Trusted Computing Engineering for Resource Constrained Embedded Systems Applications

Deliverable D2.1

Use Cases Application Viewpoint
(Internal Document)

Project:	TERESA
Project Number:	IST-248410
Deliverable:	D2.1
Title:	Use Cases Application Viewpoint (Internal document
Version:	v2.0
Confidentiality:	Public
Author:	David Gonzalez & Manuel Blanco (Ikerlan)
Date:	16 June 2011

Part of the Seventh Framework Programme Funded by the EC - DG INFSO

Table of Contents

Document History

Version	Status	Date
V0.1	draft	19/01/2010
V.1.0	Major version	06/14/2010
V1.1	Black Channel pattern added	25/03/2011
V1.2	More detailed version of Secure Software Download pattern description added	07/04/2011
V1.3	HMAC pattern added	08/04/2011
V1.4	New version of Voter pattern description added	26/05/2011
V2.0	Taking into account comments from the EU reviewers	16/06/2011

Approval		
	Name	Date
Prepared	Antonio Kung	10/06/2011
Reviewed	All Project Partners	16/06/2011
Authorised	16/06/2011	16/06/2011
Circulation		
Recipient	16/06/2011	
Project partners	16/06/2011	
European Commission	16/06/2011	

1 Executive Summary

This document is divided into two main parts. In the first part the process for use cases definition is described, and in the second part the main use cases are detailed by following this process.

In order to ease the definition of the use cases, the actors and roles that take part in the engineering process have been identified for each application domain (industrial, automotive, home control, and metering). After this analysis, the identification of the roles involved in each use case becomes an easier task, as well as the definition of the relations and interdependencies between them.

To identify the actors, roles and tasks that may be involved in the different application domains, the V-Modell XT part 4 [V-Modell XT, 2006] of the German government has been taken as a reference. This is a way to unify criteria

Apart from identifying the different roles that take part in the engineering process, this document aims at revealing the different use cases where they are involved in.

Comments from the EU. reviewers:

1. **Comment:** Do and document the necessary work to increase the number and representative of the identified patterns. (Safety & Security at the same time is also a concern).

   Response: With the advisory board, we decided not to add more patterns, because in that way we can spend more time improving the ones identified, and our common engineering process. For the industrial domain, we have added two extra patterns, the first one focused on dependability (see section 2.2.1.3 and 2.2.1.4 point 3), and the other one required to introduce security patterns in the Industrial domain's case study (see section 2.2.1.3 and 2.2.1.4 point 4).

   This case study will deal with dependability and security patterns at the same time.

2. **Comment:** Template needs to be extended further to enable the formal validation of patterns and their usage.

   Response: D2.1 was submitted in M6 to provide a high level overview of the use cases.

   WP4.T3 defines that the final patterns will be specified subsequently in D4.2, D4.3 and D4.4.

   We are continually making internal updates of D2.1 to enhance the description of the use cases. This version of D2.1 addresses several of the reviewers' comments.

   We added a reference in the section 2.1.6 to the deliverable D4.2, where an improved description of the patterns has been defined.

3. **Comment:** Classification between Security and Dependability Patterns seems to be important, in order to support validation tasks in cross-integration scenarios.

   Response: Apart from the table in section 3.4, where all patterns are listed with the corresponding type of focus, a new field in the use case template was defined to collect the type of focus of each pattern.

4. **Comment:** What needs to be made much clearer for the next review is which of these use cases will be used to develop a demonstrator for the TERESA project results.

   Response: With the advisory board, it was decided to implement the Industrial domain's case study in a first term. Right now, in the WP6 – D6.1 we have selected the use cases for the Industrial application. For the other domains, the selection of the use cases will be carried out in D6.2. At this stage we can not decide which use cases will be used in each specific application. For that reason, we decided not to cover this decision in this deliverable.

5. **Comment:** Explain how completeness of the described aspects in the scenarios was determined and checked.

   Response: We used an informal process based on our own domain experience to check the completeness of the described aspects in each scenario.

   Completeness is not crucial at this moment, because the repository and engineering process will provide mechanisms for extension.

6. **Comment:** What about implications of functionality split (MES vs. ERP) in the Industrial Control scenario.

   Response: The implication of the functionality split is out of the scope of this deliverable. This kind of functionality split is more related to manufacturing industries where the quantity of information exchange between different departments and people is higher. Right now, in the TERESA project, we are focused only on the realization part of the process, where the quantity of the information and the participants are less than in a manufacturing process.

   On the other hand, although in our engineering process we handled less information than in a manufacturing industry, we will consider the management of the information, and the relationship between the specification, design, verification and validation teams for the development of our trusted engineering process. This part of the process could be seen like a portion of an ERP system, but the definition of this information will be collected in the deliverable D3.2, where the meta-model of the engineering process is defined.

7. **Comment:** What about considering the metering system as a decomposable instead of a monolithic / tamper proof systems => Aspects of tampering with subcomponents / sensors or communication links in the metering scenario.

   Response: In the metrology use cases and patterns, the metering system has been considered as a monolithic system. This reflects the way a metering system is seen in the metrology requirement catalogues like the MID. Before a meter can be put on the market, it has to pass the type approval, which includes the compulsory calibration. One consequence of the compulsory calibration is that the case when the meter is closed and provided with a calibration seal. The case has to be constructed in a way that the opening of the case leads to permanent damage on

the calibration seal or the case itself. Once the case is opened, the calibration of the meter is void and the measurement values are no more considered valid. That's why the whole metering system is seen as one monolithic part.

Of course, special care must be taken, e.g. which subcomponents are used for the storage of private parameters like cryptographic keys or software images in a practical implementation. But in the scope of this document, the patterns should be described from an abstract, platform independent view. For that reason, the abstract "Secure Storage" has been used for the storage of sensitive data in the metrology use case and pattern descriptions, which leaves space for different implementations. Especially when these patterns are adapted for the use in other domains, it could become interesting to take a closer look at the subcomponent level – particularly when no sealing of the devices is intended in the application domain.

8. **Comment:** What about interaction with other systems / aspects of overlapping multi-purpose component usage in the home control scenario.

   Response: We assume that this question is about the Secure Service Discovery pattern.

   Obviously all systems interacting within the scenario must include the pattern (it is an assumption in the Teresa pattern description)

   It raises an interesting point on the home control domain ecosystems which could separate the application part from the platform part. Thus we could have patterns at the application level and patterns at the platform with different assurance level and schemes

9. **Comment:** Explain where you will discuss (feature) interactions between individual aspects and give at least some practical examples for the investigated domains where this interaction might occur, which implications it has and how you approach it.

   Response: Feature integration issue is not relevant because TERESA focuses on integration during the process and not off the shelf.

2 Patterns derivation

2.1 Process guidelines for pattern derivation

This section describes the set of actions that have been defined in order to analyze each application sector, and identify and define the most common development strategies (patterns) related with security and dependability. It aims at being a guide for the TERESA project participants.

2.1.1 Description of the domain application

At first, it is necessary to carry out a description of the actual context of each application domain. This description will provide a guide to the rest of the TERESA project participants to know the different functionalities related to security and dependability within each application domain.

2.1.2 Identify common security and dependability (S&D) requirements

One of the first tasks in the TERESA project is to identify the requirements related to security and dependability in each application sector. Furthermore, having identified the requirements, it is required to identify the mechanisms to use in order to achieve these requirements. These mechanisms could be defined as patterns. To collect and monitor this information, the following table has been defined.

ID	Title	Description	Mechanisms	Selected mechanism	Type of focus

Table 1. Requirement traceability

Following a brief description of each field of the previous table:

- ID. Identifier of the requirement
- Title. Name of the requirement
- Description. A brief description of the requirement.
- Mechanisms. A list of the mechanisms that address the requirement.
- Selected mechanism. The selected mechanism to be specified as a pattern.
- Type of focus. Define the focus of each selected mechanisms. The mechanism selected in the TERESA project can be focused through security or dependability.

2.1.3 Identify the mechanisms that address the S&D requirements

There may be several mechanisms that can be used to achieve the same desired functionality. These mechanisms must be listed in the previous table.

2.1.4 Select the mechanism to be specified as a pattern.

The selected mechanism will be collected in the last column of the previous table.

2.1.5 Instantiate the required use cases to describe the functionality of the mechanism within a specific application

In this section the functionality of the mechanism must be implemented in a concrete scenario. To represent it a UML use case diagram must be defined. The way the Use Case should be defined is presented in the diagram below.

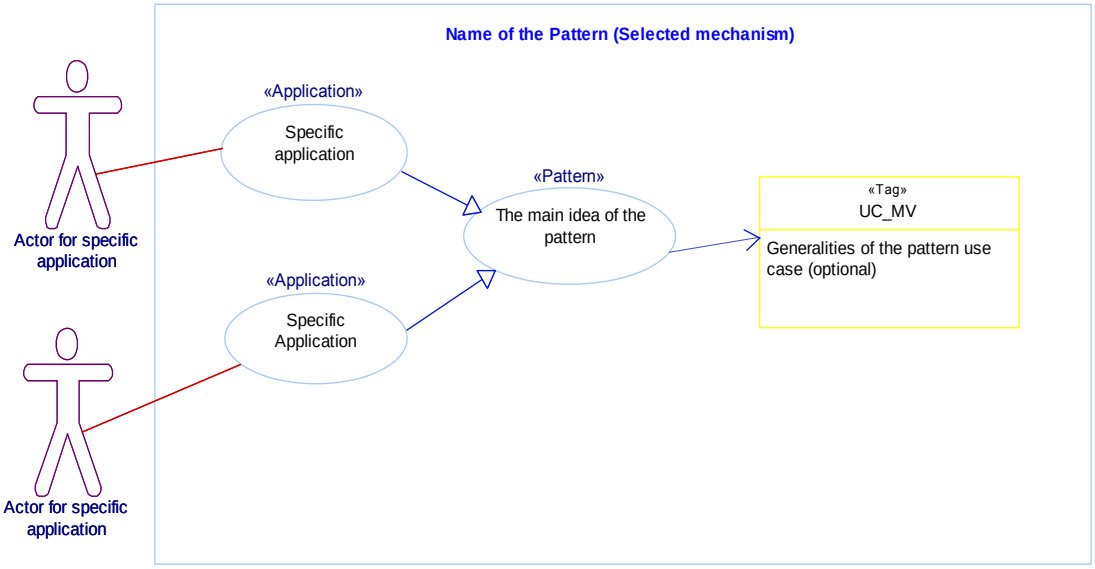

Figure 1. Use Case diagram

The Figure 1 shows how the selected mechanism (*Scenarios inspired from today practices*) will achieve the desired functionality and at the same time, it represents the possible specific applications (*Scenarios relevant to future trends in engineering*) where this mechanism could be used (defined as use cases with "application" stereotype). Besides, it identifies the actors that use the selected mechanism in each application.

2.1.6 Derive the pattern

The following figures are required to describe the functionality of the pattern.

- *Pattern template*. It summarizes the main ideas of the pattern (Name, problem, solution, and others).
- *Structure diagram (optional).* It provides a brief description of the structure of the pattern and the relationships among the different modules that comprise it.
- *Sequence diagram.* It provides a graphical representation of the functionality of the pattern. A rough description of the interaction among the modules (identified in the structure diagram) that are part of the pattern.
- *Use case template.* It is a summary of the use case of the pattern and at the same time a general description of the functionality of the pattern. To collect this information the following template must be filled in.

Use Case Label:	**Type of focus:**
Date:	**Author:**
Use Case Name:	
Actors:	•
Triggering Event (optional):	

Relations with other use cases (not always applicable)	**Extended by**: **Includes**: **Threatened by**: **Generalization**: **Mitigates or prevents**: **Detects**:
Description:	
Preconditions:	•
Post-conditions:	•
Normal scenario:	1.
Alternative scenario:	
Open issues:	

Table 2. Template for describing a Use Case

The "generalization" use cases are set within the use cases template shown above along with "includes", and "extended" use cases, to give a complete overview of each use case context.

In this template the necessary information to understand it is also collected, such as the use case label and name, actors, scenarios, description, pre/post – conditions, and others. In the points below, there is a little explanation of the sections that comprise the template:

- *Use Case Label*: Use Case identifier or reference number.
- *Type of focus*: To identify which is the focus of the pattern. It could be *"Dependability"* or *"Security"*.
- *Date*: The date when the use case was initially documented.
- *Author*: The partner that documented the use case.
- *Use Case Name*: *Concise name for the use case
- *Actors*: List of the actors involved in the use case
- *Triggering Event*: The event that initiated the use case.
- *Extended by*: List of other use cases that could extend this use case.
- *Includes*: List of other use cases that are included in this use case.
- *Threatened by*: List of the actions that threaten this use case.
- *Generalization*: List of the use cases that adapt and use the primary use case to address the desired functionality.
- *Mitigates or prevents*: The actions that this use case prevents.
- *Detects*: The misuse case detected by this use case.
- *Description*: Goal to be achieved by the use case and sources for requirements.
- *Preconditions*: The conditions to be met before the use case can start.
- *Post-conditions*: Describe the state of the system at the conclusion of the use case execution.
- *Normal scenario*: The interactions between actors and system necessaries to achieve goal.
- *Alternative scenario*: Description of other scenarios. Some variations in the interactions described above.
- *Open issues*: List of any remaining open issue that must be resolved.

This template will be extended in the deliverable D4.2 – section 3.3.1, in order to take into account domain specific aspects of the patterns. Besides, in this deliverable in the section 5 a formal representation of the pattern will be defined in order to establish a flexible structure that overcome the lack of formalism of the classical pattern form (textual) and to allow an easy use of the pattern (for reuse or for use in the repository).

2.2 Identification and definition of the patterns

2.2.1 Industrial Control System

2.2.1.1 Industrial Control system domain

In most of the application domains, the development of Safety Critical Embedded Systems (SCES) that must satisfy a certain set of constraints (e.g., dependability, availability, integrity, etc.) leads to a vast growth in the complexity of design, verification, validation and certification effort [Kop07, PC06].

Faced with the existence of this complexity challenge and with the aim of bridge the gap from specification to execution [PC06], a set of strategies such as abstraction, partitioning, segmentation, composability, and others [KS03a], and a collection of design techniques, measures and tools that support the development of SCES must be followed [PC06, THR+06].

In the Industrial domain and in the context of the TERESA project one of the principal concerns is the dependability of the system. In SCES that must provide a given service despite the occurrence of faults (dependable-embedded systems) requires the distribution of functions and the use of specific techniques to achieve fault containment, error containment and fault tolerance [Kop97].

An embedded system consists of a series of components HW and SW that interacts between them with the aim of provide a specific functionality or a set of functionalities. In the context of SCES some of these functionalities can be safety related.

From the point of view of TERESA project and inside the definition of the SCES some considerations must be taken into account in order to increasing the dependability of the system. These considerations must be taken directly on each of the components involved in the definition of the safety related functionality.

In first place, the main components of an embedded system that could be involved in the development of safety related functionality are collected in the following list:

- Processing unit
- I/O units interfaces
- Data paths
- Clock
- Memories (variable and invariable memories ranges)
- Electrical components
- Electronic components
- Power supply
- Sensor
- Actuators

In second place, once identified the main components that could be involved in the development of a safety related system, it has analyzed which considerations related to the components previously defined must be taken into account to increase the dependability of the embedded system:

- Monitoring and reporting about the status of the safety function.
- Identifying failures in the processing unit that leads to an incorrect result of the safety function.
- Ensuring that the information extracted from the invariable memories range is correct.
- Controlling the storage of the information in the invariable memories ranges.
- Detecting failures during addressing, writing, storing and reading information from a variable memory range.

- Prevent sending of inadmissible outputs to the system.
- Check that the information transferred does not contain any defect.
- Assure a correct program execution.

These considerations provide a guide in order to establish which functionalities of a specific application could be implemented in such a way that the system will be more dependable. Besides, these considerations give an idea of what mechanisms can be consider for the implementation of these functionalities.

2.2.1.2 Requirements and mechanisms identification

ID	Title	Description	Mechanisms	Select mechanisms	Type of focus
IN01	Availability of information provided to the Actuator	Information provided by the source must be available to the actuator.	Voter Comparator Reciprocal comparison	Majority Voter	Dependability
IN02	Reliability of the information provided to the Actuator	Information provided from the source to the actuator must be reliable.	Voter Comparator Reciprocal comparison	Majority Voter	Dependability
IN03	Reliability of the information shared in a distributed system	Information provided by the slaves to the Master in a distributed system must be reliable.	Communication Ring	Communication Ring	Dependability
IN04	Availability of the information shared in a distributed system	Information provided by the slaves in a distributed system must be available to the Master.	Communication Ring	Communication Ring	Dependability
IN05	Reliability of the information shared through a non-safety-related communication mechanisms	Information shared between safety-related elements through a non-safety –related communication mechanism must be reliable.	Black Channel	Black Channel	Dependability
IN06	Integrity of the information transmitted over an unreliable mechanism	The integrity of the information shared between safety related elements through an unreliable mechanism must be assured.	HMAC	HMAC	Security

Table 3. Industrial requirements and mechanisms identification

2.2.1.3 Use case definition

The following use cases diagrams will describe the functionality of the mechanisms previously selected.

1. **Voter**

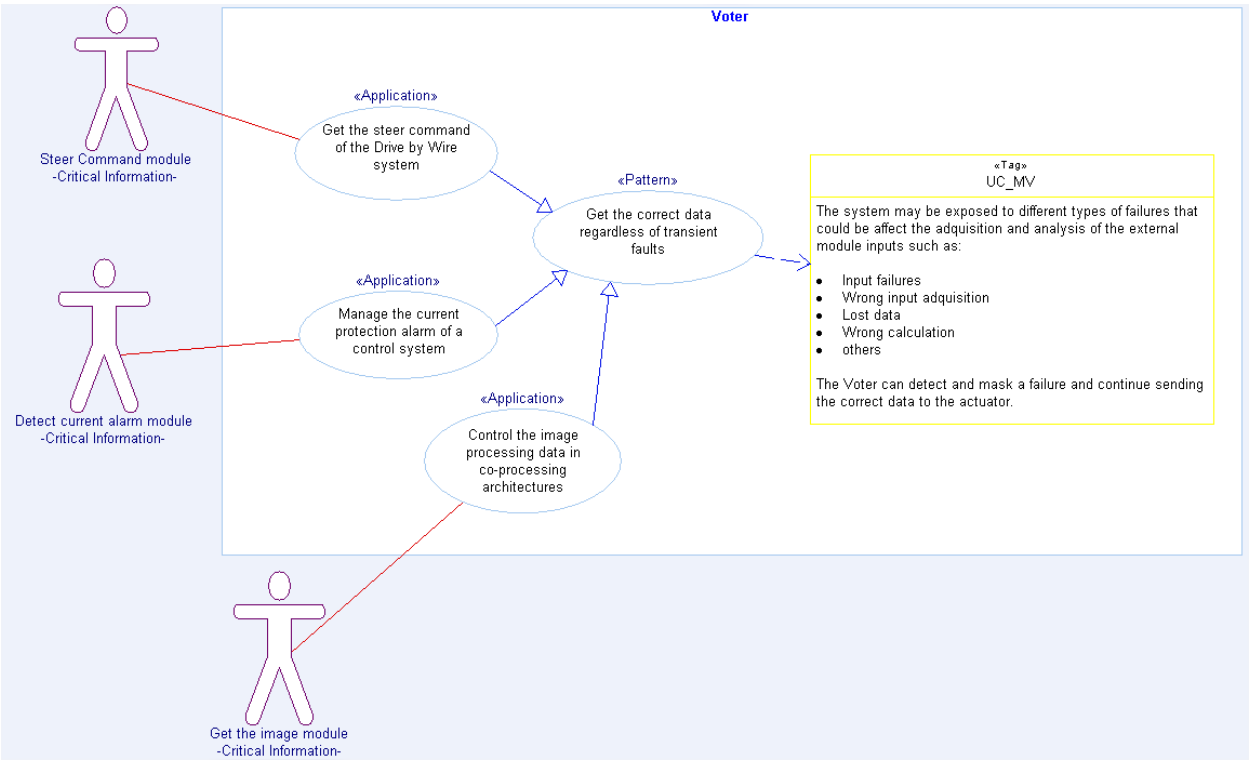

Figure 2.Voter Use Case

2. **Communication Ring**

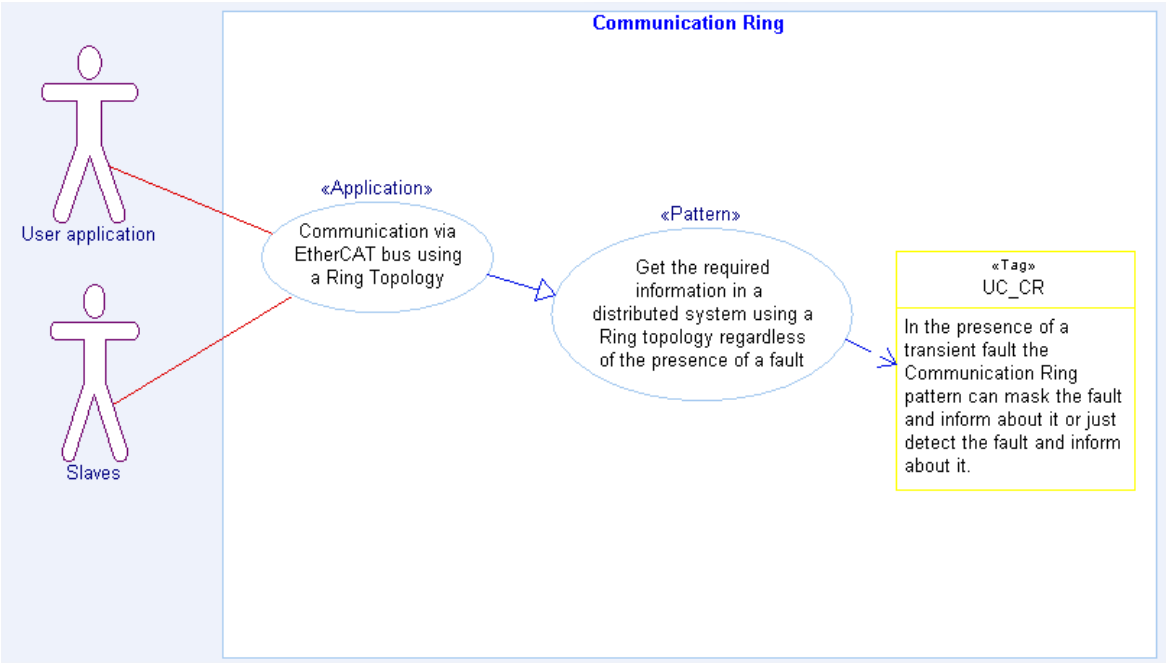

Figure 3. Communication Ring Use Case

3. **Black Channel**

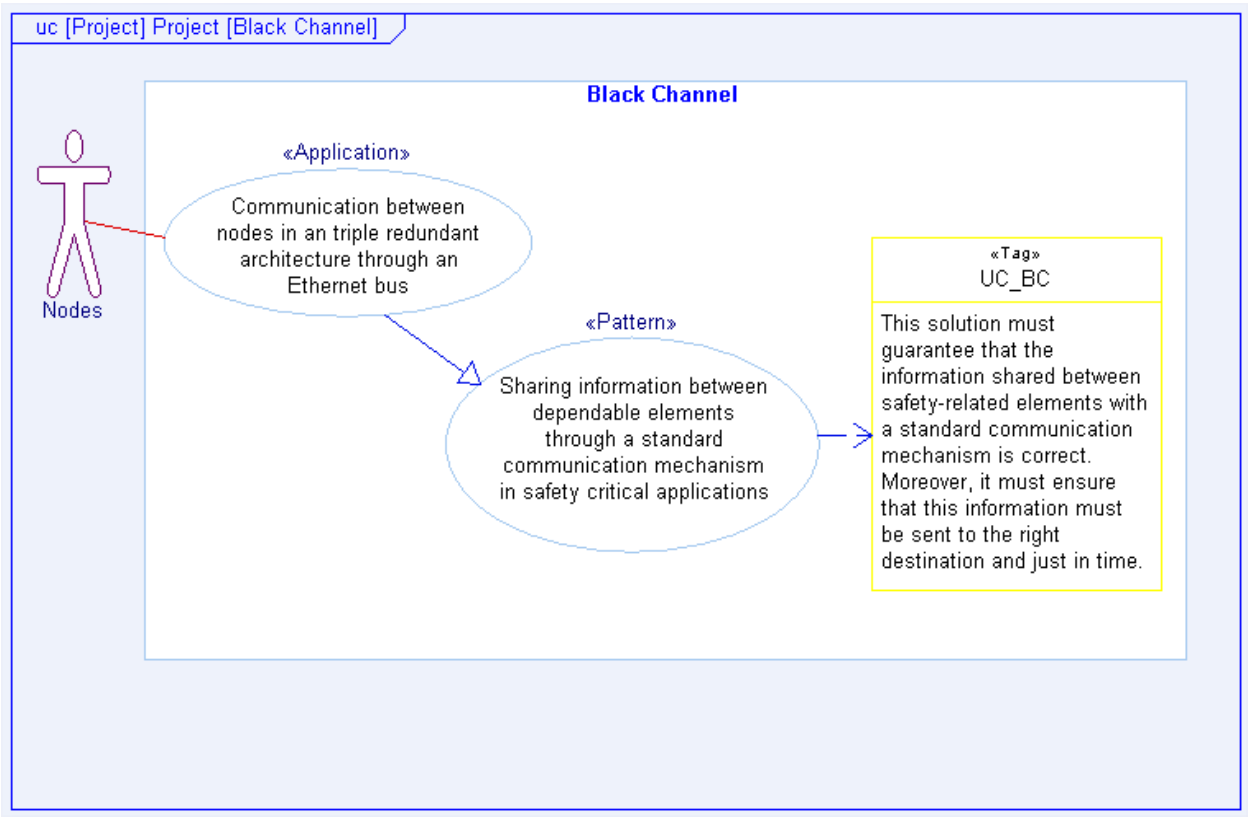

Figure 4. Black Channel Use Case

4. **HMAC**

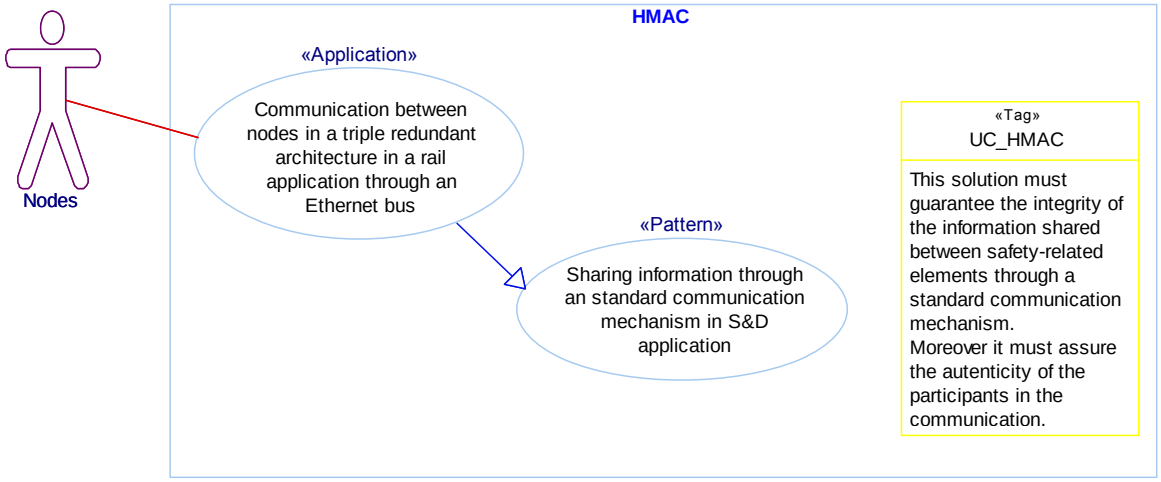

Figure 5. HMAC Use Case

2.2.1.4 Pattern definition

Based on the information derived above, the following pattern have been identified and defined for the different application sectors.

1. **Majority Voter Pattern**

Name:

- Voter

Also know as:

- Voter

Example:

- Triple redundancy architect

Context:

- A critical application that must perform a safety-related functionality regardless the presence of a transient fault.

Problem:

- A transient fault in a critical application which leads to a system failure is not detected.

Solution:

To properly apply this solution, the system must be replicated to generate the number of independent inputs required to carry out the comparison. This action also increases the reliability and availability of the function implemented by the system. Once the proper architecture has been defined, the *"Decision maker"* detect if any system is running incorrectly and provide the accorded data of systems which is running correctly to the next functional block.

Structure:

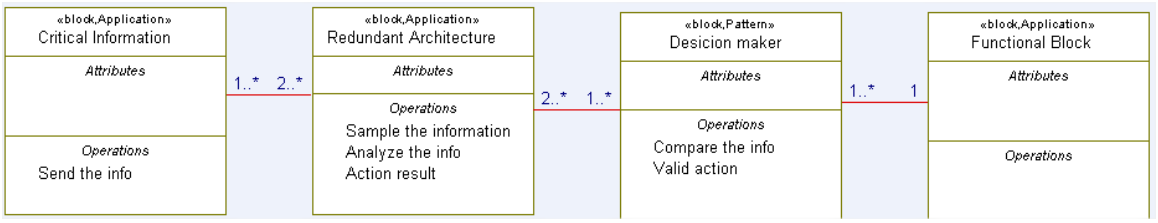

Figure 6. Voter Structure Diagram

Dynamics:

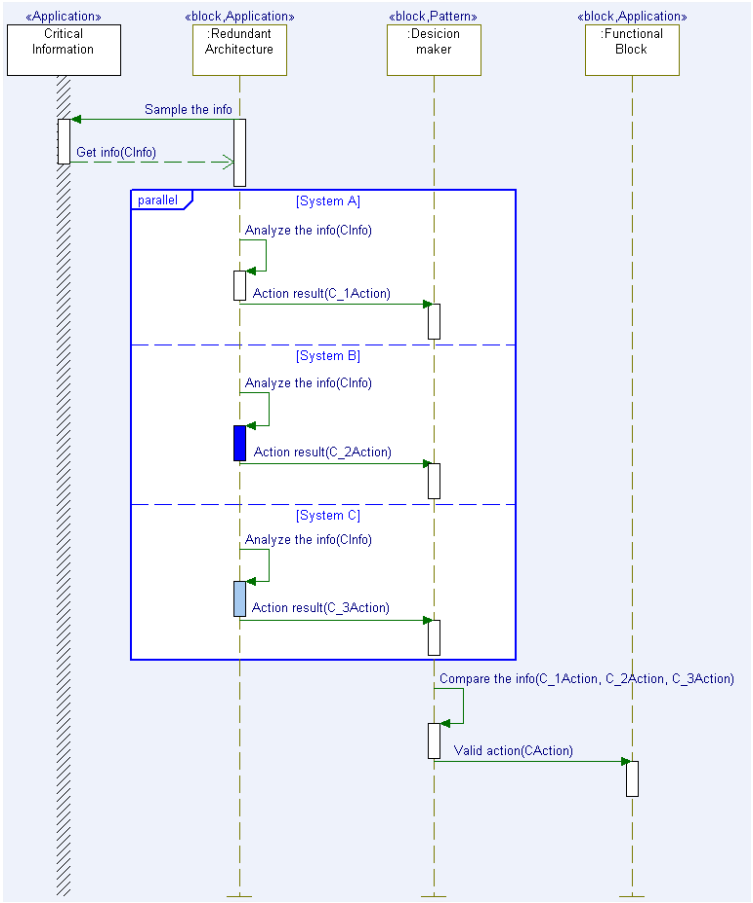

Figure 7. Voter Sequence Diagram

In the Sequence Diagram above is important to remark the concurrency of the operation within the voter. To express this concurrency the ***Interaction Operation*** artefact has been used

along with the ***Parallel*** stereotype and a different colour has been selected for each input data to emphasize this issue.

Besides, it is important to remark that the redundant architecture provides independent inputs to the decision maker. This independence is represented with the different names of the actions (C_NAction) provide to the *"Decision Maker"*. The *"Critical Information"* could also be independent. This may come from different sources. For simplicity of the previous diagram this case has not been represented.

Voter Use Case Template:

Use Case Label: *AL0.1*	***Type of focus:*** *Dependability*
Date: *10/02/2010*	***Author:*** *David González & Manuel Blanco*
Use Case Name:	*Voter*
Actors:	*Critical Information (Provide by the application)*
Triggering Event (optional):	*None*
Relations with other use cases (not always applicable)	***Extended by:*** *None* ***Includes:*** *None* ***Threatened by:*** *None* ***Generalization:*** • *Get the steer command of the Drive by Wire system* • *Manage the current protection alarm of a control system* • *Control de image processing data in co-processing architectures* ***Mitigates or prevents:*** *None* ***Detects:*** *None*
Description:	*Get the correct data regardless of transient faults*
Preconditions:	• *System architecture has been defined.* • *The safety function has been identified.*
Post-conditions:	• ***Without redundancy*** - *IN01 – p(IN01) = (1-p(S'))*(1-p(N'))*(1-p(R`))* - *IN02 – p(IN02) = (1-p(S'))\*(1-p(N'))\*(1-p(R`))* *; p(S') = Probability that a sensor fails* *p(N') = Probability that a network fails* *p(R') = Probability that a Receiver fails* • ***With redundancy*** - *IN01 – p(IN01) = $(3*x^2* – 2*x^3)*(1-p(D_M'))$, which is better than x in case x > 0.5* - *IN02 – p(IN02) = $(3*x^2* – 2*x^3)*(1-p(D_M'))$, which is better than x in case x > 0.5* *; x = (1-p(S'))*(1-p(N'))*(1-p(R`))* *D_M' = Probability that the Decision Maker fails*
Normal scenario:	• *Get the critical information from the application* • *Analyze the information in each system of the Redundant Architecture.* • *Send the data to the Decision maker* • *Compare the information* • *Send the valid data to the Actuator*
Alternative scenario:	
Open issues:	*Take into account the synchronism of the data of each channel that*

	comes to the voter. *Monitor the status of the "Decision maker".*

Table 4. Voter Template

Consequences:

- The system can provide the safety-related functionality regardless the presence of a transient fault.
- The system can react in the presence of a fault.
- Liabilities:
  - *Overhead and cost could increase owing to the "Redundant Architecture" of the system and the actions taken to monitor the status of the "Decision maker".*

2. Communication Ring

Name:

- Communication Ring

Also Know As:

- None

Example:

- EtherCAT bus in distributed systems using Ring topology

Context:

- A communication critical application that must guarantee that the information share between the different components "Master / Slaves" of the system is received by the "Master" of the system regardless the presence of a random failure.

Problem:

- The presence of random hardware failure in the system is not detected and this can lead to a loss of the information needed to perform the safety-related functionality.

Solution:

- Fault tolerance systems use information redundancy strategies to increase the system availability in the presence of a HW or SW fault. The Ring topology "Master /Slaves" supports sending information in both directions and at the same time decides if all the required information has been received.

Structure:

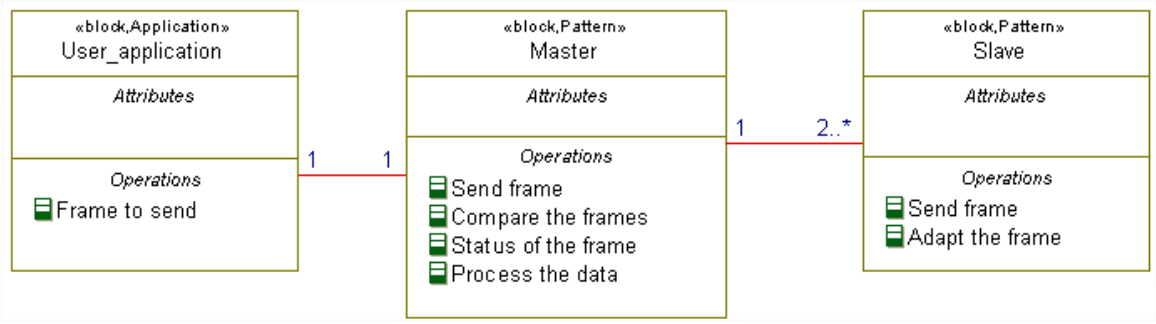

Figure 8. Communication Ring Structure Diagram

Dynamic:

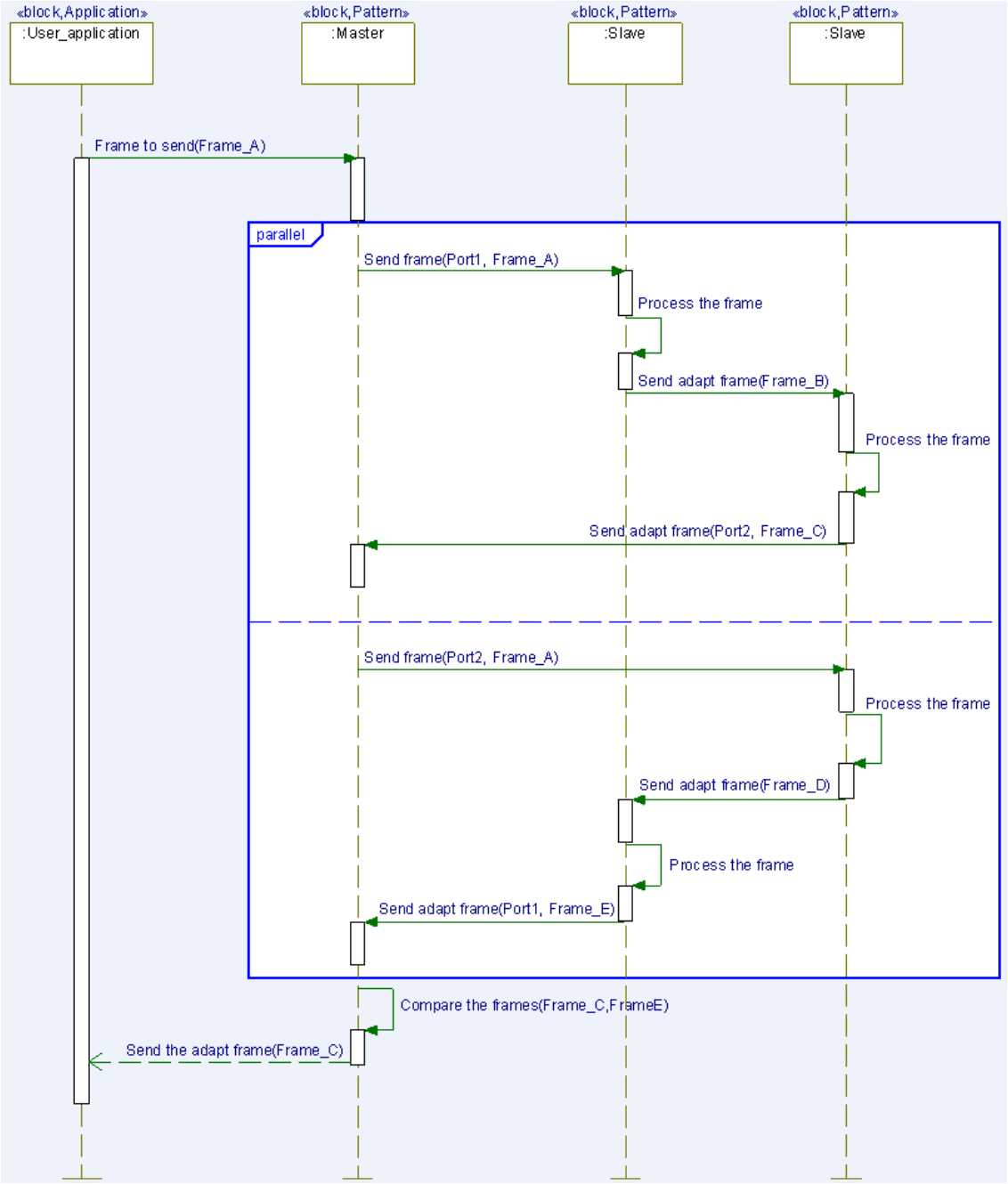

Figure 9. Communication Ring Sequence Diagram

Communication Ring Use Case Template:

Use Case Label: *AL0.2*	***Type of focus:*** *Dependability*
Date: *10/02/2010*	***Author:*** *David González & Manuel Blanco*
Use Case Name:	*Communication Ring*
Actors:	*User application* *Slaves*
Triggering Event (optional):	*A request of some information required for the User application*

Relations with other use cases (not always applicable)	***Extended by:*** *None* ***Includes:*** *None* ***Threatened by:*** *None* ***Generalization:*** • *Communication via EtherCAT bus using a Ring Topology* ***Mitigates or prevents:*** *None* ***Detects:*** *None*
Description:	*Get all the required information for user application regardless of system faults.*
Preconditions:	• *The Ring topology must be defined (Master and Slaves identified).* • *The selected bus for the application must support a bi-directional communication, and a way to send the frame in opposite directions of the line at the same time (e.g. two independent lines, Ethernet line, etc).*
Post-conditions:	• ***Without redundant information*** - $IN03 – p(IN03) = p(S_{i/M}(i)) = (1-p(L_e))^i*(1-p(n_e))^i$ - $IN04 – p(IN04) = p(S_{i/M}(i)) = (1-p(L_e))^i*(1-p(n_e))^i$ $; Si$ = Any slave in the ring topology $p(L_e)$ = Probability that a link fails $p(n_e)$ = Probability that a node fails • ***With redundant information*** - $IN03 – p(IN03) = p(S_{i/M}(i)) = [(1-p(L_e))*(1-p(n_e))]^i + (1-[(1-p(L_e))*(1-p(n_e))]^i)* [(1-p(L_e))*(1-p(n_e))]^{n-i}$ - $IN04 – p(IN04) = p(S_{i/M}(i)) = [(1-p(L_e))*(1-p(n_e))]^i + (1-[(1-p(L_e))*(1-p(n_e))]^i)* [(1-p(L_e))*(1-p(n_e))]^{n-i}$ $; n$ = Total numbers of slaves
Normal scenario:	• *Receive the frame from the User application.* • *Master sends the frame through two different ports in opposites directions to the Slaves.* • *The Master receives the frames from the Slaves in both directions.* • *The Master compare the frames* • *The Master process the frame* • *The Master sends the frame to the User application*
Alternative scenario:	• *Receive the frame from the User application.* • *The Master sends the frame through two different ports in opposites directions to the Slaves.* • *The Master receives the frames from the Slaves in both directions.* • *The Master compare the frames* • *The Master detect a failure* • *The Master process both frames in order to get all the information of the slaves.* • *The Master sends the frame to the User application*
Open issues:	

Table 5. Communication Ring Template

Consequences:

- The "*User application"* request can be carried out in a distributed system regardless the presence of a transient fault.

- The information resulting from the actions performed in each component "*Master/Slaves*" of the distributed system can be collected and analyzed in the "*Master*" regardless the presence of a transient fault.
- Liabilities:
  - *The application of this pattern is restricted to a type of bus that supports a bi-directional communication. Besides, the bus must support a way to send the information in opposite direction of the communication line at the same time.*
  - *The overhead of the application is increased because the information has to be sent twice in both direction of the Ring topology.*

3. Black Channel

Name:

- Black Channel

Also Know As:

- None

Example:

- Triple redundant architecture (three individual nodes) communicating through an Ethernet bus

Context:

- The communications could be at different levels: On chip, inter boards, systems, others.

Problem:

- The use of standard/normal communication mechanism in order to communicate two or several safety-related elements in a safety critical application.
- Several errors may occur when a message is transferred in complex network topologies. Different sources could be the origin of these errors as for example hardware failures, electromagnetic interference or others.

Solution:

- The black channel provides a safety interface/layer that allows different elements with a normal/standard communication mechanism to communicate in safety manner. This solution must guarantee that the data shared is correct and that they are sent to the right destination and just in time.

Structure:

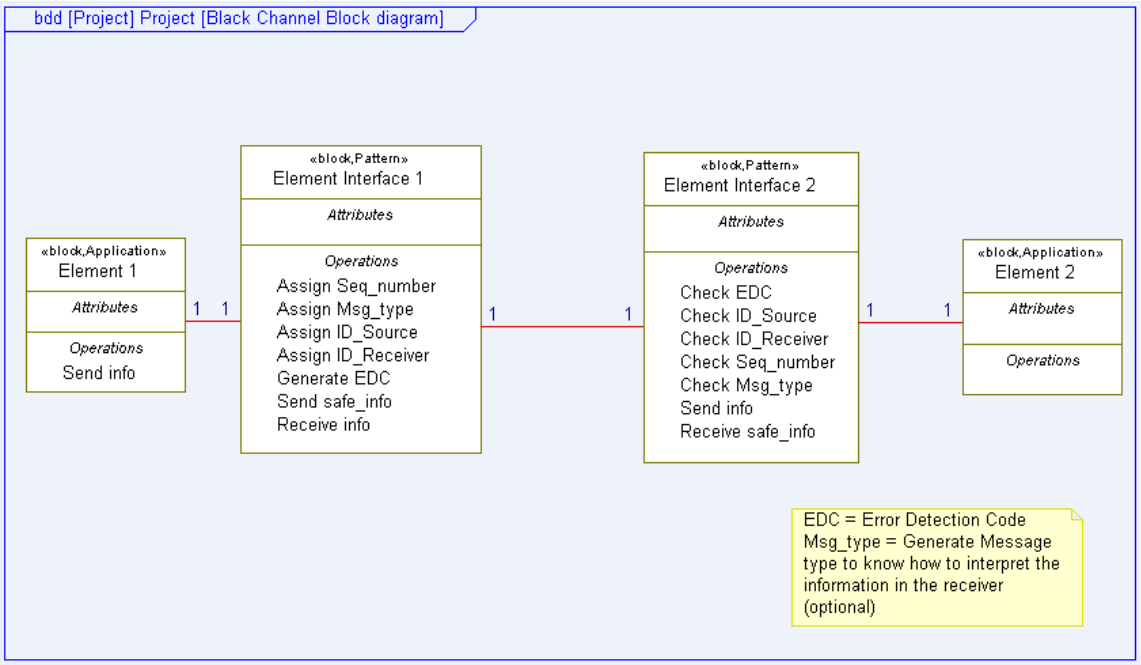

Figure 10. Black Channel Structure Diagram

Dynamic:

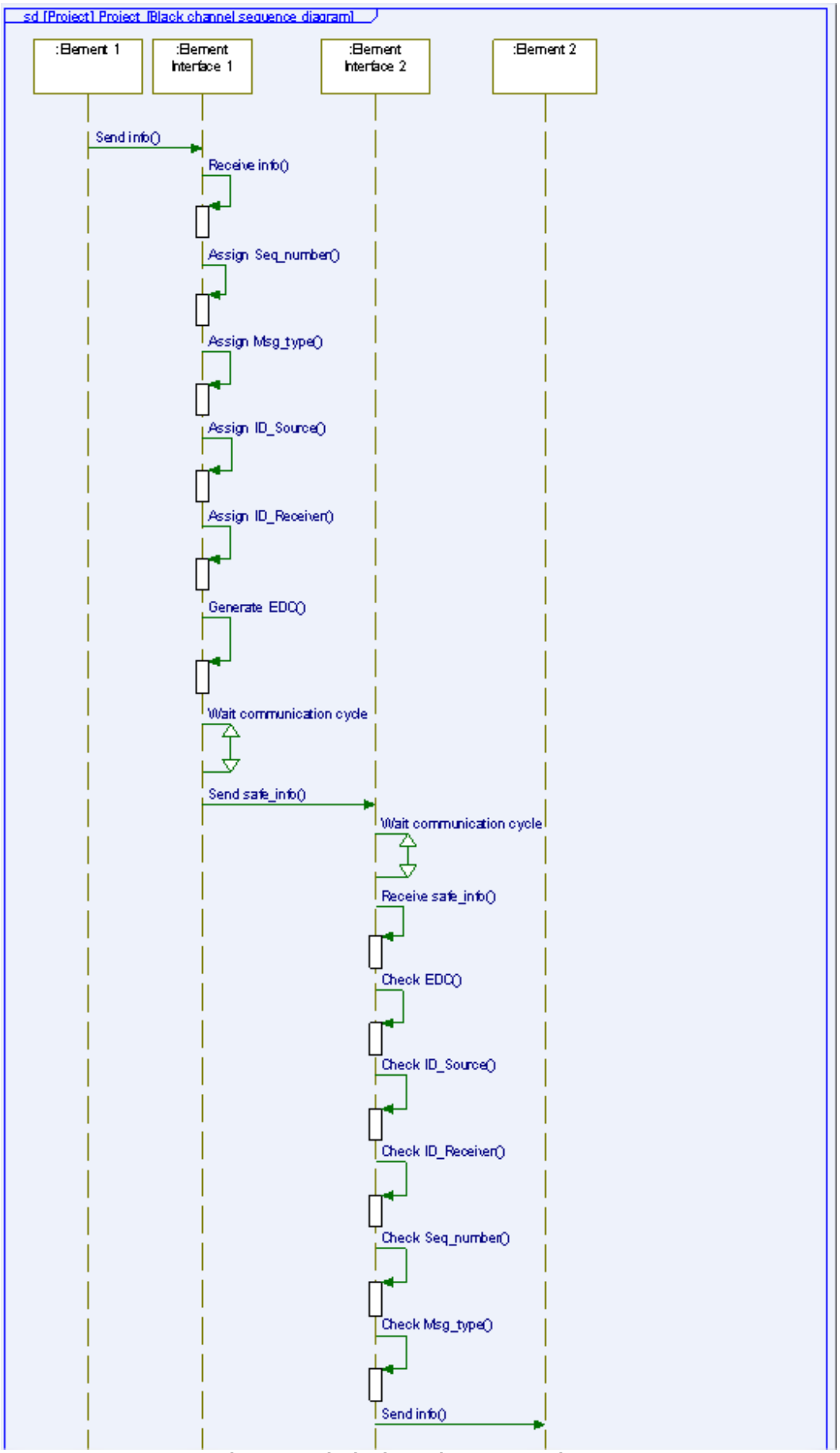

Figure 11. Black Channel Sequence Diagram

In the sequence diagram above is important to remark that the actions of sending and receiving the safety message must be cyclical and periodical because to the fact that elements involved in the communication must always know when the message should be sent and when it should be be received.

Black Channel Use Case Template:

Use Case Label: *AL0.3*	***Type of focus:*** *Dependability*
Date: *10/02/2011*	***Author:*** *David González & Manuel Blanco*
Use Case Name:	*Black Channel*
Actors:	*Nodes*
Triggering Event (optional):	*A data must be transmitted from a node to another*
Relations with other use cases (not always applicable)	***Extended by:*** *None* ***Includes:*** *None* ***Threatened by:*** *None* ***Generalization:*** • *Communication between independent nodes in an triple redundant architecture through an Ethernet bus* ***Mitigates or prevents:*** *None* ***Detects:*** *None*
Description:	*Share safety information between nodes through a standard communication mechanism ensuring the correct transmission of data, to the correct destination and at the right time.*
Preconditions:	• *The elements that share the information must be safety-relevant* • *The platform must support the implementation of an error detection code.*
Post-conditions:	• ***Without black channel*** - *IN05 – p(IN05) = (1-p(ch))* *; p(ch) = Probability that the channel fails* • ***With black channel*** - *IN05 – p(IN05) = (1-p(I_1))*(1-p(I_2))*(1-p(R(t_{ch}, t_c)))* *; p(I_n) = Probability that an interface fails* *p(R(t_{ch}, t_c))) = Probability of a residual error happens. This probability depends on the type of the channel, and the EDC used to diagnostic the failure.* *t_{ch} = type of channel* *t_c = type of EDC used*
Normal scenario:	• *Receive the data from the node.* • *The Node interface to the standard communication mechanism adapt the data* - *Assign the sequence number* - *Assign message type* - *Assign ID Source* - *Assign ID Receiver* • *Generate the EDC** • *Wait for the communication time* • *Send the safety information* • *Wait for the communication time* • *Receive the safety data* • *The Node interface to the standard communication mechanism*

	check the information - *Check the EDC** - *Check the sequence number* - *Check ID Source* - *Check ID Receiver* - *Check the message type* • *Send the data to the node* *EDC = Error Detection Code
Alternative scenario:	• *Receive the data from the node.* • *The Node interface to the standard communication mechanism adapt the data* - *Assign the sequence number* - *Assign message type* - *Assign ID Source* - *Assign ID Receiver* • *Generate the EDC** • *Wait for the communication time* • *Send the safety information* • *Wait for the communication time* • *Receive the safety data* • *The Node interface to the standard communication mechanism check the information* - *Check the EDC** • *The EDC of the data received is different from the one calculated by the node interface* • *Cancel the transmission* *EDC = Error Detection Code
Open issues:	*Use an EDC code that take into account the dependability and the security of the information transmitted.*

Table 6. Communication Ring Template

Consequences:

- The information shared in a safety critical application at different levels can be transmitted in a safely manner without having to implement a safety communication mechanism.
- In order to communicate two elements in a safety critical application, it is only required to implement a safety interface in each of the elements that support the detection of errors in the data. Moreover, it is required to guarantee that the data are sent to the right destination and just in time.
- Liabilities:
  - *The implementation of this pattern increases the computational cost of the application. This increase is caused by the calculation of the EDC.*
  - *The amount of information through the communication mechanism is increased because the transmitted data size is larger. This increase is caused by the extra information needed for identification and error detection of the data.*

4. HMAC

Name:

- HMAC

Also Know As:

- None

Example:

- Triple redundant architecture (three individual nodes) communicating through an Ethernet bus

Context:

- A critical application that must guarantee the integrity of the information shared between different components “Nodes, channels, etc” through an unreliable medium.

Problem:

- The use of standard/normal communication mechanism in order to communicate two or several elements in security-related applications.
- Several potential attackers can manipulate the content of the information when a message is transferred through an unreliable medium. Different sources could be the origin of these attacks e.g. malware, man-in-the-middle, others.

Solution:

- The HMAC provides a secure layer that allows different elements with a normal/standard communication mechanism to communicate in secure manner. This solution must guarantee the authenticity of both the source of the message and its integrity without use of any additional mechanism.

Structure:

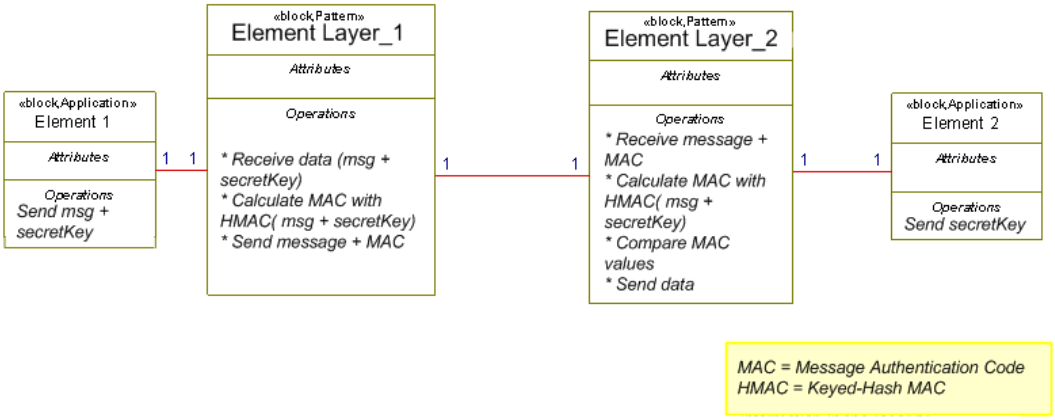

Figure 12. HMAC Structure Diagram

Dynamic:

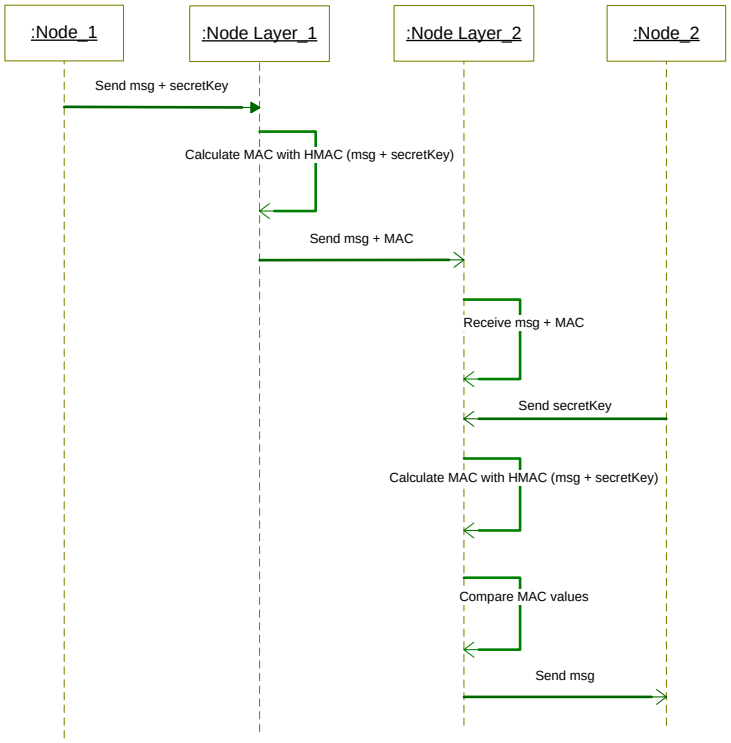

Figure 13. HMAC Sequence Diagram

Black Channel Use Case Template:

Use Case Label: *AL0.4*	***Type of focus:*** *Security*
Date: *06/04/2011*	***Author:*** *David González & Manuel Blanco*
Use Case Name:	*HMAC*
Actors:	*Nodes*
Triggering Event (optional):	*Information must be shared between nodes.*
Relations with other use cases (not always applicable)	***Extended by:*** *None* ***Includes:*** *None* ***Threatened by:*** *None* ***Generalization:*** • *Communication between independent nodes in an triple redundant architecture through an Ethernet bus* ***Mitigates or prevents:*** *None* ***Detects:*** *None*
Description:	*Share critical information between nodes through a standard communication mechanism ensuring the authenticity of both the source of the message and its integrity.*
Preconditions:	• *The elements that share the information are part of the community of users, and each one knows the secretKey.* • *The platform must support the implementation of a hash function.*
Post-conditions:	• *Data shared between nodes cannot be modified undetectably.* • *It is assure that the data shared between nodes is genuine and the authenticity of the parties involved in the communication has*

	been guaranteed.
Normal scenario:	• *Receive the data + secretKey from the Node_1.* • *The Node layer_1 to the standard communication mechanism adapt the data.* - *Calculate the MAC value with HMAC function (msg + secretKey).* • *Send the message + MAC.* • *Receive message.* • *Receive the secretKey from Node_2.* • *The Node layer_2 to the standard communication mechanism check the information.* - *Computes MAC with HMAC function (msg + secretKey).* • *Compare calculated MAC with MAC transmitted with the message.* • *Send the message to the Node_2*
Alternative scenario:	
Open issues:	

Table 7. HMAC Template

Consequences:

- The information shared in an application at different levels can be transmitted in a secure manner without implement a secure mechanism.
- Liabilities:
  - *The implementation of this pattern increases the computational cost of the application. This increase is caused by the calculation of the MAC with the hash function.*
  - *The amount of information through the communication mechanism is increased because the transmitted data size is larger. This increase is caused by the extra information needed by the message authentication code (MAC).*

2.2.2 Automotive System

2.2.2.1 Automotive system domain

While safety measures of today's automotive computing systems are already well-established for achieving dependability we also need to consider security. According to [Wol09], the majority of current vehicular systems are not or only hardly protected against attacks. This is not at least due to the fact that security is usually considered after achieving the core functionality but also due to the obvious fact that establishing the necessary organizational and technical measures as well as the infrastructure for securing IT-systems is highly cost-intensive. In addition the IT-security mechanisms in the vehicle cannot directly be advertised to the end user. Actually, most vehicular electronics are evolved in a bottom-up manner. Hence the vehicle is developed from simple and isolated microcontrollers to highly complex, interconnected, and software-driven distributed systems. Since implementing IT security afterwards is normally doomed to failure, it is necessary to have efficient ways of establishing security already in the very beginning of the design of IT systems as proposed by the Serenity project [MSD08].

The vehicular IT environment has some helpful properties that could ease the integration of security requirements in the design process. As described in [Wol09] amongst others these advantages are:

- *Periodic inspection* (reveal manipulations)
- *Moving target (*limited time for external attackers*)*
- *Physical protection (*complicates attacks up to a certain extent*)*
- *Centralized controller topology (*eases the implementation and maintenance of powerful, flexible, and sophisticated security measures*).*

On the other hand this environment suffers from some constraints that restrict the realization of security and safety requirements:

- *Physically challenging environment* (e.g., high variations in temperature, moisture or particular mechanical loads may not affect the maintenance effort)
- *Comprehensive liability* (legally binding warranties for the operating safety must be ensured for highly safety-critical applications, e.g., driving assistance, crash prevention)
- *Long product life cycles* (up to 20 years)

- **A Selection of Possible Attacks**

  In the automotive domain there are many different attack points. Attackers usually have full physical access. In fact almost every built-in component can be manipulated or replaced and its actual physical environment and (physical) inputs can be manipulated. The following list of possible attacks illustrates the strong need for vehicle security mechanisms.

  - ***Steal*** the vehicle or a valuable component (e.g., airbag)
  - ***Circumvent restrictions*** in hardware or software functionality (e.g., speed locks, feature activation, software updates)
  - ***Manipulate*** financially, legally, or warranty relevant vehicular components (e.g., electronic tolling devices, digital tachograph, chip tuning)
  - ***Spy on*** manufacturer's expertise and intellectual property (e.g., counterfeits, industrial espionage)
  - ***Violate*** privacy issues (e.g., contacts, last trips)
  - ***Impersonate*** (e.g., electronic license plate)
  - ***Misuse*** external communication facilities (e.g., disturb, misuse, harm)
  - ***Harm*** passengers, destroy OEM’s reputation (e.g., safety attacks)

  Due to these vehicle electronics intrusions the definition of a trust engineering approach that helps the engineer to achieve security and dependability is of high interest. In the future, problems will be made worse when open platforms are integrated in automotive systems.

- **Security requirements**

  Exemplary security requirements in the automotive area are amongst others component identification and authentication, strong isolation, secure communication, and secure initialization [Wol09]0. Component identification and authentication is used to ensure that original, legitimate components cannot be exchanged by unauthorized installations. Strong isolation of subsystems, components and individual applications prevents unauthorized access (i.e., data, functionality) or even affect (e.g., performance) among each other without proper authorization. Secure Communication is important to ensure confidentiality, integrity, and non-repudiation of the communicated information and the authenticity of the communication endpoints (secure channel). Via a trusted channel secure communication also allows to verify the configuration of the communication endpoints in order to determine its trustworthiness. Secure initialization ensures the integrity (authenticity, non-repudiation, and freshness) of a vehicular (sub-) system.

- **Use cases related to security**

  Current and future automotive applications with IT security requirements can be classified in the following categories:

  - Theft protection
  - Counterfeit and intellectual property protection
  - Software updates
  - After-sales applications

- Vehicular communication
  - *In-Vehicle communication*
  - *Vehicle-to-Device communication (V2D)*
  - *Vehicle-to-Infrastructure communication (V2I)*
  - *Vehicle-to-Vehicle communication (V2V)*
- Secure data storage
- Diagnosis

- **Secure Communication – Use Cases**

Until now, vehicular communication meant mainly in-vehicular communication between different electronic control units (ECU) and their respective sensors, which are distributed all over the vehicle. Here an interesting application is the component recognition. This is not only a safety critical application with regard to the availability of the components but also with regard to their authenticity. It should be ensured that only valid components are accepted.

The Communication to the outside world was normally restricted to some vehicle-to-device (V2D) interfaces. This usually means interfaces for fault diagnosis, ECU software updates but also the integration of some of the driver's mobile devices such as cellular phones to enable hand free calling. Beside the ECU software updates, in terms of secure communication also the feature activation, which can for instance be personalized, is a very interesting use case as well as applications like the remote door unlock and the electronic immobilizer.

However vehicle manufactures has already started to integrate first external communication channels that enable their vehicles to communicate wirelessly with their surrounding infrastructure (V2I) using available base stations. Based on the integration of GSM or UTMS receivers and the vehicle's current position, V2I communication enables various so-called location-based services (LBS), such as emergency call, location-based billing, stolen vehicle location or wrong way warning.

Furthermore applications based on electronic traffic signs such as Adaptive Cruise Control (ACC), intelligent shifting and emergency breaking are of great interest.

Future applications assume that vehicle will even communicate directly to each other using dedicated short-range communications to exchange for instance information about current emergency events, road conditions or traffic information.

For applications like lane change or lane merge assistance, a collision avoidance system or even for the mutual regulation of the right way it is of high importance that the messages are transferred securely. Another possible future application would be the realization of a priority signal for enabling emergency vehicles or police cars to send a priority message to automatically set traffic lights and adapt the right of way regulations in order to reach their emergency destinations safely and quickly. In approaches for multi-hop data routing vehicles are used as nodes in a dynamic ad-hoc or mesh network to enable various multi hop routing protocols. This allows vehicles to exchange information over their actual transmission range, but without relying on available infrastructures.

A further use case is the safety reaction as consequence of an emergency message. For instance such a safety reaction could be active breaks or cooperative ACC. The following list summarizes all interesting automotive use cases mentioned requiring secure communication.

For TERESA we have identified the following interesting use cases with regard to secure communication. Anyway, there are a lot of further use cases that involve secure communication but are inadequate to present it in a vivid manner.

- In-Vehicle communication (between sensors and ECUs)
  - *Component recognition*

- Vehicle-to-Device (V2D) communication
  - *ECU (Electronic Control Unit) software updates*
  - *Feature activation, e.g. personalized feature activation*
  - *Remote door unlock / electronic immobilizer*
- Vehicle-to-Infrastructure (V2I) communication - using available base stations (today, this means using a GSM or UMTS Receiver)
  - *Electronic traffic signs, e.g. Adaptive Cruise Control (ACC), intelligent shifting, emergency breaking*
  - *LBS (location based services), e.g. emergency call, location based billing (eTolling)*
  - *Stolen vehicle location*
  - *Wrong way warning*
- Vehicle-to-Vehicle (V2V) communication - using dedicated short range communications
  - *Local danger warning in emergency events from/ to other cars*
  - *Traffic information from/ to other entities*
  - *Lane change/merge assistance*
  - *Right of way regulation and cooperative driving*
  - *Collision avoidance systems*
  - *Priority signal*
  - *Multi-hop data routing*
  - *Safety reaction as consequence of an emergency message, e.g. active brake, cooperative ACC*

In order to obtain measures that help to prevent vehicle electronics intrusion for automotive applications, the implementation of security and dependability patterns is particularly interesting for secure communication and remote attestation.

Secure communication, occurring between electronic control unit and other parties, is important for software and policy administration, and vehicle to infrastructure communication. Remote attestation is important to verify the consistency of a vehicle configuration. The categories above comprise several use cases that are related to secure communication and remote attestation, which are described in the following.

- **Remote Attestation – Use Cases**

In the automotive domain one of the use cases for which remote attestation is important is the feature activation that was already mentioned before. For example, it is possible to deactivate the comfort features of a vehicle as reaction, if the electronic unit for component recognition notices that one of the integrated components is not original. Likewise, if the installed breaks are not from the original manufacturer, the maximum speed can be restricted.

Another application is an after-sales application, the Digital Rights Management. However, an application which is especially useful for the TÜV or the insurance company is the detection of manipulations such as chip tuning and mileage. In addition tolling and payment systems such as road charges are interesting in conjunction with remote attestation.

Beside the interesting use cases identified for secure communication there are also a lot of exciting use cases for remote attestation mainly with regard to payment scenarios:

- Feature (de)activation based on vehicle configuration (via backend)
  - *Component recognition*
  - *Deactivation of comfort functionalities*
  - *Restriction of the maximum speed*
- Digital Rights Management (DRM) - After-Sale Applications; DRM technologies attempt to control use of digital media by preventing access

- Detection of manipulations, e.g., chip tuning, mileage (esp. useful for TÜV (Association for technical inspection) / insurance company)
- Tolling (On-board unit with information of the vehicle) and payment systems, e.g. road charges depending on the payload (extra charge for dangerous goods, car type, weight) – where third parties are involved

- **Requirements for the Development of a New Functionality**

  For the implementation of a new function different aspects have to be taken into account. In the majority of cases both new hardware and software is needed. Before a sustainable solution can be established new and existing elements needed to be grouped to modules. Here we are confronted with the mapping problem to assign functions to different nodes of the vehicle's distributed system and their implementation in software and/or hardware. Besides a distinction between event-triggered, time-triggered functions/activities and combination of them is necessary.

  Since the vehicular software systems grow and hidden interrelationships are introduced the complexity increases. Managing the introduced product complexity is becoming a crucial problem and a challenge for the automotive industry today. Multiple networks with different protocols responding to different requirements are used.

  A problem here is the issue of mixing components with different criticality on shared resources in a system, like software components sharing a processor. This requires that non-interference between the objects of different criticality can be assured. But the large number of components have not only operational relations such as communication, synchronization, sharing resources, allocation and execution, also analytical relations established by the functions they attempt to implement such as performance, errors and reliability exist. Therefore each design decision can have some implications on other design decisions.

  Implementations of automotive functions are highly resource constrained. Small cost increase for one unit multiplies to a huge total cost. For this reason a trade-off between the system behavior (quality of service) and the resources required (processing, memory and power) is essential. Thus a fundamental issue in the development of automotive functionalities is the handling of the conflicting requirements: functionality, performance, and cost.

  Based on the intended functional content of the partial architecture and the geometrical distribution of components the allocation of functions to components has to be designed with the aim of

  - Minimization of costs
  - Minimization of cabling
  - Minimization of the number of connectors and
  - Minimization of hardware components

  to obtain an easier production and increased reliability of the system.

Today, in the automotive domain secure communication and remote attestation are important functionalities, each having a central meaning for a number of use cases. Therefore, we focus on the definition of appropriate corresponding patterns.

2.2.2.2 Requirements and mechanisms identification

Secure communication can have very different characteristics. It involves several different requirements like authentication, confidentiality, integrity, freshness, non-repudiation, accountability etc., which makes it hard to be described for the whole domain. For this reason, we consider toll collection as certain use case for describing the requirements of the secure communication mechanism.

As described detailed in [KFL09], most currently used car tolling systems are based on an extra On-Board Unit (OBU) that is integrated in the car. Equipped with such an OBU, a car is

able to communicate with the Road Side Units (RSU) of the toll provider. For an automatic accounting toll system the communication between these two instances uses the positioning technologies: GSM (Global System for Mobile Communication) and GPS (Global Positioning System). The RSU is continually broadcasting a control signal including necessary data for the accounting that allows the OBU to calculate the toll fee. The car's Communication Unit (CU), which includes the OBU, sends the data needed for accounting the driver: the type of the car, toll contract identification, pay method, and the signed bill of the last paid toll.

Table 6 lists the security and dependability requirements associated with this use case.

ID	Title	Description	Mechanisms	Select mechanisms	Type of focus
SC01	Authenticity	A mutual authentication should be performed between the RSU and the OBU of the car. This ensures that the toll amount charged by the RSU is authentic verifying the origin of the RSU's message, and on the other side that the driver is correctly accounted.	Secure Communication	Secure Communication	Security
SC02	Data Integrity	It is necessary to make sure that the account of data sent from the vehicle and the RSU is not manipulated.	Secure Communication	Secure Communication	Security
SC03	Freshness	It is important that messages are transferred directly to the RCU without being recorded.	Secure Communication	Secure Communication	Security
SC04	Confidentiality	It should be ensured that the communication cannot be eavesdropped, because transferred information like the payload, pay method, the charged toll, toll contract identification, and the signed bill of the last paid toll are confidential (privacy concerns).	Secure Communication	Secure Communication	Security
SC05	Non-repudiation	Non-repudiation has to be ensured since the driver should not be able to contest a correctly generated bill.	Secure Communication	Secure Communication	Security
RA01	Verification of platform integrity	Integrity of the car's system state has to be verifiable by a remote party. This ensures that the sensors and ECUs integrated in the car are not manipulated, e.g. to prevent attacks that aim to reduce the toll payment.	Remote Attestation	Remote Attestation	Security
RA02	Secure platform initialization	In order to detect manipulations at start-up the platform integrity is measured at system initialization and stored within a secure hardware component.	Secure Boot	Secure Boot	Security

Table 8. Automotive requirements and mechanisms identification

2.2.2.3 Use case definition

1. **Secure communication**

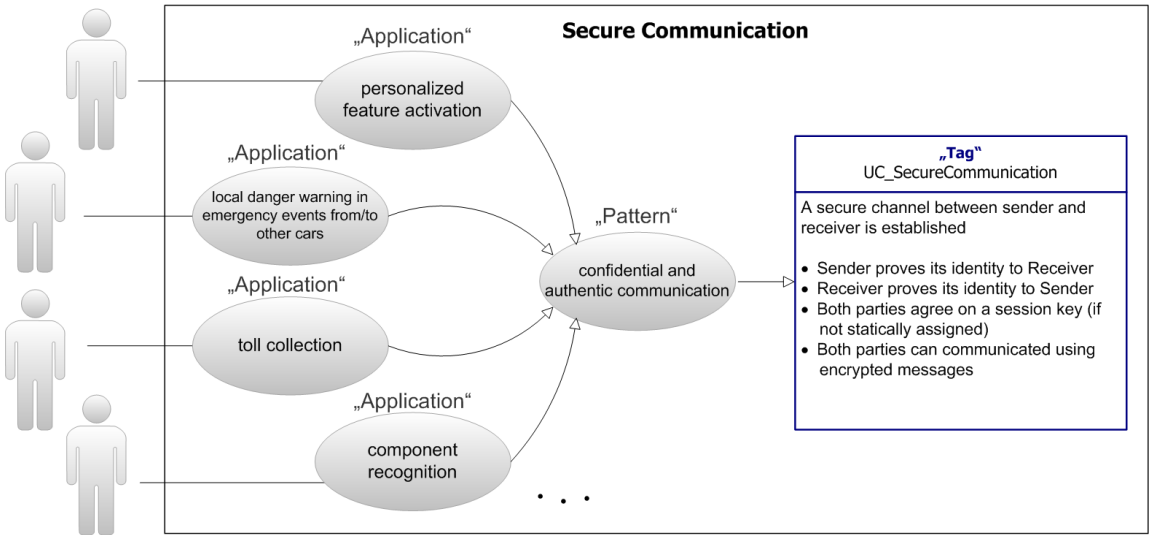

Figure 14. Secure Communication Use Case

2. **Remote Attestation**

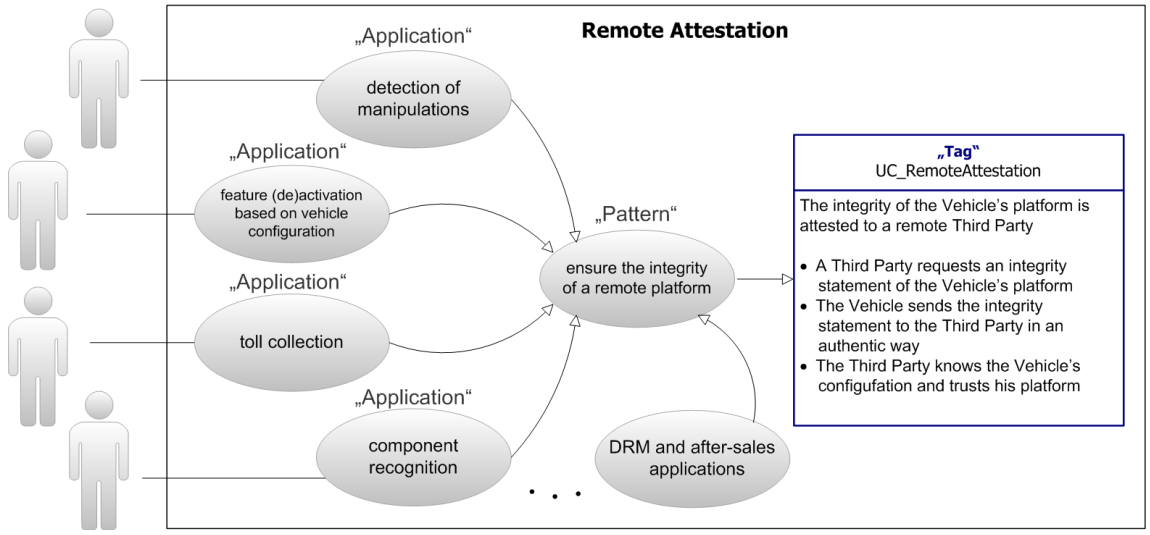

Figure 15. Remote Attestation Use Case

2.2.2.4 Pattern definition

1. **Secure Communication Pattern**

Name:

- Secure Communication

Also Know As:

- None

Example:

- Today’s vehicles are highly interconnected, e.g., for toll collection.

Context:

- Regarding the use case toll collection, the communication of the car’s OBU with the RSU of the toll provider has to be protected. Besides avoiding the transferred messages from being eavesdropped (Confidentiality) the identity of the communicating entities has to be verified (Mutual Authenticity). Furthermore it has to be ensured that the messages are transferred directly without being recorded (Freshness) and to make sure that the data sent from the vehicle and the RSU is not manipulated (Data Integrity).

- The intent of the Secure Communication Pattern is to ensure that mutual security policy objectives are met when there is a need for two parties to communicate in the presence of threats. It follows the secure weakest link principle, since the communication link is the weakest link of the system in this case [HChS04].

Problem:

- Messages need to be protected from being eavesdropped or manipulated.

Solution:

- The vehicle's CU signs and encrypts all messages using a session key that is only known by the RSU provider and vehicle itself to ensure that the driver is correctly accounted and that only an allowed RSU provider is able to decrypt the data needed for accounting.

Related Patterns:

- Mutual Authentication: How to prove its own identity to the communication partner.
- Key Agreement: How to agree on a shared session key.
- PKI: Public key infrastructure

Structure:

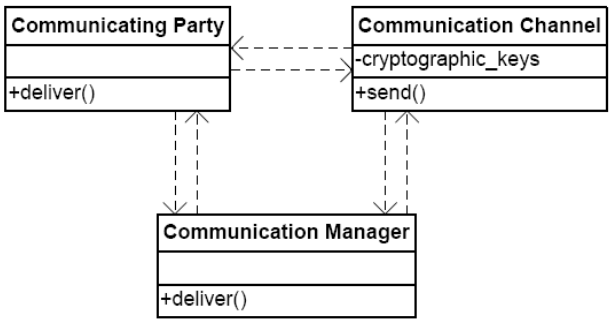

Figure 16. Secure Communication Structure

Dynamic:

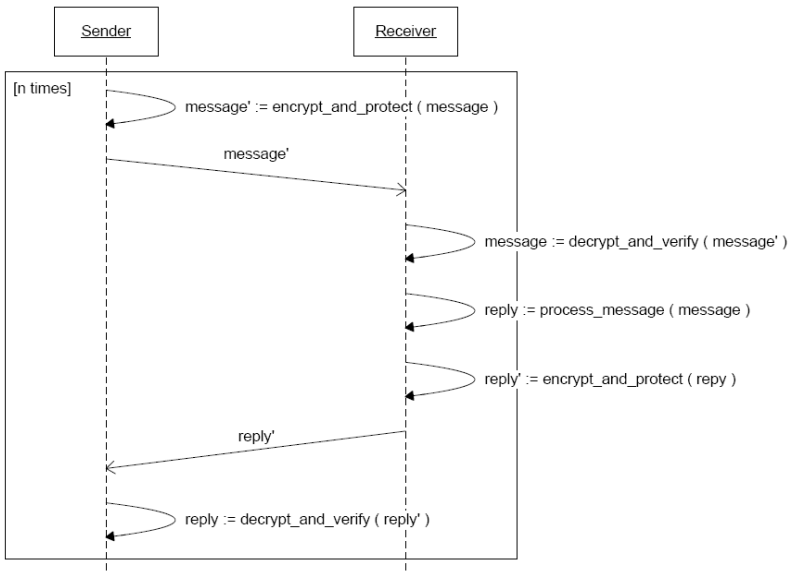

Figure 17. Secure Communication Sequence Diagram

Secure Communication Template:

Use Case Label: *UC_AD_SC*	*Type of focus:* ***Security***
Date: *01/03/2010*	***Author:*** *escrypt GmbH*
Use Case Name:	*Secure Communication*
Actors:	*OBU: extra On-Board Unit integrated in the vehicle* *RSU: Road Side Unit of the toll provider*

Triggering Event (optional):	*The communication between the vehicle's OBU and the toll provider's RSU should be carried out in a secure way.*
Relations with other use cases (not always applicable)	***Extended by:*** *None* ***Includes:*** *Mutual Authentication, Key Agreement, PKI* ***Threatened by:*** *None* ***Generalization:*** *None* ***Mitigates or prevents:*** *None* ***Detects:*** *None*
Description:	*The secure communication pattern allows two entities to communicate in a confidential and authentic way.*
Preconditions:	*Both entities can prove their identity.*
Post-conditions:	*A secure channel between OBU and RSU is established.*
Normal scenario:	• *OBU proves its identity to RSU* • *RSU proves its identity to OBU* • *Both parties agree on a session key (if not statically assigned)* • *Both parties can communicated using encrypted messages*
Alternative scenario:	
Open issues:	
Process requirements (exemplary)	• *In production process step INITIALIZATION, the platform has to be initialized with an asymmetric key pair*

Table 9. Secure Communication Template

2. *Remote Attestation Pattern*

Name:

- Remote Attestation

Also Know As:

- None

Example:

- Toll collection

Context:

- The integrity of the car's system state has to be attested. Therefore one central aspect is the evaluability of the system state.

Problem:

- For toll collection as well as for other applications that are related to Remote Attestation the integrity verification of the remote platform of the vehicle is important to ensure that it is non-malicious. This includes the integrity check of the sensors and ECUs such as the communication unit integrated in the vehicle in order to avoid attacks that for instance aim to reduce the toll payment or allow a victim to receive personal data like bank account or credit card number.

Solution:

- The remote attestation indispensably involves the secure platform initialization (e.g., authenticated boot) to determine the system state and hence to make statements regarding the integrity of the whole platform. This requires a security anchor (root of trust) in hardware, which stores the integrity measurements in protected registers (e.g., PCR). A chain of trust is achieved by consecutively measuring each component before it is being executed. To attest

the trustworthiness of the platform to a remote entity the secure hardware signs and issues the previously measured platform configuration. The platform integrity is measured at system initialization and stored within a secure hardware component. These measurements are reported to the external entity.

- In-vehicular remote attestation: At conventional remote attestation scheme, it is necessary to communicate with a verification server for attestation. However, satisfying such a requirement is hard because of a possible limitation of the mobility capability, e.g., in tunnels or underground parking spaces. The purpose this particular attestation scheme is keeping software of vehicle system healthy. In this scheme, there is at least one rich-asset ECU a master ECU. A master ECU verifies other ECUs in the role of a verification server [GHA+09].

Related Patterns:

- Secure boot: Detect manipulations at start-up

Structure / Dynamic:

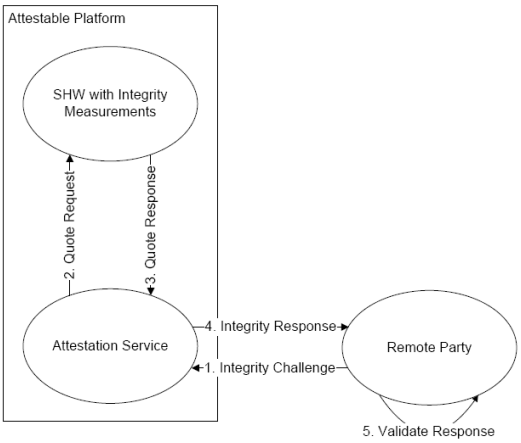

Figure 18. Remote Attestation Structure and Dynamic

Remote Attestation Template:

Use Case Label: *UC_AD_RA*	***Type of focus:*** *Security*
Date: *01/03/2010*	***Author:*** *escrypt GmbH*
Use Case Name:	*Remote Attestation*
Actors:	*Third Party* *Receiver*
Triggering Event (optional):	
Relations with other use cases ***(not always applicable)***	***Extended by:*** *None* ***Includes:*** *Secure / Authenticated Boot* ***Threatened by:*** *None* ***Generalization:*** ***Mitigates or prevents:*** *None* ***Detects:*** *None*
Description:	*A Third Party requires a statement regarding the integrity of the Vehicle's.*
Preconditions:	*The Vehicle's platform is securely initialized using secure boot.*
Post-conditions:	*The Third Party trusts the Receiver's platform.*
Normal scenario:	• *A Third Party requests an integrity statement of the Vehicle's platform.* • *The Vehicle sends the integrity statement to the Third Party in an authentic way.* • *The Third Party knows the Vehicle's configuration and trusts his platform.*

Alternative scenario:	
Open issues:	

Table 10. Remote Attestation Template

2.2.3 Home Control System

2.2.3.1 Home Control system domain

- **Overview of home systems (inspired by [Kun+95])**

An increasingly wide range of electrical, electronic and informatics products contribute to the comfort in today's home. Goods such as washing machines, refrigerators, hi-fi equipment, television, telecommunication systems, heating products, lighting systems, security systems, and so on. Recently wireless communication and mobile embedded computing devices have raised the interest and the panel of products that can be connected and the ease of use and connection between them.

These home systems products provide an increasing number of functions of growing complexity, at the same time opening up the possibility of entirely new applications. There is also an increasing trend towards functions covering the entire home e.g.: home energy management system connected to future smart grid, home lighting control, home security systems, home monitoring of assisted person. This trend towards integrating individual home products together into systems forms the basis for Home Systems. But to integrate these products together automatically requires that the various home appliances co-operate together without human intervention.

Home systems products will have to be easy to use, to install, and to move, following the plug and play concept. In order to achieve this, the following requirements will have to be met:

- Network sharing.
- Interoperability.
- Expandability.
- Device sharing.
- Flexible positioning of devices.
- Automatic configuration.

- **Network Sharing**

  It allows applications to share the same medium. This is a crucial requirement from a consumer point of view, as it allows any extra wiring or wireless device costs to be shared by different products and applications.

- **Interoperability**

  It allows products to be replaced by alternative competing products. From a consumer point of view, this is an equally crucial feature of home systems, since existing products will need to be replaced when they wear out, or when better alternatives become available.

- **Expandability**

  It allows the consumer to start from single isolated home applications and expand step by step towards a fully-fledged home systems environment.

- **Device Sharing**

  It allows home occupiers to use products that are shared by different applications, resulting in significant cost savings.

- **Flexible Positioning of Devices**

Products connected on the same sub network and on different sub networks need to communicate together. This corresponds to a requirement called ‘flexible positioning of devices’. Any purchased product can be located anywhere in the home as long as the wiring is compatible to the given product or the wireless communication protocols used are the same.

- **Automatic Configuration**

  Modification to configurations due to installation of new products or to repositioning of existing products must be handled by the Home System automatically. Just plug in the products, and they start working. This is called ‘automatic configuration’.

- **Security**
  - **Embedded systems generality**

    Embedded computers and computing devices increasingly permeate our lives. Security issues are not new for embedded systems and are critical. Security could prove a more difficult long-term problem than security does for desktop and enterprise computing since wireless communication are more and more often used. As more embedded systems are connected to the Internet, potential damages from such vulnerabilities scale up.

  - **Home control domain**

    Classical security techniques developed for enterprise do not satisfied all embedded application requirements. In the Home Control domain most of the problem belong to the Privacy which widely cover a lot of areas in security.

    For example, a remote thermostat setting control could also determine whether you are asleep, at home or out of the house. Therefore a burglar could prepare is attack according to the information that you provided. Another instance of this kind of issues is the automatic connexion of a new device to a Zigbee [Zigbee Alliance] home control network. It has to ensure that the new devices can be trusted.

    Even if all solutions do not already exist, most of them – their main principles at least - already are either in the classic computer engineering or in another field of application of embedded systems. This solution can be found for example in Cryptography, Access Control, and Authentication or Data integrity field.

    For instance, assuming that a key is provided by a trusted authority, the security of communication could be enforces through the classical “secure channel” patterns.

2.2.3.2 Requirements and mechanisms identification

ID	Title	Description	Mechanisms	Select mechanisms	Type of focus
HCA1	*ServiceBrowser* Authenticity	*ServiceBrowsers* must be authentic to belong to the group of trusted devices before acting upon *Service Request* (i.e. *Service data*).	Secure Service Discovery	Secure Service Discovery	Security
HCA2	*ServiceProvider* Authenticity	*ServiceProviders* must be authentic to belong to the group of trusted devices before issuing a *Service Request*.	Not provided by Secure Service Discovery	Secure Service Discovery	Security
HCC1	Confidentiality of Service Interaction	The Service Interactions (consisting of *service request* and *service data*) must be confidential to the authentic *ServiceBrowser* from the set of trusted devices, to the non-authentic *ServiceProvider* and those agents granted access by the *ServiceProvider*.	Secure Service Discovery	Secure Service Discovery	Security
HCI1	Integrity of Service Interaction	The integrity of Service Interactions (consisting of *service request* and *service data*) must be assured such that they can only be altered by the authentic *ServiceBrowser* from the set of trusted devices, from the non-authentic *ServiceProvider* and those agents granted access by the *ServiceProvider*.	Secure Service Discovery	Secure Service Discovery	Security

Table 11. Home Control requirements and mechanisms identification

2.2.3.3 Use case definition

1. Secure Service Discovery

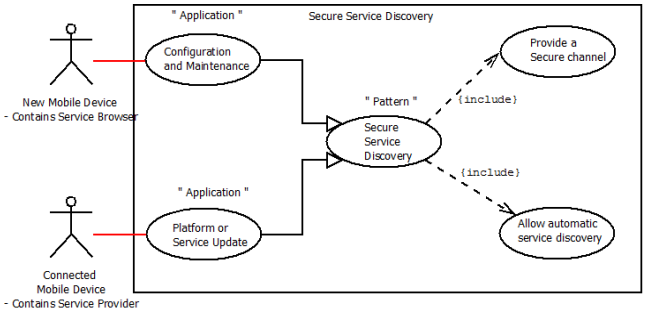

Figure 19. Secure Service Discovery Use Case

2.2.3.4 Pattern definition

1. Secure Service Discovery Pattern

Name:

- Secure service discovery

Also Know As:

- None

Example:

- Terminal (could be TV, Security camera, etc …) need to discover and select available services in the house according to the user profile (child, old person, nurse, etc …)

Context:

- Ad-hoc networked service environment which have potentially large numbers of services
- Service browser to find and utilize services.
- Some services have to be secure and the network could be unsafe.

Problem:

- Communication are not secure
- Discovery services relying on user profile and reactive (other use case could be the proactive one) announcement of possibly large amount of services
- Encryption has overhead

Solution:

- Use of a combination of Secure Channel and Intelligent Service patterns

Structure:

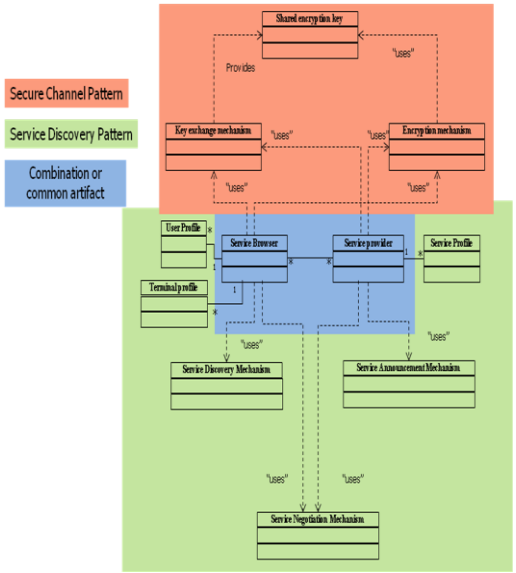

Figure 20. Secure Service Discovery Structure

Dynamic:

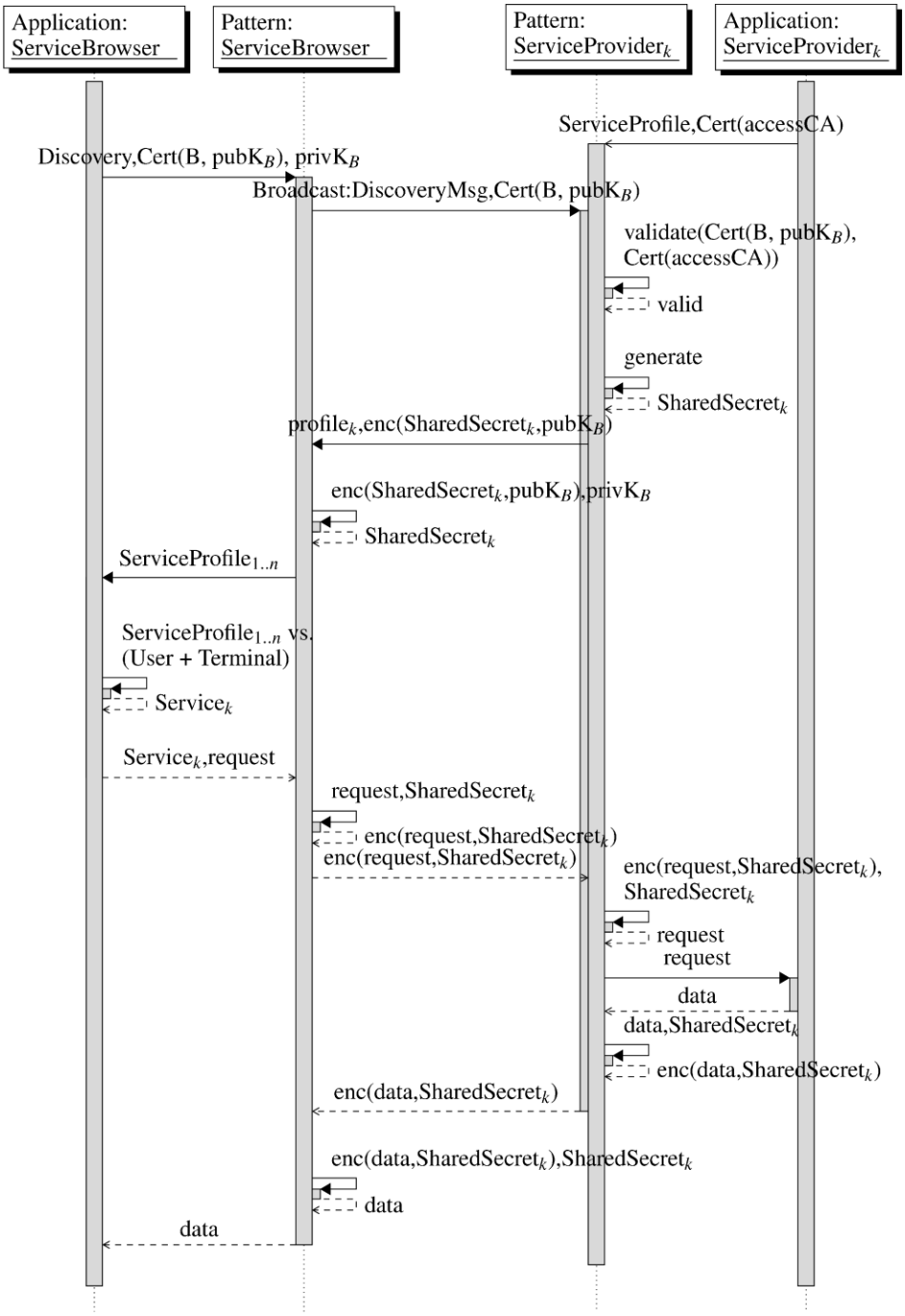

Figure 21. Secure Service Discovery Sequence Diagram

Secure Service Discovery Template

Use Case Label: *SSD-1*	***Type of focus:*** *Security*
Date: *10/02/2010*	***Author:*** *C.Grepet, C.Jouvray & A.Kung*
Use Case Name:	*Secure Service Discovery*
Actors:	*Service Browsers* *Service Providers*
Triggering Event (optional):	*The Service Browser enters in the home network area and wants to send a discovery message.* *The Service Provider is already in the home network area and wants to answer the new Service Browsers discovery message.*

Relations with other use cases ***(not always applicable)***	***Extended by:*** *None* ***Includes:*** *Certificate Based one-way Authentication | Asymmetric Encryption | Symmetric Integrity Preserving Encryption | Intelligent Service Discovery* ***Threatened by:*** *None* ***Generalization:*** ***Mitigates or prevents:*** *None* ***Detects:*** *None*
Description:	*Allow a Service Browser to request fitting services according to user and terminal profile whilst being authenticated and securing the service interaction.*
Preconditions:	• *Existing Home network* • *Access to Home network for Pattern* • *accessCA-Certificate predeployed to all Service Providers* • *accessCA-Certificate cannot be exchanged by rogue entity* • *Device Certificate issued by accessCA and corresponding private keys predeployed to all trusted Service Browsers* • *Private key is confidential to Service Browser application and pattern* • *Communication between respective Applications and Pattern instances on each device cannot be eavesdropped or altered* • *Data must be generated and handled confidentially on the devices.*
Post-conditions:	• *ServiceBrowsers are authentic to belong to the group of trusted devices before acting upon Service Request (i.e. Service data).* • *The Service Interactions (consisting of service request and service data) are confidential to the authentic ServiceBrowser from the set of trusted devices, to the non-authentic ServiceProvider and those agents granted access by the ServiceProvider.* • *The integrity of Service Interactions (consisting of service request and service data) are being assured such that they can only be altered by the authentic ServiceBrowser from the set of trusted devices, from the non-authentic ServiceProvider and those agents granted access by the ServiceProvider.*
Normal scenario:	• *ND requests list of services and sends its personal key certificate* • *CDs check access through validation of certificate* • *CDs send their service profiles and encrypted fresh shared secrets* • *ND select the needed services according to the user profile* • *ND encrypted specific service request with shared secret* • *CDs encrypted provide the service data with shared secret*
Alternative scenario:	
Open issues:	*Service Provider Authentication; ServiceProviders are not authentic to belong to the group of trusted devices before issuing a Service Request.* *Any Service Provider can impersonate any other Service Provider and thereby gain any valuable information from requests.*

Table 12. Secure Service Discovery Template

Implementation:

- Rely on both patterns information

Example resolved:

Variants:

- Use of server for both services and encryption
- Proactive announcement of services

Known uses:

Consequences:

- Provide services only to authenticated ServiceBrowsers in a confidential way with integrity.
- Limited overhead by using this mechanism only for services that need security
- Liabilities : standardization is required, overhead and cost could dramatically increase

2.2.4 Metering System

2.2.4.1 Metering system domain

Liberalization of the European energy market led to a change of roles in the sector of producing and consuming energy. Formerly the market was dominated by large energy producing companies and end customers consuming the generated energy. In the liberalized energy market every market participant is able to buy, sell or trade energy. Nowadays the energy consumer is enabled to choose between different energy providers.

This fact caused a change in metering sector. Formerly large energy providing companies had their own mechanically operated energy meters.

EU Directive 2006/32/EC states that for new installations only smart meters may be used. The customer should be able to view his actual energy consumption. This should make it more attractive to reduce the energy consumption. If the customer wishes, a monthly billing of the consumed energy has to be made available by the energy provider. This makes remote readout of measurement data using modern communication techniques economically reasonable.

Modern measuring devices for gas, water or electricity are embedded systems. Nowadays there is the role of the measuring point operator, who is the owner of the measuring device and provides the service of measuring consumed or generated energy to other energy market participants. As embedded systems run software, there has to be a possibility for updates to remove errors or add new features. Concerning security and dependability the following design patterns for metering applications are of great importance: Secure Remote Readout and Secure Software Download.

The following parts give a short overview of the roles and actors in the liberalized energy market and points out interesting use cases from the metering domain.

- **Actors in the metering sector application domain**
  - **Measuring point operator (MPO)**
    The measuring point operator is the owner of the metering device.
  - **Metrology institute (MI) / notified body**
    The (federal) metrology institute is responsible for the compulsory calibration guidelines. The type approval for new meters can be accomplished either by the Metrology institute itself or by a so called Notified Body. The Notified body is allowed to represent the Metrology institute regarding the type approval. In this document we will only use the term Metrology institute.
  - **Software publisher**
    The software publisher develops software/firmware for embedded metering systems.
  - **Remote meter readout center (RRC)**
    The remote meter readout center collects counter readings of different types of meters (Electricity, water, heat, gas) and forwards them to the corresponding energy providers.
  - **Energy provider**

Affiliated groups like RWE, EON and Vattenfall in Germany are typical energy providers.

- **Energy producer**

  The energy producer sells generated energy

- **Service technician**

  The service technician is responsible for all service issues regarding the meter. This predominantly includes installation and commissioning.

- **Network operator**

  The network operator provides the transport network for the different types of energy (Electricity, water, heat, gas).

- **Meter manufacturer**

  The meter manufacturer develops and produces the hardware of the metering device, programs it with the appropriate software from the software publisher and sells it to the measuring point operator.

- **Entity authorized for software download (EASD)**

  This entity is authorized to launch the software update mechanism on the metering device. E.g. this can be the Metrology Institute or the Meter Manufacturer, depending on the chosen MID module for type approval.

- **Energy consumer**

  The energy consumer is the person, whose energy consumption is measured by the meter, collected by the remote meter readout center and billed by the energy provider(s).

- **Readout entity**

  This entity represents the actor who wants to read out data in the Secure Remote Readout pattern. E.G. this can be the Metrology Institute or the Remote meter readout centre.

- **Use Case examples overview**

Use case	Description
Set metering system time	The metrology point operator, technician or another authorized actor needs to set the meters internal RTC. This action has to be noted in the compulsory relevant log file (PTB requirement) and is security relevant (authenticity, integrity, access control).
Time synchronization	Time synchronization is needed to ensure the accuracy of the timestamps of the measurement information. The internal RTC is synchronized to an external clock source. There are requirements (PTB) regarding the synchronization frequency and the amount of time. As long as they are applied to, the process of time synchronization is not noted in the compulsory relevant log. If these restrictions cannot be applied to, the set metering system time process has to be used to set the correct system time.
Set tariffs (electricity, water, etc.)	The energy provider sets tariff information to inform the energy customer about the current pricing. This service is security relevant (authenticity, integrity, access control).
Change of Measuring Point Operator	In case of change of owner (Measuring Point Operator), the security relevant information (Keys, passwords, certificates, etc.) needs to be updated/changed. This service is security relevant (authenticity, integrity, access control).
Change of Remote Meter Readout Center	If the energy provider changes the Remote meter readout center, this change needs to be send to the meter. The new public key or certificate has to be stored securely. This service is security relevant (authenticity, integrity, access control).
Change of Energy Provider	The liberalized energy market allows every consumer to freely choose his energy provider. The new public key or certificate has to be stored securely. This service is security relevant (authenticity, integrity, access control).
Software Update	A software update on the metering device can be necessary to fix software errors and support a long system lifetime. This feature can also be used to add new functionality to the meter. The software update of compulsory relevant software has to be distinguished from the update of non compulsory relevant software (-parts). The Measuring Point operator sends the command to the meter to enable the update functionality. This means the meter is now ready for the software download to the designated memory. The second step is to download the software and check the Measuring Point Operator's signature. In case of compulsory relevant software, the Metrology Institute's signature has to be additionally verified. The third step is to start the new device after a successful verification process.
Remote readout of measurement data	One central use case in the Metering sector is the remote readout of measurement data. The measurement data is the basis of the calculation of the amount that will be billed to the energy consumer. Therefore the measurement data is subject to compulsory calibration. The measurement data is read out by the remote readout

	center. The integrity and authenticity of the data must be ensured.
Remote readout of software version	When a meter allows the download of newer software versions it must also provide the possibility to read out what version is running. One precondition for a software download is that the downloaded software version must be newer than the currently running version. The integrity and authenticity of the data must be ensured.
Remote readout of status	To readout the status of the metering device no special access level is needed.
Remote readout of compulsory calibration log	When a parameter of the metering device or its software is changed that is subject to compulsory calibration this change must be made retraceable. The changing of parameters that are subject to compulsory calibration is only allowed, when it either implies breaking of a calibration seal or when the change is written into the compulsory calibration log. When the log is read out it must be made sure that the integrity and authenticity of the data is provided. The integrity and authenticity of the data must be ensured.
Remote readout of load profile	For some corporate customers the billing is based on registration periods of e.g. 30 or 15 minutes. In this case a load profile meter has to be recorded over a longer period of time. The remote readout of a load profile results in a large response to the request. The integrity and authenticity of the data must be ensured.
Remote readout of system log	A metering device may have an additional log file that is not subject to compulsory calibration.

Table 13. Description of Metering Use Cases

- **Functional overview of metering use cases**

  The relevant use cases and needed patterns are chosen by building up a functional overview of exemplary use cases. For the TERESA-project, security and dependability functions are in the center of interest.

  Only use cases dealing with compulsory relevant parts of the metering system software will be selected.

Data direction	Type of data	Security relevance	Compulsory calibration relevance	Example Use Cases
Read from meter	Single data	Security relevant	Relevant for compulsory calibration	Read measurement data
			Non relevant for compulsory calibration	Read software version
		Non security relevant	Non relevant for compulsory calibration	Read status
	Record/ data stream	Security relevant	Relevant for compulsory calibration	Read compulsory calibration log
				Read load profile
			Non relevant for compulsory calibration	Read system log
Write to meter	Single data	Security relevant	Relevant for compulsory calibration	Set time
			Non relevant for compulsory calibration	Set tariffs
				Time synchronization
				Change Measuring Point Operator
				Change Remote Meter Readout Center
				Change Energy Provider
	Record/data stream	Security relevant	Relevant for compulsory calibration	Software download

Table 14. Metering Use Cases Selected

2.2.4.2 Requirements and mechanisms definition

ID	Title	Description	Mechanisms	Select mechanisms	Type of focus
SD01	Authenticity of the person initiating a secure software download	The authenticity of the person that initiates the download (entity authorized for software download) must be verified, before a software image is transmitted and stored in the target devices memory.	Secure Software Download	Secure Software Download	Security
			Manual update by trustworthy person with physical access to the device.		
SD02	Integrity of the software image transmitted via Secure Software Download	The integrity of the software image transmitted to the target device must be verified.	Secure Software Download	Secure Software Download	Security
			Manual update by trustworthy person with physical		

			access to the device.		
SD03	Authenticity of the software images origin transmitted via Secure Software Download	It must be possible for the measurement device to verify the authenticity of the software image's origin.	Secure Software Download Manual update by trustworthy person with physical access to the device.	Secure Software Download	Security
SD04	Software version check in Secure Software Download	The currently running Software must not be replaced with an older or the same version of the software.	Secure Software Download Manual update by trustworthy person with physical access to the device.	Secure Software Download	Security
SD05	Software Image transmitted via Secure Software Download must have been approved by Metrology Institute.	In all cases legally relevant software is updated, the software image must have been approved by the Metrology Institute before.	Secure Software Download Manual update by trustworthy person with physical access to the device.	Secure Software Download	Security
RR01	Secure Remote Readout	Measurement Data has to be read out of the metering device. It must not be possible to manipulate the measurement values after being read from the calibrated sensor.	Secure Remote Readout Customer fills out Postcard with meter reading and sends it by mail Employee of the energy producer locally collects meter readings	Secure Remote Readout	Security

Table 15. Metering requirements and mechanisms identification

2.2.4.3 Use case definition

The following use cases diagrams will describe the functionality of the mechanisms previously selected.

1. **Secure Software Download**

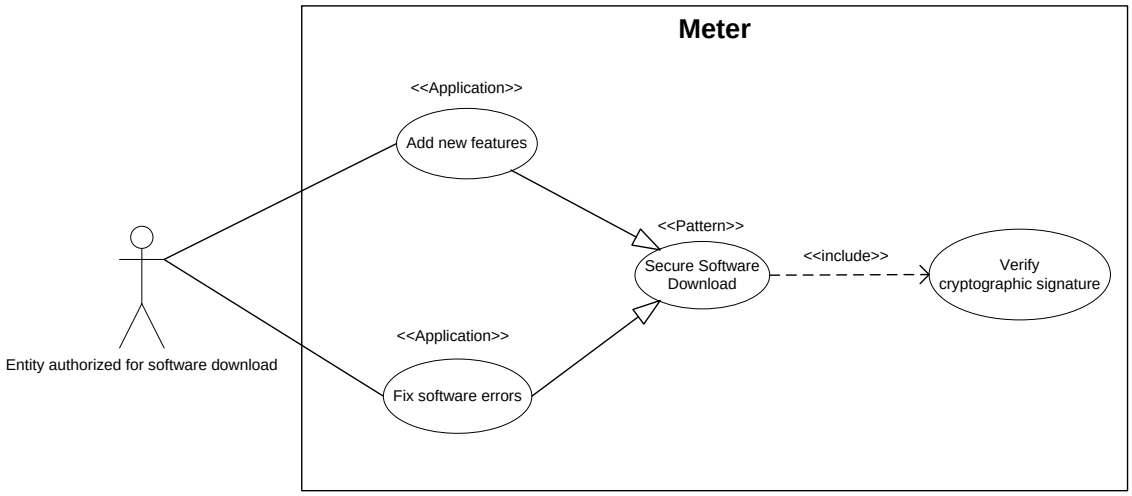

Figure 22. Secure Software Download Use Case

2. **Secure Remote Readout**

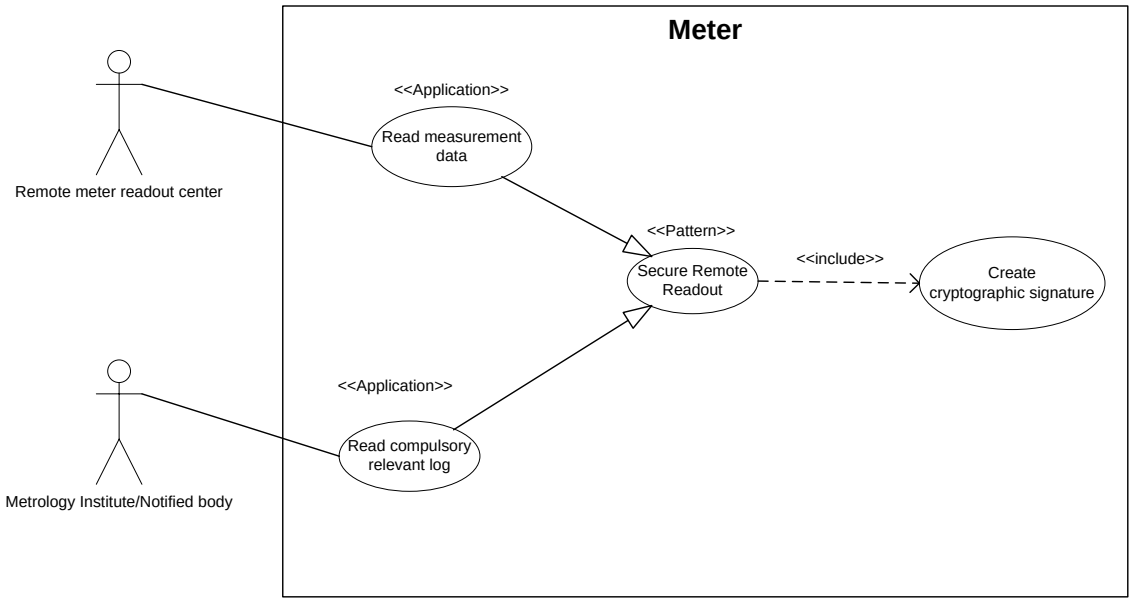

Figure 23. Secure Remote Readout Use Case

2.2.4.4 Pattern definition

Based on the information derived above, the following pattern have been identified and defined for the different application sectors.

1. **Secure Software Download Pattern**

Name:

- Secure Software Download

Also known as:

- Software Firmware Update
- Software Update

Example:

- Secure software download for embedded systems like meters, electronic control units et cetera, wherever a software update can be necessary without exchanging the physical system.

Context:

- The target device stays in use at the customer for a longer period of time, e.g. 20 years without direct access by service personnel.
- The target device is connected to a possibly unsafe communication network
- The target device has restricted memory resources. There is only enough space for two software images: the currently running version and the new software version.
- The bootloader shall be as small as possible. E.g. It does not contain any cryptographic functions or libraries.
- A secure storage is available which guarantees authenticity of the saved software image.

Problem:

- Guarantee authenticity and integrity of the firmware image and access control to the update interface. Ensure that only certified software images (in regard to compulsory relevance) are downloaded and installed.

Solution:

- The update takes place in three steps. At first the client that provides the new software has to authenticate at the target system, so the target system can enable the download functionality. Second, the software is transferred into a designated secure memory region at the target system and the integrity and authenticity of the software image is checked on the target system. Third, the target system is restarted and the bootloader loads the new software image from the designated memory and starts it.

Dynamics:

- The dynamics of the software update process is shown in the two sequence diagrams below. The first one gives an overview about the whole process starting at the provision of the software update until the successful start of the new system software. The second diagram shows the interaction between the Entity Authorized for Software Download and the Metering Device in more detail and highlights the data transmitted during the process.

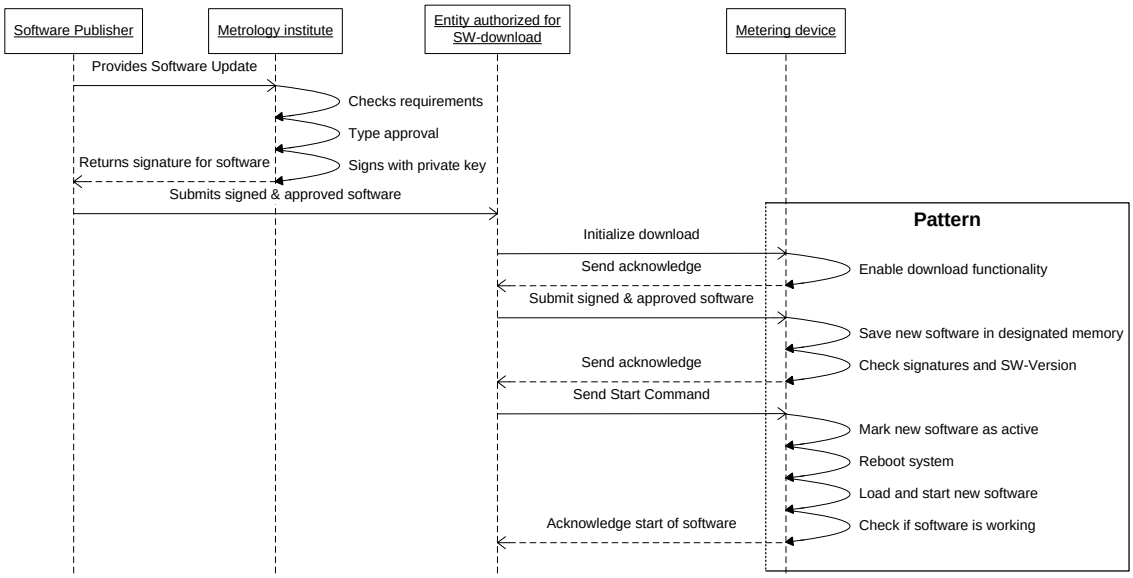

Figure 24. Secure Software Download Sequence Diagram Overview

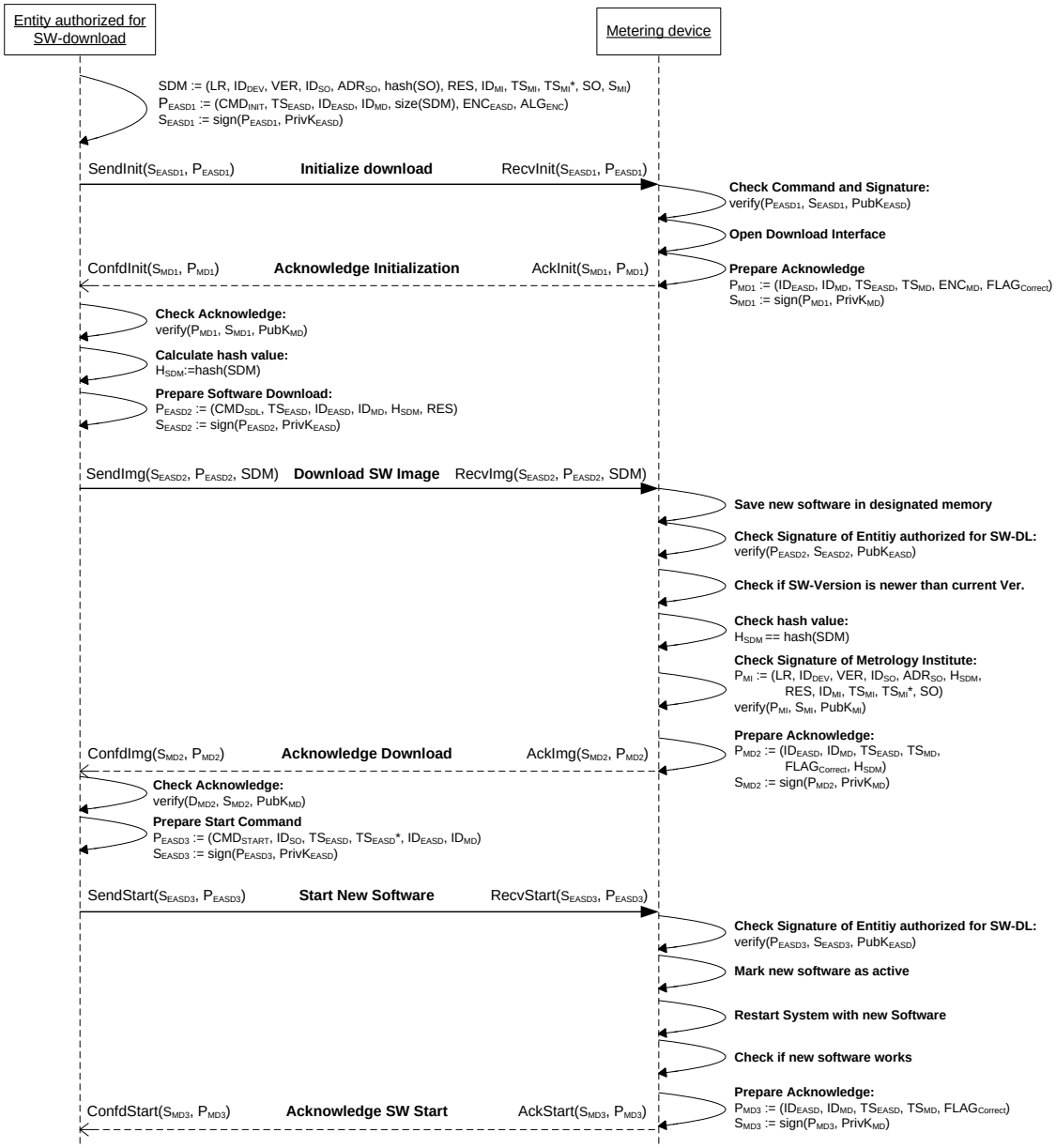

Figure 25. Secure Software Download Sequence Diagram Overview

Symbol	Description
ADR_{SO}	Start Memory Address
ALG_{ENC}	Symmetric encryption algorithm and parameters
CMD_{INIT}	Command to initialize Software Download
CMD_{SDL}	Command to download the Software
CMD_{START}	Command to start the new Software Image
ENC_{EASD}	Part of secret key K from EASD (Diffie Hellman Key Exchange)
ENC_{MD}	Part of secret key K from Meter Developer (MD) (Diffie Hellman Key Exchange)
$FLAG_{Correct}$	Flag: last operation was successful
hash(SDM)	Hash value calculated over the Software Download Module
hash(SO)	Hash value calculated over Software Image
ID_{DEV}	Unique Identifier of the meter developer
ID_{EASD}	Unique Identifier of EASD

ID_{MD}	Unique Identifier of Metering Device
ID_{MI}	Unique Identifier of the Metrology Institute
ID_{SO}	Unique Identifier of the Software
LR	Flag: legally relevant software
P_{EASDn}	Data Payload sent from Entity Authorized for Software Download to the Metering Device
P_{MDn}	Data Payload sent from Metering Device to the Entity Authorized for Software Download
P_{MI}	Data Payload signed by the Metrology Institute
$PrivK_{EASD}$	Private Key of the Entity Authorized for Software Download
$PrivK_{MD}$	Private Key of the Metering Device
$PrivK_{MI}$	Private Key of the Metrology Institute
$PubK_{EASD}$	Public-Key of the Entity authorized for Software Download
$PubK_{MD}$	Public Key of the Metering Device
$PubK_{MI}$	Public Key of the Metrology Institute
RES	Reserved field for future use
SDM	Software Download Module
S_{EASDn}	Cryptographic Signature calculated over Payload P_{EASDn}
size(SDM)	Size of Software image
S_{MDn}	Cryptographic Signature calculated over Payload P_{MDn}
S_{MI}	Signature from the Metrology Institute calculated over all previous fields
SO	Software Image
TS_{EASD}	Timestamp by Entity Authorized for SW-DL before the download started
TS_{EASD}*	Timestamp by Entity Authorized for SW-DL after the download finished
TS_{MD}	Timestamp by the Metering Device
TS_{MI}	Timestamp when the Software Image has been received by the MI
TS_{MI}*	Timestamp: Start of validity
VER	Software – Version

Table 16. Description of Symbols used in the Secure Software Download Sequence Diagram

The sequence diagram above shows the dynamics of the software update process from provision of the software update until the successful start of the new system software.

Secure Software Download Use Case Template:

Use Case Label: *UC_MD_SSD*	***Type of focus:*** *Security*
Date: *09/04/2010*	***Author:*** *Christian Bodenstedt & Donatus Weber*
Use Case Name:	*Secure Software Download*
Actors:	*Software publisher,* *Metrology Institute,* *Entity authorized for software download (e.g. measuring point operator, service technician),* *Metering Device*
Triggering Event (optional):	*Embedded system of a metering device needs software update*

Relations with other use cases (not always applicable)	***Extended by:*** *None* ***Includes:*** *Creation and verification of cryptographic signatures* ***Threatened by:*** *None* ***Generalization:*** • *Add new features* • *Fix software errors* ***Mitigates or prevents:*** *None* ***Detects:*** *None*
Description:	*The template allows a software update process which guarantees authenticity and integrity of a software image downloaded via an insecure communication channel. The launch of the update process is protected by an access control. The integrity and authenticity of the downloaded software is checked at the end of the transmission of the whole image. Storage of the image until the new software is started takes place in a designated secure memory, so that a lightweight boot loader does not have to check the authenticity again.*
Preconditions:	• *Cryptographic keys and the unique identifiers of the involved parties must be available in the Metering Device (see "Parameters to be provided").* • *Secure storage for keys and identifiers: For an unauthorized person, it must not be possible to manipulate the cryptographic keys. It must not be possible to read out the private key of the metering device.* • *Secure storage for software image: When the software image is loaded from the designated memory into the program memory of the MCU/CPU, only a very small boot loader is running. The bootloader is not able to check cryptographic signatures, thus the software image itself has to be saved in a secure storage in order to provide authenticity of the software image.* • *Cryptographic Library providing a one way hash function and algorithms for asymmetric en- and decryption.*
Parameters to be provided	*The following parameters have to be provided to the embedded system:* • *$PrivK_{MD}$: Private Key of the Metering Device (to sign parts of the responses).* - *Must only be known to the Metering Device. It must not be possible for external agents to read out or change the Private Key.* • *$PubK_{EASD}$: Public key of the Entity authorized for software download (to check the authenticity of this actor for access control and to check the authenticity of the origin of data of the Software Image).* - *Must authentically originate from the Entity authorized for software download* • *$PubK_{MI}$: Public key of the Metrology Institute (to check the cryptographic signature of compulsory relevant software images).* - *Must authentically originate from the Metrology Institute* • *ID_{EASD}: The unique identifier of the Entity authorized for software download.* - *Must authentically originate from the Entity authorized for software download* • *ID_{MI}: The unique identifier of the Metrology Institute.* - *Must authentically originate from the Metrology Institute* • *ID_{MD}: The unique identifier of the Metering Device.* *The following parameters have to be provided to the Entity authorized for software download:*

	• *$PubK_{MD}$: Public Key of the Metering Device (to check cryptographic signatures in responses of the MD).* - *Must authentically originate from the Metering Device* • *$PrivK_{EASD}$: Private Key of the Entity authorized for software download (to create cryptographic signatures).* - *Must authentically originate from the Entity authorized for software download* • *ID_{EASD}: The unique identifier of the Entity authorized for software download.* - *Must authentically originate from the Entity authorized for software download* • *ID_{MI}: The unique identifier of the Metrology Institute.* - *Must authentically originate from the Metrology Institute* • *ID_{MD}: The unique identifier of the Metering Device to address.* - *Must authentically originate from the Metering Device*
Post-conditions:	• *Embedded system runs with downloaded software update* • *The authenticity of the person initiating the software download has been validated.* • *The integrity of the downloaded software image has been checked.* • *The authenticity of the software images origin has been checked.* • *The version number of the new software image is greater than the version number of the previously running software.* • *For the update of legally relevant software only software images approved by the Metrology Institute were accepted.*
Normal scenario:	• *Software publisher provides a new version of a software* • *Metrology institute approves compulsory calibration of relevant software parts* • *Entity authorized for software download sends command to launch the update process* • *Meter downloads software to designated memory and checks integrity & authenticity* • *After having received the appropriate command, the system is restarted and the bootloader checks the integrity of the software update and starts the new system software.* • *In case of errors/problems, the systems falls back to the last working system software*
Alternative scenario:	*None*
Open issues:	*None*

Table 17. Secure Software Download Template

2. Secure Remote Readout Pattern

Name:

- Secure Remote Readout

Also known as:

- Secure Readout

Example:

- Remote readout of data stored in an embedded device ensuring integrity and authenticity. In the metering domain this is used for transferring measurement data or compulsory relevant log files.

Context:

- The lawful requirements for metering devices comprise the demand of measuring data being non manipulable from recording until billing (Transparency from measurement recording until billing for all market participants). The Secure Remote Readout is used to ensure integrity and authenticity of the transferred data and prevent manipulation.

Problem:

- Ordinary measurement tuples or log file entries are easily manipulable. Authenticity and integrity have to be assured when transferring measurement data from the meter to the Remote Meter Readout Center. Also the compulsory relevant log file needs to be securely read out (guaranteeing authenticity and integrity of every single entry) by the Metrology Institute/ Notified body.

Solution:

- The data to be remotely read out (Measurement data and compulsory relevant log for meters) is protected against manipulation with the help of digital signatures. In the moment the data becomes available, a timestamp is appended and a digital signature is created over the data and the timestamp. These three parts (data, timestamp, signature) are concatenated to a tuple. Any of the later processing steps (e.g. saving to persistent memory, transferring to the remote readout center) works on the whole tuple.
- ***Dynamics:***

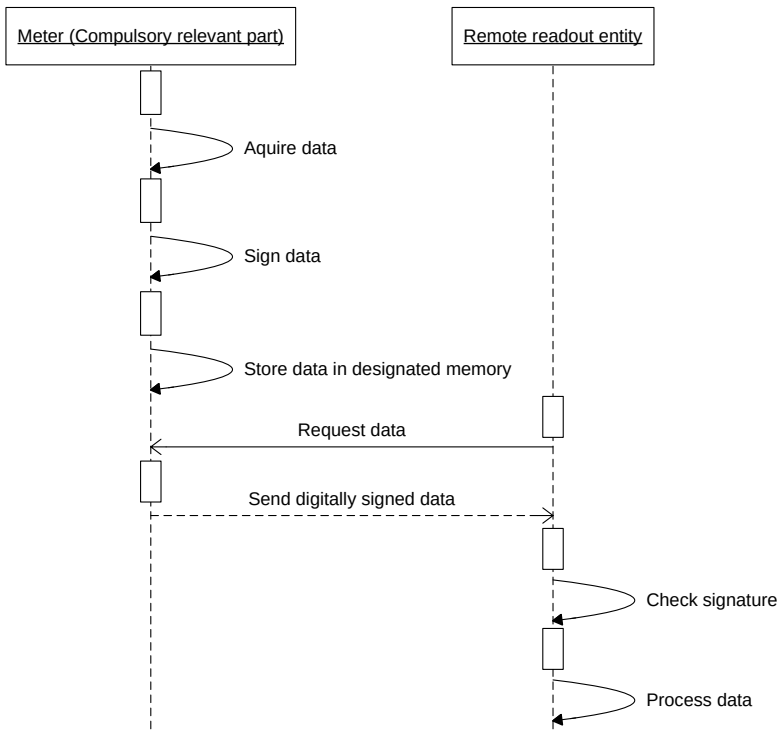

Figure 26. Secure Remote Readout Sequence Diagram

Secure Remote Readout Use Case Template:

Use Case Label: *UC_MD_SRR*	***Type of focus:*** *Security*
Date: *09/04/2010*	***Author:*** *Christian Bodenstedt & Donatus Weber*
Use Case Name:	*Secure Remote Readout*
Actors:	*Metrology Institute/notified body, Remote Meter Readout Center*
Triggering Event (optional):	*None*

Relations with other use cases (not always applicable)	***Extended by:*** *None* ***Includes:*** *Creation of cryptographic signature* ***Threatened by:*** *None* ***Generalization:*** • *Read measurement data* • *Read compulsory relevant log* ***Mitigates or prevents:*** *None* ***Detects:*** *None*
Description:	*Data to be protected is handled in the following way: The moment the data becomes available, a timestamp is appended and a digital signature is created over the data and the timestamp. These three parts (data, timestamp, signature) are concatenated to a tuple. Any of the later processing steps (e.g. saving to persistent memory, transferring to the remote readout center) works on the whole tuple.*
Preconditions:	• *Cryptographic keys must be available to be able to add a digital signature to the sensitive data. E.g. each metering device generates its own private/public key pair when it is started the first time. The private key of the metering device is used to create the cryptographic signature.* • *It must not be possible to read out the private key of the metering device. Thus it has to be put into a secure storage, e.g. a temper resistant memory on the MCU chip that can only be read by the MCU itself but not from external devices.*
Post-conditions:	*Authenticity and integrity for transferred data is ensured*
Normal scenario:	• *Metering system acquires the relevant data (e.g. measurement value from the sensor)* • *Data is digitally signed in the secure environment* • *Signed data is stored in the designated memory* • *Data is requested by an external entity (e.g. Metrology Institute/notified body)* • *Signed data is sent to the external entity by the meter* • *External entity checks signature and processes data*
Alternative scenario:	*None*
Open issues:	*None*

Table 18. Secure Remote Readout Template

3 Pattern integration

3.1 Process guidelines for pattern integration

Due to the variability of the development environment in the different application sectors and because of the nature of the patterns, it is important to identify for each application domain the entire set of actors that participate in a standard project. Some of these actors might interact with the identified patterns during pattern integration process.

A complete analysis of each application sector must be carried out to identify the set of actors, roles, and actions related with patterns integration in a new design. The procedure to analyze application domains to perform this analysis and ease pattern integration is described in this section.

3.1.1 Actors definition table

To establish a set of standard names for the actors within the TERESA project, the main roles from the V-Modell XT part 4 [V-Modell XT, 2006] have been taken as a reference.

The goal is to identify the actors within each application sector that develop the roles described in the V-Modell XT. First of all, the partners must identify the actors that participate in a standard project of their own application domain from the list of actors proposed by the V-Model XT (Annex A). Afterwards, each partner must fill the roles classification table shown below by marking in which roles the actors are involved.

This procedure can be summarized as follows:

- Identify the actors in the development process of a specific application sector.
- Map the identified actors with those extracted from the V-Modell to unify criteria.
- Identify the roles and tasks for each actor.

To collect this information the Role classification table below has been defined (the table for each application domain has been defined in an excel file).

Industrial Control System																	
Roles / Actors	Assesor	Change Request Manager	CM Manager	Hardware Architect	Hardware Designer	Verification Inspector	Validation Inspector	Process Engineer	Project Leader	Quality Manager	Requirement Engineer	Safety Manager	Security Manager	Software Architect	Software Developer	System Architect	System Integrator
Client											X						
Executive																	
Project Leader			X					X	X								
Assesor	X																
Hardware Architect		X		X							X						O
Software Architect		X									X			X			O
Hardware Developer		O			X												C
Software Developer		O													X		C
System Architect		X						X			X					X	X
Verification Inspector		O				X	X										
Quality Manager		C								X							
Security / Safety Manager												X	X				

Table 19. Roles Classification

To discern between main roles and roles in which an actor only cooperate some special marks have been defined:

X : Mandatory participation

O : Mandatory participation (optional)

C : Cooperation

3.1.2 *Pattern integration Use Case*

This action is required to know which actors participate in the integration of the pattern in a given design and what are the activities in which they are involved in this process. Once the actors that will take part in the project are identified, it is easier to describe pattern integration process.

The following diagrams are required to describe the pattern integration in a new design:

- *Use Case diagram.* This diagram is used to identify all the actions required to integrate the pattern in a specific design and at the same time to establish all the actors involved in these actions.
- *Sequence diagram.* It provides a description of the activities carried out in the integration process.
- *Use case template.* It details the use cases identified in the use case diagram. (The structure of the template is the same as the one used in the pattern definition)

Note: The process for pattern integration in each application sector should be similar, but some considerations must be taken into account. For that reason, an independent process for pattern integration for each application sector must be defined.

3.2 Engineering Roles Classification

3.2.1 Industrial Control System

Industry Control Systems																	
Roles / Actors	Assesor	Change Request Manager	CM Manager	Hardware Architect	Hardware Designer	Verification Inspector	Validation Inspector	Process Engineer	Project Leader	Quality Manager	Requirement Engineer	Safety Manager	Security Manager	Software Architect	Software Developer	System Architect	System Integrator
Client											X						
Executive																	
Project Leader			X					X	X								
Assesor	X																
Hardware Architect		X		X							X						O
Software Architect		X									X			X			O
Hardware Developer		O			X												C
Software Developer		O													X		C
System Architect		X						X			X					X	X
Verification Inspector		O				X	X										
Quality Manager		C								X							
Security / Safety Manager												X	X				

Table 20. Industrial Roles Classification

3.2.2 Automotive System

Automotive Systems														
	Small													
Roles / Actors	Change Request Manager	Ergonomics Manager	Hardware Architect	Hardware Developer	Inspector	Process Engineer	Project Leader	Quality Manager	Requirements Engineer	Security/Safety Manager	Software Architect	Software Developer	System Architect	System Integrator
Hardware Architect	X		X										C	
Hardware Developer				X									C	
Inspector					X									
Project Leader		X				X	X						C	
Quality Manager		C						X					C	C
Requirement Engineer									X				C	
Security/Safety Engineer										X			C	
Software Architect	X										X	C	C	
Software Developer											C	X	C	
System Architect			C	C			C	C	C	C	C	C	X	
Electrical Engineer			X	X										
Cost Controller							X							

Table 21. Automotive Roles Classification

3.2.3 Home Control System

Home Control System												
Roles / Actors	Hardware Architect	Hardware Developer	Inspector	Process Engineer	Project Leader	Purchaser	Quality Manager	Security Manager	Software Architect	Software Developer	System Architect	System Integrator
Hardware Architect	X	O				O						
Hardware Developer		X										
Inspector			X				C					
Project Leader					X							
Quality Manager							X	O				
Security / Safety Manager	C			C			O	X	C		C	
Software Architect									X	O		
Software Developer								C		X		
System Architect	C	C		C		O			C		X	
System Integrator								C				X

Table 22. Home Control Roles Classification

3.2.4 Metering System

Metering Systems																	
	Large																
Roles / Actors	Assessor	Change Request	CM Manager	Executive	Hardware Architect	Hardware Developer	Inspector	Process Engineer	Project Leader	Quality Manager	Requirements Engineer	Safety Manager	Security Manager	Software Architect	Software Developer	System Architect	System Integrator
Client											X						
Executive				X													
Project Leader			X					O	X								
Assesor	X																
Hardware Architect		X			X	C					O						O
Software Architect		X									O			X	C		O
Hardware Developer		O				X											C
Software Developer		O													X		C
System Architect		X							O		X					X	X
Verification Inspector		O					X										
Quality Manager		C								X							
Security / Safety Manager												X	X	C		C	

Table 23. Metering Roles Classification

The actor called “assessor” is domain specific for the metering sector, especially for Germany. Every new developed metering device needs a type examination for the final type approval before it can be placed on the market. In the table above the actor “assessor” has been used to represent the function of an external institution, the metrology institute (or a notified body who represents the metrology institute).

The notified body / metrology institute is responsible for the accomplishment of the type approval. Only then a meter may be placed on the market.

3.3 Use case pattern integration

3.3.1 Industrial Control System

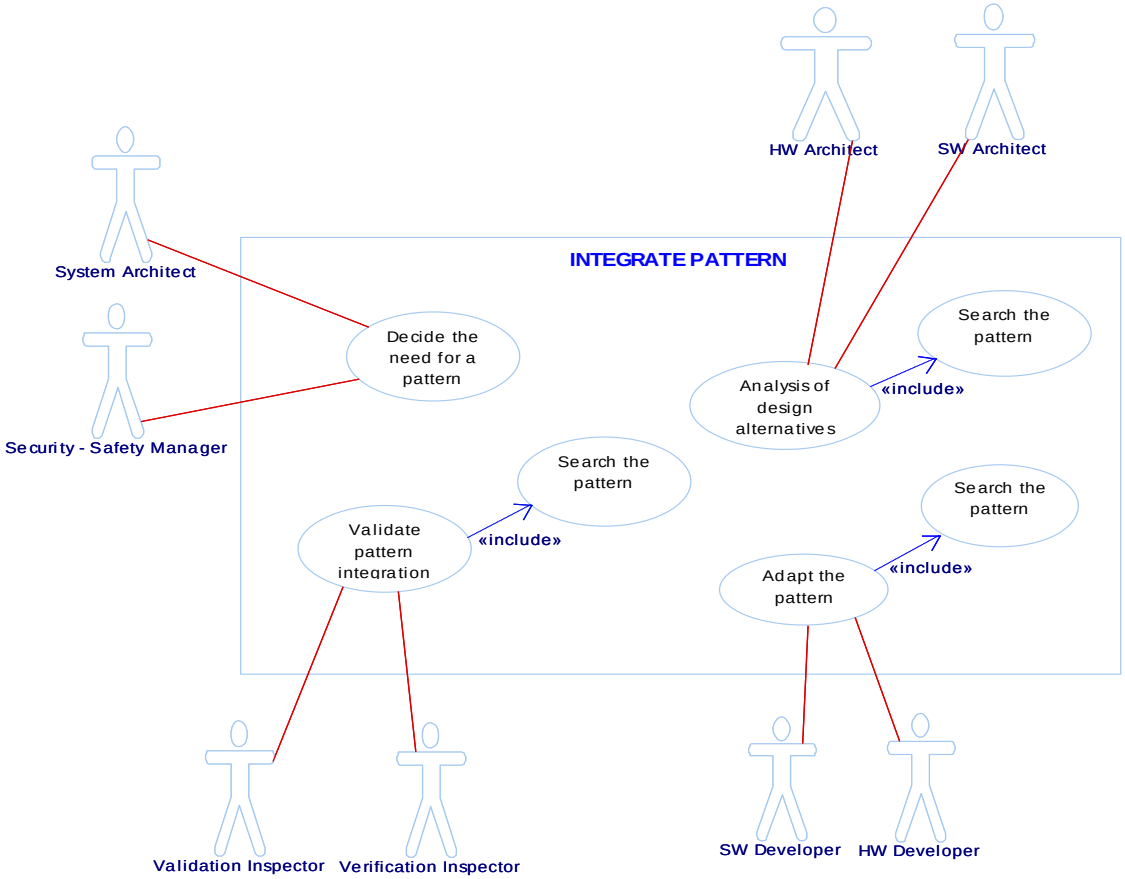

Figure 27. Industrial pattern integration Use Case

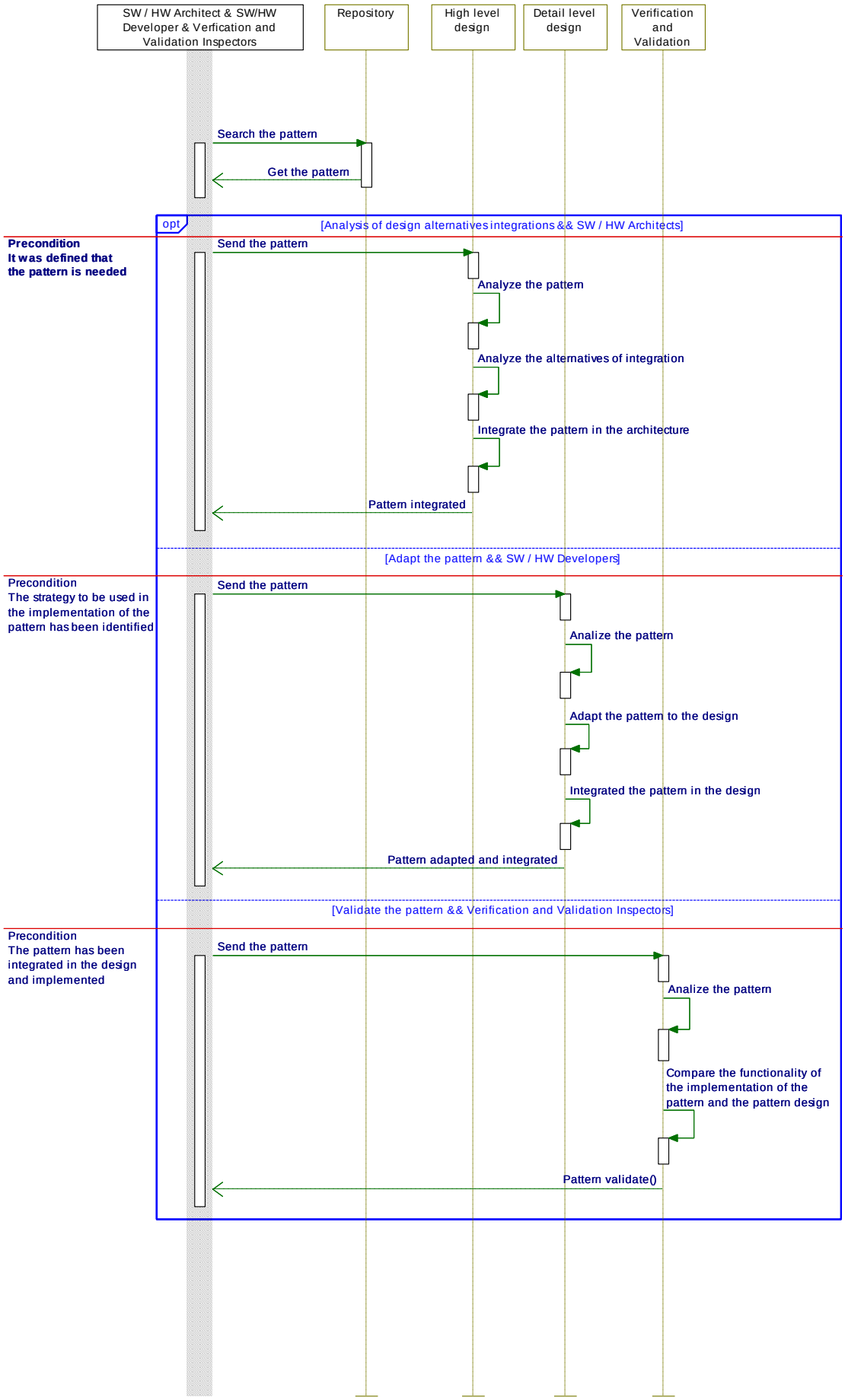

Figure 28. Sequence diagram for Analysis, adaptation and validation of Majority voter pattern.

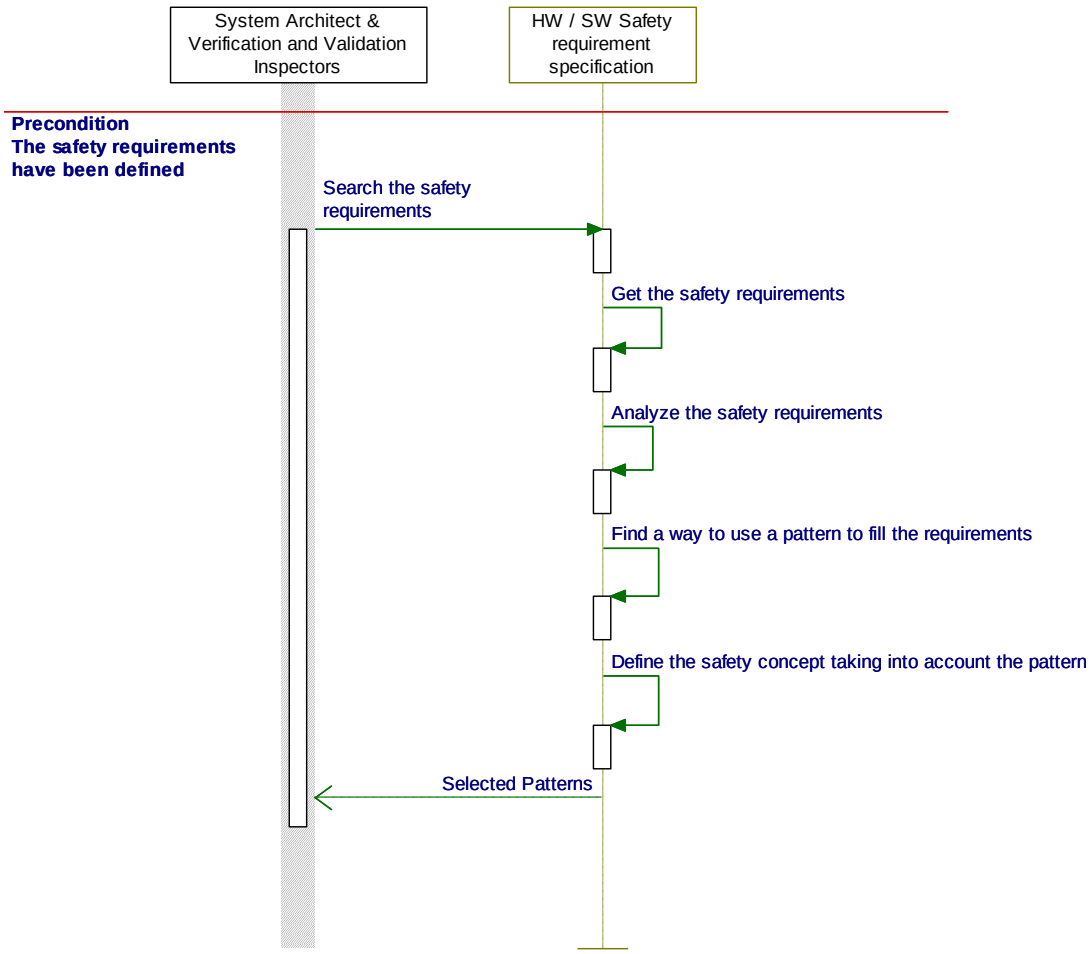

Figure 29. Sequence diagram to decide the need of a pattern.

Decide the need for a pattern Use Case template:

Use Case Label:	*DP0.1*
Date: *10/02/2010*	***Author:*** *David González & Manuel Blanco*
Use Case Name:	*Decide the need for a pattern*
Actors:	*System Architect* *Security/safety Manager*
Triggering Event (optional):	*None*
Relations with other use cases ***(not always applicable)***	***Extended by:*** *None* ***Includes:*** *None* ***Threatened by:*** *None* ***Generalization:*** *None* ***Mitigates or prevents:*** *None* ***Detects:*** *None*
Description:	*Analyze if some of the patterns in the repository could be useful in the design*
Preconditions:	*The safety requirements have been defined*
Post-conditions:	*The patterns that would be part of the system are identified and are part of the safety concept.*
Normal scenario:	• *Look for the safety requirements* • *Get the safety requirements*

	• *Find a way to use a pattern to fill the requirements* • *Define the safety concept taking into account the pattern*
Alternative scenario:	
Open issues:	

Table 24. Decide the need for a pattern template.

Analysis of the design alternatives Use Case template:

Use Case Label:	*DP0.2*
Date: *10/02/2010*	***Author:*** *David González & Manuel Blanco*
Use Case Name:	*Analysis of design alternatives*
Actors:	*Hardware Architect* *Software Achitect*
Triggering Event (optional):	*None*
Relations with other use cases ***(not always applicable)***	***Extended by:*** *None* ***Includes:*** *Search the pattern* ***Threatened by:*** *None* ***Generalization:*** *None* ***Mitigates or prevents:*** *None* ***Detects:*** *None*
Description:	*Analyze how the pattern selected has to be implemented.*
Preconditions:	*It was defined that the pattern is needed*
Post-conditions:	*It is defined if the pattern selected is implemented in SW or HW or in a combination of both strategies.*
Normal scenario:	• *Look for the pattern in the repository* • *Get the pattern from the repository* • *Analyze the pattern* • *Analyze alternatives of integration* • *Integrate the pattern in the architecture*
Alternative scenario:	
Open issues:	

Table 25. Analysis of design alternatives template.

Adapt the pattern Use Case template:

Use Case Label:	*DP0.3*
Date: *10/02/2010*	***Author:*** *David González & Manuel Blanco*
Use Case Name:	*Adapt the pattern*
Actors:	*Hardware Developer* *Software Developer*
Triggering Event (optional):	*None*
Relations with other use cases ***(not always applicable)***	***Extended by:*** *None* ***Includes:*** *Search the pattern* ***Threatened by:*** *None****Generalization:*** *None* ***Mitigates or prevents:*** *None* ***Detects:*** *None*

Description:	*Analyze the pattern and the design to detect which part of the pattern has to be modified for a easy integration in the design.*
Preconditions:	*The strategy to be used in the implementation of the pattern has been identified*
Post-conditions:	*The actions that has to be done in the adaptation of the pattern have been identified*
Normal scenario:	• *Look for the pattern in the repository* • *Get the pattern from the repository* • *Analyze the pattern* • *Adapt the pattern to the design* • *Integrate the pattern in the design*
Alternative scenario:	
Open issues:	

Table 26. Adapt the pattern template.

Validate pattern integration Use Case Template:

Use Case Label:	*DP0.4*
Date: *10/02/2010*	***Author:*** *David González & Manuel Blanco*
Use Case Name:	*Validate the pattern integration*
Actors:	*Verification Inspector* *Validation Inspector*
Triggering Event (optional):	*None*
Relations with other use cases ***(not always applicable)***	***Extended by:*** *None* ***Includes:*** *Search the pattern* ***Threatened by:*** *None****Generalization:*** *None* ***Mitigates or prevents:*** *None* ***Detects:*** *None*
Description:	*Verify that the pattern has been implemented in the design in a correct way.*
Preconditions:	*The pattern has been integrated in the design and implemented*
Post-conditions:	*The actions that has to be done in the adaptation of the pattern have been identified*
Normal scenario:	• *Look for the pattern in the repository* • *Get the pattern from the repository* • *Analyze the pattern* • *Compare the functionality of the implementation of the pattern and pattern design*
Alternative scenario:	
Open issues:	

Table 27. Validate the pattern integration.

3.3.2 Automotive System

In the automotive domain, different roles are concerned with the pattern integration (see Figure N). A pattern may include requirements to the whole process integrating it. For example, the secure communication pattern requires keys to be stored in a certain phase of the production process. Hence, a concrete and especially structured description of the whole production process from the very beginning to the ready-made product is needed. This pattern

description shall reference process steps that have to be adapted. So using a simple pattern may influence the whole process.

In general, structural patterns (e.g., secure communication) and procedural patterns (describing a process) are to be made available for the engineers to achieve efficient pattern integration.

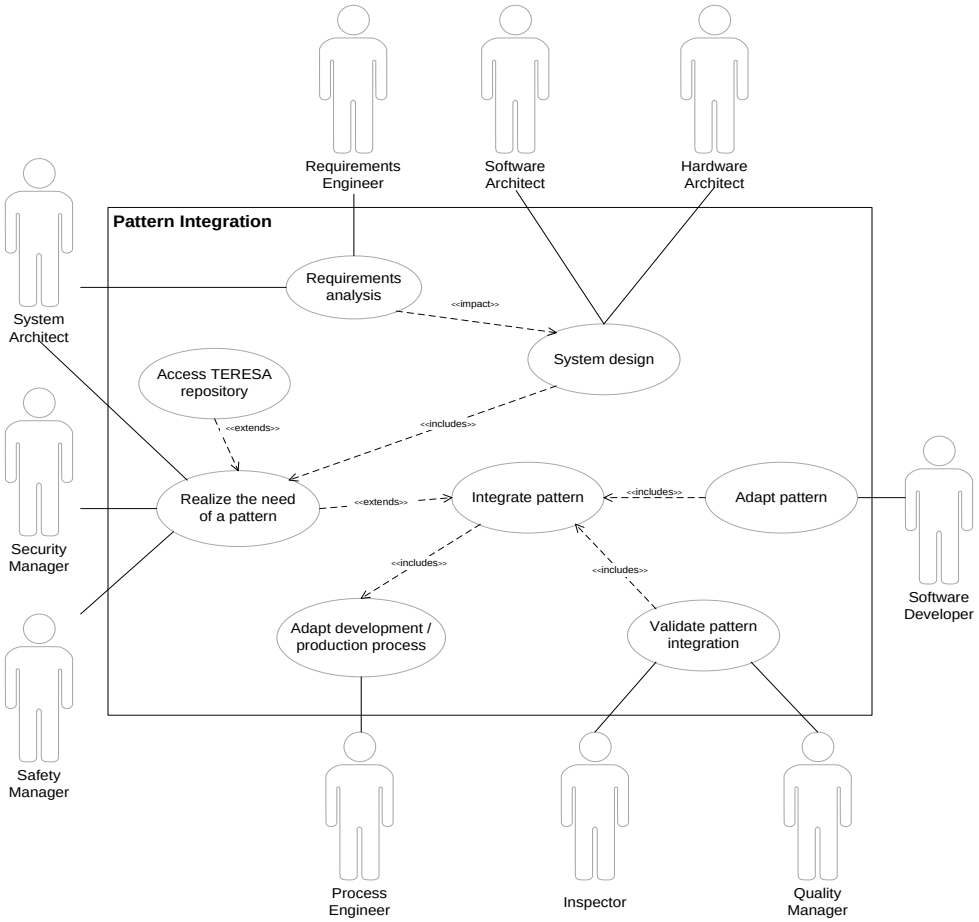

Figure 30. Automotive pattern integration Use Case

3.3.3 Home Control System

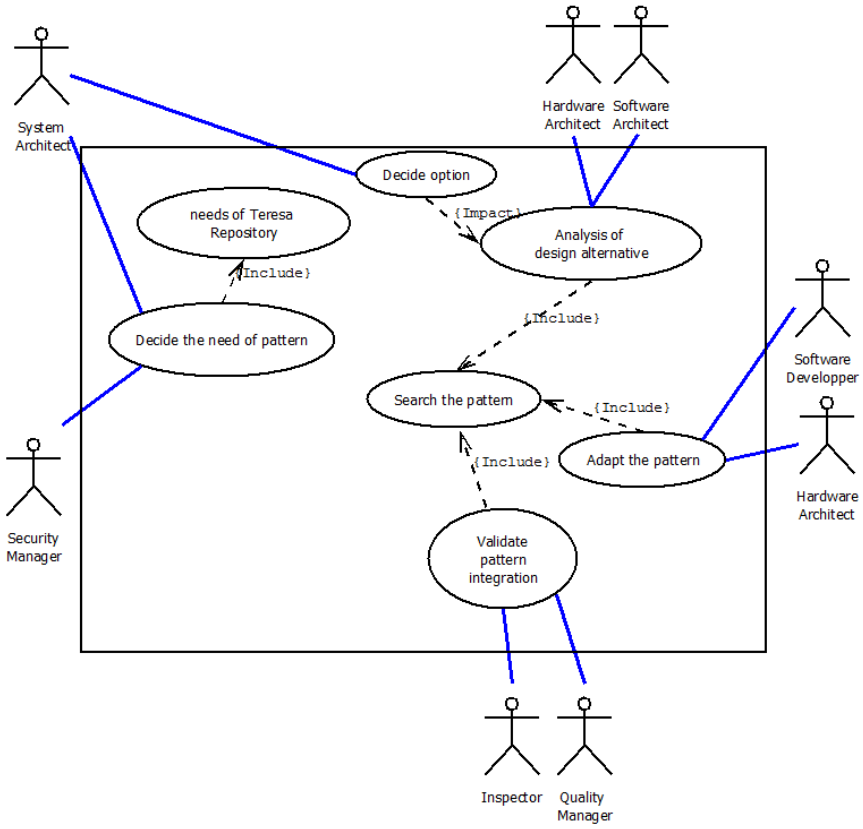

Figure 31. Home Control pattern integration Use Case

3.3.4 Metering System

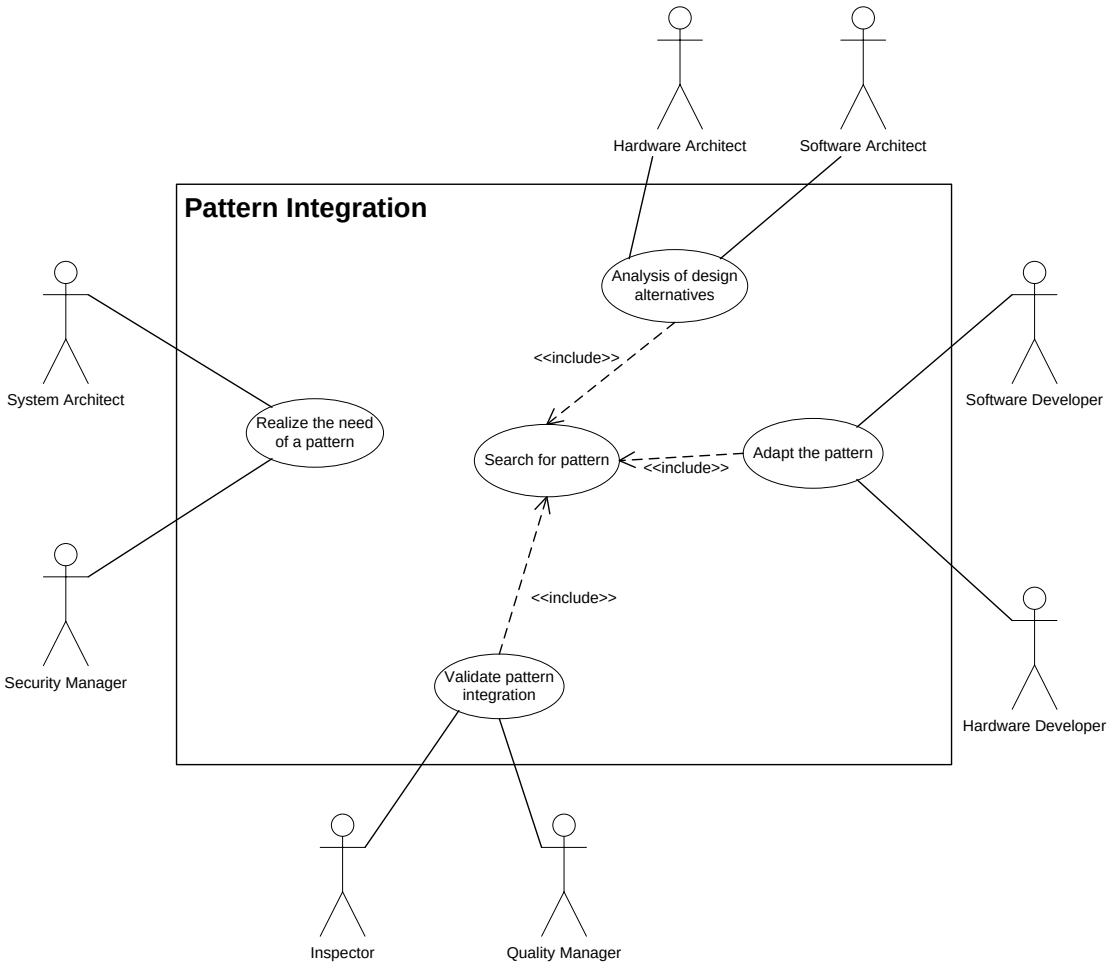

Figure 32. Metering pattern integration Use Case

3.4 List of identified patterns

To summarize the collection of design patterns the table below has been defined. In this table each partner has to add the patterns related to their sectors. When all the patterns have been identified and defined, each partner has to decide which of these patterns can be used in their application sector.

Application Sector / Patterns	Industrial	Automotive	Home Control	Metering	Type of focus
Majority Voter	***X***				*Dependability*
Communication Ring	***X***				*Dependability*
Black Channel	***X***				*Dependability*
HMAC	***X***				*Security*
Secure Communication		***X***			*Security*
Remote Attestation		***X***			*Security*
Secure Service Discovery			***X***		*Security*
Secure Software Download				***X***	*Security*
Secure Remote Readout				***X***	*Security*

4 Table 28. Summary of collected patterns.

Measures of Success of Project Objectives

4.1 General Measures of Success

Teresa investigates various criteria, and a major point is genericity since these criteria will be applied to four specific domains: automotive, home control, industry control, and metering. It is necessary to define some points that will highlight the success of the application of general criteria to specific cases. To do this, two main objectives have been set in for the TERESA project.

The first objective is to provide guidelines for the specification of sector specific RCES trusted computing engineering. These guidelines will help software process engineers to define a trusted computing engineering process that is tailored to the usual software development process in the RCES sector.

The second objective is to define a trusted computing engineering approach that is suited to the four sectors of TERESA but also that could be extended to other sectors.

It is necessary to define criteria to measure the success of these two objectives. This section defines these criteria.

4.1.1 Objective from Generic to Domain Specific

The first objective concerns guidelines that are oriented from generic to domain specific. The criteria used to measure the success of this objective are:

- The level of generality of guidelines
  - the same guidelines can be used successfully to instantiate the four TERESA sector specific engineering processes
  - the same guidelines can be used to instantiate other processes
- The level of extendibility to which extent the guidelines can be enhanced
- The level to which tools can support the integration
  - the consistency verification between different models.

4.1.2 Objective from Domain Specific to Generic

The second objective concerns an approach that is oriented from domain specific to generic. The criteria used to measure the success of this objective are:

- The level of reusability of patterns developed in the Teresa project for an application sector
  - developed patterns can be successfully integrated in an application sector
- The level of extendibility of new patterns to a specific application sector
  - patterns not developed in the project can be integratable in the process
- The level to which tools can support the application sector
  - the level to which tools can support the development process of a specific sector

4.2 Refined Measures of Success

In this section, more refined measures of success are described. Each one is linked to one or more specific deliverables that will fulfil the requirements described. Five measures are defined:

- **M1:** A TERESA process must be able to be hooked to other processes.
- **M2:** The TERESA process should enforce the separation of concerns between the different engineering roles.

- **M3:** The TERESA process should support the requirements for various engineering roles according to their assurance levels.
- **M4:** As a consequence of the two previous objectives, TERESA must provided tools and a repository that also fit the role oriented view.
- **M5:** Throughout the product lifecycle, a TERESA process must ensure both the consistency and traceability between each phase.

4.2.1 Compliance with Other Processes

M1*: A TERESA process must be able to be hooked to other processes.*

In domain specific development, some processes are used to provide interoperability. For instance, AUTOSAR compliance is becoming mandatory in automotive domain. Another example is the SIL-series that offers certification in the industrial domain.

The TERESA meta process definition must deal with the issue of hooking to other processes. This measure will be addressed in deliverables D3.2, D3.3, D3.4, and D3.5.

4.2.2 Support for Role-focused Features

4.2.2.1 Separation of Concerns and Engineering Roles

M2: *The TERESA process should enforce the separation of concerns between the different engineering roles.*

To do this, the process should allow the user to see information that is relevant for a specific role. In addition, all irrelevant information should be hidden from the user to help him access only the information that is relevant.

This measure will be addressed in deliverables D4.1 and D4.2.

M3: *The TERESA process should support the requirements for various engineering roles according to their assurance levels.*

For instance, a software engineer can request a low assurance level, whereas a security engineer can require high assurance for Security and Dependability.

Separation of concerns will be addressed in deliverables D4.1, D4.2, and different engineering roles in deliverables D5.1, D5.2 and D5.3.

4.2.2.2 Feature for Role-focused Activity

M4: *As a consequence of the two previous objectives, TERESA must provided tools and a repository that fit the role oriented view.*

Functionalities, options and views depend on who (i.e. the role of the engineer) uses a TERESA tool or browses the repository. For instance, in the repository, a security engineer would be allowed to use the function the “add a new S&D pattern” although a software engineer would not have access to it.

This measure will be addressed in deliverables D4.5, D4.6

4.2.3 Consistency and Traceability

M5: *Throughout the product lifecycle, a TERESA process must ensure both the consistency and traceability between each phase.*

The product lifecycle is described as Design – Development – Test – Deployment – Execution.

On one hand, consistency and traceability have to be preserved between the design and development phases. As TERESA will rely on design by model, consistency and traceability must also be maintained between the model-model and the model-implementation phases. A model can be a refinement of another one at a finer granularity.

On the other hand, the refinement of Platform Independent Models (PIM) as Platform Specific Models (PSM) is an objective for TERESA. Thus consistency and traceability must be maintained between models. For instance, if model 2 is a refinement of model 1 and model 1 is modified, then model 2 must be modified to keep consistency, or at least a warning must be sent to the user.

Moreover, traceability is mandatory in several domains, e.g. metrology.

This measure will be addressed in deliverables D4.1, D4.2, D4.3 and D4.4.

5 Annexes

5.1 Annex A: Actors definition

In the following paragraphs, there is a brief description of each actor in order to provide insight into the roles that each actor has to do. For more information about the tasks in which the actors are involved use the V-Modell XT like reference. The specifications of the actors have been made in alphabetic order.

Note: It is important to clarify that the following actors list is based on just a portion of the roles list described in the V-Model XT part 4 *[V-Modell XT, 2006]*. Because of this reason, some actors defined in the list below have been assigned with more tasks that belong to other actors in the original list.

5.1.1 Account Manager

The Account Manager is responsible for planning the order intake of his sales department and for maintaining the existing and developing new acquirer relations. He is also responsible for Request for Proposal with are of relevance for the submission of a proposal.

5.1.2 Assessor

The Assessor is an independent consultant. He assesses the documented and practiced processes of the process model of an organization.

5.1.3 Change Request Manager

The Change Request Manager is the person who is the responsible for dealing with any problems detected or change request made. He is the responsible to analyze the problem, develop the solutions of the problem, assess these solutions and give the recommendation to fix the problem.

5.1.4 CM Manager

The CM manager manages, coordinates and controls the Configuration Management and specifies all necessary project-specific conditions in the Project Manual. He could be also responsible for monitoring the quality in the project, and thus the quality of the results.

5.1.5 Executive

The Executive is responsible for the economically and technically successful planning, execution and completion of a project.

5.1.6 Hardware Architect

The role of Hardware Architect includes particularly the engineering and integration of hardware systems. He is responsible for hardware architecture, hardware specification, external hardware module specification and the hardware implementation, integration and evaluation concept.

5.1.7 Hardware Developer

The Hardware Developer comprises the realization of hardware elements. He cooperates in the hardware specification; moreover he collaborates in the development of hardware architecture, and in the development of the hardware implementation, integration and evaluation concept.

5.1.8 Verification Inspector

The Verification Inspector prepares the evaluation specifications, which he uses as basis for testing the project results. He records the evaluation results in an evaluation report. He

participates in other different roles such as system specification, the specification of external and internal units and others, in order to get a better idea to test the system.

5.1.9 Validation Inspector

The Validation Inspector prepares the evaluation specification, which he uses as a basis for validating the safety requirements of the project. He participates in other different roles such as system specification, the specification of external and internal units and others, in order to get a better idea to validate the system.

5.1.10 Project Leader

The Project Leader assumes the operational management of the project. He plans, coordinates, monitors and controls the project sequence, the project team and the project as a whole.

5.1.11 Quality Manager

The Quality Manager is responsible for preparing, maintaining, coordinating and distributing quality management regulations in the entire organization. He is in charge of implementing the quality policy and of all multi-project quality aspects in the system/software/hardware development.

5.1.12 Requirement Engineer “Acquirer / Supplier”

The Requirement Engineer is responsible for requirement definition and for the evaluation of the collection and preparation of user requirements. He shall ensure the quality of user requirements and create the prerequisites for ensuring traceability and changeability of the requirements throughout the entire life cycle. He is also responsible for the preparation of the overall system specification. In order to fulfill this task he must ensure the quality of the requirements and created the prerequisites for tracking the requirements during the entire life cycle.

5.1.13 Security/Safety Manager

The Security / Safety Manager is responsible for the compliance with and the implementation of Security aspects (preparation and updating of project-related security concepts, monitor the implementation of security concept, others). He is also is responsible for the compliance with and the implementation of the Safety requirements of a system to be developed or to be used.

5.1.14 Software Architect

The Software Architect is responsible for designing and developing all Software Units and products of the type External Software Module of a System.

5.1.15 Software Developer

The Software Developer is responsible for realizing the software elements (Software components, modules and units) in accordance with the Software Specification.

5.1.16 System Architect

The System Architect has the central role in system design and specification. Based on the Overall System Specification, he designs the System Architecture. Besides he defines the system elements based on the System Specification, External Unit Specification and the respective System Implementation, Integration and Evaluation Concept.

5.1.17 System Integrator

The System Integrator has the central role in the system realization phase. Based on the System Implementation, Integration and Evaluation Concept, he integrates system elements into Segments and into the System.

5.1.18 User

The User is in charge of use the system for fulfilling his tasks. Based on this analysis he derives requirements for the overall system and introduces appropriate changes proposals. He also collaborates in other task such as requirement specifications, identification of function to be realized, development of test and acceptance procedures and others.

6 Bibliography

[GHA+09]. Gyesik Lee, Hisashi Oguma, Akira Yoshioka, Rie Shigetomi, Akira Otsuka, and Hideki Imai. Formally Verifiable Features in Embedded Vehicular Security Systems, 2009.

[HChS04]. Spyros T. Halkidis, Alexander Chatzigeorgiou, and George Stephanides. A Qualitative Evaluation of Security Patterns, 2004.

[KFL09] Enno Kelling, Michael Friedewald, Timo Leimbach et al. E-safety vehicle intrusion protected applications: Specification and Evaluation of e-Security relevant use cases EVITA Deliverable 2.1, 2009. http://www.evita-project.org/Deliverables/EVITAD2.1.pdf.

[Kop97]. Hermann Kopetz. Design Principles for Distributed Embedded Applications. Kluwer Academic Publisher, 1997.

[KS03b]. Hermann Kopetz and Neeraj Suri. Compositional design of rt systems: A conceptual basis for specification of linking interfaces (presentation). In Proc. 6th IEEE International Symposium on Object-oriented Real-Time Distributed Computing (ISORC), Hokkaido, Japan, 2003.

[Kun+95]. The EHS: European Home Systems Network. Paris : Trialog, 1995.

[MSD08]. Antonio Maña, Daniel Serrano, and Athanasios-Dimitrios Sotirious. Towards Precise and Certified Security Patterns, 2008.

[PC06]. Juan M. Perez Cerrolaza. Safety-critical transportation embedded-systems (state of the art). Technical report, Ikerlan, 16th June 2006.

[THR+06] Martin Trngren, Dan Henriksson, Ola Redell, Christoph Kirsch, Jad El-Khoury, Daniel Simon, Yves Sorel, Hanzalek Zdenek, and Karl-Erik rzn. Co-design of control systems and their real-time implementation - a tool survey. Technical Report TRITA - MMK 2006:11, Royal Institute of Technology, KTH, 87 2006.

[V-Modell XT, 2006]. *DAS V MODELL*. Retrieved from http://v-modell.iabg.de/index.php?option=com_docman&task=cat_view&Itemid=&gid=15&orderby=dmdate_published&ascdesc=DESC

[Wol09]. Marko Wolf. Security Engineering for Vehicular IT Systems: Improving the Trustworthiness and Dependability of Automotive IT Applications, 2009.

[ZigbeeAlliance]. Zigbee Alliance site. *Zigbee Alliance site.* [On line] http://www.zigbee.org/.