SECRET

SECurity of Railways against Electromagnetic aTtacks

Grant Agreement number: 285136
Funding Scheme: Collaborative project
Start date of the contract: 01/08/2012
Project website address: http://www.secret-project.eu

Deliverable D 4.2

Final specification of the dynamic protection system

Deliverable on Final specifications of the dynamic protection system
Date: 21/08/2014
Distribution: All partners
Manager: Trialog

Document details:

Title	Final specification of the dynamic protection system
Work package	WP4
Date	21/08/2014
Author(s)	E Jacob (EHU), M Higuero (EHU), C Pinedo (EHU), C Gransart (IFSTTAR), M Heddebaut (IFSTTAR), A Kung (TRIALOG), M Sall (TRIALOG)
Responsible Partner	Trialog
Document Code	SEC- D4.2-C-082014-Final dynamic protection system.docx
Version	C
Status	Final

Dissemination level:

Project co-funded by the European Commission within the Seventh Framework Programme

PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission) Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Document history:

Revision	Date	Authors	Description
0.1	30/09/2013	EHU IFSTTAR TRIALOG	Creation of the document. Contribution from EHU on architecture and component specification
0.2	02/10/2013	IFSTTAR TRIALOG	Contribution from IFSTTAR and Trialog on architecture.
0.3	07/01/2014	EHU IFSTTAR TRIALOG	Refinement of specification: sections 3.2 and 5 (EHU), sections 4.1 and 4.2 (IFSTTAR and Trialog) Contribution on architecture evaluation (Trialog).
1.0	26/02/2014	TRIALOG	Executive summary added
1.1	09/06/2014 27/06/2014 29/06/2014	EHU IFSTTAR Trialog	Minor modification after internal review by Alsthom
1.2	07/07/2014	TRIALOG	Final version

Table of content

1 Executive summary 5

2 Introduction 6

2.1 Purpose of the document 6

2.2 Definitions and acronyms 6

3 Overview of the Protection System 8

3.1 Detection System 8

3.2 Multipath Communication System 10

4 Specification of the Detection System 14

4.1 Components of the detection system 14

4.1.1 Acquisition System Sensor 14

4.1.2 Acquisition System Analyser 14

4.1.3 Health/Attack Manager (HAM) 15

4.1.4 Central Health/Attack Manager (CHAM) 19

4.2 Interfaces 22

4.2.1 HAM-CHAM Interface 22

4.2.2 ASA-HAM Interface 22

4.2.3 MCS-HAM Interface 27

5 Specification of the Multipath Communication System 28

5.1 Components 28

5.1.1 Multipath Communication Manager (MCM) 28

5.2 Interfaces 36

5.2.1 Control Interface 36

5.2.2 Input interface 39

5.2.3 Output interface 40

Annex A Approach to Evaluate SECRET Resilient Architecture 44

A1 Introduction 44

A2 Methodology Used 45

A2.1 Architecture Decisions 45

A2.2 ATAM 47

A2.4 CBAM 49

A3 Evaluation using CBAM 51

A4 Attack Centred Assessment in Architecture Evaluation 56

A5 Applying ATAM to SECRET 56

A5.1 List of ASR Train Level 56

A5.2 List of ASR Track Level 58

A6 Towards SECRET Guidelines for Architecture Resiliency Evaluation 59

1 Executive summary

The objective of the dynamic protection system is to detect and dynamically cope with different EM attack conditions that may affect the communication among devices of the railway system.

This deliverable completes D4.1 (preliminary specification). It provides

- An overview of the protection system
- A component and interface specification of the detection system
- A component and interface specification of the multipath communication system

An annex on the evaluation of the architecture has been added. This annex will be used for the validation of the implementation through a use case (D4.5)

2 Introduction

2.1 Purpose of the document

The purpose of the Deliverable 4.2 is to present a final and complete view of the resilient architecture in order to face EM attacks. This deliverable will be the basis for the implementation of the resilient architecture during the remaining work of the Work Package 4.

The document contains 2 important sections which present the detailed architecture which is a refinement of the preliminary architecture described in D4.1. This constitute the last step before the implementation. The first one of these sections is related to the detection system while the second one is related to the multipath communication system.

The annexes contain an approach to the evaluation of a resilient architecture for SECRET. While the last annex gives some guidelines for this evaluation when facing to another architecture.

2.2 References

D4.1: Preliminary specification of the dynamic protection system

2.3 Definitions and acronyms

	Meaning
API	Application Programming Interface
AS	Acquisition System
ASR	Architecturally Significant Requirement
BSC	Base Station Controller
BTS	Base Transceiver Station
CHAM	Central Health/Attack Manager
DS	Detection System
DSS	Detection SubSystem
EDGE	Enhanced Data Rates for GSM Evolution
EM	Electromagnetic
ERTMS	European Railway Traffic Management System
ETCS	European Train Control System
ETML	European Traffic Management Layer
EVC	European Vital Computer
FMEA	Failure Mode and Effect Analysis
GSM	Global System for Mobile communications
GSM-R	Global System for Mobile communications - Railway

GPRS	General Packet Radio Service
HAM	Health/Attack Manager
HIP	Host Identity Protocol
HSPA	High-Speed Packet Access
ICT	Information and Communications Technology
IP	Internet Protocol
LEU	Lineside Electronic Unit
LTE	Long Term Evolution
MCS	Multipath Communication System
MIP	Mobile IP
MPTCP	Multipath TCP
MSC	Mobile Switching Centre
OHAM	On-board Health/Attack Manager
PDU	Packet Data Unit
RBC	Radio Block Centre
RIU	Radio In-fill Unit
RMS	Railway Management System
TETRA	TErrestrial Trunked RAdio
TCP	Transmission Control Protocol
THAM	Trackside Health/Attack Manager
TVRA	Threat and Vulnerability Risk Assessment
UMTS	Universal Mobile Telecommunications System
WiMAX	Worldwide Interoperability for Microwave Access
WP	Work Package

3 Overview of the Protection System

The protection system consist of two elements: the detection system and the multipath communication system

3.1 Detection System

The main objective of the Detection System (DS) is to continuously monitor the overall network in order to detect EM attacks that maybe performed on the network. When such situation happens, it sends information related to the attack allowing the system to overcome it. In this section, we described in details the architecture of this subsystem.

The general architecture of the detection sub-system is shown on the figure below.

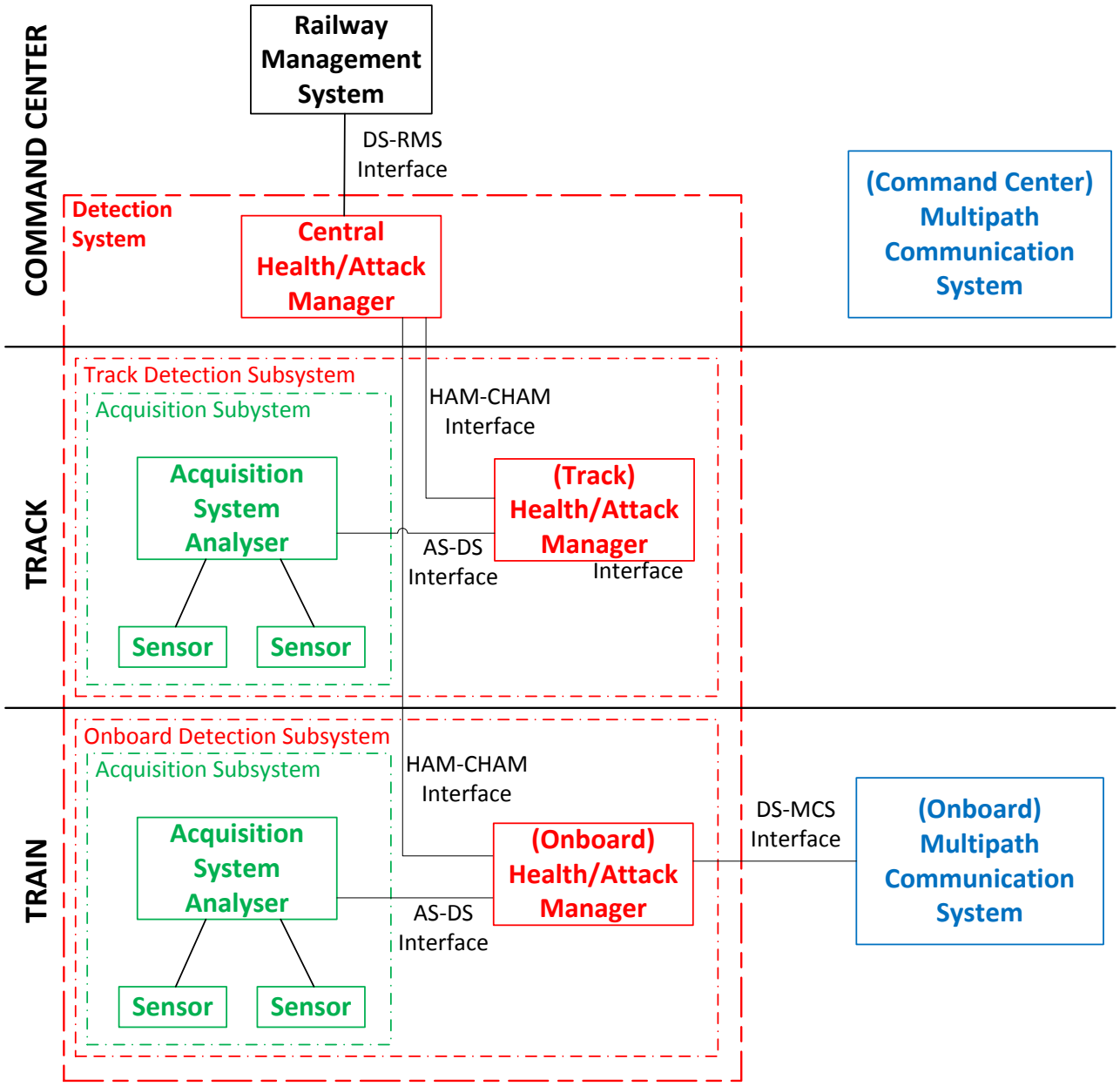

Figure 1: Architecture of the detection subsystem

The system is composed of several distributed detection subsystems located on the train and along the track and one Central Health/Attack Manager which manages the different detection subsystems and provides the management staff with the complete information about the EM status of the railway infrastructure that is monitored.

Each detection subsystem is composed of one Health/Attack Manager (HAM) with several sensors that monitor the EM status of a specific and well-defined geographical area. There are two types of detection subsystem:

- The Track Detection Subsystem, composed of a (Track) Health/Attack Manager (THAM), an Acquisition System Analyser and several sensors located on the track or on certain sections of the track.

- The Onboard Detection Subsystem, composed of a (Onboard) Health/Attack Manager (OHAM), an Acquisition System Analyser and several sensors located inside the train.

The sensors of a detection subsystem are continuously monitored and the information they provided is computed in order to determine if there are normal EM conditions or an attack is ongoing. All changes like new attack, end of attack, fail of one sensor are reported by the HAM to the CHAM.

At least one detection subsystem should be deployed in the railway infrastructure although depending on the number of sensors and their location, it could be interesting to deploy multiple detection subsystems.

The basic operations of the onboard detection system are very similar to those of the detection subsystems installed along the track. However, the OHAM has the following characteristic:

- The attack information is not only provided to CHAM but also to the Multipath Communication Subsystem (MCS) in order the MCS to be able to adjust the communication paths depending on the detected attacks. Furthermore, it would be also advisable to inform the driver of the train about any EM problem that could cause a loose of communication with trackside, thus one interconnection with the onboard ERTMS is also proposed.
- The CHAM may inform the OHAM about a still undetected attack that is taking place in a section of the track the train is going to cross. So, the MCS can be notified before the communication with trackside has been disrupted and the MCS can perform the convenient actions to guarantee communications even before suffering the EM attack.

3.2 Multipath Communication System

The objective of the Multipath Communication System (MCS) is to provide resilient communications between trains and the command centre located at ground. The communications to protect are considered to be based on the IP protocol stack and to be always initialized by trains towards ground. Firstly, although current ERTMS is not based on IP, as it was pointed out in the Deliverable 4.1 the current trend is to evolve ERTMS towards IP in order to be able to replace GSM-R with other more advanced wireless technologies like GPRS or LTE. Thus, this system could be a valid proposal so as to strengthen the communications of the next generation ERTMS that are being designed currently. Secondly, current ERTMS communications are initiated by the train towards the RBC and rarely from RBC to the train. This is due to the difficulty to know the active trains at every moment and to maintain mappings of trains and their telephone numbers. With IP protocol the problem is similar or even worst because depending on the wireless technology used trains could change their IP address during their trip and so the RBC requires an updated mapping of trains and IP addresses to be able to initiate the communication with the train. Although there are multiple solutions to maintain an updated mapping between trains and IP addresses (Dynamic DNS, Rendezvous servers, …), we consider an excessive complexity, which is not required and is out of the interest of the Multipath Communication System, and consequently we initially propose to limit the initialization of communications from trains.

The MCS, like the Detection System, has been considered as an autonomous system that might be deployed in a railway infrastructure autonomously. However, the deployment of the MCS with the Detection System offers a synergistic solution to face EM attacks. On the

one hand, the Detection System could dynamically manage the resilient communication provided by the MCS based on the status of the EM environment and thus the MCS could provide a more robust communication against EM attacks due to quicker response time. On the other hand, the communications of the Detection System might also benefit from the additional reliance provided by the MCS.

For the point of view of one overall resilient system, the Detection System would be the engine of the resilient communication architecture whereas the MCS is one possible method Detection System could use to overcome EM attacks. The MCS is not the unique reactive system may be designed or deployed against EM attacks. For example, another approach could be a system connected to the BTSs to increase the transmission power in case of one EM attack and this system would be also managed by the Detection System to increase the transmission power only in case of necessity.

In the Figure 2 the main actors that may interact with the MCS are displayed: the ERTMS and the Detection System.

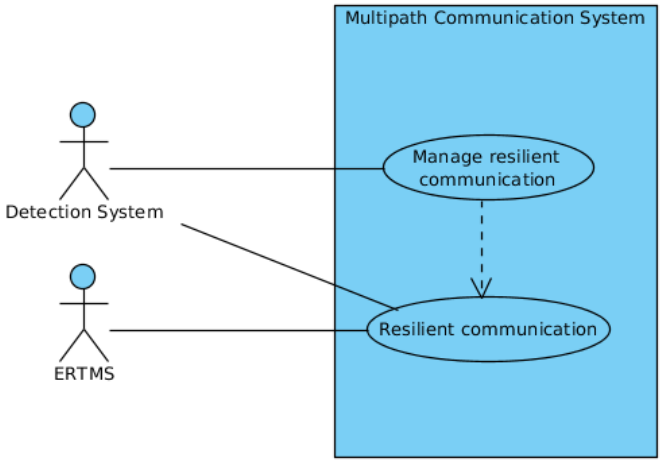

Figure 2: Use case of the Multipath Communication System

Obviously, the main objective of the MCS is to provide more robust communications and the ERTMS will use the MCS to strengthen the communication between the ERTMS subsystem located in the trains and the ground ERTMS subsystem. However, not only could the ERTMS use the MCS, but also any other distributed system that requires high available communications between trains and ground, for example, a remote telemetry system or a remote video surveillance system. In fact, the Detection System is supposed to also use the MCS to strengthen communications between the onboard HAM and the CHAM located in ground.

Apart from that, the MCS can be governed by the Detection System and so the resilient communication provided by the MCS may be customized according to the EM attacks detected by the Detection System.

The architecture of the MCS consists of several Multipath Communication Managers (MCMs), as it is shown in the Figure 3 Each MCM provides resilient communications for inner devices that require establishing a remote IP communication with other devices which will be similarly accessible behind other MCMs. Thus, the MCMs can protect the path among them thank to the use of a multipath communication approach. The MCMs are required to be installed on trains and on trackside in front of the services to protect.

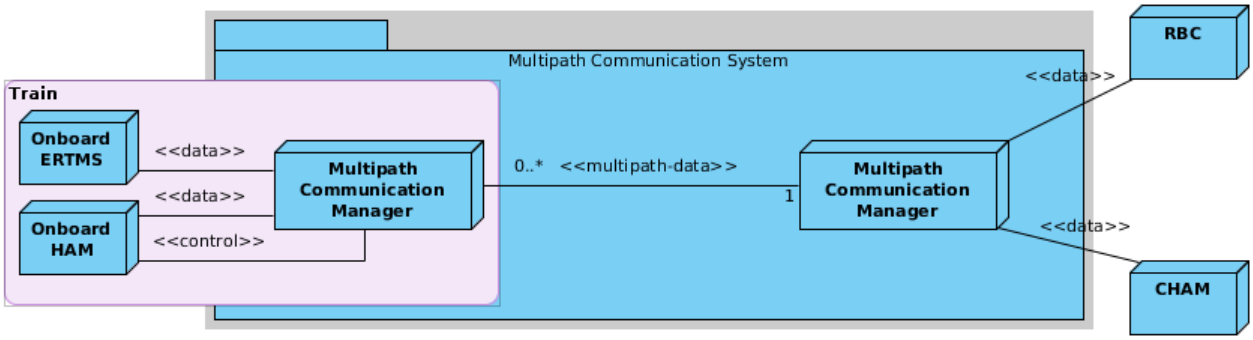

Figure 3: Architecture of the Multipath Communication System

One of the key design points was the use of proxies MCM (Multipath Communication Manager) instead of implementing the multipath functionality directly in the end-devices such as the onboard-ERMTS equipment and the Radio Block Centre (RBC). The proxy approach is considered to offer several advantages.

To begin with, the proxy approach can provide a reliable communication service to multiple legacy IP devices. In addition, gathering all the wireless interfaces in the MCM reduces the number of interfaces required per train. For example, using only three wireless interfaces in the MCM for providing service to all critical services of the train instead of requiring three wireless interfaces per critical service: three interfaces for the multipath-Onboard-ERTMS, three interfaces for the multipath-Onboard-HAM and so on.

Another point to consider is the control signalling between the multipath functionality and the Detection System. This implies not only a customized multipath technology but also a control interface to allow the dynamic modification of the behaviour of the multipath technology by an external device, which should be implemented in every multipath end-device and even in embedded systems.

Thus, the proxy approach is architecturally cleaner and the proxy approach based on the use of MCM devices was considered.

MCM can be located in trains or in trackside. The MCM located in the train offers redundant communications to the onboard ERTMS device and other possible sensitive systems such as the Health/Attack Manager (HAM) of the detection system via a wired interface. The MCM of the train is required to have and manage multiple wireless communication interfaces with ground. Although, the MCM works with only one interface, the advantage of its use is obtained from two or more interfaces. This communication interfaces must be based on IP technologies like GPRS, WiMAX, WiFi, LTE and so on. The number and type of interfaces will depend only on the desired objective. In order to efficiently face EM attacks it should be advisable to use different and disjointed wireless technologies that use different frequency bands. However, the technology used is trivial for the MCM as long as it is an IP-capable technology. In fact, it would be possible to deploy exclusively multiple interfaces of the same technology with antennas located in different points of the train. This approach won't be the best solution against EM attacks but it could provide a more robust approach against hardware failures apart from providing a little more resiliency against EM attacks.

Apart from the MCM located in the trains, the ground devices or services that are accessed by these resilient trains are required to have their own MCM in front of them, because the multipath communication can only be established between two MCMs This required ground MCM will have wired interfaces and so it is not strictly required to have multiple wired interfaces because the increased number of communication paths provided

by these wired interfaces are not going to improve the resiliency against EM attacks, which are focused on the radio environment. However, it may be desirable to have multiple wired interfaces to improve the resiliency against hardware failures or networking problems.

Finally, it is worth pointing out that the communication between a protected device with one MCM and a legacy IP device is possible. This communication will be a legacy IP communication, in order words, it will consist of only one path. This is really interesting for allowing the coexistence of one path and multipath communications and thus, for example, in order to allow coexistence scenarios.

4 Specification of the Detection System

4.1 Components of the detection system

In this section, the different elements that compose the detection system are described in more details.

4.1.1 Acquisition System Sensor

The sensors are an output of WP3. They provide information like the bit error rate or the number of retransmissions to be processed by the Acquisition System Analyser which must detect the presence of EM attacks.

4.1.2 Acquisition System Analyser

As said in D4.1, the Acquisition System Analyser (ASA) is mainly responsible for processing the information provided by sensors to detect the presence of EM attacks. It must also control sensors and establish the communication with the Health/Attack Manager (HAM) to inform about changes detected in the state of the EM environment detected.

In this subsection, the components that composed the ASA are presented on the Figure 4. Each physical sensor (like Wi-Fi equipment, GSM modem) has a logical representation named generically Logical_Sensor. An interface is defined between Acquisition_Sink and Health_Attack_Manager (which is not in the Acquisition Subsystem) to transmit data generated by the sensors.

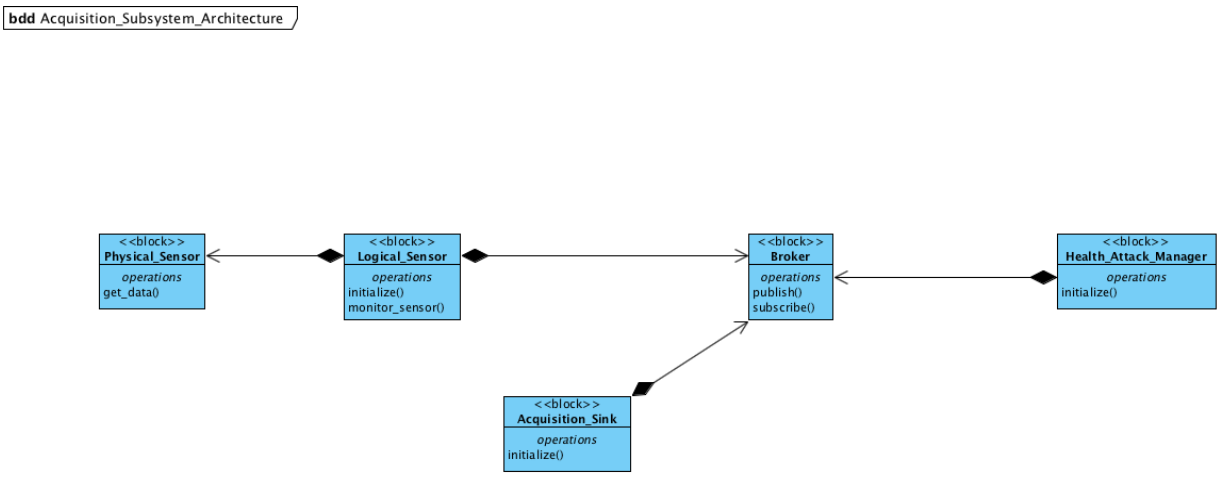

Figure 4: Architecture of the Acquisition System

The role of this logical sensor is to realize a physical connection with the sensor in order to get data from it. In other words it is a computer representation of a physical device. This interface is sensor dependant. Next, the Logical_Sensor sends data to the Acquisition_Sink using a Broker. This Broker manages asynchronous communications. The Acquisition_Sink also communicates with the Health_Attack_Manager thanks to the Broker. The Acquisition_Sink centralizes at its level all the data generated by the different Physical_Sensors.

The advantage of this architecture is that it is easy to add a new sensor into the subsystem. Moreover, the sensors can be installed on various equipements that are distributed.

The Figure 5 presents the context of the subsystem including various kind of sensors. In SysML modelling, the Domain block is used as the root block to model the context of the Acquisition Subsystem. During the implementation step, this block will disappear.

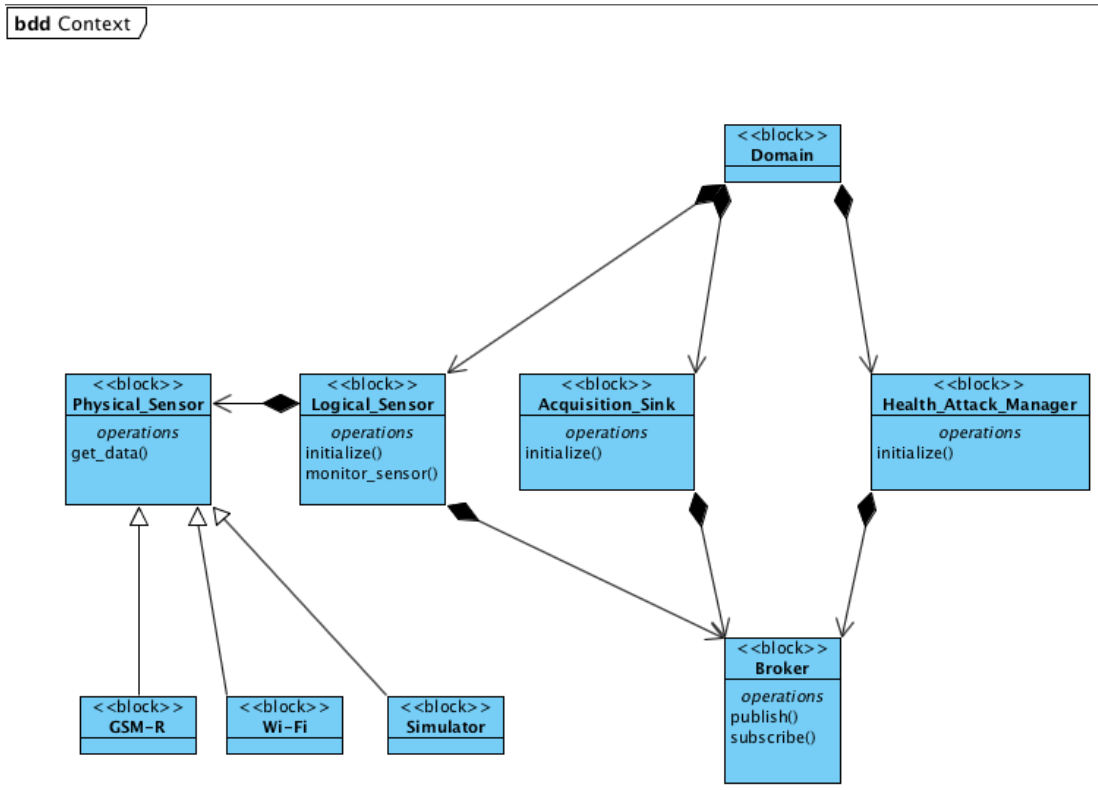

Figure 5: Context of the acquisition subsystem

4.1.3 Health/Attack Manager (HAM)

The detection capabilities depend on all the elements that set up the DS and that allow the detection of EM attacks. It consists of sensors deployed along the railway infrastructure and inside trains, and HAMs that manage these sensors and process the information provided by them.

A detection subsystem provides information about its capabilities of detecting EM attacks and informs about any change that affect its detecting capabilities.

The HAM has been designed in order to meet all the functional requirements described in D4.1. There are two kinds of distributed HAMs: On board Health/Attack Managers (OHAMs) that are located inside a train and controls the sensors deployed also inside the train, and the Track Health/Attack Managers (THAMs) that manage sensors deployed statically near the track. It is worth mentioning that both architectures are the same, in other words, both architecture have exactly the same modules. However, some modules of the OHAM involve more complexity than the equivalent ones of the THAM, because the OHAM may be required to interact with the MCS and because the location management is more complex in a moving HAM. These differences will be detailed in the description of the modules. The figure below shows the modular architecture of the OHAM.

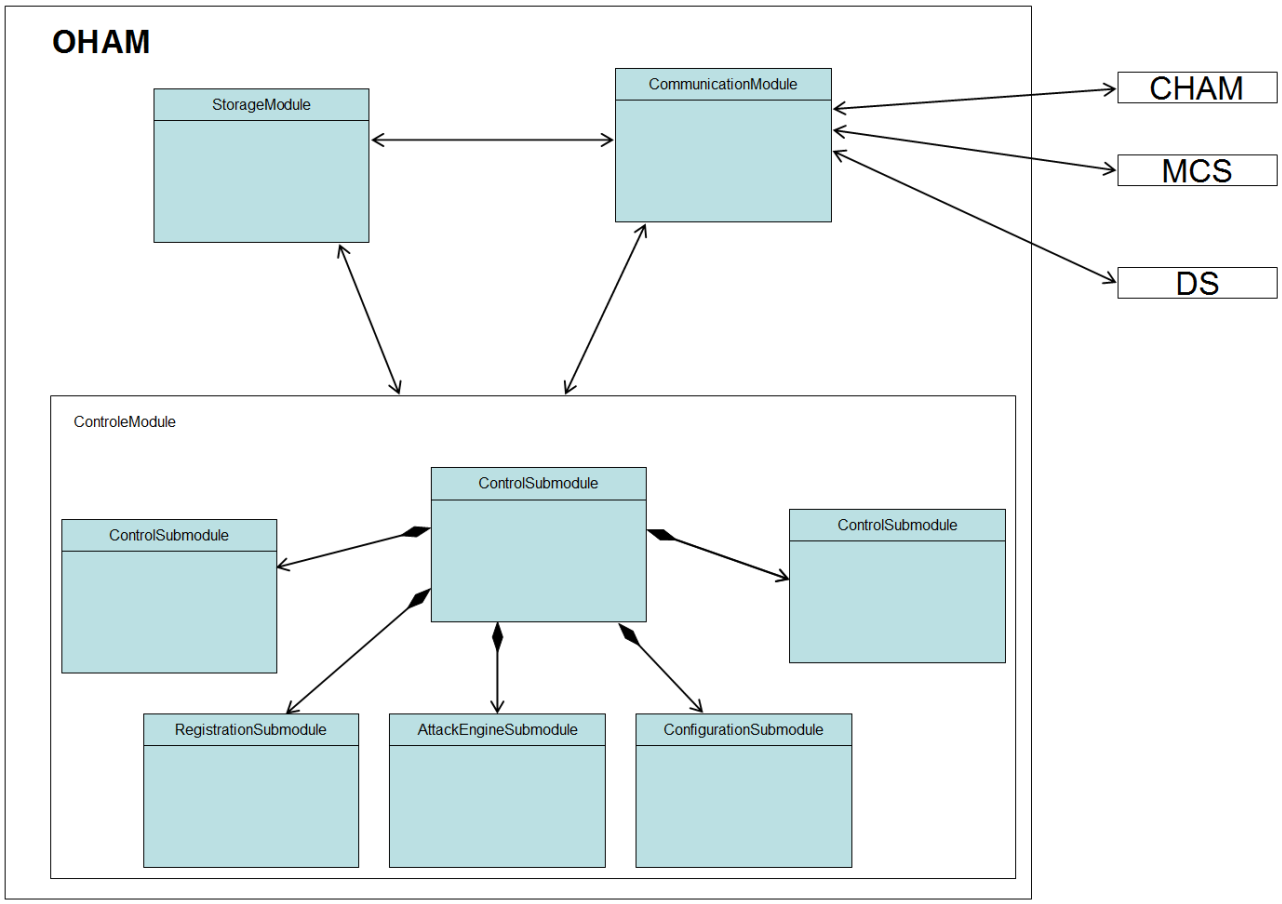

Figure 6: Architecture of the Onboard Health/Attack Manager (OHAM)

In general, the HAM has fewer modules than the CHAM. It doesn't need to have a display and the storage needs are less complex than those of the CHAM. On the other side, the control module of the HAM needs new modules to face new functionalities in the DS:

- Manage the position of the HAM and sensors (location submodule)
- Manage sensors (sensors submodules)

The storage module is still needed here for storing the credentials, the configuration of the HAS, and the reports that are waiting to be sent to the CHAM.

The Communication Module offers functionalities very similar to those present in the CHAM (see 4.1.4). But in addition, it is able to establish a second connection with the onboard MCS (i.e. located inside the train), besides the connection with the CHAM. Therefore this communication module is responsible for managing communications with both the CHAM and the MCS. It provides control module with some common basic services such as network technology abstraction and security in the communications.

The communication module exchanges packets between the control module of the HAM and the remote CHAM or if available, the train MCS in a secure way. For that, it first authenticates the CHAM and MCS and then supplies integrity and optionally privacy communications with them.

The communication module implements an advertising protocol to allow the HAM to discover the CHAM and MCS without requiring a previous manual configuration. This feature is especially important for Onboard Health/Attack Subsystem due to its dynamic nature and mobility. A DS in a train may be powered on and off several times a day and because of its mobility can participate in different Detection Subsystems managed by different CHAMs.

The communication module has one internal interface with the control module and another with the storage module.

The control module of the HAM is quite similar to the control module of the CHAM. However, its main function is to govern the logic of the DS based on the indications provided by the CHAM. The HAM interprets the output of sensors and, using the configuration specified by the CHAM, it is able to determine if an EM attack is being performed, in which case the HAM will notify it to the CHAM and the MCS.

The module is composed of 6 submodules, four of them are quite similar to the submodules developed in the control module of the CHAM:

- Control submodule
- Registration Submodule
- Configuration Submodule
- Attack Engine Submodule
- Location Submodule
- Sensors Submodule

The control submodule performs basically the orchestration between these submodules. It launches new actions in a submodule when an output from another submodule has been obtained. When new packets are received through the communication module it verifies that the packets are received from the CHAM and MCS in the correct state and if it is the case, it forwards them towards the adequate submodule.

It would be preferable that this verification was done by the communication module.

The registration submodule establishes the registration of the DS and keeps this registration active while the HAM remains powered and the current configuration allows it. This registration is performed immediately after the communication submodule has found the CHAM or the MCS. With this registration process, CHAM and MCS are aware of the availability of the HAM and register the address in order to be able to contact it. The registration submodule updates this registration as soon as a change in the contact address is changed (perhaps due to a roaming process) or periodically as a “keep-alive” method to inform that the HAM is still available at the same contact address. The periodicity of the updates may be dynamic depending on the probabilities of losing connectivity with the CHAM.

This configuration submodule is responsible for the configuration of the HAM. It sends capability changes to the CHAM and attends capability requests from it. Furthermore, it manages configuration requests from the CHAM and applies the desired configuration in the HAM.

When the CHAM requests the current capabilities of the DS, the HAM replies with a capability report updated. Each time there is a change in the capabilities of the DS, the HAM sends a capability report to the CHAM for informing about the changes that affect its detection capabilities.

When the CHAM requests the current configuration of the DS, the HAM replies with a configuration report updated. When the CHAM requests a change in the current configuration of the DS, the HAM applies it immediately and replies with an acknowledge.

The attack engine submodule of the HAM performs quite differently from the attack engine submodule of the CHAM. This submodule processes all the information received from the sensors (through the sensors submodule) and based on this information decides if there is any EM attack or if normal conditions are met.

The EM attacks detection depends on the configuration of the algorithms that are deployed in the HAM. This is performed remotely by the CHAM, allowing the CHAM to remotely adjust the detection capabilities of the DS.

This submodule constantly analyses the information provided by the sensors submodule and establishes if an EM attack is being performed in the DS according to the current detection configuration. Once an EM attack has been detected, the submodule reports the EM attack to the registered endpoints (CHAM and MCS) unless the configuration of the HAM avoids it. Attack reports are continuously sent about the detected EM attack to detail the time and spatial variations of the EM attack until the attack finishes. If the communication is lost, the attack will be reported as soon as the communication is re-established.

The submodule is active only once the HAM has successfully registered in one CHAM or MCS. When, the CHAM remains unregistered the attack engine submodule remains inactive because no CHAM or MCS is available to report the attacks to. Although this could be customized by the configuration of the HAM, allowing processing the attacks during some time when there is no connection to CHAM or MCS in order to report the attacks once re-registration takes part.

The location submodule is responsible for managing the location of the HAS so that the CHAM can know the current position of the HAS and where the detected EM attacks are being performed.

This module is only required to be active once the HAM has registered in the CHAM or in the MCS unless the current configuration of the CHAM determines something different. When, the CHAM remains unregistered the location submodule remains inactive because no CHAM or MCS is available to use the information. However, as it may be required to process the attacks even when there is no registration, the location management can also be performed even if there is no registration when the configuration of HAM specifies it.

The management of the location of the DS may be really simple or complex depending on the case. The location of all the sensors can be the same or may be different depending on the area size covered by the DS. For example, the location of all the sensors of a single DS inside a train may be considered to be the same whereas the location of the sensors of a DS deployed in the track along a lineal area of 20km are be necessary different. Another aspect to consider is the mobility of the location. For example, the location of sensors of a DS located inside a train are moving and thus the location submodule should be able to determine the current location of the train for any method (odometry, GPS, …). Whereas, in the case of sensors deployed statically along the track the location of the sensors might be easily statically set in the location submodule because it is not going to change. Thus, the management of the location by this module can be more or less complex depending on the specific case of the DS.

The location management is carried out by this submodule, but as this information is considered as a capability of the DS, it is supplied to the configuration submodule that will put it in the capability reports that are sent to the CHAM when there are changes in the location.

The sensors submodule is connected to the Acquisition System through the communication module. It receives the status of all of them and obtains the measurements that needed to determine the presence or not of an EM attack. Changes in the status of any sensor is immediately reported to the configuration submodule so that the configuration submodule can notify the CHAM about the capability change in the DS.

The storage module offers an abstraction from the storage method used to save and retrieve persistent data. This module is used by all the modules of the HAM and provides them with a method for exchanging information.

4.1.4 Central Health/Attack Manager (CHAM)

The main objective of the CHAM is to provide to the management staff information in realtime about ongoing EM attacks. The information provided contains the location of EM attack, the start time of the attack and severity of the attack. This information is obtained from the different HAM through secured communication system. It can be clearly displayed to the management staff so that the required reactive actions could be performed in order to re-establish the normal operation of the railway infrastructure.

The CHAM has been designed in order to meet all the functional requirements described in D4.1. The next figure shows a detailed view of it and how it is connected to the other components of the architecture.

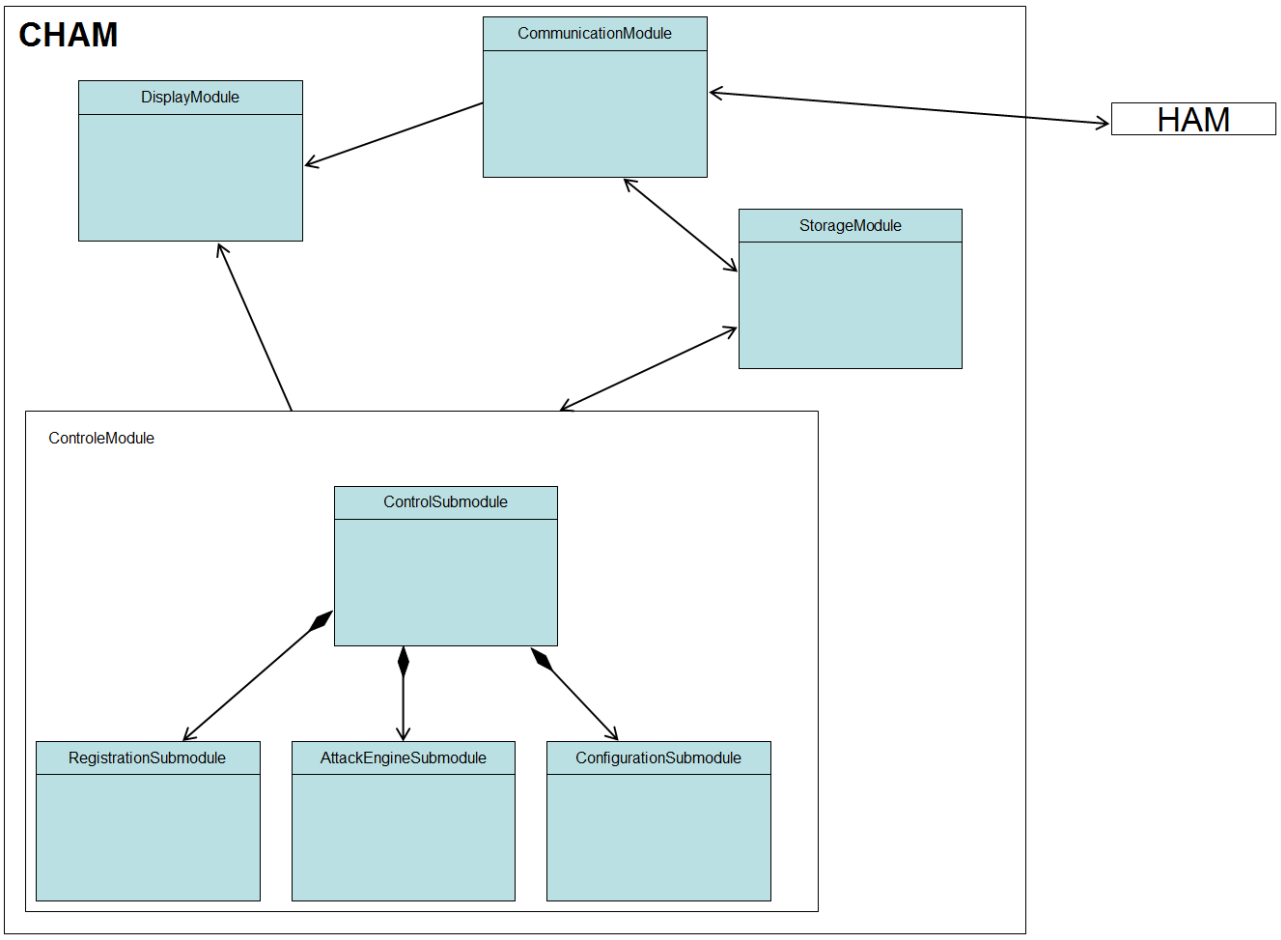

Figure 7: Architecture of the Central Health/Attack Manager (CHAM)

The communication module is responsible for the management of the communications with the distributed HAS. It offers to the control module some common basic services such as network technology abstraction and secure communications. It supports all the necessary technologies to establish communications with the HAM that may use different technologies. Thus, this module caches the complexity of the underlying technology to the upper modules.

The control module is one of the most important component in the DS because it supplies most of the logic of the DS. It configures remote HAS and attends the requests received from them according to the preferences set by the management staff. In this document, only the 4 main submodules are described:

- control submodule
- registration submodule
- configuration submodule
- attack engine submodule

The control submodule performs the orchestration between the remaining submodules, allowing launching new actions in a submodule when an output from another submodule has been obtained. Some examples:

- when a HAS is appropriately registered via the registration submodule the control submodule will require the configuration submodule to launch a capability request and later a configuration request to get the current capabilities and configuration of the HAS.
- when a HAS is deregistered, the control submodule will notify the rest of the submodules to discard the state of the already unregistered HAS.

When a new packet is received through the Communication Module from a HAS, the Control Module verifies that it can be taken into account according to the registration state of this HAS:

- A packet whose final destination is the registration submodule is always accepted.
- A packet whose final destination is the time service submodule is only accepted if it comes from a previously registered HAS
- A packet whose final destination is the configuration submodule is only accepted if it comes from a previously registered HAS.
- A packet whose final destination is the attack engine submodule is only accepted if it comes from a previously registered HAS with current capabilities and configuration known.

The registration submodule supplies registration services to all the HAMs of the DS whose packets have been previously authenticated and authorized by the Communications Module. Each HAS must register to the CHAM so that the latter is informed about its existence and availability. During this registration process all the required contact information (e.g. status and address) is provided and stored allowing the CHAM to establish a communication with the new-registered HAS. Afterwards, the registration submodule is noticed of every change concerning the HAM.

The configuration submodule processes the capability and configuration reports from HAMs already registered. These reports are passed to the configuration submodule once they have been verified by the Communication Module. Reports from unregistered HAM are ignored by the Communication Module. The capability reports may be explicitly requested by the CHAM. They are sent systematically by the HAM when there is a change of its capabilities. The configuration reports are sent by a HAM only upon request by the CHAM. Finally, these reports are maintained and stored in the storage submodule. Through the configuration submodule, the CHAM has the possibility to change the configuration of a HAM.

The attack engine submodule receives all the EM attack reports from the currently active HAMs. As it is for all other information, attack report packets are received by the

Communication Module which has verified the identity of the HAS and the integrity of the packets. The submodule verifies that the current configuration, stored in the storage module, allows the notification of EM attack reports. If not, the EM attack report is discarded. If the EM attack report is received from a valid HAM, it is immediately stored in the storage module.

Generally, the attack engine submodule receives reports about ongoing EM attacks from a registered HAS. However, the attack engine is also able to receive information about a past attack that could not have been reported at the time it was detected perhaps because of a lack of communication between the CHAM and the HAS for whatever reason. The submodule studies and is able to detect the relationship between attack reports from different HASs. So, for example, the submodule may consider that all attack reports received from different HASs are related to the same attack. These attacks and the relations to EM attack reports are stored in the database.

The attack engine submodule may decide to inform other HASs about an ongoing EM attacks that are being performed near their location. This information could be interesting to easily detect the impact of the attack in the closer HAMs or to warn a moving HAM that is getting closer to the area of the attack.

The storage module offers an abstraction from the storage method used to save and retrieve persistent data. This module is used by all the modules of the CHAM and provides them with a method for exchanging information. The storage module is able to store the following information:

- Attacks
- Attacks reports
- Configuration reports
- Capabilities reports
- Registration status
- Users of the display module.

The Display Module is the interface between the management staff and the overall DS. It supplies access to operational and management tasks that must be carried out by the management staff in the Detection Subsystem covering the CHAM and the registered HAMs. The operational tasks are related to the everyday usage of the DS. For the operational tasks, the display module provides accurate information about the ongoing EM attacks that are being performed in the railway infrastructure and about the own capabilities of the DS in order to detect attacks. This module offers the possibility to filter adequately the displayed information and to perform searches of historical attacks. The management tasks are related to the configuration of the DS. The display module eases the configuration of the parameters of the HAM and the configuration of the CHAM. This configuration is essential to modify the detection capabilities of the overall DS.

In the case that the operational and management tasks should be available for other external programs, this API should be offered by the display module. Initially, this is not considered due to the great number of railway management systems and the lack of standardization.

4.2 Interfaces

4.2.1 HAM-CHAM Interface

The first thing the HAM does at startup is to register to the CHAM. For that, the HAM implements a service advertising protocol, which allows it to discover the CHAM it must be connected to. Once the CHAM has been discovered, the HAM can register to it. This is shown in the state diagram below:

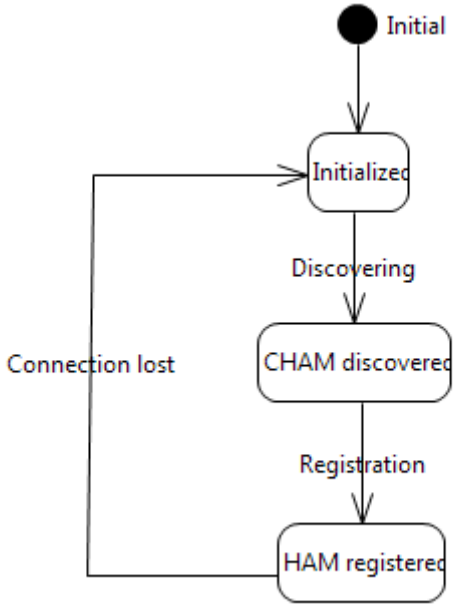

Figure 8: State diagram of the registration process

Once the registration with the CHAM has been done, the HAM periodically sends keepalive requests to the CHAM in order to inform it that it is still alive.

All the other exchanges between the HAM and the CHAM have been discussed in the previous sections.

4.2.2 ASA-HAM Interface

The Acquisition subsystem is composed of severals actors: the *Acquisition_Sink* which is the main actor, some *Sensors* according to different wireless technologies and the Acquisition_Sink communicates with a *Health_Attack_Manager* to send the current state of the various sensors.

The behavior of the Acquisition_Sink is a loop to Get_Sensor_Value.

According to the different data grabbed, it computes the new status of the sensor. Then, It sends to the Health_Attack_Manager an update of the sensor status.

Moreover, periodically, the Acquisition_Sink sends an heartbeat to the Health_attack_Manager to signal that it is always online. Figure 9 presents this use case.

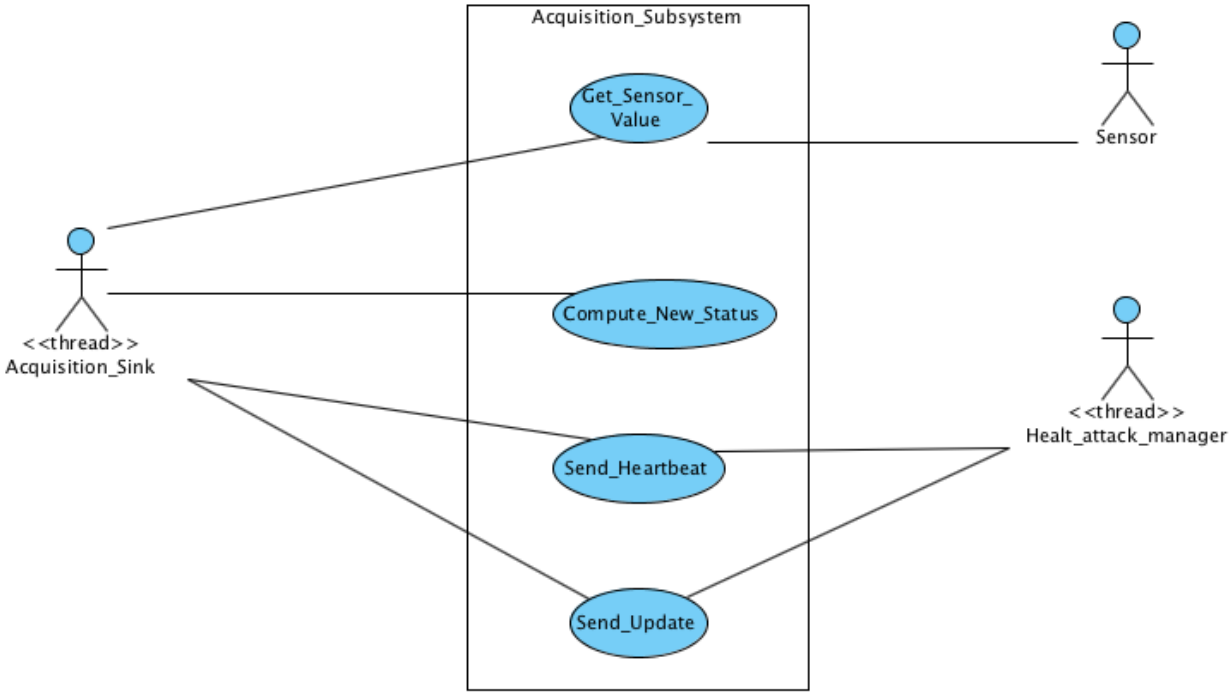

Figure 9: Acquisition Subsystem use case

4.2.2.1 Behavior of the acquisition subsystem using lifelines

The Logical_Sensor monitors a Physical_Sensor. For that, it get_data periodically and send an update of that data to the broker. Periodically, the Logical_Sensor publishes a heartbeat value to inform that it is still alive.

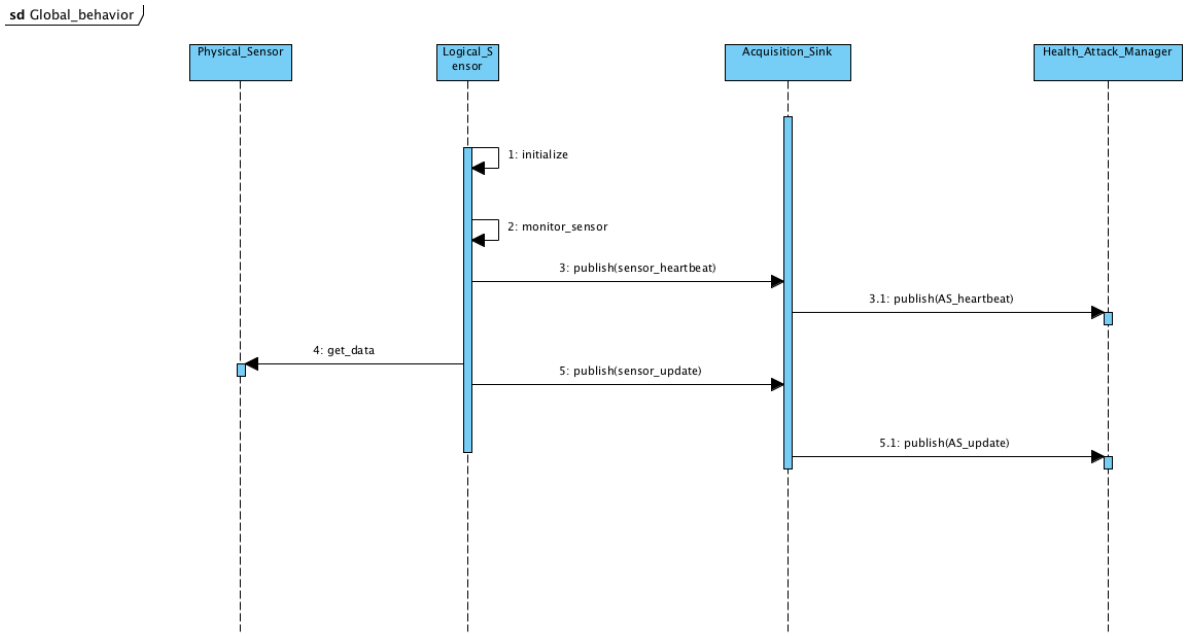

Figure 10: Sensor behavior

The Acquisition_Sink receives the heartbeat messages from the sensors. It can aggregate them and send its own heartbeat to the Health_Attack_Manager to say that everything is OK.

When the Acquisition_Sink receives a sensor_update message, it processes it and add for instance the location (if this data can be computed locally), … and then it sends it to the Health_Attack_Manager. The data will sent to the corresponding submodule (e.g. location data will be sent to the location submodule).

4.2.2.2 Message formats

The Figure 11 presents the content of the messages that will be exchanged between the differents components. The types are defined with the YDL language. An up to date version of the messages is given into an external file.

```
package SECRET_Acquisition is


--
-- Generic Message
--
      type Current_State is
            Sensor_ID : String;
            Degree_Of_Attack : Integer;
            Detect_Delay : Float;
            Attack_Evolution : Float;
            Under_Attack : Boolean;
           Type_Of_Jammer : optional Integer; -- 5 or 6 categories
      end Current_State;


--
-- For GSM
--
    type I_Q_EVM is
```

```
        State : Current_State;
        Constellation_Radius : Float;
        Constellation_Standard_Deviation : Float;
        Constellation_Shape : String; -- two values "internal" or "external"
        EVM : Float_Array;
    end I_Q_EVM;


    type Spectral_Analysis is
        State : Current_State;
        SNR : optional Float; -- Useful at base station
        Energy : Float; -- average level on the specta
        Decision_Value : Float;
        Processing_Delay : Float;

        Frequency_Min : Float;
        Sample_Size : Float;
        Number_Of_Samples : Integer;

        Samples : Float_Array; -- size = Number_Of_Samples
        Critical_Samples : Boolean_Array; -- size = Number_Of_Samples
    end Spectral_Analysis;

end SECRET_Acquisition;
```

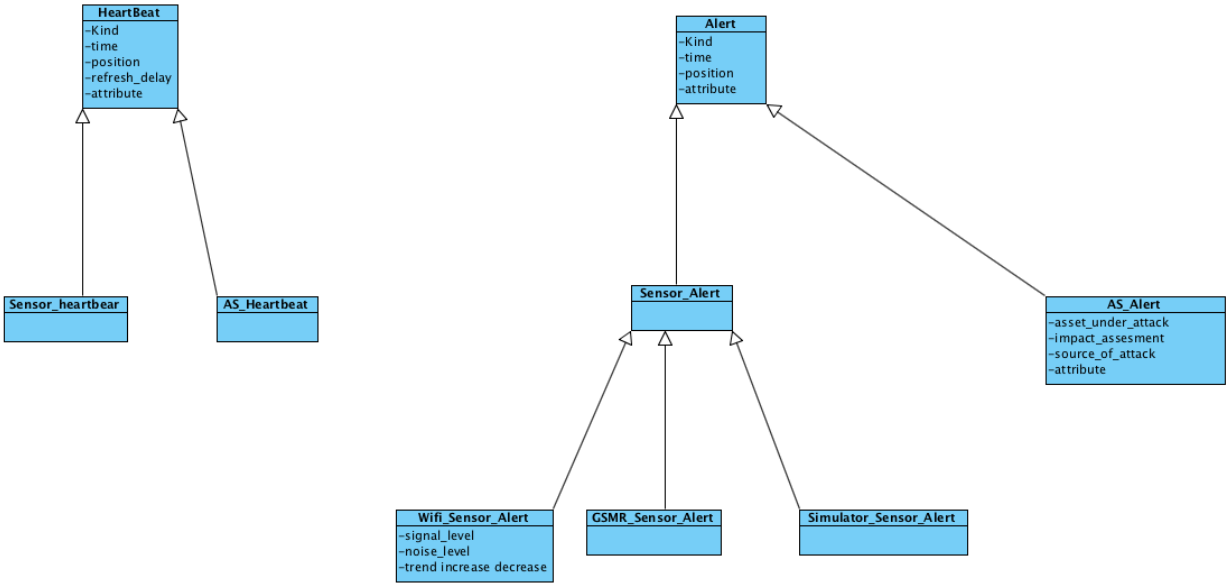

Figure 11: Message formats

The contents of the messages are generated by the algorithms for WP3.

4.2.2.3 Implementation

A broker (Yami4 - http://inspirel.com/yami4/) is currently under evaluation. Publishers and Subscribers can be developed in C++, Java, Ada, Python, …

YDL Type Mapping in implementation languages:

Types from YDL are mapped to structure types in C++.
Type fields are mapped to member fields.
Field types are mapped to standard C++ types in the following way:

- Boolean is mapped to bool.
- Boolean_Array is mapped to std::vector<bool>.
- Integer is mapped to int.
- Integer_Array is mapped to std::vector<int>.
- Long_Long is mapped to long long.
- Long_Long_Array is mapped to std::vector<long long>.
- Float is mapped to double.
- Float_Array is mapped to std::vector<double>.
- String is mapped to std::string.
- String_Array is mapped to std::vector<std::string>.
- Binary is mapped to std::vector<char>.
- Binary_Array is mapped to std::vector<std::vector<char> >.
- fields of user-defined types are implemented as member fields of already generated structure types – that is, record nesting is mapped directly to struct nesting.

Types from YDL are mapped to classes in Java.
Type fields are mapped to member fields.
Field types are mapped to standard Java types in the following way:

- Boolean is mapped to boolean.
- Boolean_Array is mapped to boolean[].
- Integer is mapped to int.
- Integer_Array is mapped to int[].
- Long_Long is mapped to long.
- Long_Long_Array is mapped to long[].
- Float is mapped to double.
- Float_Array is mapped to double[].
- String is mapped to String.
- String_Array is mapped to String[].
- Binary is mapped to byte[].
- Binary_Array is mapped to byte[][].
- fields of user-defined types are implemented as member fields of already generated classes – that is, record nesting is mapped directly to class nesting.

If this broker is not suitable, a solution based on message queuing or Data Distribution Service could be evaluated.

4.2.2.4 Publisher naming convention

Message routing can be defined with higher flexibility thanks to the hierarchical matching. A hierarchical tag has at least one dot character. The whole routing scheme with hierarchical tags can be described based on naming examples known from Usenet discussion groups.

```
AS.GSM.IQ.HB.Name_of_Equipment – for heartbeat
AS.GSM.IQ.Update.Name_of_Equipment – for value update


AS.GSM.Spectral.HB.Name_of_Equipment – for heartbeat
AS.GSM.Spectral.Update.Name_of_Equipment – for value update

AS.AS.HB – for heartbeat
```

4.2.3 MCS-HAM Interface

Once the OHAM has been registered to the CHAM, it must also register to the MCS. This process is very similar to the one described for the registration of the HAM to the CHAM.

The other exchanges between the OHAM and the MCS are described in the section 5.

5 Specification of the Multipath Communication System

5.1 Components

5.1.1 Multipath Communication Manager (MCM)

One MCM will interact with multiple components belonging to different systems, as it is shown in the Figure 12.

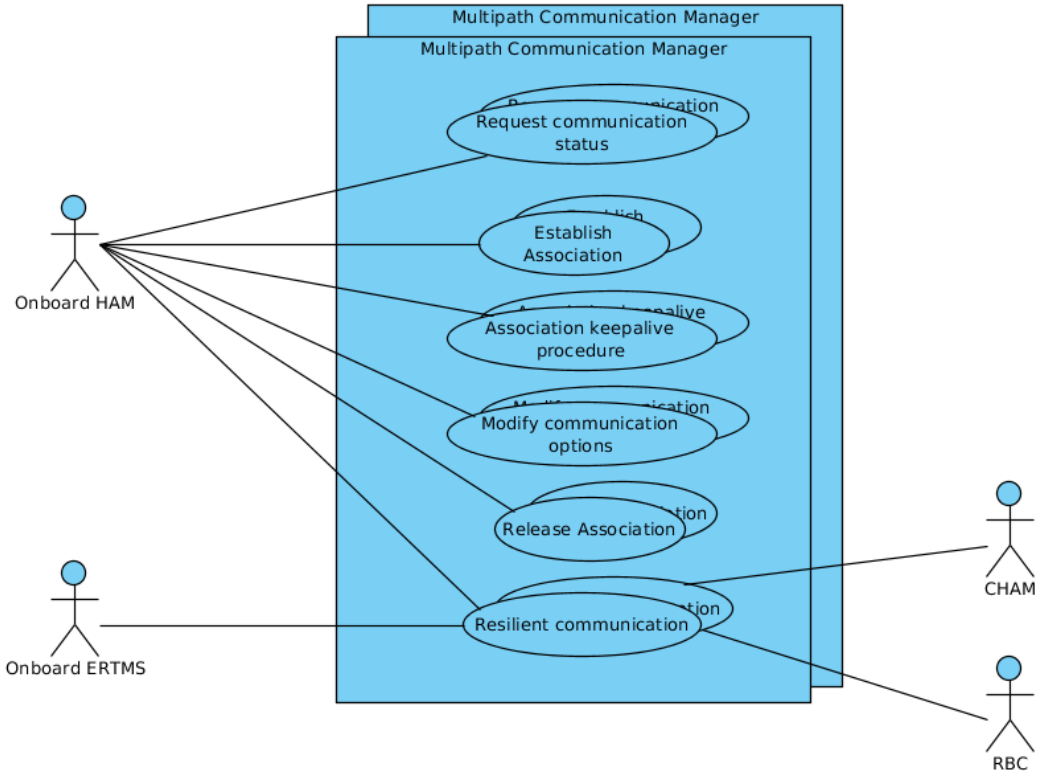

Figure 12: Detailed use case of the Multipath Communication Manager

There are two main kinds of MCM: the onboard MCM and the trackside MCM. The onboard MCM is the MCM located in the train. It provides resilient communications to the onboard ERMTS and the onboard HAM. On the other hand, the trackside MCM is located at ground and it is the multipath communication endpoint to reach the CHAM and the RBC. The actors of the onboard MCM are placed on the left of the, Figure 12 whereas the actors of the onboard MCM are placed on the right.

Multiple devices use the MCM to get the “Resilient communications” functionality. This is the case of ERMTS: onboard ERTMSs located in the trains and the RBC located in ground use MCMs to strengthen the IP communication. In addition, as it was pointed out before, other systems that require robust communication between the train and ground might also benefit and make use of the “resilient communication” functionality of the MCM such as the Detection System.

Without any additional configuration, the MCM already provides a robust multipath configuration for the devices using the “resilient communication” functionality. In this initial state, the MCM is in the “Automatic Resilient Communication” mode according to the state diagram of the MCM detailed in Figure 13. In this mode, the MCM “resilient

communication" functionality replicates the packets by all the available wireless interfaces of the logical output interface. The idea is to replicate all the packets by all the available wireless interfaces since there is not information provided by the CHAM to know is one interface is suffering an EM attack. The remote MCM will have to discard all the duplicated IP packets before transmitting them to the destination devices. In this "Automatic Resilient Communication" mode, one HAM can perform a limited set of control actions: request information about the current communication status of the MCM ("Request communication status" use case) and establish an association between one HAM and the MCM ("Establish association" use case).

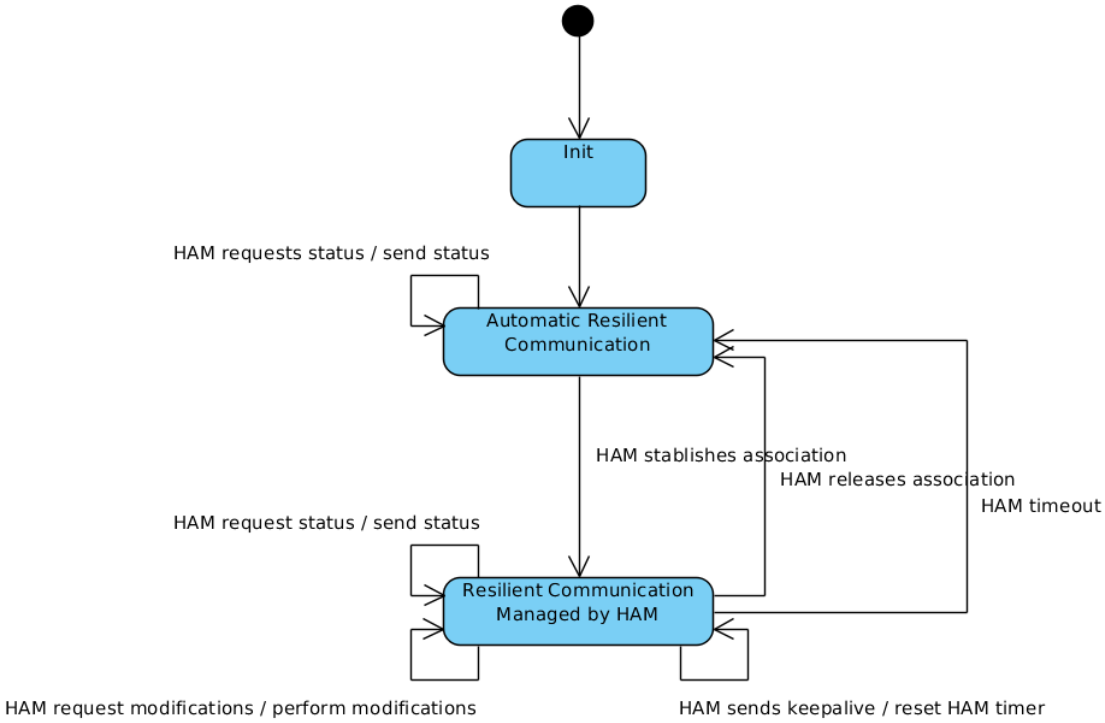

Figure 13: State diagram of the Multipath Communication Manager

In the case of establishing an association between one HAM and the MCM, the MCM gets in "Resilient Communication Managed by HAM" state. In this state, the associated HAM is responsible for governing the multipath communication and thus can completely modify in real-time the multipath communication provided by the "Resilient communication" functionality. Thus, in this state the HAM is able to modify the communication paths according to non-networking related information like information about EM attacks. For example, the HAM may decide to use by default one kind of interface such as LTE but if the HAM detects attacks in the frequency of LTE, it may decide to enable WiMAX and shutdown LTE or even enable all the available wireless interfaces to replicate the packets by all the available interfaces. This depends on the policies configured in the HAM.

In this state, the HAM may request information about the current communication state of the MCM ("Request communication status" use case) and request the modification of the multipath behaviour of the MCM ("Modify communication options" use case). Furthermore, the HAM must periodically send keepalive messages to the MCM so that the MCM could verify the HAM is still alive ("Association keepalive procedure" use case). If the MCM does not receive keepalive messages, it changes to "Automatic Resilient Communication" state in order to assure a reliable communication in case HAM has failed. The most suitable

procedure to release the association and enter “Automatic Resilient Communication” state is that the HAM performs a “Release association” use case.

This explained behaviour is the general behaviour of a MCM and is specifically the case of the onboard MCMs (MCMs located at trains). The behaviour of trackside MCMs is much simpler. They do not enter in “Resilient Communication Manager by HAM” and only work in “Automatic Resilient Communication” because these trackside MCMs are not going to be managed locally by a HAM due to their lack of necessity of dealing with EM interferences. The trackside MCMs have only wired interfaces and are not affected by the EM attacks.

Before explaining in more detail the use cases defined in the Figure 12, the modular architecture of the MCM is going to be presented. As it is shown in the Figure 14, the Multipath Communication Manager (MCM) consists of **two main modules:** the proxy module and the control module.

The **proxy module** is the main component of the MCM. It provides a more robust data transport method for inner IP devices by converting traditional IP data flows based on one unique path into communications based on multiple paths. Thus, the proxy is the module responsible to implement the multipath approach selected. This proxy module is closely related to the “Resilient communication” use case.

The **control module** is responsible for delegating the control of the proxy module and consequently the control of the behaviour of the multipath approach to the detection system. One Health/Attack Manager (HAM) may decide to register in the MCM and govern its behaviour. This decision may be triggered because the HAM has been configured to govern a local MCM. Once registered, the HAM should be able to allow and disallow the wireless network interfaces of the MCM depending on the EM attacks detected or order inputs the Detection System may have. This module is closely related to all the previous use cases associated with the control of the MCM: “Request communication status”, “Establish association”, “Association keepalive procedure”, “Modify communication options” and “Release Association”.

The MCM has three main external interfaces, which will be explained in more detail in section 4.2 of this document:

- The input interface is the interface for providing resilient communications to legacy IP devices such a next generation ERTMS system. This interface is implemented on top of a wired interface.
- The output interface is the interface that is going to implement the multipath approach. Depending on the kind of the MCM, the physical interfaces associated to this logical output may vary. For example, in the case of a MCM located in a train, the output interface would consists of several wireless physical interfaces; whereas in the case of a MCM located in ground, the output interface could consists of one or even several wired physical interfaces.
- The control interface will allow the management of the MCM by the Detection System.

Apart from these external interfaces, there is an internal interface between the control module and the proxy module that will allow the control module to reconfigure the behaviour of the proxy module.

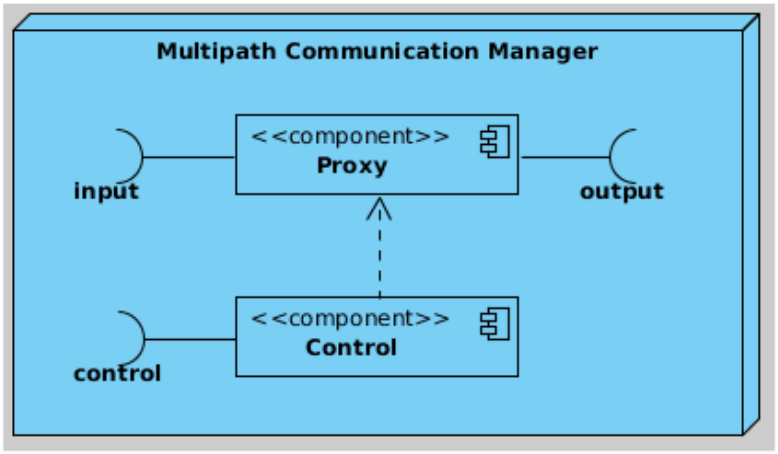

Figure 14: Architecture of the Multipath Communication Manager

Once described the internals of the Multipath Communication Manager, following the operation of the MCM is detailed by describing with UML diagrams its principal use cases.

Use Case: Resilient Communication

Devices using the MCM for resilient communications will use the “input interface” of the proxy module of the MCM to transfer legacy IP packets towards a remote device IP device. The MCMs will transfer IP packets among them through the output interface using a multipath approach in order to strengthen this section of the path.

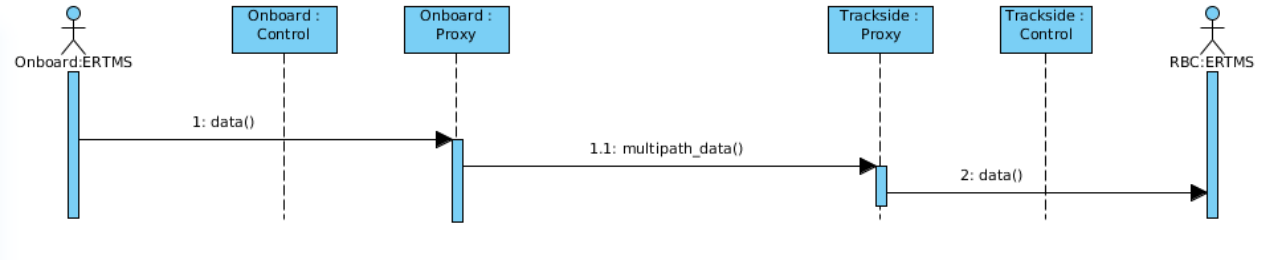

Figure 15: proxy behaviour

The specific behaviour of the proxy modules will depend on the configuration performed by the control module. Two main traffic policies are considered:

- Replication of packets through all the interfaces in case of “Automatic Resilient Communication” state
- Replication depending on the instruction provided by the HAM in case of “Resilient Communication Managed by HAM” state.

Use Case: Establish Association

The HAM may request to establish an association with the MCM through the control interface. The control module must verify the identity of the HAM and that the MCM is not

already managed by another HAM. If both checks are correct, the control module will force the proxy module in manual state instead of automatic mode. The sequence diagram is detailed in the Figure 16.

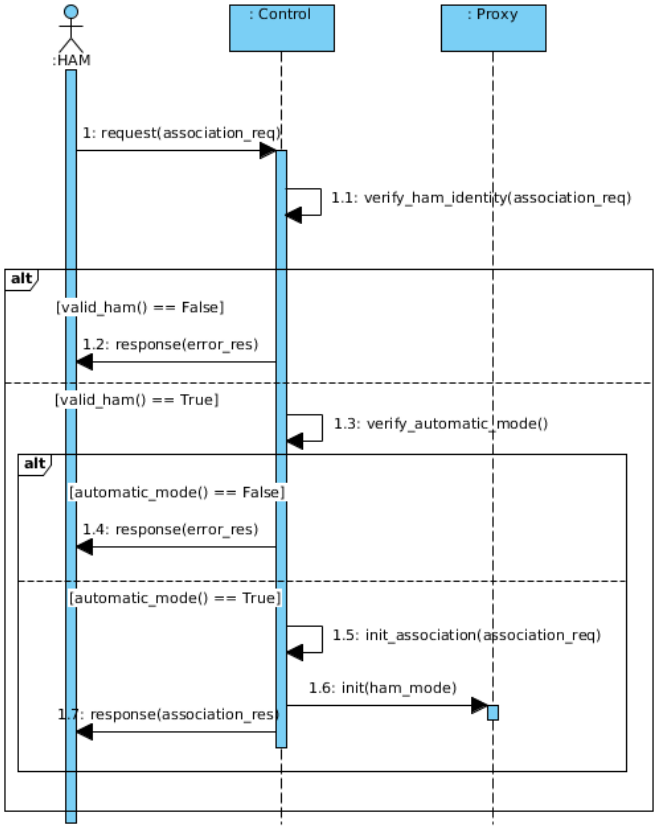

Figure 16: Sequential diagram for establishing a new association

Once established the association between the MCM and the HAM, if there is any change of state of any of the interfaces of the proxy module, it will be notified immediately to the HAM. Thus, the HAM may react and instruct the MCM to take the required reactive actions.

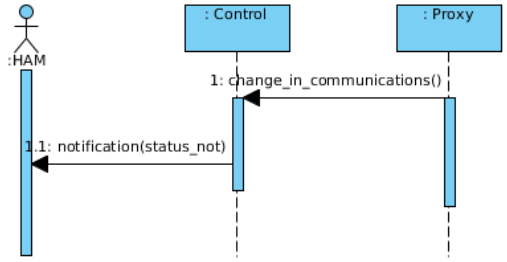

Figure 17: Sequential diagram for events on proxy interfaces once established the association

Besides, once established the association, the HAM is required to send periodically (according to the periodicity set by the MCM in the response to the association request) keepalives to the MCM. Thus, the MCM is aware that the HAM is still governing the behaviour of the multipath system.

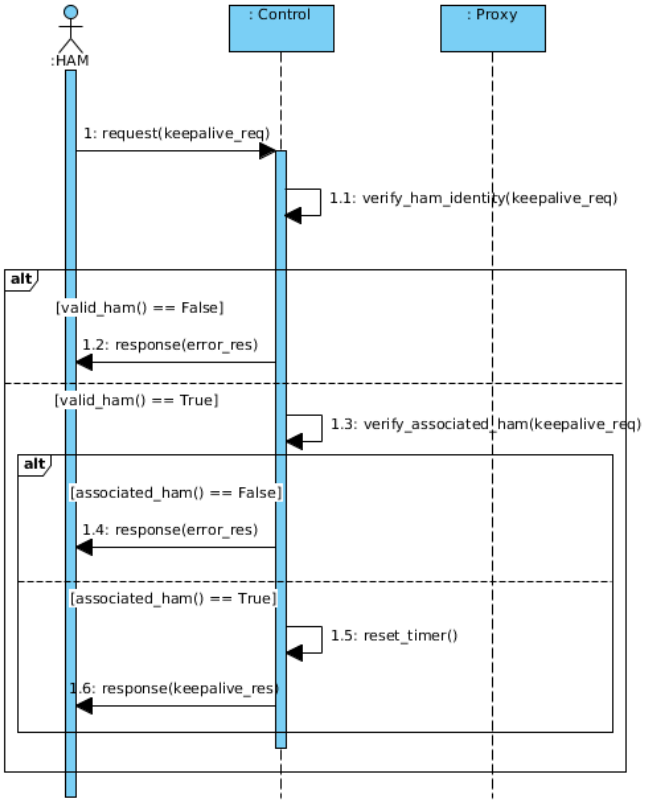

Figure 18: diagram for the keepalive procedure

The lack of keepalives will suppose the release of the association and the MCM enters automatically in the “Automatic Resilient Communication” state without sending any kind of notification to the HAM.

Use case: Request status

One valid HAM, even one HAM not associated with the MCM, can request through the control interface the status of the MCM. The response will provide information about the operational status and the configuration status because they may differ. For example, one interface may be configured as backup by the HAM but because of a problem in the currently configured active interface nowadays it is the active interface.

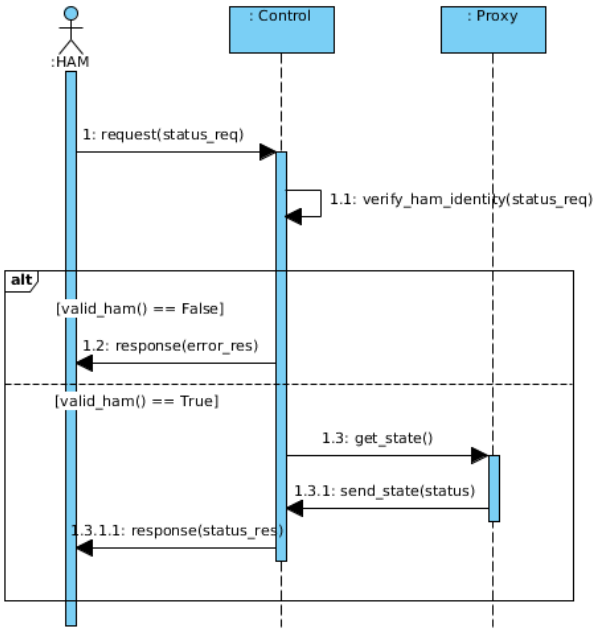

Figure 19: Sequence diagram for the request state procedure

Use case: Modify configuration options

One valid and already associated HAM with a MCM may accomplish the modification of the communication options of the MCM through the control interface. This allows a HAM to enable or disable one interface of the MCM and to establish that interface as active or backup. This procedure is shown in the Figure 20.

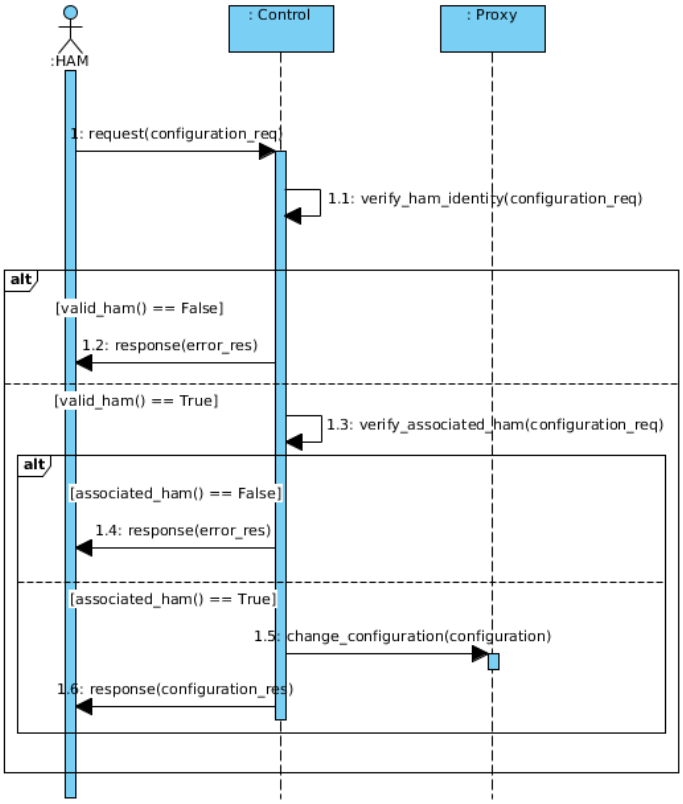

Figure 20: Sequence diagram for modifying the multipath options

Use case: Release association

One HAM associated with a MCM may request the release of the association to the MCM. The MCM will automatically enter in the “Automatic Resilient Communication” state. In the “Automatic Resilient Communication” state, all the available interfaces will be enabled and set in active mode to perform packet replication through all the interfaces.

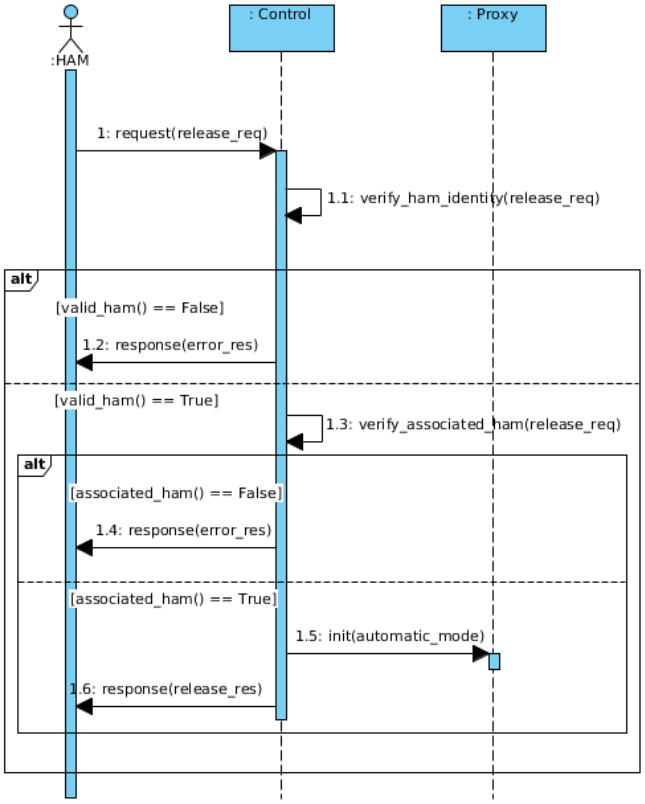

Figure 21: Sequence diagram for releasing an established association

5.2 Interfaces

5.2.1 Control Interface

The control interface allows the management of the MCM by a HAM. This control interface must implement the messages between the HAM and the control module of the MCM required for enabling the use cases detailed in the previous section

This interface supports three main types of messages: request messages, response messages and notification messages. In order to specify the type of several message attributes some datatypes must be defined and these are listed in the Figure 22.

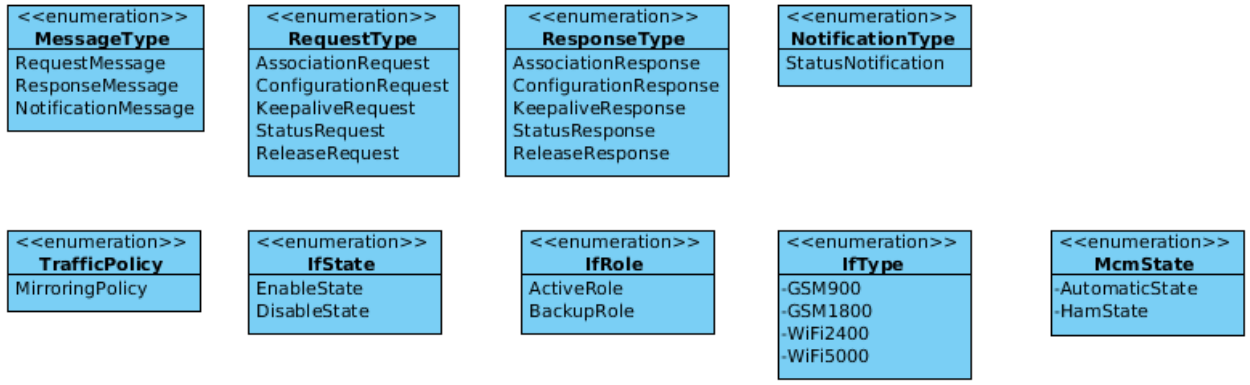

Figure 22: Datatypes used for the definition of the control interface messages

The request messages are messages launched from HAM to the MCM. There are five different request messages whose contents are defined in Figure 23

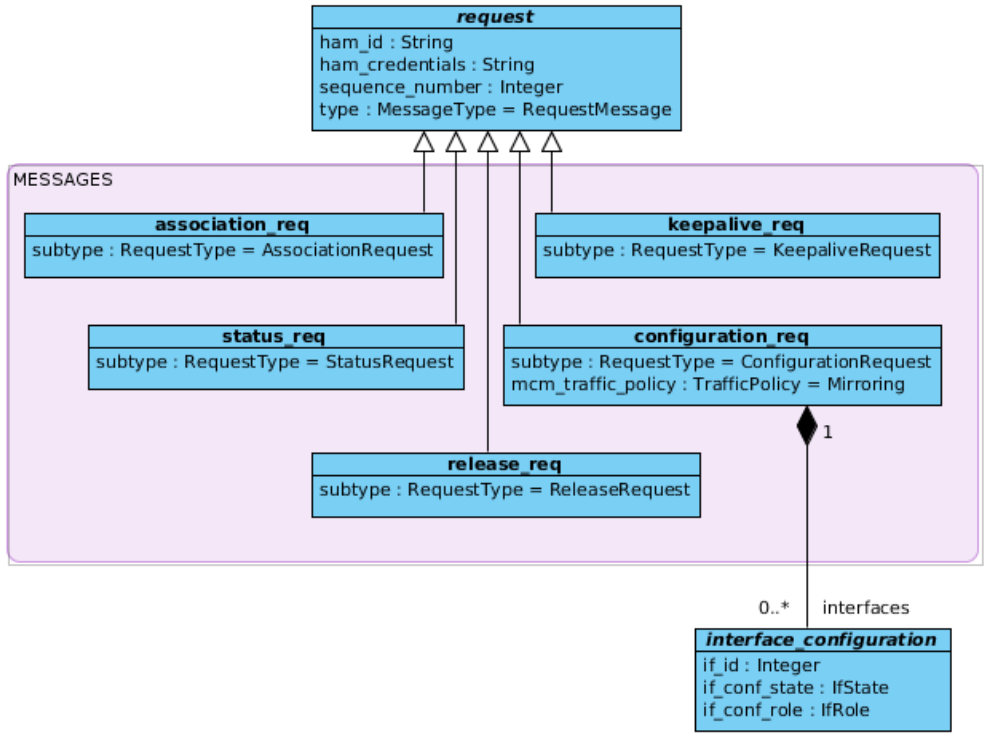

Figure 23: Request messages of the control interface

There is a response message associated to each request messages. Furthermore, there is response message for report errors in case the execution of a request produces an error. These response messages are detailed in the Figure 24.

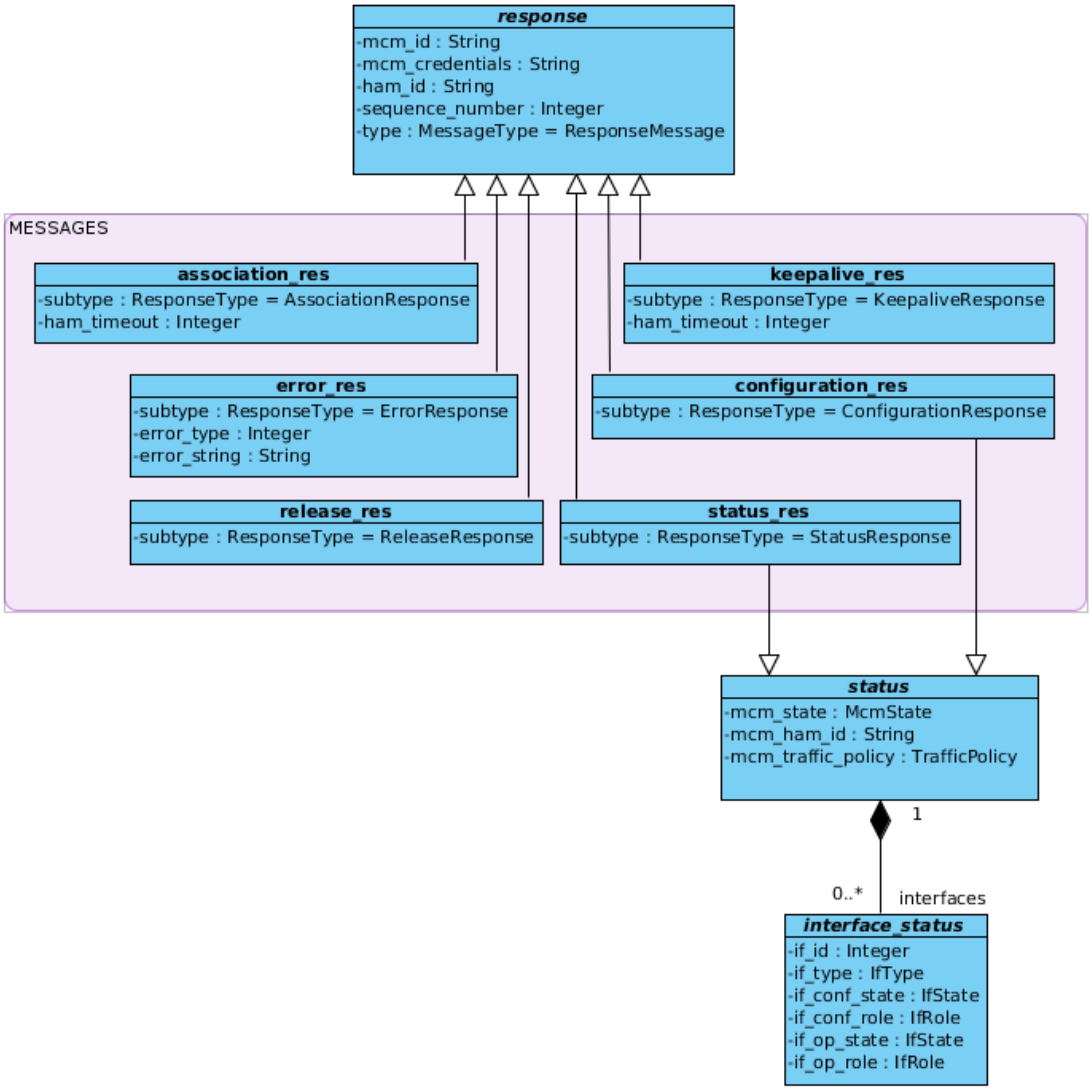

Figure 24: Response messages of the control interface

Finally, there are some asynchronous notifications that may be sent by the MCM to the registered HAM without a previous request of the HAM. The notifications are detailed in the Figure 25.

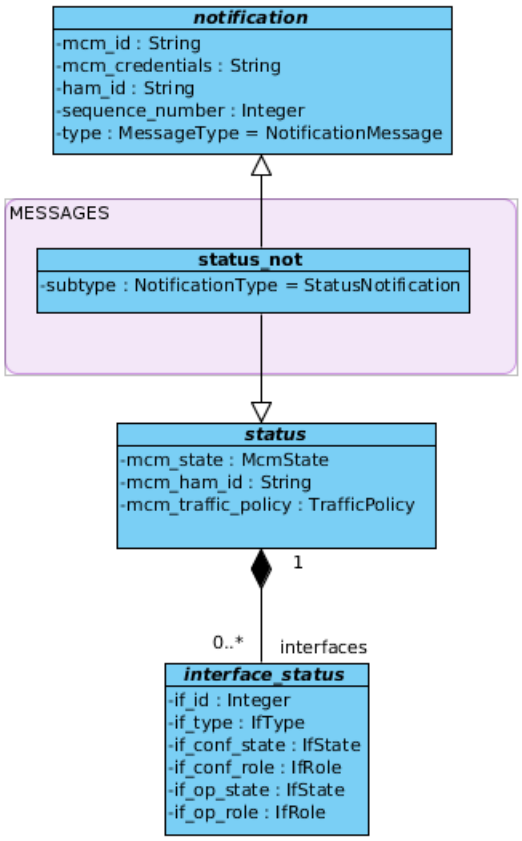

Figure 25: Notification messages of the control interface

The control interface will be deployed on top of an Ethernet 100Base-T compatible wired interface. The use of a wired and shielded interface is mandatory to avoid the disruption of the communication between the HAM and MCM due to EM attacks. In order to ease the auto-discovery of the MCM by the HAM, the MCM should publish the MCM service as a Zeroconf service. Thus, the HAM may find the MCM without any previous configuration in case it could be interested in managing the MCM manually, for example, because there are some rules in the MCM that require the use of a local MCM. Furthermore, the IP address of the control interface for the MCM could also be configured statically or dynamically via one DHCP option or Zeroconf procedure.

5.2.2 Input interface

The input interface is the logical interface used to provide service to IP devices that want to benefit from the resiliency provided by the MCM.

This input interface will be deployed on top of an Ethernet 100Base-T compatible wired interface. The wired interface is mandatory to reduce the impact of EM attacks. In addition, that interface is recommended to be shielded against EM interferences.

In order to ease the configuration of the inner IP devices, the MCM will provide an address autoconfiguration service such as DHCP. However, it would be also possible to configure the inner devices with static IP addresses. The IP address of the MCM will be provided to the inner devices by the autoconfiguration service as the “default router” in order to route all their IP packets through the MCM.

Finally, the control interface defined in the Section 5.2.1 and this input interface may be deployed on top of the same physical interface or may use different physical interfaces.

5.2.3 Output interface

The output interface is the logical interface where the multipath approach is carried out among MCMs to strengthen the IP communications against EM attacks.

This logical interface will be composed of several physical interfaces. In the case of the one onboard MCM located at a train, the output interface will be deployed on top of several wireless IP interfaces of the same or different technology (GPRS, LTE, ...). The more number of wireless interfaces per train and more disjointed technologies used, the more resiliency will be able to provide the MCS. The only requirement for these interfaces is that they need to be IP-capable. For example, current ERTMS and GSM-R are not valid for the MCM because IP protocol is not used.

On the other hand, in the case of the MCM located at the command centre (in the trackside), the output interfaces of the trackside MCM are wired. The number of interfaces is variable, but increasing the number of wired interfaces of the trackside MCM does not provide more resiliency against EM attacks because a greater number of interfaces in the trackside MCM does provide a greater number of paths but these paths are not different in their wireless sections, which are exposed to EM attacks. This is explained with an example in the Figure 26Figure 1. In this example it is detailed how the wireless resiliency is only achieved increasing the number of wireless interfaces in the onboard MCM.

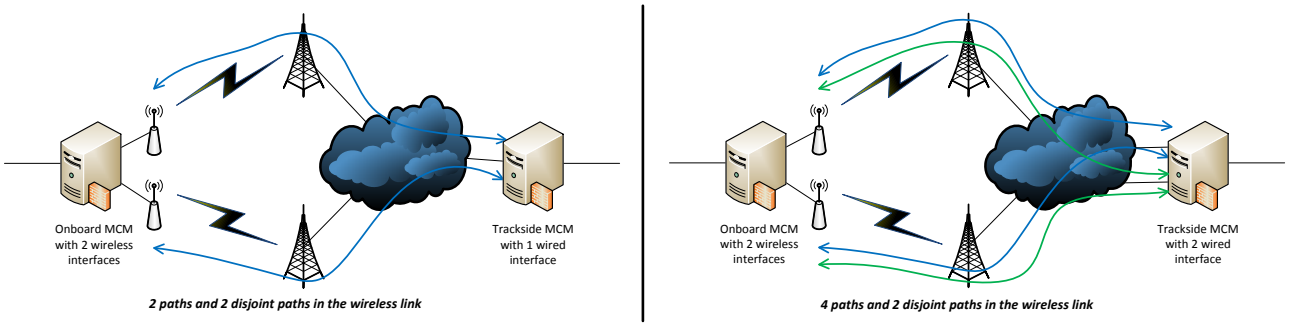

Figure 26: Example of disjointed wireless links

In conclusion, multiple interfaces in the onboard MCM supplies paths with different wireless links to face EM attacks; whereas multiple interfaces in the trackside MCM increases the number of paths but not the number of different wireless links. However, increasing the number of interfaces in the trackside MCM may provide resiliency against other type of failures such as hardware failures on the MCM or networking failures.

One of the most challenging issues of this MCM is to perform the multipath communication smoothly. In order to get it several technical solutions has been considered along the project and have been evaluated against the following requirements:

- **Mobility**. It would be advisable to support IP mobility because the train, and consequently the onboard MCM, is in movement and depending on the used wireless technology, the IP address assigned to the wireless interface of the MCM could change. In this sense, it would be advisable that the multipath approach could maintain the established IP connections instead of requiring to reinitialize them every time there is a change of IP address.

- **Multihoming**. A multihomed device is a device connected to several IP networks using one or several network interfaces. It is often used to overcome hardware failures. Furthermore, it is interesting that multihoming protocols supports also mobility to dynamically add and delete IP addresses without losing the established connections.
- **Multipath Transfer**. Multipath transfer is a communication technique that consists on using multiple alternative paths through a network to enable the communication between two devices with the objective of increasing the fault tolerance, throughput or security. In fact, to allow multipath communications it is required to have multihomed devices.

  Multipath communications is usually related to Concurrent Multipath Transfer (CMT), which is a technique for increasing the throughput by load-balancing the data through all the available paths. For the secret project, the interest is focused more on replication than on CMT. The signalling protocols of ERTMS does not require a high throughput and the objective is to replicate the packets through all the available paths to increase the possibility of successfully transferring data in case of EM attacks.
- **Proxy solution**. The MCM is a proxy that translates the traditional IP communication into a multipath communication and viceversa. The design decision of a proxy was considered mainly to ease the integration of the MCS with the Detection System and to provide resilient communication to multiple devices simultaneously. Thus, the multipath approach must support a proxy-based solution.
- **Transparency**. The multipath approach must be transparent for devices that do not implement already the multipath solution. For example, although the RBC should initially support the multipath solution perhaps not all the trains will have one onboard MCM installed and these trains with legacy IP communications must be able to communicate with a multipath-capable RBC (one RBC behind one trackside MCM).
- **Standardization and availability**. Another key factor for the selection of the multipath approach is the level of standardization and maturity of the technological solution.

All the above points taken into account, initially a solution based on **Software Define Networking (SDN)** (1) or **Openflow** (2) was considered. Nowadays, SDN is heated topic in computer networks. The idea of SDN is to consolidate the control of the network in one device called controller and to extend the programmability of software to the networking devices. The advantages of this approach are the deployment of new protocols and disruptive networking solutions that it would be slow or even impossible to perform with legacy switches and routers, which has a default behaviour and where deploy a new standard protocol may take years or decades.

However, SDN is focused on control network devices called datapaths and not end hosts. Thus, a SDN-based solution to get multipath communications would be applied on the core devices of the IP network. Such a great change in the communication network of the current Railway agents was not considered to be affordable or realistic and was directly discarded.

Another approach for SDN was to focus only on the edges of the network and thus deploy SDN-capable devices in the edge of the core network and relay on the current IP networks of the Railway agent. In order to get the multipath communication, an overlay technology such as IP tunnels could be used to dodge the characteristics of the underlying legacy IP

network and connect the datapaths located in the edge of the network. However, this approach has also its drawbacks.

To begin with, current Openflow specifications are heavily bound to Ethernet frame format. Nevertheless, not all the wireless technologies used for the MCM may be based in an Ethernet frame. GPRS-R that seems to be the first technology used for ETCS over IP does not forward Ethernet frames. Thus, this is an important limitation.

Secondly, due to the distributed architecture of the MCS with MCM in trains and in the trackside it would be difficult to control all the datapaths (MCM) with one unique Openflow controller. We should consider the possibility of losing the connection with the train. That could be possible and in that case it would be advisable to have a local controller in each train to always maintain the control of the datapath (MCM) and be able to signal new instructions to the datapath (MCM) in order to re-establish the remote communication. This breaks the concept and the benefit from having one unique controller to manage all the datapaths (MCM). Although, there is research on the interconnection of Openflow domains each of them managed by one Openflow controller, it is not mature enough.

Due to all these reasons, the SDN approach was set aside and other approaches were considered. We focused on protocols used on top of legacy IP networks to get multipath transfers. There is a huge list of candidates and the main alternatives were considered: Stream Control Transmission Protocol (SCTP) (3), Multipath TCP (MPTCP)(4) and Host Identity Protocol (HIP) (5). Other alternatives were quickly discarded. For example, the Site Multihoming by IPv6 Intermediation (SHIM6) (6) is a protocol only valid for IPv6 protocol and not for IPv4.

The Stream Control Transmission Protocol (SCTP) is a connection oriented transport protocol that replaces TCP. SCTP, opposite to TCP, transmits sequence of messages instead of raw bytes. It is thus considered a message-oriented protocol. SCTP supports multiple streams inside one connection, what allows transmitting multiple streams of messages in parallel with independent ordering of messages. SCTP also supports multihoming for failover. However, RFC 5061 (7) is required to support mobility. In other words, in order to be able to modify the IP addresses (add, delete or set a new preferred IP address) involved in one already established SCTP association RFC 5061 must be implemented. Furthermore, there are other proposals (8) to get SCTP to support multipath transfer in particular Concurrent Multipath Transfer (CMT).

However, SCTP has been discarded for the SECRET Project because applications must use the SCTP transport protocol explicitly instead of TCP or UDP protocols. This is quite unusual, because the most applications use TCP or UDP. In addition, as it has been detailed in Deliverable 4.1 of the SECRET Project, ETCS and Euroradio protocols are supposed to be encapsulated on top of TCP so SCTP is completely useless.

Multipath TCP (MPTCP) and Host Identity Protocol (HIP) are the main approaches that have been finally considered for the secret project.

MPTCP can be considered an extension of TCP protocol since it is implemented in the options field of the TCP protocol. When one TCP application establishes a new connection with a remote host, both hosts verify if they are MPTCP capable. In that case, one MPTCP session is established; if not, a standard TCP session will be established. If the MPTCP session is established and moreover the hosts are multihomed, hosts will announce their additional IP addresses to the peer and new TCP flows will be established and will be associated to the current MPTCP session. MPTCP performs Concurrent Multipath Transfer (CMT) through all the available TCP flows associated with the MPTCP connection. Furthermore, MPTCP supports mobility and the session will add or delete TCP

flows as long as IP addresses are added, deleted or changed during the lifetime of the session without losing the MPTCP connection.

One of the main advantages of MPTCP is that it can be used transparently by any TCP application. Other major advantage of MPTCP is its rapid adoption and inclusion in commercial software. Recently Apple has included support for MPTCP in IOS 7 (9). Finally, the current multipath transfer policy deployed in MPTCP is CMT and this policy should be changed for a simpler replication policy in the SECRET project.

Perhaps the only drawback of the MPTCP it that it is a solution limited to TCP applications, avoiding UDP or SCTP applications benefit from the multipath approach. This is a restriction that may affect other services interested in benefit from resiliency provided by the MCM but it is not an issue for ERTMS.

HIP is another main approach. HIP provides an abstraction between the identification of a host (Host Identity Tag) and its location (IP address). It is a protocol located between the network and transport layer, and so any transport protocol such as TCP or UDP may be used on top of HIP. This is an advantage compared to MPTCP and important in our project with the intention of deploying a proxy that should provide service to major amount of applications without imposing restrictions. HIP provides also important security features to authenticate the end devices of the connection and protect the data transferred against modifications and eavesdropping. These security features may be implemented in upper layers of the protocol stack for example by SSL protocol or in the case of ERTMS by the Euroradio protocol. Thus, these security features could be simplified although it may be an interesting feature in a security demanding environment. HIP supports multihoming and also implements mobility features to maintain HIP associations when there are handovers.

The major disadvantage of HIP is its lack of support for multipath. There are several research papers for providing HIP with CMT called mHIP (multipath HIP). However, it is not standardized and it is not implemented in official distributions of HIP. This could be a problem, but the implementation of a replication multipath transfer should not be so complicated as implementing a CMT with HIP. The other disadvantage is that HIP seems not to have so much commercial support as MPTCP. The current implementations of HIP are quite basic with no as good performance as MPTCP.

All things considered, we have finally selected the MPTCP protocol to build on top of it the multipath system in order to offer more strength to TCP applications against EM attacks.

Annex A Approach to Evaluate SECRET Resilient Architecture

A1 Introduction

The objective of this section is to specify the approach which will be used to evaluate the SECRET resilient architectures features. The approach used was explained in deliverable D4.1 (see section 4 Approach to justify SECRET Resilient Architecture). It listed a number or risk management methodologies such as the BowTie risk analysis method[1] for critical infrastructures or the TVRA analysis[2] for ICT infrastructures, as well as an architecture design methodology[3] defined by the Carnegie Mellon University Software Engineering Institute (CMU-SEI).

The TVRA methodology includes an impact analysis phase that can be used to provide some level of justification of the SECRET resilient architecture. This justification is counter measure centric, i.e. it provides the gain with respect to the introduction of a given counter measure. But the methodology does not provide a direct evaluation of the architecture in terms of resiliency.

On the other hand, CMU-SEI work on architecture also includes a methodology for architecture evaluation, involving two methods:

- An analysis method called ATAM (Architecture Trade-off Analysis method)4.
- An evaluation method called CBAM (Cost Benefit Analysis Method)

ATAM and CBAM are to our knowledge the most widely used or known architecture evaluation methods used. We suggest the application of these methods to the evaluation of SECRET resilient architecture.

This application is not completely straightforward:

- Both TVRA and ATAM are based on heuristic values and metrics. The evaluation of SECRET resilient architecture will need a preliminary proposal on such heuristics and metrics.
- ATAM methodology highlights the need for consensus, i.e. the resulting gain combines the opinion of the stakeholders involved in the design of a system (for which an architecture is specified). The evaluation of SECRET resilient architecture will therefore need an assessment by such stakeholders.
- The resilient architecture specified in SECRET must be as generic as possible. This means that some parameters of the actual architecture are not known. Further the impact of some attacks is still under study. To this end, we will integrate in the analysis some variable (something like, EMC attack pattern X). Consequently, the application of SECRET resilient architecture should always be associated **with an architecture instantiation evaluation**.

---

[1] http://www.governors.nl/bowtieexp.html

[2] ETSI. Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN); Methods and protocols; Part 1: Method and proforma for Threat, Risk, Vulnerability Analysis. s.l. : ETSI, 2011. Technical Specification. V4.2.3.

[3] http://www.sei.cmu.edu/architecture/.

[4] See also: http://www.sei.cmu.edu/architecture/tools/evaluate/atam.cfm

A2 Methodology Used

This section describes the ATAM and CBAM methods. This section is based on the following reference:

- Reference 1: Carnegie Mellon Software Engineering Institute research contribution, well explained in the following reference document: Software architecture in Practice. 3rd edition. Addison Wesley, 2013. Len Bass, Paul Clements, Rick Kazman.

This section is structured as follows:

- We first start with an introduction section on architectures and what it entails
- We then explain the ATAM method
- We then explain the CBAM method

A2.1 Architecture Decisions

There are a number of reasons on why architecture is important:

- Inhibit or enable a system's quality attribute
- Reason about and manage change
- Prediction of system's qualities
- Documented architecture for communication among stakeholders
- Allow for the earliest, most fundamental hardest-to-change design decisions
- Defines set of constraints on implementation
- Dictates structure of organisation and vice-versa
- Basis for evolutionary prototyping
- Allows architecture and project manager to reasons about cost and schedule
- Can be created as a transferable reusable model
- Architecture based development focus on assembly of components, not only their creation
- Restrict design alternatives, reduces design and system complexity

Foundation for training a new team member

Among these reasons, we single out the prediction or system's quality. This has an impact in the elicitation of the requirements of a system. Several types of requirements are identified:

- ***Functional requirements*** relate to what a system does (e.g. engine control). Another term used in the literature is r*esponsibility*.
- ***Quality attributes requirements*** relate to how well a system does it. Examples of quality attributes could be execution qualities (e.g. security, usability, dependability), evolution qualities (testability, maintainability, scalability), business qualities (time to market, cost. Another term that is close is non-functional requirement[5].
- ***Constraints*** are design decisions that are already taken (e.g. re-use something, use a given operating system, use IP V6)

[5] https://en.wikipedia.org/wiki/Non-functional_requirement.

These requirements lead to architecture decisions: functional requirements are satisfied by defining appropriate set of responsibilities within design. Quality attributes are satisfied by structures and behaviors of the architecture.

D4.1 already explained quality attributes (the section is reproduced in italics) :

The approach is to specify scenarios, which are structured means to state attribute requirements. A scenario includes the following elements:

- *Source At security level, the source is typically an attacker*
- *Stimulus. At security level, the stimulus is typically an attack e.g. an EM attack*
- *The stimulated Artefact. At security level, the stimulated artefact is typically the system being attacked.*
- *The Environment or the conditions under which a stimulus occurs (e.g. normal train operation)*
- *The Response to the stimulus. The response is influenced by architecture techniques called Architecture tactics (e.g. switch to manual mode operation)*
- *The Response Measure. This measure is needed in the design process in order to validate the architecture design (e.g. train still in operation at 200 km/h)*

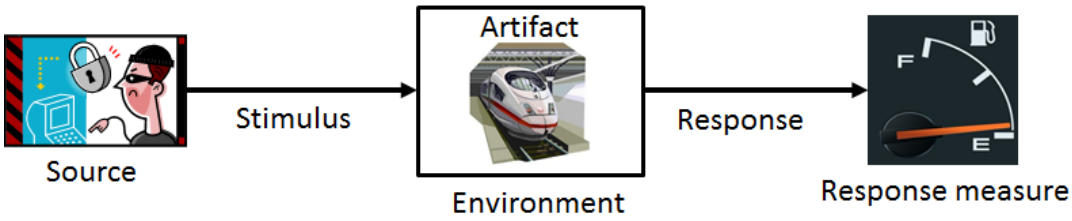

Figure 27: Scenario Model

Figure 27 shows the resulting scenario model. The whole design process can be described as involving three steps:

- *Identification of scenarios.*
- *Influencing the responses by selecting appropriate architecture techniques called in CMU jargon **architecture tactics**.*
- *Measuring the responses in order to validate the design decisions*

An ***architecture tactic*** is a design decision that includes the achievement of a quality attribute response

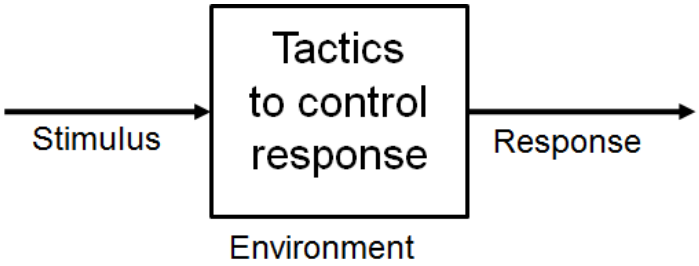

Figure 28: Architecture Tactic

Figure 28 shows where tactics artefacts are displayed in a the scenario model.

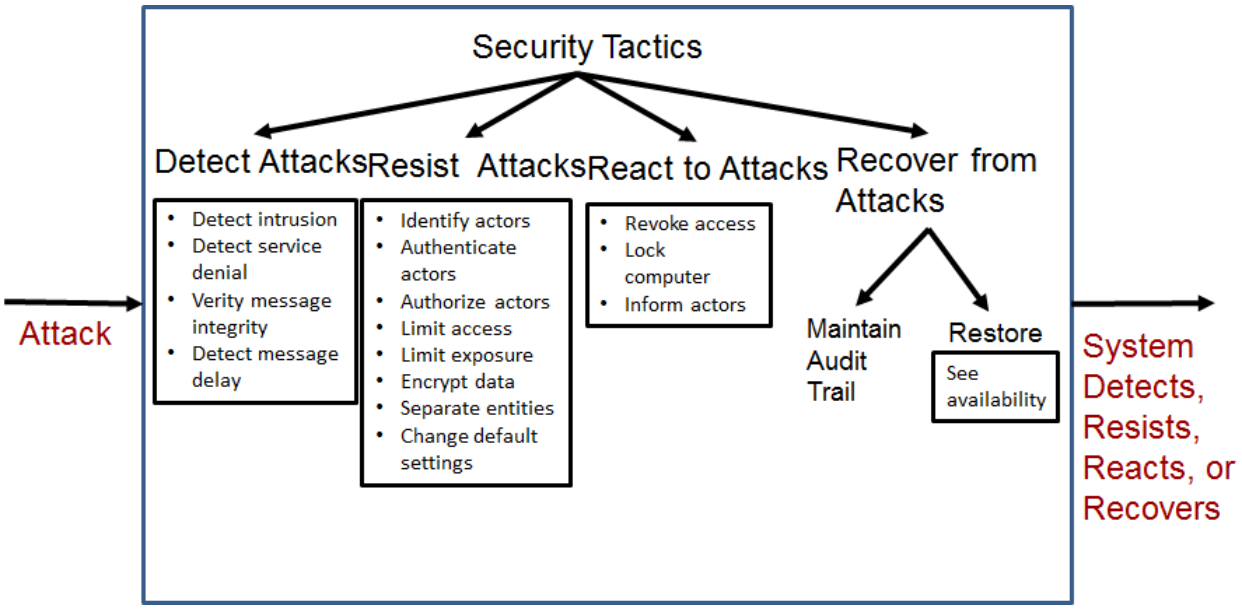

Figure 29: Example of Tactics for Security Attribute

Figure 29 shows the taxonomy of security tactics. Reference 1 provides lists of tactics for a good number of attributes.

Quality attributes are further associated with ***response models***, or functions to predict the response measures given a stimulus for a particular architecture. There are two known approaches for such models:

- analytic models which support quantitative analysis (e.g. Markov models for hardware availability, scheduling theory for predictability),
- check lists/guidelines which support scales (e.g. common criteria level, safety integrity levels).

A2.2 ATAM

In ATAM, architecture requirements are also known as ***ASR (architecturally significant requirement)***. These requirements are captured in multi-stakeholder workshops (e.g. project managers, members of development team, testers and integrators, maintainers, product line application builders, customers, end users, business managers …). These ASR are specified through ***utility trees***, consisting of 3 levels:

- Level 1 in the tree will describe quality attributes (e.g. performance)
- Level 2 in the tree will describe attribute refinements (e.g. latency)
- Level 3 in the tree will describe the ASR through the following
  - A quality attribute scenario
  - A business value (high, medium, low)
  - An architecture impact value (high, medium, low)

The table below show 3 examples of utility tree

Level 1	Quality attribute	Performance
Level 2	Attribute refinement	Throughput
Level 3	ASR	At peak load, system is able to complete 150

		transations per second
	Business value	Medium
	Impact on architecture	Medium

Level 1	Quality attribute	Security
Level 2	Attribute refinement	Integrity
Level 3	ASR	System resists intrusion and reports intrusion within 90 seconds
	Business value	High
	Impact on architecture	Medium

Level 1	Quality attribute	Configurability
Level 2	Attribute refinement	User-defined changes
Level 3	ASR	A hospital increases the fee for a particular service. Configuration team makes the change in 1 working day
	Business value	High
	Impact on architecture	Medium

The architecture design process ATAM is iteration-based. Each iteration is described with an input, an iteration process and an output. The input is a list of requirements (functional, quality, constraints) and architecture requirements. The output consists of sketches of architectural views. The iteration process includes the following steps:

- The selection of the element of the system to design
- The identification of ASR (architecturally significant requirements) for that part
- The generation of a design solution
- An inventory of remaining requirements and selection of input for the next iterations.

The resulting documentation consists of a number of models. Reference 1 suggests three types of models:

- Modules views which provide a static view or a focus on the decomposition into elements of a system.
- Component and connectors views which provide a dynamic view or a focus on the interactions between elements
- Other views dedicated to the specification of mapping issues to specific environments, i.e. organisation, development, installation. For instance a software architecture consisting of software modules must be mapped on top of a hardware architecture through an allocation view).

When the same design decision is found repeatedly, ***architecture patterns*** are used, i.e. documentation and models can be made readily available for future re-use. Examples of well-known patterns are the following: layers, client-server, publish-subscribe, shared data, isolation.

The application of ATAM involves a well-defined organisation. Participants include

- the evaluation team. It involves specific, important roles:
  - a team leader (in charge of customer relation)
  - an evaluation leader (in charge of the evaluation)
  - a scenario scribe (which lists the proposed scenarios)
  - a proceedings scribe (which lists the adopted scenarios)
  - a questioner (which raises issues of architectural interest in the area where he has expertise)
- the project decision makers
- the architecture stakeholders. These stakeholders could be many in large projects (up to 12 to 15 persons). They do not participate to the entire exercise

The application of ATAM involves a well-defined process with the following phases:

- a preparation phase. It involves the evaluation team and project decision makers. The duration is a few weeks.
- a first evaluation phase. This phase lasts 1-2 days. It involves the evaluation team and the project decision makers. The ATAM process is presented, then business drivers are presented, then the architecture is presented, approaches are identified, and utility trees are identified.
- a evaluation phase which takes place 1 to 3 weeks later. This phase lasts 2 days. It involves architect stakeholders. In this phase, brainstorming and prioritisation of scenarios take place.
- a follow-up phase where the evaluation team provides a final report.

A lightweight version of ATAM is also available (4 to 6 hours).

A2.4 CBAM

CBAM (Cost Benefit Analysis Method) takes place after ATAM. The objective is to maximise the difference between the benefit derived from system and the cost of implementing the design as showed in Figure 30.

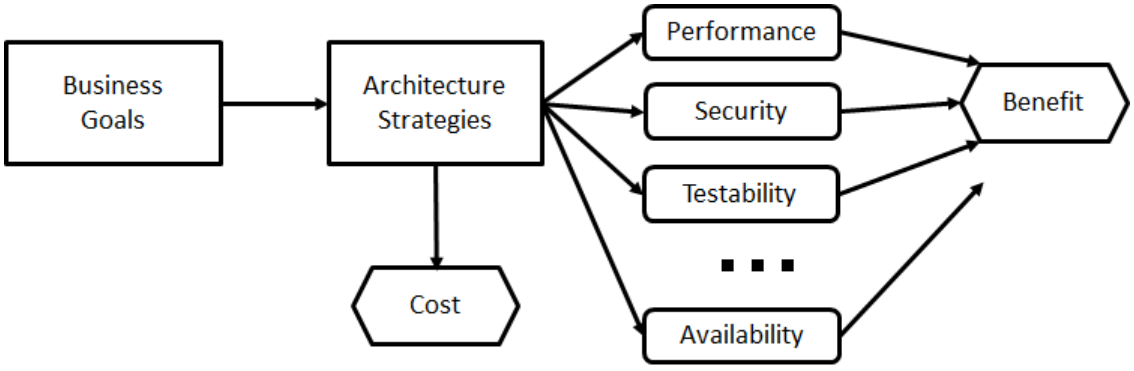

Figure 30: Cost VS Benefit

Basically the objective is to “measure” the utility of a tactic (for performance, security, testability, availability and so forth…). This necessitates the definition of a ***utility-response***

curve. Such curves are in general different from one tactic to another. They also depend on the context, i.e. they will change from one company to another. The approach in general is to vary the values of the responses (e.g. a and b in Figure 31) and to "agree" on a utility value (from 0 to 100). For instance a 99.99 percent availability could have an utility of 90/100 while a 90 percent availability could a have a utility of 10.

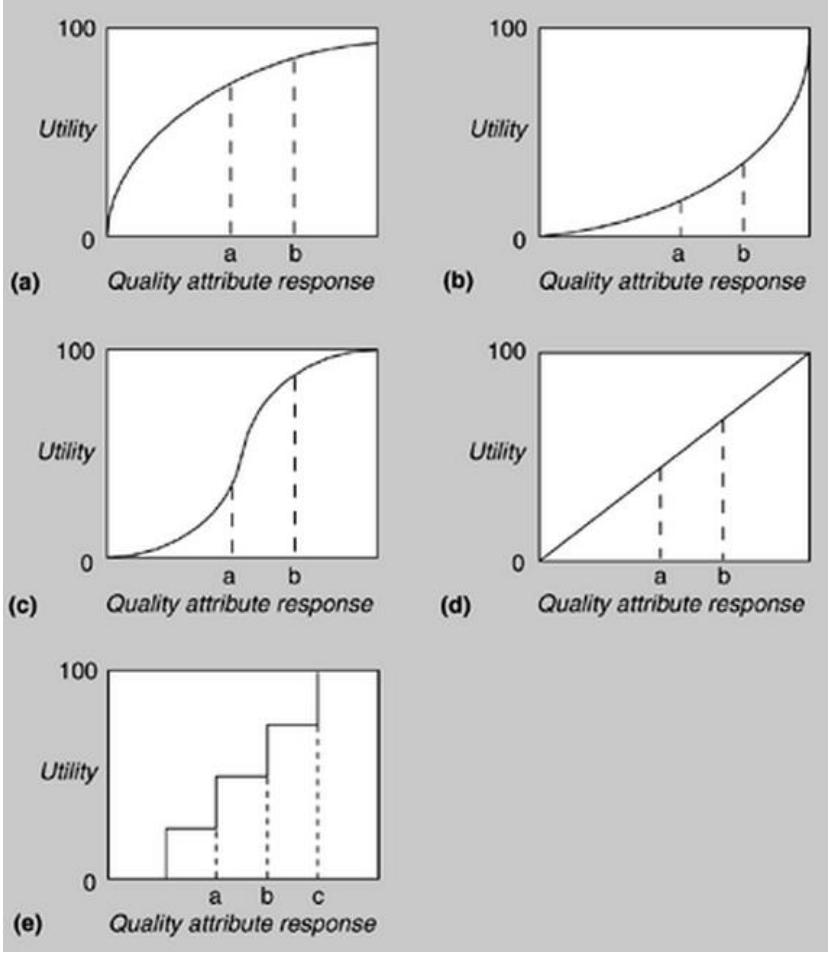

Figure 31: Utility-Response Curves (from Ref 1)

CBAM uses the following metrics:

- Benefit: $\mathbf{B_i}$, defined as follows:
  - i denotes a **strategy i**.
  - Strategy I is described through **j scenarios**.
  - Each scenario I receives a **weight $\mathbf{W_j}$**
  - $\mathbf{b_{ij}}$ is the change in utility caused by scenario j using a utility-response curve ($\mathbf{b_{ij}} = U_{expected} - U_{current}$)
  - $\mathbf{B_i}$ is the sum of all changes taking into account the weight : $\mathbf{B_i = \Sigma_j(b_{ij} \times W_j)}$
- Value for cost, **VFC**, defined as follows:
  - $\mathbf{C_i}$ is the cost of implementing architecture Strategy i
  - VFC is the ratio benefit/cost ($VFC = B_i / C_i$)

A3 Evaluation using CBAM

It is important to understand the following points in CBAM: Utility curves and weights are based on ***heuristics which depend on corporate decisions and knowledge. The overall quality and accuracy of CBAM the measure therefore depends on a good understanding of how these values are assigned.***

Here is the proposed practice proposed in reference 1:

- Utility-response curves are obtained by providing at least the following four values:
  - The best case quality-attribute level which receives value 100. For instance a response time of 0.1 second receives value 100.
  - The worst case quality-attribute level which receives value 0.
  - The current quality-attribute level
  - The desired quality-attribute level
- Weights of scenarios are determined as follows. The N scenarios are prioritized (from 1 to N). Each stakeholder (e.g. project manager, business manager, developer…) provides a priority list. The weight is the sum.
- Costs values are decided in the organization, e.g. as a scale (e.g. the cost of A is X, while the cost of B is 1.5X)
- The evaluation work includes the following phases:
- Step 1: collate scenarios and prioritise them according to business goals (high, medium, low). Select the top third for further consideration.
- Step 2: refine scenarios. In this step, worst case, current, desired, best case quality attributes level are determined
- Step 3: prioritise scenarios. Each stakeholder receives 100 votes. Choose the top 50 percent. Assign weight of 1.0 to highest rated scenario. Assign related weight to others.
- Step 4: assign utility-response curve for step 3 scenarios
- Step 5: identify architectural strategies and associated scenarios. Determine their expected QA response level
- Step 6: Determine the utility of the expected QA response levels by interpolation
- Step 7: Calculate total benefit obtained from an architectural strategy
- Step 8: Select architectural strategy based on VFC (compatible with cost and schedule constraints)
- Step 9: confirm results with intuition

Here is an example (again from reference 1), an earth observing system (constellation of NASA satellite):

Scenario	Scenario Description
1	Reduce data distribution failures that result in hung distribution requests requiring manual intervention.
2	Reduce data distribution failures that result in lost distribution requests.
3	Reduce the number of orders that fail on the order submission process.
4	Reduce order failures that result in hung orders that require manual intervention.
5	Reduce order failures that result in lost orders.
6	There is no good method of tracking ECSGuest failed/canceled orders without much manual intervention (e.g., spreadsheets).
7	Users need more information on why their orders for data failed.
8	Because of limitations, there is a need to artificially limit the size and number of orders.
9	Small orders result in too many notifications to users.
10	The system should process a 50-GB user request in one day, and a 1-TB user request in one week.

Figure 32: Step 1. Collate scenarios

Scenario	Worst	Current	Desired	Best
1	10% hung	5% hung	1% hung	0% hung
2	> 5% lost	< 1% lost	0% lost	0% lost
3	10% fail	5% fail	1% fail	0% fail
4	10% hung	5% hung	1% hung	0% hung
5	10% lost	< 1% lost	0% lost	0% lost
6	50% need help	25% need help	0% need help	0% need help
7	10% get information	50% get information	100% get information	100% get information
8	50% limited	30% limited	0% limited	0% limited
9	1/granule	1/granule	1/100 granules	1/1,000 granules
10	< 50% meet goal	60% meet goal	80% meet goal	> 90% meet goal

Figure 33: Step 2 Refine scenarios

Scenario	Votes	Worst	Current	Desired	Best
1	10	10% hung	5% hung	1% hung	0% hung
2	15	> 5% lost	< 1% lost	0% lost	0% lost
3	15	10% fail	5% fail	1% fail	0% fail
4	10	10% hung	5% hung	1% hung	0% hung
5	15	10% lost	< 1% lost	0% lost	0% lost
6	10	50% need help	25% need help	0% need help	0% need help
7	5	10% get information	50% get information	100% get information	100% get information
8	5	50% limited	30% limited	0% limited	0% limited
9	10	1/granule	1/granule	1/100 granules	1/1000 granules
10	5	< 50% meet goal	60% meet goal	80% meet goal	> 90% meet goal

Figure 34: Step 3 Prioritise scenarios

Scenario	Votes	Worst	Current	Desired	Best
1	10	10	80	95	100
2	15	0	70	100	100
3	15	25	70	100	100
4	10	10	80	95	100
5	15	0	70	100	100
6	10	0	80	100	100
7	5	10	70	100	100
8	5	0	20	100	100
9	10	50	50	80	90
10	5	0	70	90	100

Figure 35: Step 4 Assign utility

Strategy	Name	Description	Scenarios Affected	Current Response	Expected Response
1	Order persistence on submission	Store an order as soon as it arrives in the system.	3	5% fail	2% Fail
			5	<1% lost	0% lost
			6	25% need help	0% need help
2	Order chunking	Allow operators to partition large orders into multiple small orders.	8	30% limited	15% limited
3	Order bundling	Combine multiple small orders into one large order.	9	1 per granule	1 per 100
			10	60% meet goal	55% meet goal
4	Order segmentation	Allow an operator to skip items that cannot be retrieved due to data quality or availability issues.	4	5% hung	2% hung
5	Order reassignment	Allow an operator to reassign the media type for items in an order.	1	5% hung	2% hung
6	Order retry	Allow an operator to retry an order or items in an order that may have failed due to temporary system or data problems.	4	5% hung	3% hung
7	Forced order completion	Allow an operator to override an item's unavailability due to data quality constraints.	1	5% hung	3% hung
8	Failed order notification	Ensure that users are notified only when part of their order has truly failed and provide detailed status of each item; user notification occurs only if operator okays notification; the operator may edit notification.	6	25% need help	20% need help
			7	50% get information	90% get information
9	Granule level-order tracking	An operator and user can determine the status for each item in their order.	6	25% need help	10% need help
			7	50% get nformation	95% get information
10	Links to user information	An operator can quickly locate a user's contact information. Server will access SDSRV information to determine any data restrictions that might apply and will route orders/order segments to appropriate distribution capabilities, including DDIST, PDS, external subsetters and data processing tools, etc.	7	50% get information	60% get information

Figure 36: Step 5 Architectural Strategies and Determining Expected QA Response Level

Strategy	Strategy	Scenarios Affected	Current Utility	Expected Utility
1	Order persistence on submission	3	70	90
		5	70	100
		6	80	100
2	Order chunking	8	20	60
3	Order bundling	9	50	80
		10	70	65
4	Order segmentation	4	80	90
5	Order reassignment	1	80	92
6	Order retry	4	80	85
7	Forced order completion	1	80	87
8	Failed order notification	6	80	85
		7	70	90
9	Granule level order tracking	6	80	90
		7	70	95
10	Links to user information	7	70	75

Figure 37: Step 6 Utility of Expected QA Response Levels

Strategy	Scenario Affected	Scenario Weight	Raw Architectural Strategy Benefit	Normalized Architectural Strategy Benefit	Total Architectural Strategy Benefit
1	3	15	20	300	
1	5	15	30	450	
1	6	10	20	200	950
2	8	5	40	200	200
3	9	10	30	300	
3	10	5	-5	-25	275
4	4	10	10	100	100
5	1	10	12	120	120
6	4	10	5	50	50
7	1	10	7	70	70
8	6	10	5	50	
8	7	5	20	100	150
9	6	10	10	100	
9	7	5	25	125	225
10	7	5	5	25	25

Figure 38: Step 7 Benefit Obtained from an Architectural Strategy

Strategy	Cost	Total Strategy Benefit	Strategy ROI	Strategy Rank
1	1200	950	0.79	1
2	400	200	0.5	3
3	400	275	0.69	2
4	200	100	0.5	3
5	400	120	0.3	7
6	200	50	0.25	8
7	200	70	0.35	6
8	300	150	0.5	3
9	1000	225	0.22	10
10	100	25	0.25	8

Figure 39: Step8 Select architectural strategy based on VFC

A4 Attack Centred Assessment in Architecture Evaluation

TVRA was presented In D4.1. A toy example was provided based on three attacks:

- Simple EMC . The asset being attacked is a local device in charge of local function in a train (e.g. distance control). The effect of an attack is the failure of the local function. The countermeasure is an attack detection system (local health/attack system) and a subsequent reconfiguration to an secondary node.
- EMC attack combined with CPU Denial of Service (DOS) attack. The asset being attacked is a local device in charge of local function in a train (e.g. distance control), associated with a local secondary node. The effect of an attack is the failure of the local function (both local and local detection node). The countermeasure is an CPU isolation system (e.g. hypervisor).
- EMC attack combined with CPU and network DOS attack. The asset being attacked is a region consisting of several train. The effect of an attack is the failure of train operation in a region (necessitating all trains in the region to stop). The countermeasure is a global attack detection system (i.e. global health/attack manager) and a subsequent reconfiguration to isolate a local region

We suggest to integrate the attacks scenarios as quality attributes scenarios.

A5 Applying ATAM to SECRET

Two types of scenarios were identified:

- EMC based scenario attacks using the architecture description (see below).
- Associated scenarios (maintainability, survivability, interoperability…).

The scenarios will include unknown X (in bold italic) whenever appropriate.

A5.1 List of ASR Train Level

Level 1	Quality attribute	Security
Level 2	Attribute	Resilience

	refinement	
Level 3	ASR	**Train level** Sensor detects EMC attack ***pattern X1*** and reports intrusion within 10 seconds to acquisition system which reports to on-board HAM
	Business value	High
	Impact on architecture	Medium

Level 1	Quality attribute	Security
Level 2	Attribute refinement	Resilience
Level 3	ASR	**Train level** Acquisition system detects sensors error behaviors that might have been caused by multiple EMC attacks (***pattern X2***) which reports to on-board HAM
	Business value	High
	Impact on architecture	Medium

Level 1	Quality attribute	Security
Level 2	Attribute refinement	Resilience
Level 3	ASR	**Train level** Acquisition system does not behave properly with on-board HAM which detects a possible attack (***pattern X3***) which reports to on-board HAM
	Business value	High
	Impact on architecture	Medium

Level 1	Quality attribute	Security
Level 2	Attribute refinement	Resilience
Level 3	ASR	**Train level** Train communication system channel no longer operational because of EMC attack (***pattern X4***). MCS selects another communication channel within 1 second
	Business value	High

	Impact on architecture	Medium

A5.2 List of ASR Track Level

Level 1	Quality attribute	Security
Level 2	Attribute refinement	Resilience
Level 3	ASR	**Track level** Sensor detects EMC attack ***pattern X1*** and reports intrusion within 10 seconds to acquisition system which reports to on-board HAM
	Business value	High
	Impact on architecture	Medium

Level 1	Quality attribute	Security
Level 2	Attribute refinement	Resilience
Level 3	ASR	**Track level** Acquisition system detects sensors error behaviors that might have been caused by multiple EMC attacks (***pattern X2***) which reports to on-board HAM
	Business value	High
	Impact on architecture	Medium

Level 1	Quality attribute	Security
Level 2	Attribute refinement	Resilience
Level 3	ASR	**Track level** Acquisition system does not behave properly with on-board HAM which detects a possible attack (***pattern X3***) which reports to on-board HAM
	Business value	High
	Impact on architecture	Medium

A6 Towards SECRET Guidelines for Architecture Resiliency Evaluation

For the following reasons:

- SECRET resiliency architecture must be generic
- New attack patterns can be found out in the future

We believe that SECRET guidelines for architecture resiliency evaluation must be provided. The objective of WP4 is therefore described in Figure 40: From SECRET Architecture to SECRET Recommendations:

- D4.2 describes SECRET resilient architecture
- D4.5 provides a proof of concept validation
- D5.5 includes an ATAM and CBMA-based evaluation. This yields a technical recommendation document that includes guidelines for evaluation of instantiated architecture.

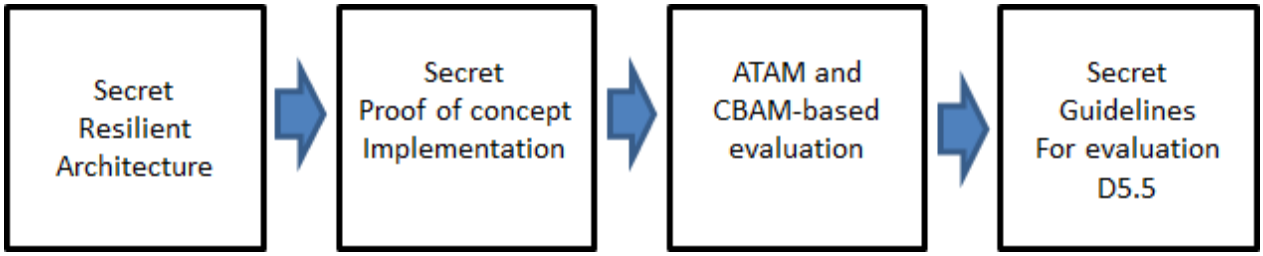

Figure 40: From SECRET Architecture to SECRET Recommendations

D5.5 will be used in the future as follows

- It includes a methodology description
- The instantiated architecture integrates SECRET architecture features (and possibly future extensions).
- An evaluation of the instantiated architecture is applied.

Figure 41: How to use SECRET Recommendations