Requirements Engineering for Trusted Embedded Systems

José L. Fernández
Industrial Enginering School
Madrid Technical University (Madrid, Spain)
jose.fernandez@incose.org

Scope of the presentation

- We present the approach followed to specify trust requirements for the ITEA2 project named TECOM (Trusted Embedded Computing).

- Using these requirements, the project partners develop embedded systems in diverse areas such as home control, video surveillance, digital content delivery, automotive, and communications gateways for remote maintenance.

- The approach for the engineering of trust requirements presented here is based on
  - using a common conceptual model to identify the trust concepts for embedded systems,
  - a quality model for the decomposition of each trust dimension into quality factors and subfactors,
  - and the usage of a defined process to elicit and specify trust requirements for the diverse demonstrators to be developed in the project.

- Trusted Embedded Computing is the ITEA 2 project 06038.
- The project partners are Atego (France), EADS DS (France), Fagor Electrodomesticos (Spain), Ikerlan (Spain) Technikon (Austria), Thomson (France), Trialog (France), Universidad Politécnica de Madrid (Spain) Universidad Politécnica de Valencia (Spain), and Visual Tools (Spain).
- Project began September 2007 and will be finished October 2010

TECOM

- The strategic objective of TECOM is to investigate solutions and architectures for embedded systems platforms meeting the specified trust requirements.
- The TECOM research approach applies the concept of trusted platforms to real-time embedded systems.
- The following results will be made available at the end of the project:
  - Security solutions based on two different approaches (hypervisors and middleware)
  - Demonstrators using the security solutions (mobile applications, home control, video-surveillance)
  - A study in two sector domains (automotive, avionics).

Trusted embedded systems

What we mean by trust and which are the particularities of embedded systems?

Trust and its properties

Muskens defines trust as the degree to which a trustor has a justifiable belief that the trustee will provide the expected function or service.

- Trust is directed. It is an oriented relationship between the trustor and the trustee.
- Trust is subjective. It is based on a person's confidence and opinions.
- Trust is context-dependent.
- Trust is measurable. Levels or values of trust can be computed.
- Trust is not necessarily symmetrical. A may trust B, but this does not necessarily imply that B will trust A.
- Trust can evolve. Trust value can change over time.
- Trust can be history-dependent. The present level of trust may be affected by previous experience.
- Trust can be a composite property. In TECOM, integrity, availability, safety, reliability, security and survivability are considered as trust dimensions.

Particularities of embedded systems

- Storing data on an embedded system creates privacy concerns. Storing sensitive information on a single embedded system rather than on multiple servers minimizes the number of locations where an attack can occur.
- Embedded systems have resource constraints in terms of memory, computational capacity and energy, a situation which can pose several risks.
- In terms of memory, sophisticated public key cryptography techniques might be infeasible for embedded systems.
- An energy intensive trust mechanism can cause an embedded system to perish from battery exhaustion before it can perform useful work.
- Unlike transaction-oriented business computing, embedded systems often perform periodic computations with real-time deadlines, so the embedded system becomes vulnerable to attacks designed to disrupt system timing.

Trust requirements engineering

The approach used for elicitation and specification of trust requirements for TECOM applications

Trust requirements engineering

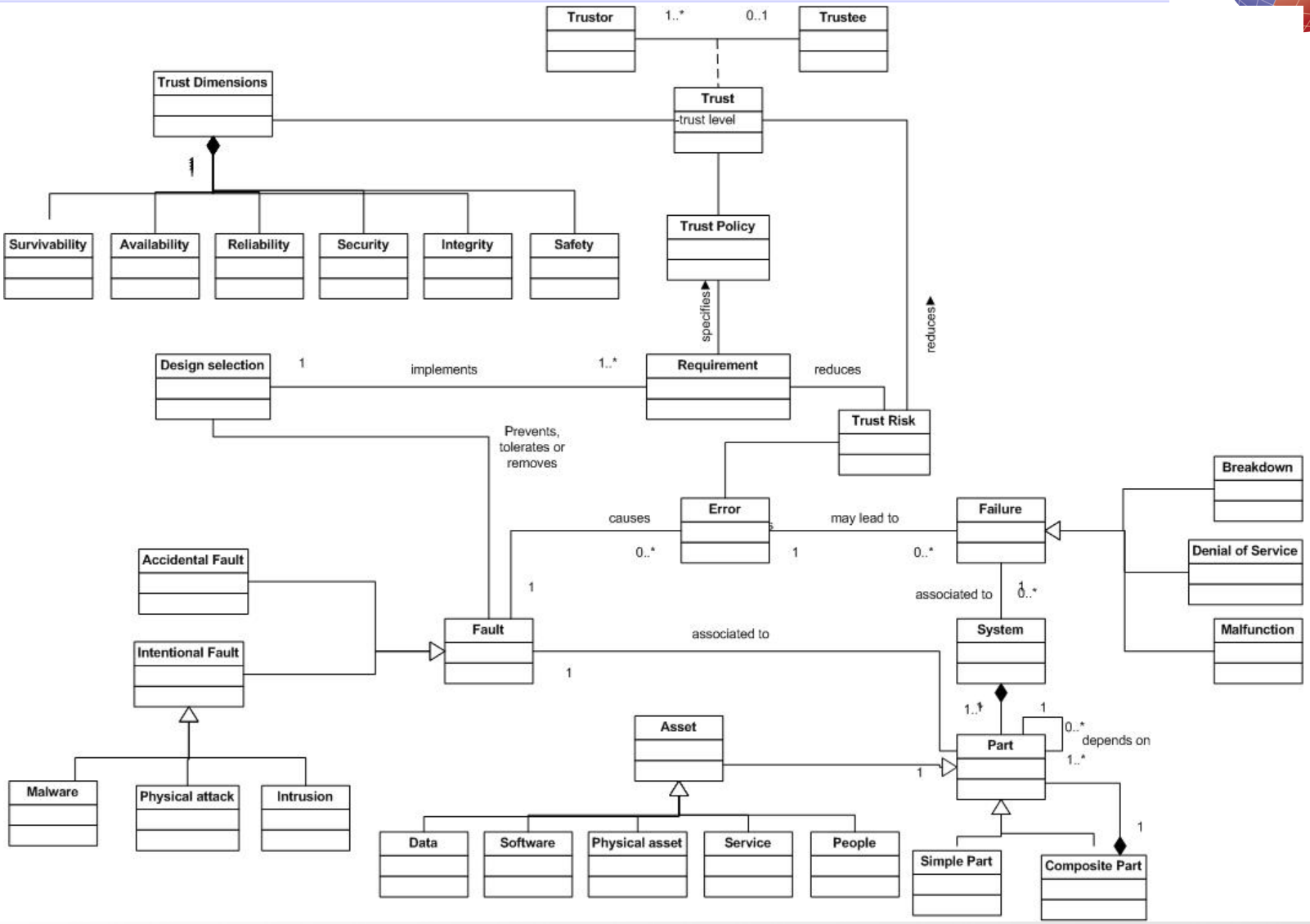
- The approach for the engineering of trust requirements presented here is based on:

1. Using a common conceptual model to identify the trust concepts for embedded systems and their relationships.
2. Decomposing each trust dimension into quality factors and subfactors.
3. A defined process to elicit and specify trust requirements for the demonstrators to be developed in the project.
4. Finding commonalities among trust requirements specified for the diverse TECOM applications

Trust conceptual model

The conceptual model represents the trust concepts and their relationships while avoiding the ambiguities relative to similar concepts.

Trust conceptual model for embedded systems (I)



Trust conceptual model for embedded systems (II)

- The conceptual model uses the Avizenis fault⇒error⇒failure paradigm, frequently called AVI fault model, and extends it with new concepts, e.g. trust, trust dimension, requirement, design selection, and system.
- The conceptual model also represents some important relationships between the concepts by various lines and symbols. The following symbols represent relationships in the UML notation.
  - Associations that signify the relationship between two concepts that need to know each other: a line.
  - Composition that is the relationship between an element and its parts: a black diamond.
  - Specialization that is used when a concept is specialized in more specific ones: a hollow triangle.
- The design selections to be considered in trusted systems are not represented in the conceptual model. This is part of the architecture phase of embedded system development, and for this reason the design selection concept is not expanded in the model represented in previous slide.

Trust conceptual model for embedded systems (III)

- Some important concepts are:
  - **Asset:** Anything of value that should be protected from harm. An asset is a specialization of a system part.
  - **Trust Dimensions:** These dimensions can be aggregated in the so-called dependability quality factor. From the conceptual model point of view the following trust dimensions are taken into consideration:
    - **Integrity:** The absence of improper, intentional, or accidental system alterations. Integrity here is a more general concept that the integrity concept of security that considers only intentional faults.
    - **Availability:** Ensures that the system continues to operate in the face of certain anticipated failures.
    - **Reliability:** The capability of a system to perform consistently and precisely what it is expected to do.
    - **Security:** Ensures that the system resists intentional faults. Security is a complex and important dimension that also contains diverse factors, for example access control, attack detection, security integrity, freshness, physical protection, privacy and other factors described in slide 15.
    - **Survivability:** The capability to provide a level of functionality or service in adverse or hostile conditions.
    - **Safety:** The absence of catastrophic consequences on the user(s) and the environment.

Trust quality model

A quality model decomposes each trust dimension into quality factors (i.e. aspects, attributes or characteristics) and subfactors.

Trust quality model

To illustrate this, the requirements of the trust-security dimension for each TECOM system are grouped using the factors defined by Firesmith from the SEI:

- **Access Control** is the degree to which the system limits access to its resources only to its authorized externals (e.g., human users, programs, processes, devices, or other systems). The following are quality subfactors of the access-control quality factor:
  - **Identification** is the degree to which the system identifies (i.e., recognizes) its externals before interacting with them.
  - **Authentication** is the degree to which the system verifies the claimed identities of its externals before interacting with them. Thus, authentication verifies that the claimed identity is legitimate and belongs to the claimant.
  - **Authorization** is the degree to which access and usage privileges of authenticated externals are properly granted and enforced.
- **Attack/Harm Detection** is the degree to which attempted or successful attacks (or their resulting harm) are detected, recorded, and notified.
- **Availability Protection** is the degree to which various types of Denial of Service attacks are prevented from decreasing the operational availability of the system. This is quite different from the traditional availability quality factor, which deals with the operational availability of the system when it is not under attack.
- **Freshness** is the degree to which information is not a copy of another one received in the past. This factor prevents replay attacks.
- **Security Integrity** is the degree to which system assets are protected from intentional and unauthorized corruption. These system assets could be divided in several groups (data, equipment, people, software and service) as seen in a previous slide.
- **Nonrepudiation** is the degree to which a party to an interaction (e.g., message, transaction, transmission of data) is prevented from successfully repudiating (i.e., denying) any aspect of the interaction.
- **Physical Protection** is the degree to which the system protects itself and its assets from physical attack.
- **Privacy** is the degree to which unauthorized parties are prevented from obtaining sensitive information. Privacy includes the following subfactors:
  - **Anonymity** is the degree to which users' identities are hidden away from unauthorized parties.
  - **Confidentiality** is the degree to which sensitive information is not disclosed to unauthorized parties.
- **Prosecution** is the degree to which the system supports the prosecution of attackers.
- **Recovery** is the degree to which the system recovers after a successful attack. Recovery is also a factor of survivability.
- **Security Auditing** is the degree to which security personnel are enabled to audit the status and use of security mechanisms by analyzing security-related events.
- **System Adaptation** is the degree to which the system learns from attacks in order to adapt its security countermeasures to protect itself from similar attacks in the future.

Requirements engineering process

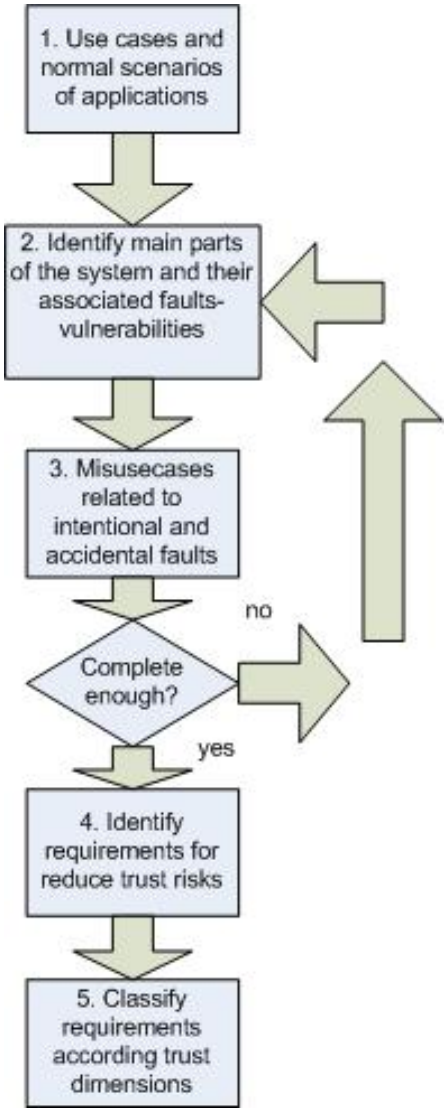
A defined process to elicit and specify trust requirements for the demonstrators to be developed in the project.

Requirements engineering process (I)

- This process encompasses requirements derived from real-life scenarios, as well as requirements based on the trust conceptual model presented in a previous slide.
- The requirements elicitation technique used is based on the "scenario" paradigm.
- The word scenario refers to the response of the system to a domain event.
- Scenarios may be of different classes: normal case scenarios, alternative case scenarios, exception case scenarios, and what-if scenarios.
- Misuse cases are a relatively new technique for describing a negative form of a use case. Misuse cases are useful in systems where there are concerns about security and/or safety
- Misuse cases are created for describing the fault–error scenarios by pairing system threats with system parts faults or vulnerabilities. This subprocess is represented as a loop in the next slide.

Requirements engineering process (II)

- This process is based on the identification of “positive” and “negative” scenarios and grouping them in use and misuse cases
- The subprocess of threats-vulnerabilities pairing ends when a complete enough criteria that is all system assets considered, is met.
- Step 4 consists of the specification of requirements that reduce the risk of each threat-vulnerability pairing.
- In step 5 these requirements are classified according to the trust dimensions defined in the conceptual model.



Finding commonalities among trust requirements specified for the diverse TECOM applications

Requirements commonalities and variabilities

- Although the embedded systems developed by TECOM partners had a wide range of scopes and functionalities, from the point of view of trust they were treated as a product line.
- Since the goal was to discover commonalities on trust requirements, an approach based on the identification of commonalities and variabilities was used.
- The TECOM requirements were classified into three groups:
  - **Common** trust requirements which are common to all TECOM demonstrators.
  - **Partial** trust requirements which are specified in more than one TECOM demonstrator.
  - **Unique** trust requirements where are specified in only one TECOM demonstrator.

Example (I)

For example, take three TECOM demonstrators, each in a different application domain (i.e. home control, video surveillance and automotive) and these systems have the following trust-security requirements:

- **Sys1:**
  - S1-R1: User authentication/identification
  - S1-R2: Boot time integrity check
  - S1-R3: Data replication in different locations
  - Others not described for the sake of brevity
- **Sys2**
  - S2-R1: Data integrity check
  - S2-R2: Backups of software and data
  - Others not described for the sake of brevity
- **Sys3**
  - S3-R3: Integrity check of stored snapshots
  - Others not described for the sake of brevity

Example (II)

The quality attributes of these requirements are organized in a table

ID	(Sub) Factors	Sys1	Sys2	Sys3	Property
1	Access control	√			Unique
1.1	Identification	√			Unique
1.2	Authentication	√			Unique
…	Others				
3	Integrity	√	√	√	Common
4	Availability	√	√		Partial
…	Others				

Example (III)

After the quality attributes are organized in the table, we know what (sub) factors have in common the different TECOM demonstrators. We could go one step further by trying to find similar requirements among these demonstrators. In this step we could use a requirements table for the traceability of specified requirements and the demonstrators implementing them.

ID	Requirement	Sys1	Sys2	Sys3	Property
1	User authentication/identification	**S1-R1**			Unique
2	Integrity checks	**S1-R2**	**S2-R1**	**S3-R1**	Common
3	Backups and data replication	**S1-R3**	**S2-R2**		Partial
4	Others…				

To Conclude

To conclude

- Frequently requirements specification is not seen as a complex issue, so the effort spent by the industry in specifying good requirements is insufficient for a successful project.
- We present an approach for the specification of non-functional requirements, mainly for those related to trust in embedded systems. The trust dimensions include reliability, integrity, availability, survivability, safety and security.
- The approach includes a well established conceptual model of the terms used to build the requirements specification, and a well defined process to elicit and specify trust requirements for embedded systems and to identify their commonalities.
- This approach was used in the ITEA2 TECOM project which has the goal of developing solutions for trust in embedded systems.
- Finding commonalities among the trust requirements specified for the different embedded systems was one of the main project issues.