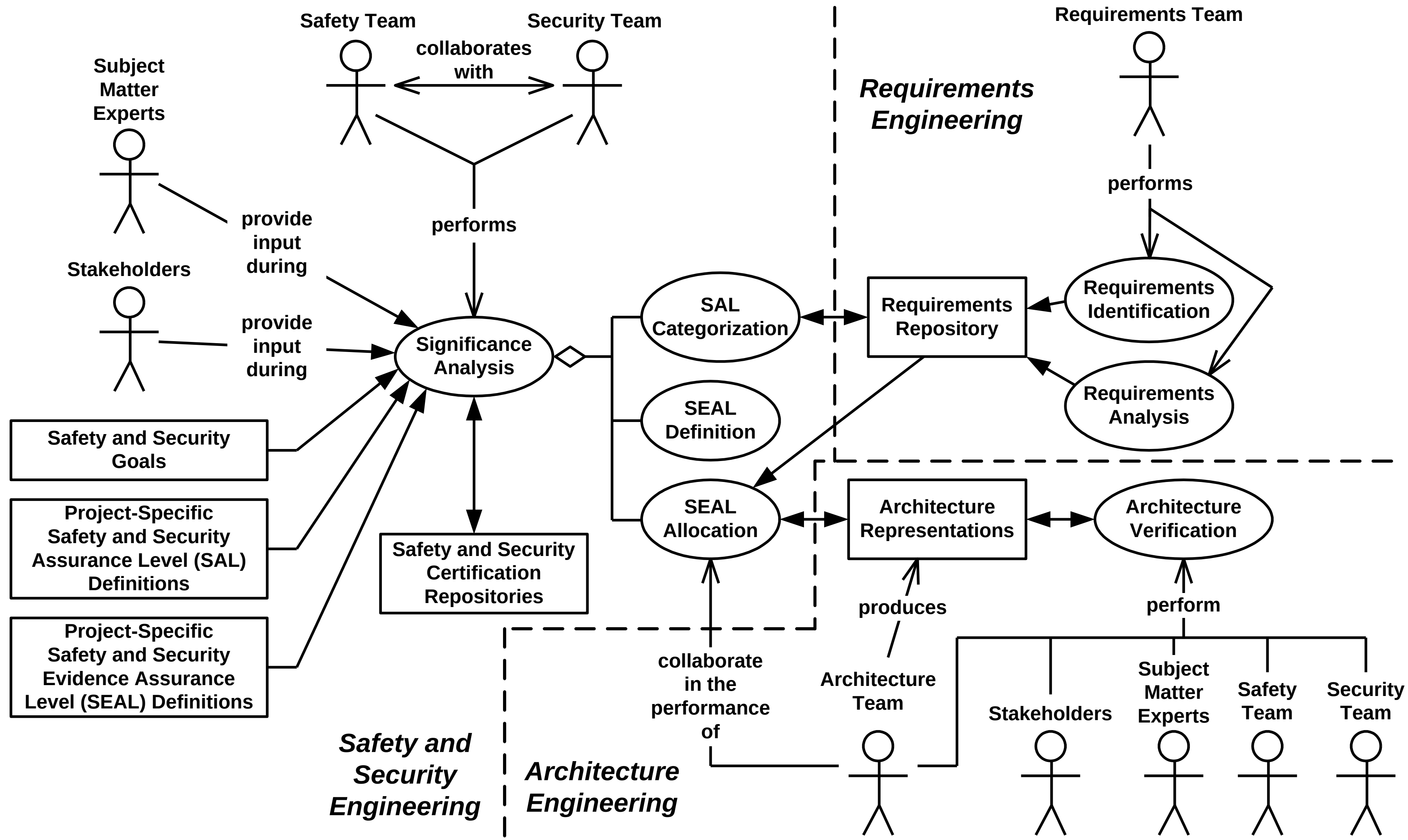
Defensibility Risk Goal

- The risk of fires in ZATS buildings will be acceptable to zoo management and the Metropolitan Zoo Authority.

Defensibility Risk Requirement

- The risk of fires in ZATS buildings shall be less than or equal to the fire risk to other zoo buildings.

Defensibility Significance Analysis



Safety/Security Assurance Levels (SALs)

Safety/Security Assurance Level (SAL)

a category of requirements based on their maximum associated potential harm severity

SALs categorize *requirements.*

SALs can be determined for:

- Individual requirements.
- Groups of related requirements (e.g., features or functions)

SALs should be clearly and unambiguously defined.

SEALs

Safety/security evidence assurance level (SEAL)

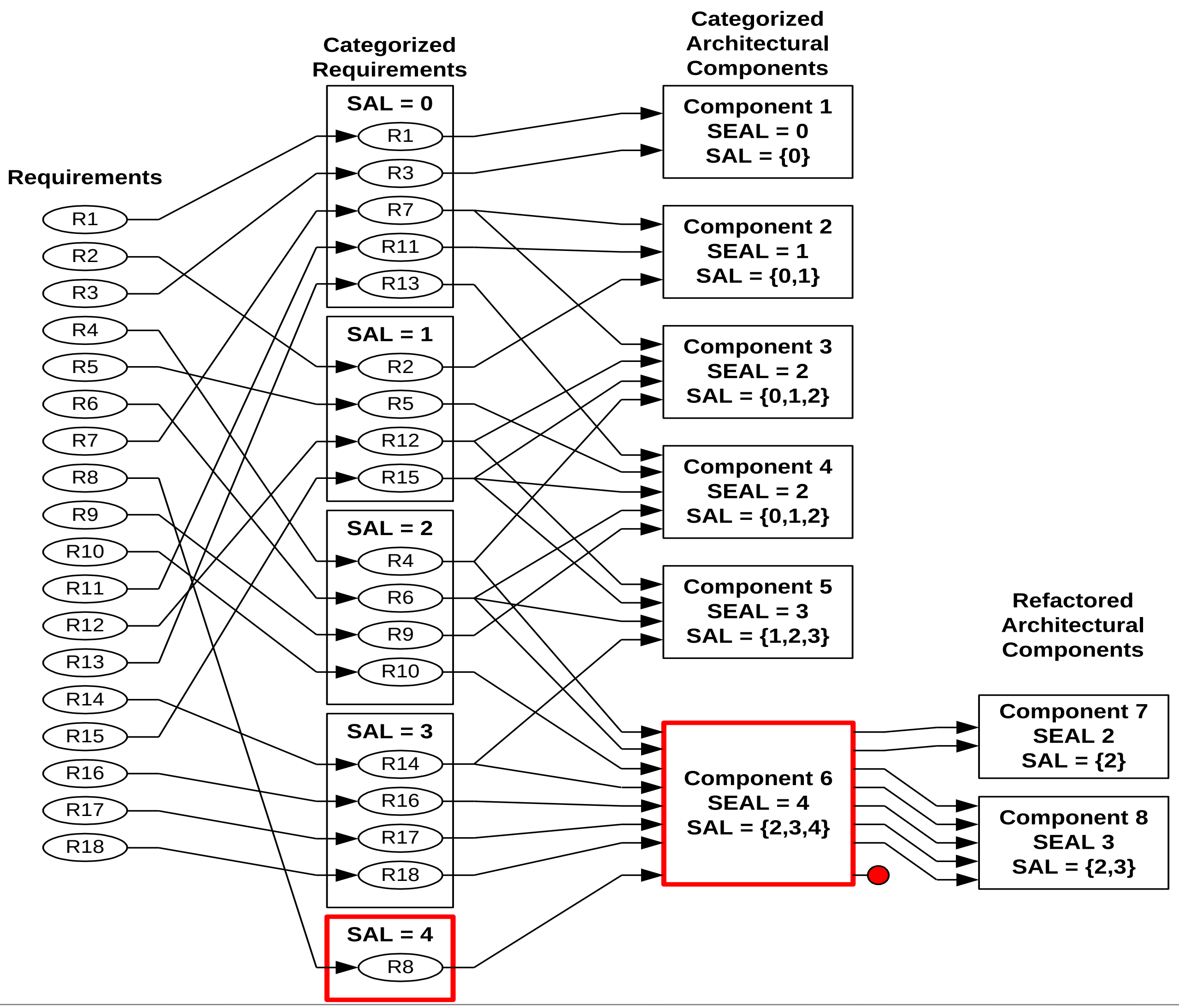
> a category of architectural components based on the highest SAL of the allocated and derived requirements they implement

SEALs categorize *architectural components* that helps fulfill these requirements.

SEALs define increasingly strict associated development methods needed to assure fulfillment of the highest associated SAL requirement.

SEALs should be clearly and unambiguously defined.

SAL versus SEAL



Safety/Security Evidence Assurance Level (SEAL)

High SEALs require more rigorous development method (including better requirements and architecture engineering):

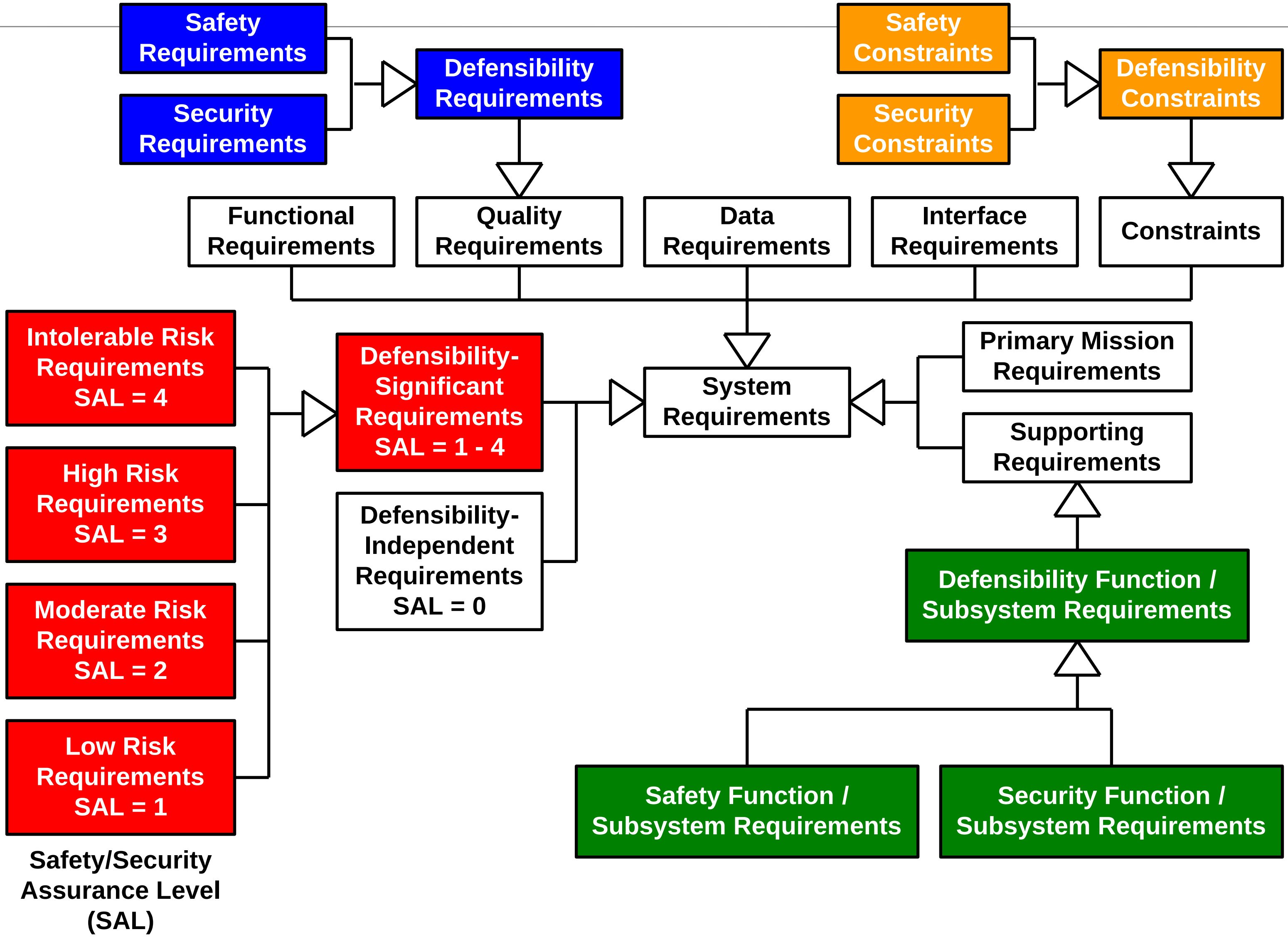
- Formal specification of requirements
- Fagan inspections of requirements
- Quality assessments of the architecture

Often SEALs only apply to design, coding, and testing:

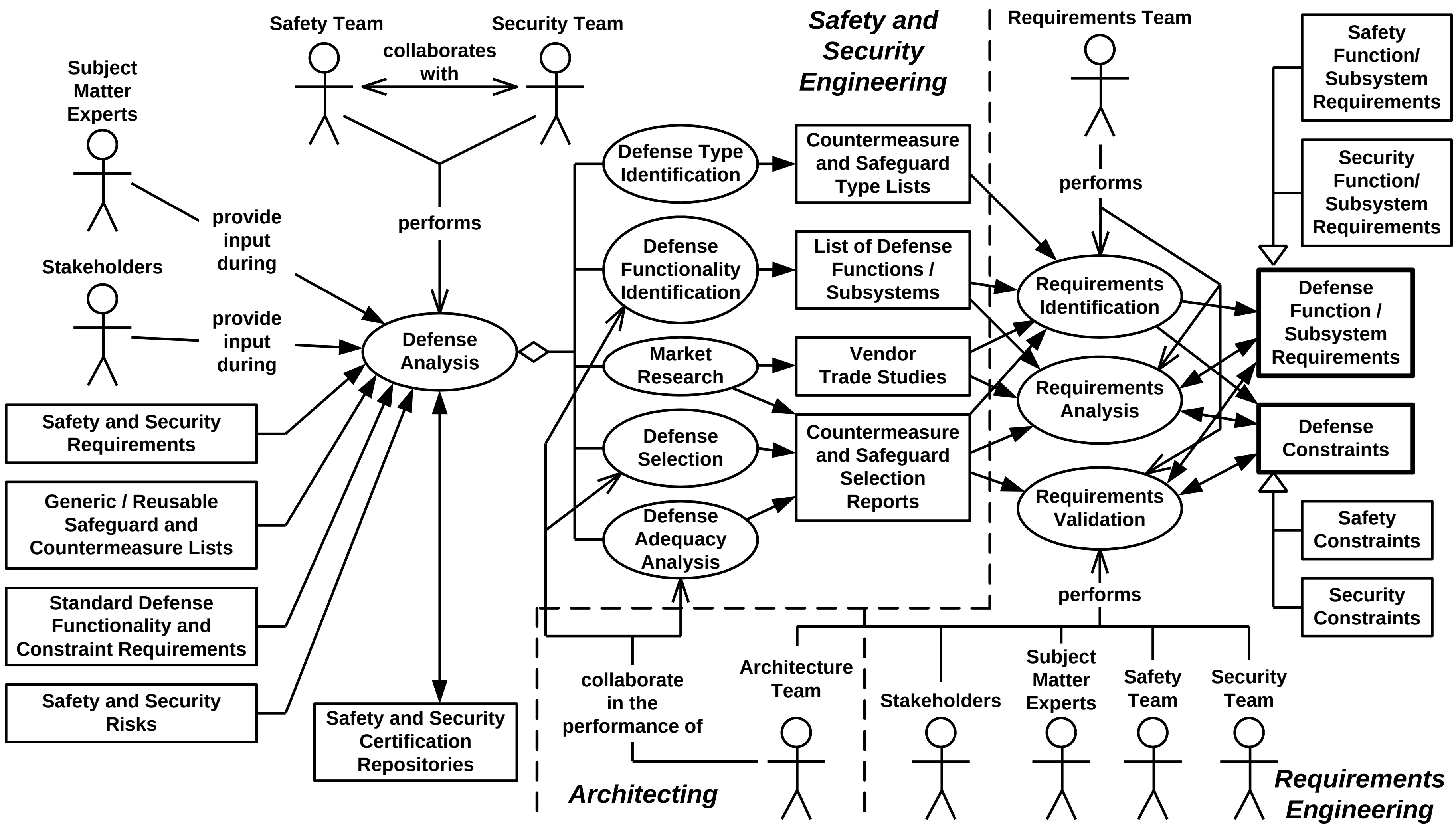
- Design inspections
- Formal derivation of code and proof of correctness
- Model-Based Development (MBD) of software from models
- Safe subset of programming language
- Extra testing (test types and test completion criteria)

Because of the added cost and schedule, architects often attempt to minimize the scope of components having high SEALs.

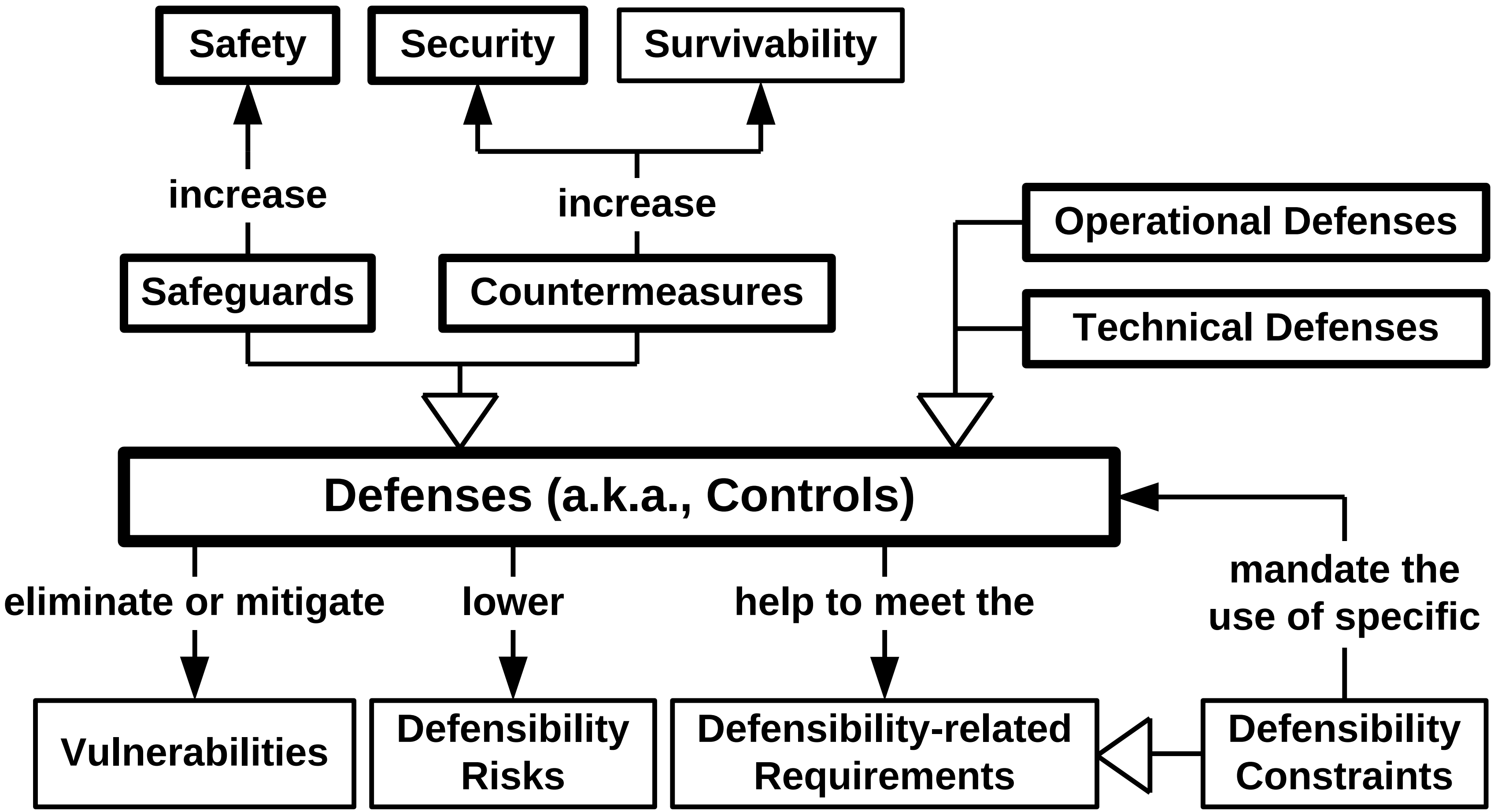
Defensibility-Significant Requirements (in red)



Defense Analysis



Types of Defenses



Representative ZATS Safety Constraints

Each ZATS taxi shall contain a taxi door subsystem consisting of two doors, two door position sensors, two door locks, two door lock sensors, one door motor, and associated computer hardware and software.

ZATS guideways shall be constructed from pre-stressed reinforced concrete that can support at least 150% of the maximum expected loading.

> *Rationale*: This constraint practically prevents the risk of guideway collapse due to credible excessive loading.

The bottom of ZATS guideways shall be a minimum of 17 feet above ground level.

> *Rationale*: Elevation of the guideways provides patrons with good visibility of the animals in their habitats, safely separates zoo patrons from the animals, eliminates the possibility of taxis running down patrons on zoo walkways, and eliminates the possibility of collision between taxis and patrons' vehicles in the parking lots. The minimum height of the bottom of the guideway was also chosen to exceed the American Zoological Association (AZA) recommendation of 16 feet.

Representative ZATS Security Constraints

ZATS shall use a COTS public key encryption/decryption product to protect sensitive data.

Rationale: Encryption and decryption is the most effective countermeasure for ensuring the confidentiality of sensitive data. Using a COTS product is best in terms of cost and schedule, and COTS products tend to use public key encryption.

ZATS shall use of a COTS antivirus product to prevent server infection by malware.

Rationale: ZATS servers are threatened by the existence of software malware (e.g., viruses, worms, and Trojans). COTS antivirus products maintain current definitions of such malware and are of higher quality and much less expensive than developing and maintaining such a subsystem in-house.

We Are Here

Conclusion$_1$

Engineering safety- and security-related requirements requires *appropriate*:

- Concepts
- Methods
- Techniques and tools
- Expertise

There are four types of safety- and security-related requirements:

- Safety and security quality requirements
- Safety- and security-significant requirements
- Safety and security function/subsystem requirements
- Safety and security constraints

These different types of requirements need to be identified, analyzed, and specified differently.

Conclusion$_2$

Processes for requirements engineering, safety engineering, and security engineering need to be:

- Properly interwoven.
- Consistent with each other.
- Performed collaboratively and in parallel (i.e., overlapping in time).

Final Thoughts

Look for my upcoming book by the same title to be published by Auerbach in 2010.

Questions?

For more information contact:

Donald Firesmith
Acquisition Support Program (ASP)
Software Engineering Institute (SEI)
Pittsburgh, Pennsylvania, USA 15213
412-216-0658 (cell)
dgf@sei.cmu.edu