Pilot SHIELD

pilot embedded Systems archItecturE for multi-Layer Dependable solutions

SEVEN FRAMEWORK PROGRAMME

Project no: 100204

p-SHIELD

pilot embedded Systems architecture for multi-Layer Dependable solutions

Instrument type: Capability Project

Priority name: Embedded Systems /Rail Transportation Scenarios

Pshield Pilot Demonstrators

For the pSHIELD-project

Deliverable D6.3

Partners contributed to the work:

ASTS, Italy
CWIN, Norway
SELEX-ELSAG, Italy
SESM, Italy
UNIROMA1, Italy

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2012)		
Dissemination Level		
PU	Public	
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)	X

Document Authors and Approvals			
Authors		Date	Signature
Name	Company		
Mariana Esposito	ASTS		
Francesco Flammini	ASTS		
Alfio Pappalardo	ASTS		
Przemyslaw Osocha	SESM		
João Cunha	SESM		
Fabio Giovagnini	SESM		
Andrea Fiaschetti	UNIROMA1		
Vincenzo Suraci	UNIROMA1		
Elisabetta Campaiola	SELEX-ELSAG		
Zahid Iqbal	CWIN		
Reviewed by			
Name	Company		
Name	IRCTR		
Approved by			
Name	Company		
A.Di Marzo	SESM		

Modification History		
Issue	Date	Description
Draft A	26 /10/2011	First ToC proposal for comments
Draft B	29/11/2011	First Issue for comments

Contents

Figures

Tables

This Page is intentionally left blank

1 Executive summary

The aim of this task is to validate the proposed architecture in an industrial relevant application scenario: m*onitoring of freight trains transporting hazardous material*. The goal of this use-case is to demonstrate how information from sensors in ad-hoc environments will require actions depending on the constellations.

As witnessed by the results of risk assessment and accidents happened in recent past, this is an important problem to be addressed in the context of critical infrastructure protection and railway security.

Therefore, in order to provide resiliency against both random and malicious threats the use case focuses on:

(i) providing SPD functionalities to off-the-shelf smart-sensors, measuring environmental parameters (e.g temperature, vibration level, etc.)

(ii) developing a monitoring application which detects abnormal operating conditions and testing the overall system for SPD functionalities like node authentication, checksum and cryptography

In pSHIELD focus will be on access to the services provided through the sensors. One application might be the run-time information for drivers of hazardous waste on where and how material has to be placed in order to avoid collocation of reactive components.

In this specific use case, the following requirement has to be fulfilled:

- Secure handling of the critical information of the transported material;
- Secure and dependable monitoring of the transport.

The structure and content of the document are the following:

- Chapter 1 – Purpose of the document and its structure
- Chapter 2 – Brief introduction on the document and its contents
- Chapter 3 – Taxonomy
- Chapter 4 – Description of application scenario in which the Pshiedl concepts will be demonstrate
- Chapter 5 – Power Node prototype
- Chapter 6 – Radio network prototype
- Chapter 7 – Semantic model prototype and demonstration of composability
- Chapter 8 – Description of platform for the security integration
- Chapter 9 – Hardware and software implementation of the prototype

2 Introduction

The aim of deliverable 6.3 is to present the multi-technology demonstrators in order to prove the SPD (Security, Privacy and Dependability) concepts developed in WP 2-5. There are several pilot demonstrators for different objectives:

- A demonstration of composability of SPD functionality;
- Security integration across heterogeneous platforms;
- Hardware prototypical implementations;
- SPD levels.

As pSHIELD is a pilot project, prototypical solutions were implemented. These contain:

- FPGA Power Node involves modular system reconfiguration, self-dependability at node layer hardware and software security and privacy service provider and management of power sources.
- Cognitive radio network prototype which includes reconfigurable radio components with waveform Tx parameters, sensing mechanisms to acquire awareness about resources, cognitive algorithms elaborating available resources and embedded platform adaptation for validation of algorithms.
- pSHIELD semantic model prototype (ontology) and middleware prototype.
- Monitoring trains with WSN which identifies requirements of real-world applications and SPD functions in an integrated embedded sensor testbed, opens for SPD metrics based composability.

3 Terms and definitions

Availability	Readiness for correct service. The correct service is defined as delivered system behaviour that is within the error tolerance boundary.
Common Criteria	The Common Criteria for Information Technology Security Evaluation (abbreviated as Common Criteria or CC) is an international standard (ISO/IEC 15408) for computer security certification. It is currently in version 3.1. Common Criteria is a framework in which computer system users can specify their security functional and assurance requirements, vendors can then implement and/or make claims about the security attributes of their products, and testing laboratories can evaluate the products to determine if they actually meet the claims. In other words, Common Criteria provides assurance that the process of specification, implementation and evaluation of a computer security product has been conducted in a rigorous and standard manner.
Confidentiality	Property that data or information is not made available to unauthorized persons or processes.
Control system	A system that uses a regulator to control its behaviour
COPS	The GCommon Open Policy Service (COPS) Protocol is part of the internet protocol suite as defined by the IETF's RFC 2748. COPS specifies a simple client/server model for supporting policy control over Quality of Service

	(QoS) signalling protocols. Policies are stored on servers, and acted upon by Policy Decision Points (PDP), and are enforced on clients, also known as Policy Enforcement Points (PEP). There are two models of COPS: The Outsourcing Model and the Provisioning Model, considered from the view of the client or PEP.
Fault	Normally the hypothesized cause of an error is called fault [2]. It can be internal or external to a system. An error is defined as the part of the total state of a system that may lead to subsequent service failure. Observing that many errors do not reach a system's external state and cause a failure, Avizienis et al. [2] have defined active faults that lead to error and dormant faults that are not manifested externally.
Integrity	Absence of malicious external disturbance that makes a system output off its desired service.
OWL	The Web Ontology Language OWL is a semantic markup language for publishing and sharing ontologies on the World Wide Web. OWL is developed as a vocabulary extension of RDF (the Resource Description Framework) and is derived from the DAML+OIL Web Ontology Language. This document contains a structured informal description of the full set of OWL language constructs and is meant to serve as a reference for OWL users who want to construct OWL ontologies.
PAP	The Policy Administration Point (PAP) component is used by the administrators of an enterprise to define fine-grained authorization policies for the enterprise users who need to access various software components to carry out their day-to-day tasks.
PDP	A Policy Decision Point (PDP) is the technical entity capable of taking a Policy Decision based on a set of policies that an administrative domain has defined. The administrative domain does the translation of policies defined by business rules into technical policy rules that is understood by the PDP. The PDP stores the technical policy rules in a repository.
PIP	The Policy Information Point (PIP) is a repository of information to help make the access decision. It could be a database of device IDs, a user directory such as the *Lightweight Directory Access Protocol* (LDAP), a *one-time password* (OTP) token server, or any other system that houses data relevant to a device or user access request.
Privacy	The right of an entity (normally a person), acting in its own behalf, to determine the degree to which it will interact with its environment, including the degree to which the entity is willing to share information about itself with others.
Reliability	Continuity of correct service even under a disturbance.
Safety	Absence of catastrophic consequences on the users and the environment.
SNMP	Simple Network Management Protocol (SNMP) is an "Internet-standard protocol for managing devices on IP networks. Devices that typically support SNMP include routers, switches, servers, workstations, printers, modem racks, and more."It is used mostly in network management systems to monitor network-attached devices for conditions that warrant administrative attention. SNMP is a component of the Internet Protocol Suite as defined by the Internet Engineering Task Force (IETF).

System	A composite constructed from functional components. The interaction of these components may exhibit new features/functions that none of the composite components possess individually.
XML	Extensible Markup Language (XML) is a set of rules for encoding documents in machine-readable form, produced by the W3C. The design goals of XML emphasize simplicity, generality, and usability over the Internet. It is a textual data format with strong support via Unicode for the languages of the world. Although the design of XML focuses on documents, it is widely used for the representation of arbitrary data structures, for example in web services.

4 General description of application scenario

4.1 Context

Recent interest to development in the context of embedded systems, have encouraged the conceiving of new integrated systems for the infrastructure monitoring. Overall railway infrastructure has been developed rapidly in the last two decades, including its communication systems. In the past wired communication systems were used for signalling and data communication in the railway industry. Recently wireless communication systems have emerged as alternatives to substitute wired systems in the railway industry. Today they can be used as protection systems to monitor asset within a railway infrastructure, in order to assure a reliable, safe and secure operation.

At the same time, the need for facing common threats associated in particular to freight transportation is increasing for many railway transportation operators. As witnessed by the results of risk assessment and accidents happened in recent past, this is an important problem to be addressed in the context of critical infrastructure protection and railway security. Therefore, the detection of abnormal operating or environmental conditions on board of vehicles as well as threats of burglary represents an example application of great interest for the freight train monitoring.

In case of mobile assets monitoring, train health indicators of interest include vibrations, smoke, tilt ambient temperature, and humidity in wagons. For instance temperature monitoring safeguards wagons against fire outbreak, while vibration and tilt monitoring proactively prevents potential accidents, which could be very dangerous in case of transportation of hazardous material. In addition, for security reasons, other important indicators come from the access control devices, in order to prevent cargo thefts and manumissions of the material inside the railcar.

There are several issues and challenges which are addressed by innovative features of the project:

- Since most freight cars are actually unpowered, there is the need to provide a power-aware and power-autonomous system architecture;

- Since a railway is geographically distributed system and cars are mobile entities, there is the need of providing a connection to the central monitoring system through a wireless WAN (Wide Area Network);

- Since the application needs low-cost, easy to install and easy to maintain devices, the system is based on a small number of cheap components including wireless smart sensors not requiring connection cables;

- The overall monitoring system is highly heterogeneous in terms not only of detection technologies but also of embedded computing power and communication facilities. So, the sensors and/or embedded systems can differ in their inner hardware-software architecture and thus in the capacity of providing information security, privacy and dependability.

4.1.1 Transportation of hazardous material

Hazardous materials can be solid, liquid or gas that can pose an unreasonable health, safety and/or properties risk of individuals or which may cause serious environmental damage, including cost of radioactive materials, flammable, explosive, corrosive, oxidizing, asphyxiating, toxic, infectious or allergenic. Also included are compressed gases and hazardous materials in any other way or with characteristics that make them dangerous under specific conditions (for example, certain substances released flammable gases when put in contact with water). Any material are common use as oils, paints, batteries or limited use (e.g. fertilizers, chemicals). The shipping of hazardous materials can be done in different ways and means of transportation, including several typologies of danger and security issues. The handling of hazardous material includes identification, labeling, packaging, storage, transportation and disposal. Moreover is important the secure handling of environmental conditions of infrastructures.

The use case of reference for pSHIELD project is the monitoring of freight trains transporting hazardous material. As witnessed by the results of risk assessment and accidents happened in recent past (see Section 4.1.3), this is an important problem to be addressed in the context of critical infrastructure protection and railway security. Therefore, the detection of abnormal operating or environmental conditions on board of vehicles as well as threats of burglary represents an example application of great interest for the freight train monitoring.

The main objectives of this application scenario are to validate the technical concepts of SHIELD Security, Privacy and Dependability as a whole. In particular, in this use case, the following requirements have to be fulfilled:

- providing SPD functionalities to off-the-shelf smart-sensors, measuring environmental parameters (e.g. temperature, vibration level, etc.);

- developing a monitoring application which detects abnormal operating conditions and testing the overall system for SPD functionalities like node authentication, checksum and cryptography.

4.1.2 Norms and regulation

The international norms are represented by “UN Recommendations on the transport of Dangerous Goods UN Recommendation on, as “Orange Book”, published by ONU on 1957. The objective is to regulate the circulations of hazardous materials guaranteeing the security of people, environments, and goods during the transportation.
The “Orange Book” contains the common principals of all norms regarding to the transportation’s mode of hazardous materials, at international and European levels. At international levels the main structure, controlled by UN (United Nations), are: United Nations Economic and Social Council (Ecosoc) and The Transport of Dangerous Goods Sub-Committee (TDG Sub-Committee) composed by 27 countries, including Italy. The European Union (EU) guarantees that all countries adopt the ONU’s norms, for traffic of hazardous materials both European both national. This division of two levels (national and European) is critical in particular for the times, often very long, for the adoption of norms at national levels. In Europe the transportation of hazardous materials is subject to a specific directive, applicable to road, rail or sea. Based on this directive the transportation is authorized if:

- Is in compliance with the provisions laid down in agreements RID (Règlement concernant the transport ferroviaire International des marchandises Dangereuses) for the International Carriage of dangerous goods by rail;
- Respects ADR (Agreement Concerning the International Carriage of Dangerous Goods by Road), the European Agreement concerning the International Carriage of Dangerous Goods by Road;
- Observes DNA (Accord Européen relatif au transport international des marchandises Dangereuses par voie de Navigation intérieure), the European Agreement concerning the International Carriage of Dangerous Goods by Inland Waterways.

4.1.3 Threats and accidents happened in the past

The Hazardous Materials Transportation Act (HTMA), the first US law on the transportation of hazardous materials has defined as “hazardous materials incident” an unintentional emission of dangerous substances during the transportation, the loading and unloading, and storage of the material. The PHMSA (Pipeline and Hazardous Materials Safety Administration) informs on accidents occurred after the entry into force HTM; in the figure below are represented the statistics of accidents from 2001 to 2010 divided into year, transportation means.

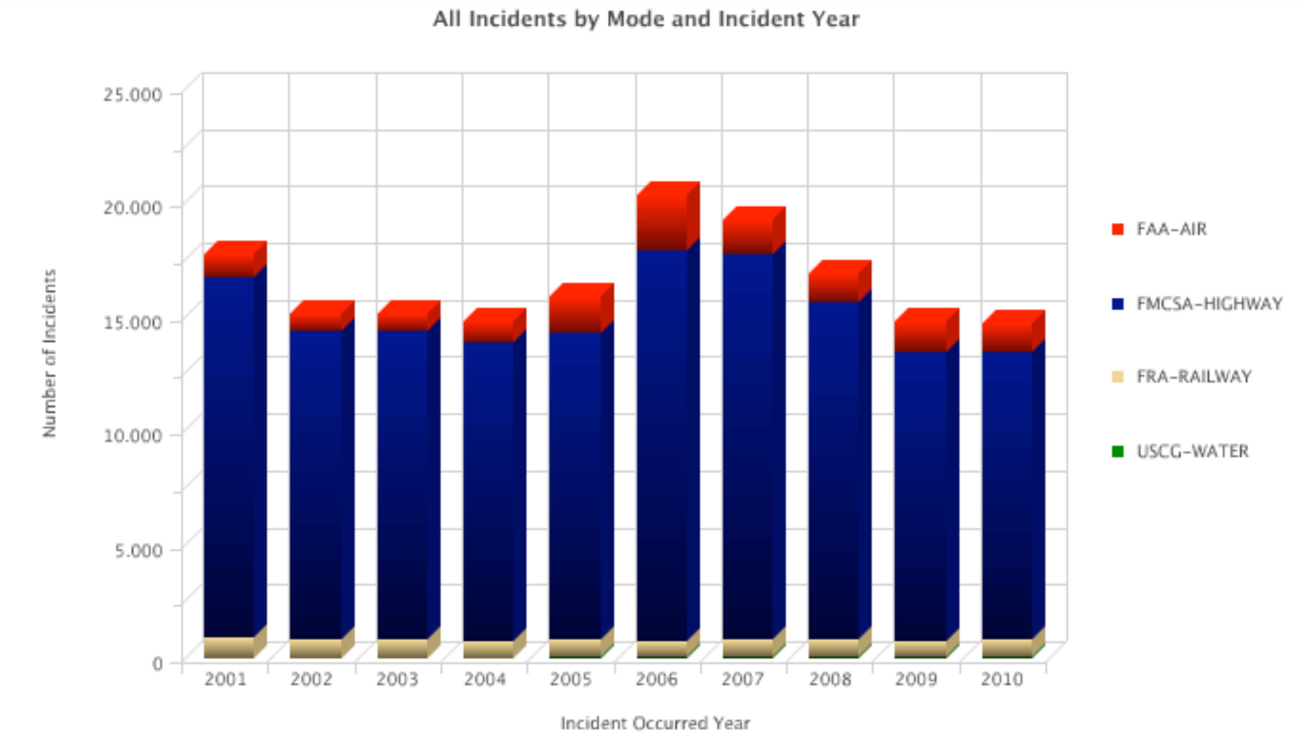


Figure 1 Statistics of accidents

Most accidents occur during transport by highway (about 87.1%), while air accidents account for 7.9%, the railway accidents are approximately 4.7% and 0.3% are marine. Most of these accidents are in the transport of petrol tankers because of the large number of implementing it. Although the majority of reported incidents hasn’t serious consequences, some can cause victims and have, thus, significant economic and environmental impacts. For example, in 2005, the derailment of a tank car carrying chlorine (poisonous gas) caused the death of nine people.

In 2010, however, the derailment of a wagon containing flammable liquids caused a fire, forcing the authorities to evacuate about 700 people.

4.1.4 Monitoring issue

To increase the safe transport of dangerous goods is necessary to monitor the following components:

- Monitoring of wagons (speed, acceleration, vibration, inclination). Through these data is possible, for example, to detect collisions and derailments and analyze the behavior of the driver (also noting any breaches that may compromise the security of cargo, such as exceeding speed limits on the way);
- Monitoring of the goods transported. If, for example, a liquid is carried, you can obtain information about pressure, the liquid level (information that allows the detection of losses) and temperature (information needed in case of flammable goods);
- GPS Location.

The monitoring of the above physical quantities will be made through a network of sensors (See section 12.1 of D6.1 for the development issues).

5 FPGA Power Node Prototype

Dependability is one of the most important aspects for many ESs' markets. Usage of hardware redundancy is frequent way to reach high dependability. But HW redundancy increases system’s cost drastically. Another solution may be selected: usage of FPGA based ESs, that are intrinsically redundant.

The concept of runtime reconfiguration is applicable to FPGAs and represents the capability to modify or change the functionality configuration of the device during fault or normal operation, through either hardware or software changes. That capability can be specialized in different way in order to reduce component count, power consumption, reusing, fault tolerance, etc. increasing the global SPD capabilities of the system. The goal of this project was to develop a new approach for FPGA runtime reconfiguration that is capable to increase the nodes dependability.

To use these FPGA specific capabilities in the pSHIELD architecture, the SPD Power Node prototype was built. The developed SPD Power Node framework benefits from FPGA dynamic partial reconfiguration functionality, increasing dependability, and is accompanied by other SotA solutions increasing security, privacy and also dependability of the system. Developed framework allows to speed up time to market for the new pSHIELD compliant nodes. In the following chapters functionalities of pSHIELD SPD Power Node prototype are presented.

5.1 Demonstration Context

In order to demonstrate the capabilities of the proposed pSHIELD SPD Power Node Layer Architecture, a case study has been implemented.

The scenario consists on the use of FSK modulation as described in deliverable D3.3 to transmit data between intrusion detection sensors placed in different cars of a freight train, to an SPD Power Node, which in turn processes the signals and sends information to a control center through the pSHIELD network.

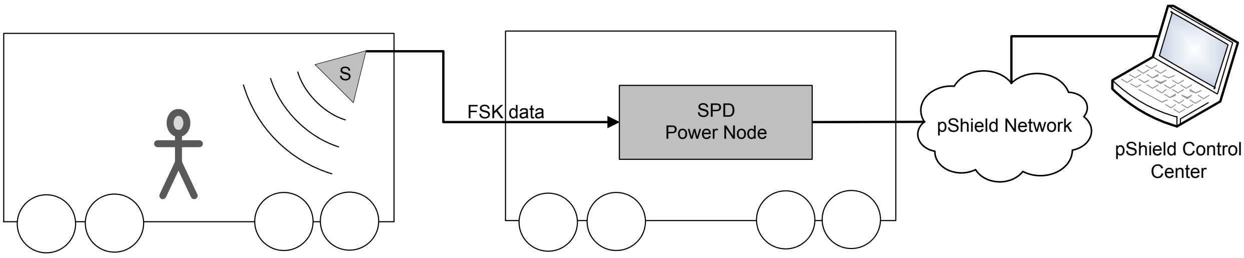


Figure 2 Power Node demonstrator context

The intrusion detection systems are embedded devices which include a remote proximity sensor, continuously measuring a distance to a nearby object, a data encryptor, and an FSK modulator. Each device modulates the signal with a different carrier, and transmits it to the power node.

The Power Node receives the signals, demodulates them, decrypts, processes the data and sends to a control center through the pSHIELD Network.

The Control Center is a remote device, which could be a personal computer, tablet or mobile phone equipped with a web browser, able to visualize data and act upon.

5.2 SPD Power Node capabilities demonstration

The SPD Power Node capabilities that are demonstrated through this use-case scenario are basically the following:

- **Dependability**, by detecting errors in the demodulator, and tolerating them, through FPGA partial reconfiguration. After a fault being injected in the FPGA, affecting the demodulator, an error is detected and the FPGA is partially reprogrammed.
- **Security**, by receiving encrypted data and being able to decrypt it.
- **Self-Reconfiguration**, by detecting when a different carrier is being used in the FM signal, and reconfiguring the FPGA for adapting to a new carrier. This situation is forced by switching in run-time the carrier that is being used by the modulator to produce the FM signal.
- **Metrics**, by collecting and providing data such as the number of messages received, errors detected, etc.
- **Composability**, by providing discovery and composability information, such as the identification of the modules and its characteristics, that build-up the SPD Power Node.
- **High performance**, by demodulating and decrypting in real-time all the received data, and applying an algorithm to alert for an intrusion.
- **Legacy component integration** in pSHIELD, by providing SPD functionalities to a legacy FM Demodulator, such as the collection of Metrics and the provision of composability information.

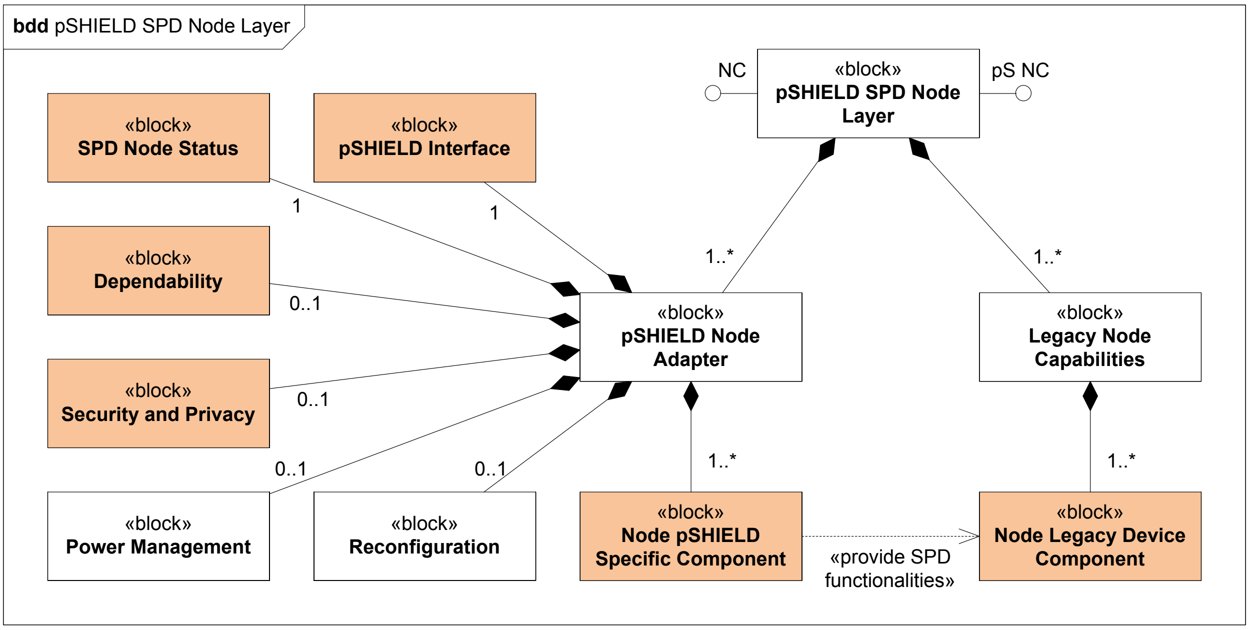

bdd pSHIELD SPD Node Layer
«block» SPD Node Status
«block» pSHIELD Interface
«block» Dependability
«block» Security and Privacy
«block» Power Management
«block» Reconfiguration
NC
«block» pSHIELD SPD Node Layer
pS NC
«block» pSHIELD Node Adapter
«block» Legacy Node Capabilities
«block» Node pSHIELD Specific Component
«block» Node Legacy Device Component
«provide SPD functionalities»
1
1
0..1
0..1
0..1
0..1
1..*
1..*
1..*
1..*

Figure 3 pSHIELD SPD Node Layer block diagram

5.3 SPD Power Node status and metrics

The status and metrics collected and provided by the SPD node allows the overlay to decide on system composition. The following metrics are collected:

- SPD Power Node identification and status
- FM signal generators identification and status
- Demodulator identification and status, including carrier frequency
- Decryptor identification and status
- Received data samples from the signal generator, with statistics:
  - Sample ID, timestamp
  - Number of valid and invalid samples
- Decryption errors (could be intrusion attempts in the connection between the FM signal generator and the SPD Power Node)
- Demodulation errors
- Self-reconfiguration (software, partial FPGA reconfiguration, or full FPGA reconfiguration)
- Error recovery (software, partial FPGA reconfiguration, or full FPGA reconfiguration)

5.4 Demonstrator composition

The demonstrator of the power node implementation has been designed and built having in mind the target to ensure a greater level of security and dependability of a system involved in mission critical scenarios.

Today there are a lot on mission critical applications that have two different but equally important expectations: security and dependability by a side; flexibility and scalability by the other side.

The above mentioned targets are today profitable achieved using FPGA based systems. Furthermore today the hardware technologies allows to achieve really great performance levels using the most recent FPGA devices.

The focus of the demonstrator is to show how the system implements an increased level of security and dependability using the Dynamic Partial Reconfiguration (DPR) features supplied by Xilinx and announced by Altera.

The name of the demonstrator is FSK-Demodulator. In fact it implements an FSK demodulator. The demodulator receives a digital signal being the result from an analogue to digital conversion and a clock being the sampling clock of the analogue to digital converter. The demodulated signal is decrypted and is sent to a System on Chip (SoC) that analyses the signal and makes it available over the network.

On the other side there is an FSK-Modulator that produces the narrow-band signal, encrypts it and modulates using FSK technology. The produced signal is kept in a digital format to make the demonstrator connection easier. Just an 8 bit wide signal and a clock signal are required.

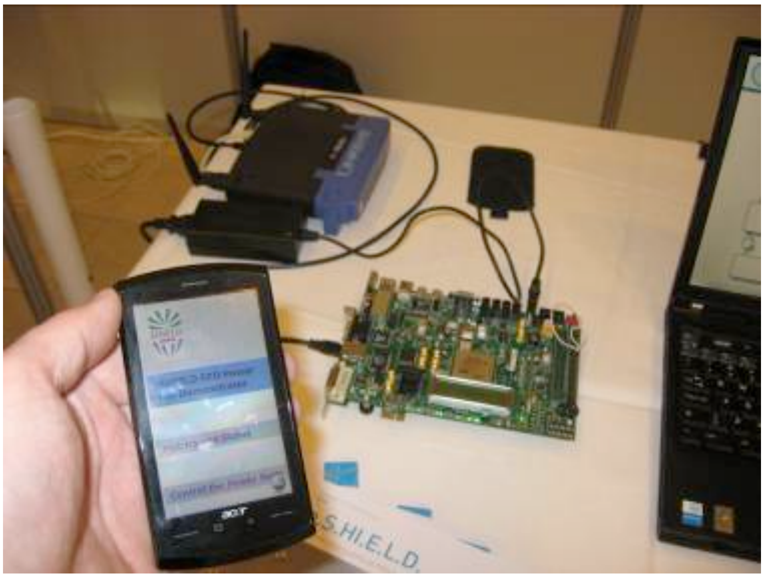


Figure 4 Running FPGA Power Node Prototype

The FM Signal Generator is the FSK-Modulator block. It produces the source signals and sent them through an 8 bit parallel synchronous bus. The <<system>> is the SPD node FSK-Demodulator and the SoC connected to it is responsible to delivery the services of the node to the network. The Fault Injection Trigger is the subsystem able to inject fault into the FSK-Demodulator to change its proper behaviour and this way to simulate failure for testing purposes.

As we have told the <<system>> contains also the SoC able to manage the demodulated and decrypted data to make them available to pSHIELD Network and finally to the control centre being in the demonstrator scenario a PC or a mobile device running a web browser.

Using the Fault Injection Trigger the demonstrator will put the FSK-Demodulator in a failure state so the SoC connected to the FSK-Demodulator will recognize that the FSK-Demodulator is not running properly and will reconfigure it.

Another way of making the FSK-Demodulator not running properly is change of the carrier frequency by FSK-Modulator. In this condition the demodulated data are not plausible anymore and again the SoC will reconfigure a different carrier FSK-Demodulator into the Partial Reconfigurable Area of the FPGA implementing “de facto” a new and different hardware system.

The FSK-Modulator is an Altera NON Partial reconfigurable FPGA. For switching the different carriers it will use a multiplexed output technique: the output lines are shared between different physical instances of the modulators and a sort of three-state logic will route the first carrier modulator outputs rather than the second. This is a classical technique that becomes not usable if the dimension and / or the number of the different multiplexed modules grows up to a big number.

Looking in deep to this second reconfiguration scenario it is possible to see an implicit security and privacy of the system, being the couple Modulator – Demodulator dynamically reconfigurable. It is very important to note that such a kind of reconfiguration is at hardware level, so using the DPR the design has the level of Security and Dependability of a generic reconfigurable system, but the performances of a hardware designed and fixed system.

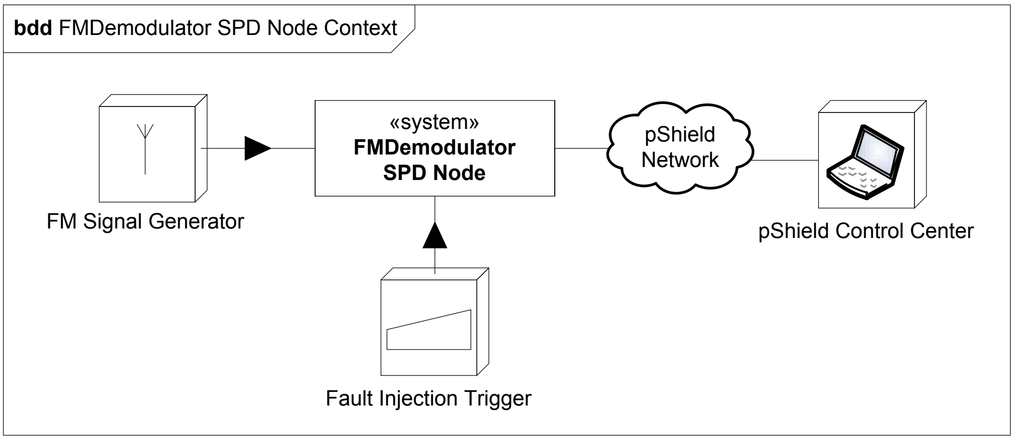


Figure 5 Synthetic scheme of the demonstrator

- A FSK modulator, implemented in an Altera **FPGA** board, with:
  - A **proximity sensor** simulator, generating data
  - A **data encryptor**
  - A **FSK modulator**, modulating the encrypted data into a FSK signal
- A parallel **8 bit wide data** bus with the synchronization clock line between the signal generator and the Power Node
- A **SPD Power Node**, built within a Xilinx Virtex-5 **FPGA**, with:
  - A **FSK demodulator, a data decryptor and a web server**, presenting the node status, metrics and received data.
  - A **fault-injector**, activated by a pushbutton, able to inject a fault into this FPGA
- A **Control Center**, which is a PC or mobile device, with a web browser
- An **Ethernet** connection between the SPD Power Node and the Control Center

5.5 Demonstration scenarios

Several scenarios have been designed in order to demonstrate the Power Node capabilities.

5.5.1 Node discovery and legacy component integration

This first scenario demonstrates the basic functioning of the SPD Power Node and the control center. It also demonstrates how a node can provide information for discovery and how a legacy device was integrated in the SPD Power Node context:

1. The SPD Power Node has a web server running, with a web page providing information about node identification, capabilities and status from the device.

2. The Control Center accesses this web page through a web browser.

3. The Control Center displays the full data received from the SPD Power Node, including the one related to the FSK demodulator (a legacy device component).

5.5.2 Metrics and high performance

The next scenario demonstrated the ability of the SPD Power Node to demodulate and decrypt the received data in real-time.

1. The sensor device periodically generates data simulating a distance to an object. This data is encrypted, modulated and sent to the SPD Power Node.

2. The SPD Power Node demodulates the signal, decrypts it, processes it to detect intrusions and stores on a local database (this requires high performance).

3. In the meanwhile metrics data are collected and stored on local database of the same node.

4. The Control Center requests and displays SPD Power Node status and metrics, including distance to object, continuously updated ever 5 seconds (frequency may be adjusted as needed).

5. The user requests for a SPD Power Node reset through the Control Center interface, and verifies that all metrics have also been reset.

5.5.3 Self-reconfiguration

In this scenario the SPD Power Node demonstrates its ability to self-reconfiguration for adapting to a change in the environment.

1. The FSK modulator from sensor device switches to a different carrier (through a simple pushbutton).

2. The SPD Power Node detects a demodulation error, and the demodulator is automatically reconfigured to this new carrier, by a partial reconfiguration of the FPGA.

3. On the Control Center, the displayed sensor data is still valid. The metrics reveal that a self-reconfiguration has been performed.

4. The Control Center operator then requests for another reconfiguration, to the other carrier. The SPD Power Node reconfigures to the other configuration, and then goes back to the previous one, as it does not match the carrier of the modulated signal. These switches are possible to notice from the status and metrics information.

5.5.4 Dependability

A fault-injector is being used in order to inject a fault into the FPGA, and then allow the SPD Power Node device to recover from itself.

1. A fault is injected in the demodulator area in the FPGA, by pressing a pushbutton.

2. An error is detected and recovered, through software reboot and hardware (FPGA reconfiguration) recovery.

3. Correct data is still presented to the Control Center. The metrics and status reveal that an error has occurred, and recovery was successful.

5.5.5 Security

This last scenario demonstrates how encryption is being used for a secure connection between the sensor devices and the SPD Power Node.

1. The Control Center operator requests the decryption in the SPD Power Node to be turned off.

2. The SPD Node is not able to understand the received signal (since it is encrypted) and then raises an intrusion detected alarm.

3. The Control Center displays invalid sensor data and the intrusion detection. The status and metrics reveal that decryption is off.

5.5.6 Communication interface with the Control Center

In the table 1, there are collected data that are provided to the Control Center.

Table 1 Control Center User Interface provided data

Identification		Responses	
ID	SPD_PN_01	Distance	123
Name	FSK Demodulator 1	Intrusion	Not Detected
Status		Metrics	
Node status	running		
SPD level	2		
Watchdog timer	on	Errors detected	0
		Error Recovery	1
		Total Failures	1
Decryption	On	Decryption Errors	1
		Decrypted frames	20
Commands	On	Reconfiguration Requests	2
		Reconfiguration Failures	1
		Service Requests	30
		Not recognized Requests	3
Demodulation	On	Samples / s	32000
		Demodulator Faults	1
		Demodulator Errors	5
		Signal Bandwidth	NaN
Capabilities			
CPU model		Error detection	Watchdog timer
CPU frequency		Error recovery	SW restart
RAM size		Error recovery	FPGA reconfiguration

The communication is based on HTTP protocol and supported by web server build-in into the Power Node.

6 Cognitive Radio Network Prototype

This demonstrator concerns the system architecture of Dynamic Spectrum Management (DSM)-capable cognitive radio networks (Figure 6). With the growing demand for different wireless systems, the radio spectrum tends to shrink while it is not utilized in most of the cases. Therefore, dynamic spectrum management, which tries to improve spectrum access and usage efficiency, is emerging as an exciting new possibility enabled by advances in radio technologies and opening up of regulatory processes. Dynamic spectrum management is one of the important applications of cognitive radio or even a more limited form of reconfigurable, adaptive, frequency agile radio. Nowadays, unfortunately, the researchers give more effort on cognitive radio networks, which are too broad, often amorphous notions. A real danger exists that the exotic and complex scenarios of cognitive radio operations that are impractical for long time to come but are rich in research problems capture the imagination of researchers.

With this demonstrator, on the contrary, we will try to present some more practical considerations about cognitive radios with concerning DSM-capable architectures. In particular, we will describe in a simpler way the architecture of a single cognitive radio terminal and in a much deeper way the architecture of DSM-capable systems, with particular emphasis on DSM capable Tactical Communication Systems (TCS). The necessity of DSM capabilities for the current tactical military systems is investigated, and, in order to incorporate such capabilities in a wireless system, our main focus will be on the management of radio parameters (e.g., transmit-power, carrier-frequency, and modulation strategy), on the network topology, on the network performances and on all issues related to the security of the system, taking account of the possible interaction even with smart jammers.

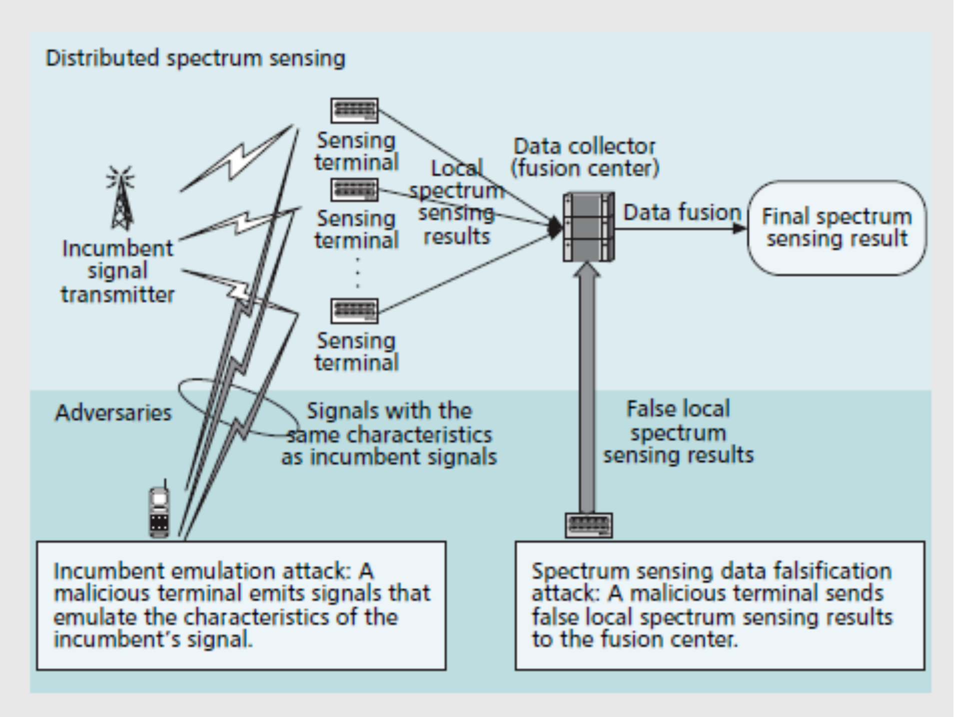

Distributed spectrum sensing
Sensing terminal
Sensing terminal
Sensing terminal
Local spectrum sensing results
Data collector (fusion center)
Data fusion
Final spectrum sensing result
Incumbent signal transmitter
Adversaries
Signals with the same characteristics as incumbent signals
False local spectrum sensing results
Incumbent emulation attack: A malicious terminal emits signals that emulate the characteristics of the incumbent's signal.
Spectrum sensing data falsification attack: A malicious terminal sends false local spectrum sensing results to the fusion center.

Figure 6 Distributed spectrum sensing

A possible threat to spectrum sensing is the data falsification attack in the group of radio spectrum sensing. Here, a rogue secondary user (or a group of rogue secondary users) may send false local spectrum sensing information, and hence may adversely affect the group detection outcome. A potential two-layer defence is proposed against spectrum sensing data falsification; i.e., a) authentication at the first layer to prevent replay or false data coming from outside the network, and b) a data fusion scheme that is robust against spectrum sensing data falsification.

The anti-jamming capability of a DSM-capable cognitive radio network is another related important topic. As it is well known, a jammer is a device that intentionally generates RF signals to disrupt the normal operation of a communication system. There are many different types of jammers according to their several applications, ranging from disrupting the receiving of signals at the target receivers, to more sophisticated, deceiving the targets into accepting false information. Traditionally, the biggest constraint on a jammer is the power constraint. If the jammer can send jamming signals with unlimited power, no communication system will survive. Therefore, in studying countermeasures one has to consider only jammers with limited transmission power. Unfortunately, the technology allowing the design of cognitive radio terminals allows also the construction of the so-called smart jammers. They can be highly sophisticated and adaptive to the environment. Specifically, smart jammers can consist of three major components:

- a spectrum sensor, which senses and locates the target physical channel,
- a spectrum analyzer, which analyzes the sensed spectrum data with the help of prior knowledge about the target signal and consequently devises an action, and
- a radio transmitter, which is dedicated to radiating jamming signals.

In this demonstrator, a cognitive radio node software prototype was implemented by means of C/C++ programs running on standard PCs. The so called "cognitive node" is able to sense signal strength of detected mobile entities; by comparing received signal strengths, the cognitive node is able to detect jammers within the monitored environment. Implemented software prototype is able to deal with fixed and mobile jammers; after a jammer is detected, the cooperative mobile entity automatically modifies its operating frequency (Figure 7).

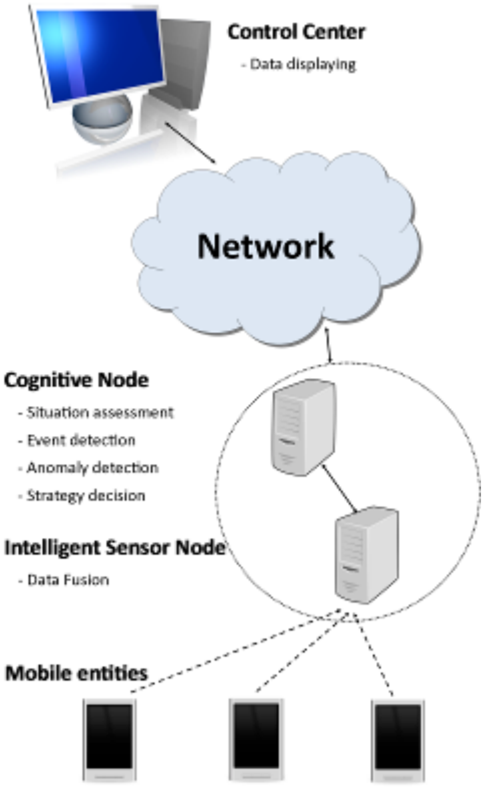


Figure 7 Example of implementation

7 Middleware Prototypes for the demonstration of SPD-oriented composability

At WP5 level, different prototypes have been delivered to demonstrate the effectiveness of the middleware technologies and algorithms to address the SPD-composition problem.

In Figure 8 the overall rationale at the basis of pSHIELD middleware is depicted: thanks to the Middleware core services and the semantic representation, the system's elements can be discovered by the overlay; then, the overlay is able to compose them to achieve the desired SPD level in two separate ways: i) by adopting policies or ii) by following the Common Criteria composition approach (defined in WP2) mixed to the context-aware Hybrid Automata approach.

The different blocks have been separately demonstrated by means of separate prototypes, mainly due to the heterogeneity of the demonstration activities (simulations for some components, architectural design for some others, software implementation for others). In particular the following prototypes have been obtained:

- **OSGI framework** to perform Middleware Core Services for discovery and composition of pSHIELD components.

- **OWL file** representing the pSHIELD ontology that, together with the pSHIELD middleware, makes the composition possible. In particular this prototype includes the **reasoner** for Common Criteria compliant composition of SPD metrics.

- **Architectural design** and performances analysis of a Policy Based approach by which the middleware composition could be driven.

- Matlab **simulation** and theoretical **formalization** of an Hybrid Automata approach to drive the SPD composition in a context-aware way.

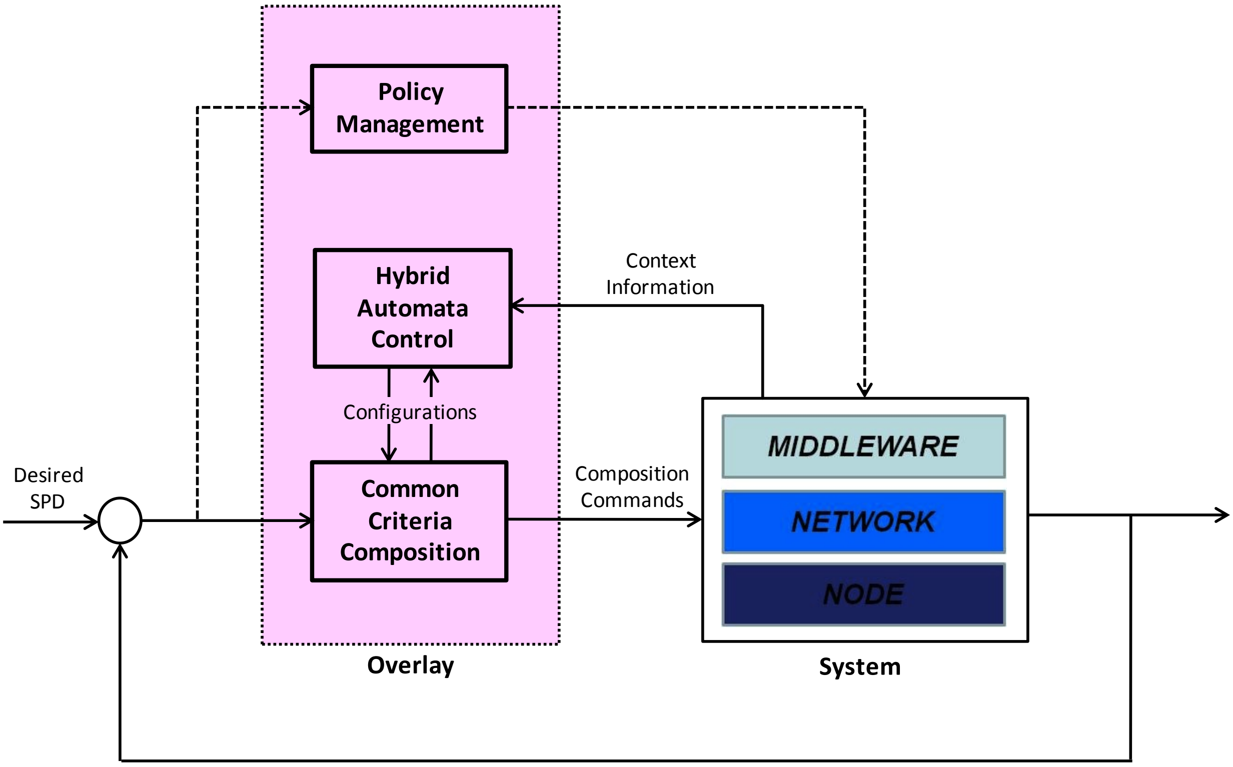


Figure 8 Rationale pSHIELD middleware

The integration of a subset of these elements will be done in the final pSHIELD demonstrator. However, at design level, it can be assured that the integration of these components will always be possible because they are all software entities and, by defining the interfaces and translating their behaviour in a software routine, they will work jointly.

In Figure 9 the rationale of Figure 8 is translated into the prototype output: the Core SPD Services (OSGI Framework), the pSHIELD Ontology (OWL) and the Common Criteria Reasoner have been implemented in a Java software environment, while the policy-based management and the hybrid automata controller have been designed and simulated in other context.

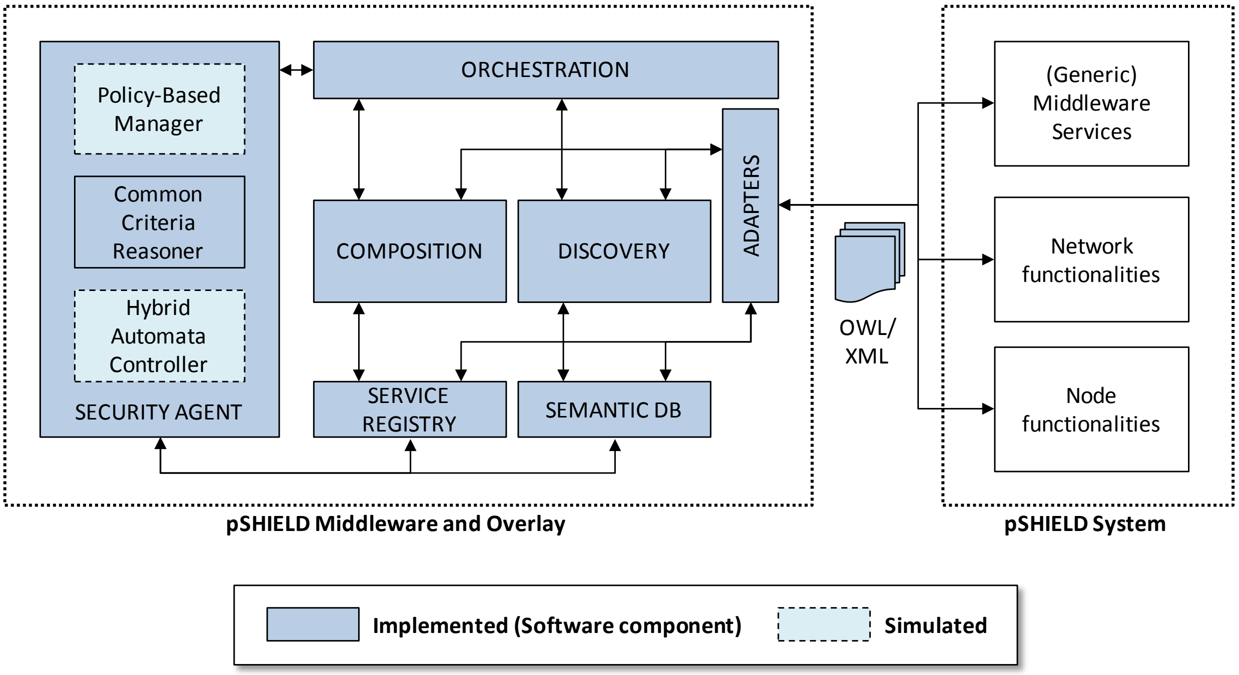


Figure 9 Middleware Prototypes Integration

7.1 Middleware Core Services

The demonstration of Composability of SPD components is based on the implementation of the pSHIELD Middleware using the OSGI framework.

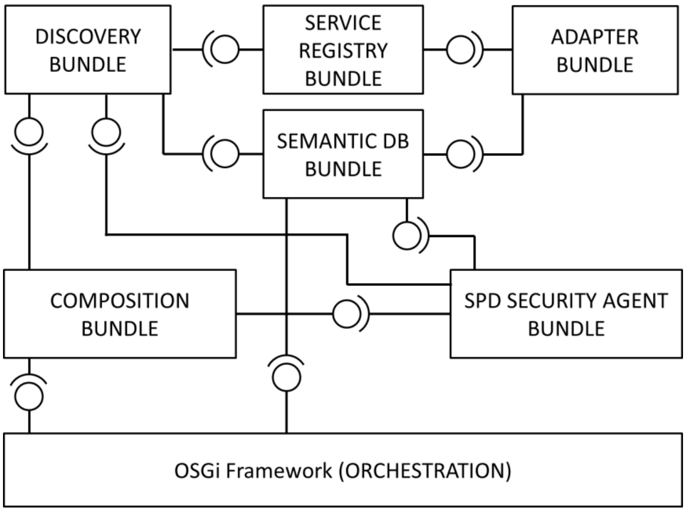


Figure 10 COre SPD services implementation in the OSGI Framework

The prototype architecture derives directly from the architecture described in D5.2. Each pSHIELD component is mapped into an OSGi bundle and, when needed, decoupled into a composition of interoperating bundles each providing a specific functionality. This modular approach simplifies the design, development and debugging of the whole system. Even the Innovative SPD Functionalities have been implemented as OSGi bundles. Each OSGi bundle has its own dependencies, provides a set of functionalities, requires a set of functionalities and is characterized by a specific SPD level. Each bundle can be registered in the Service Registry to advertise itself, to maintain updated its status in order to be discovered. Each bundle can also store its description in the Semantic Database, to be semantically composed. Each bundle interfaces the rest of the architecture providing a set of functionalities and requiring a set of functionalities, exactly as a software component does. More in particular each bundle is decoupled into two parts: the interfacing part (API) and its implementation part (IMPL). This separation between API and IMPL ease the substitution at runtime of a specific bundle, to change from one implementation to another. This substitution can be due, as an example, to the necessity to strengthen the SPD level of a specific functionality.

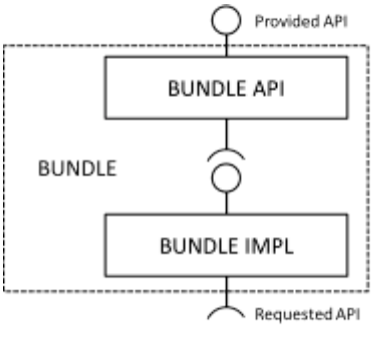


Figure 11 OSGI Bundle architecture

Applying for a top-down design approach, the Core SPD Services can be mapped in the following way:

- The Discovery and Composition are two separate bundles;

- The Orchestration is represented by the OSGi framework and orchestrates also the Discovery and Composition bundles.

To consider the interaction of the middleware layer with the rest of the architecture, the following additional bundles can be considered:

- The Service Registry bundle;
- The Semantic DB bundle;
- The SPD Security Agent bundle belonging to the pSHIELD Overlay layer;
- The pSHIELD Node, Network and Middleware Adapter that could be grouped into a single Adapter bundle.

Starting from this framework, a simple but effective demonstration has been set up to show how the pSHIELD Middleware, given a desired SPD level, is able to discover the SPD components, compute the configuration that satisfies the SPD requirements and implement it. The application scenario has been agreed with the pSHIELD partners to be coherent with the concepts discussed in the deliverable and in the railway scenario. Moreover a couple of slides will be prepared to introduce this WP5 demo, as well as some slides about the pSHIELD Ontology and Overlay concept (the other two prototypes).

7.1.1 User Story

The demonstrator has been setup to face 3 small scenarios, all related to a user story within the railway scenario.

- A railway convoy is planning to move from one station to another, the desired SPD level can change during the travel, due to different environmental condition;
- During the travel, and especially while the train stops in intermediate stations, the wagon's goods must be continuously monitored;
- The Wagon's good can be monitored only by authenticated and authorized personnel, meaning that a full and certified **auditing** process must be guaranteed during the whole travel;

7.1.1.1 Scenario #1

The train is staying at the departing station. It is equipped with all the necessary hardware and a high SPD level is asked (e.g. 4 over 5). The train is equipped with several embedded systems, each having different capabilities and supporting different implementation for the same SPD functionalities. For example the **accounting** feature is implemented via Pin, USername and Password and Token. The **authentication** mechanism has been realized using the EAP, PAP and CHAP protocols. Furthermore the **cryptography** is implemented with three different algorithms (DES, AES and BlowFish). Once the operator sets the SPD level, the overlay computes and implements the best configuration of available SPD components to compose a fully working Auditing service.

7.1.1.2 Scenario #2

When the train starts moving the operator sets a lower SPD level (e.g. 2 over 5), the overlay elaborates the best configuration of available SPD components to fit with that lower SPD level. The unnecessary SPD components are disabled and uninstalled, while the needed ones are correctly composed to guarantee the proper operation of Auditing service.

7.1.1.3 Scenario #3

When the train arrives at destination, the SPD level raises again to the maximum level (e.g. 5 over 5) to guarantee the best SPD level. The overlay reacts again to discover, compose and orchestrate those available SPD components that can guarantee the desired SPD level.

Note that pSHIELD doesn't react automatically to SPD components failures, but only to desired SPD level variations. This dynamicity is not part of pSHIELD. pSHIELD is more focused to highlight the feasibility of the composability concept.

7.2 pSHIELD ontology and Common Criteria reasoner

The second demonstrator of WP5 is the pSHIELD ontology, coupled with the Common Criteria reasoner. While the Core SPD services provide the basic functionalities of the pSHIELD Middleware, the Ontology provides the information necessary to take decisions and drive them.

The semantic Model (OWL file) that has been developed for demonstration purposes, is structured in this way:

- A section to represent *system's components*
- A section to represent *functional properties*
- A section to represent *SPD relevant information:* attributes, threats, means of mitigation

Plus:

- Attributes to identify *relations* between system and functionalities
- Attributes to quantify *SPD level*

And

- A reasoner to perform the SPD composition according to the Common Criteria rules defined in WP2.

On an implementation perspective, the following classes have been developed:

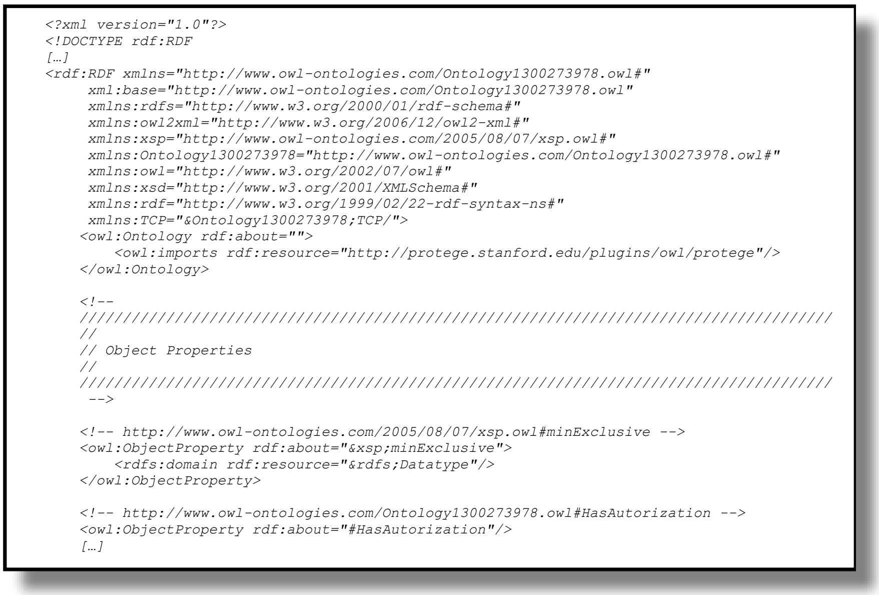
- For the structural ontology: System, Element, Hardware, SPD Component,
- For the functional ontology: SPDFunctionality, GeneralFunctionality, Connector, SPDCompositionSpecification
- For the attribute ontology: SPDConcept, SPDAttribute, SPDThreat, SPDMean

And the reasoner (semantic engine) has been structured in this way:

- At design time (offline) the semantic engine helps along the configuration of a system architecture, by discovering proper combinations of SPD modules, according to the corresponding semantic model of modules and composability rules picked out from an offline repository (catalogue); at run time (online), changes in the state of the system trigger the semantic engine to devise new compositions, based on knowledge of modules that at the moment are active in the system (possibly discovered at run time), in order to guarantee the prearranged overall SPD level. (Synthesis)
- At run time (online), the semantic engine oversees the current value of the overall SPD level as the state of the system evolves in time (Analysis)

Regarding the Common Criteria based operations that have been identified in the proposal for the aggregation of SPD metrics, and to the requirements of ontological SPD modeling, a number of suitable mixes of rules and ontology axions has been used to develop the aggregation features, including, but not limited to: *MIN, OR and MEAN operations, Redundancy configuration*.

The semantic model derived so far has been properly instantiated in the final integrated demonstrator. The prototype delivered for WP5 is in the form of OWL file, i.e. an xml file:



```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF
[…]
<rdf:RDF xmlns="http://www.owl-ontologies.com/Ontology1300273978.owl#"
     xml:base="http://www.owl-ontologies.com/Ontology1300273978.owl"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
     xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
     xmlns:Ontology1300273978="http://www.owl-ontologies.com/Ontology1300273978.owl#"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:TCP="&Ontology1300273978;TCP/">
    <owl:Ontology rdf:about="">
        <owl:imports rdf:resource="http://protege.stanford.edu/plugins/owl/protege"/>
    </owl:Ontology>

    <!--
    //////////////////////////////////////////////////////////////////////////////////////
    //
    // Object Properties
    //
    //////////////////////////////////////////////////////////////////////////////////////
     -->

    <!-- http://www.owl-ontologies.com/2005/08/07/xsp.owl#minExclusive -->
    <owl:ObjectProperty rdf:about="&xsp;minExclusive">
        <rdfs:domain rdf:resource="&rdfs;Datatype"/>
    </owl:ObjectProperty>

    <!-- http://www.owl-ontologies.com/Ontology1300273978.owl#HasAutorization -->
    <owl:ObjectProperty rdf:about="#HasAutorization"/>
    […]
```

Figure 12 pSHIELD OWL (XML File)

7.3 Policy Based approach

The third delivered prototype is an architectural design hypothesis of the application of Policy Based management to the pSHIELD context. A typical PBM architecture is mapped to pSHIELD's general architecture.

The latter includes two types of nodes at the button node layer that are categorized based on their capabilities in terms of processing power and capacity, i.e., power nodes and sensor nodes. Power nodes are described to be more resourceful while sensor nodes are typically seen as resource constrained devices. Upper supporting layers constitute network, middleware and application layers while agents in a vertical overlay monitor/tune those layers.

Given the aforementioned architecture, a PDPs and PEPs from a typical PBM architecture can be mapped naturally to power and sensor nodes respectively. Figure 13 presents the proposed PBM mapping.

On the lower layer, sensor nodes being the managed resources are considered as policy enforcers, i.e., PEPs. The latter, based on the XACML model, should enforce authorisation decisions and handle affiliated obligations specified by applicable rules. PEPs can support local policy storage in order to comply with COPS-PR mode of operation hence the provision of a local PIP although not compulsory. However, this depends on the capabilities of deployed sensor nodes whether they can afford a form of local policy storage and decision making. Moreover, power nodes are those nodes that are more resourceful than the sensor nodes which make them natural decision making points able to process/translate policies and deduce rules to be enforced by affiliated PEPs. The COPS protocol can govern the communication between PDPs and affiliated PEPs but not exclusively as SNMP is an option as well (where an LPIP is no more required).

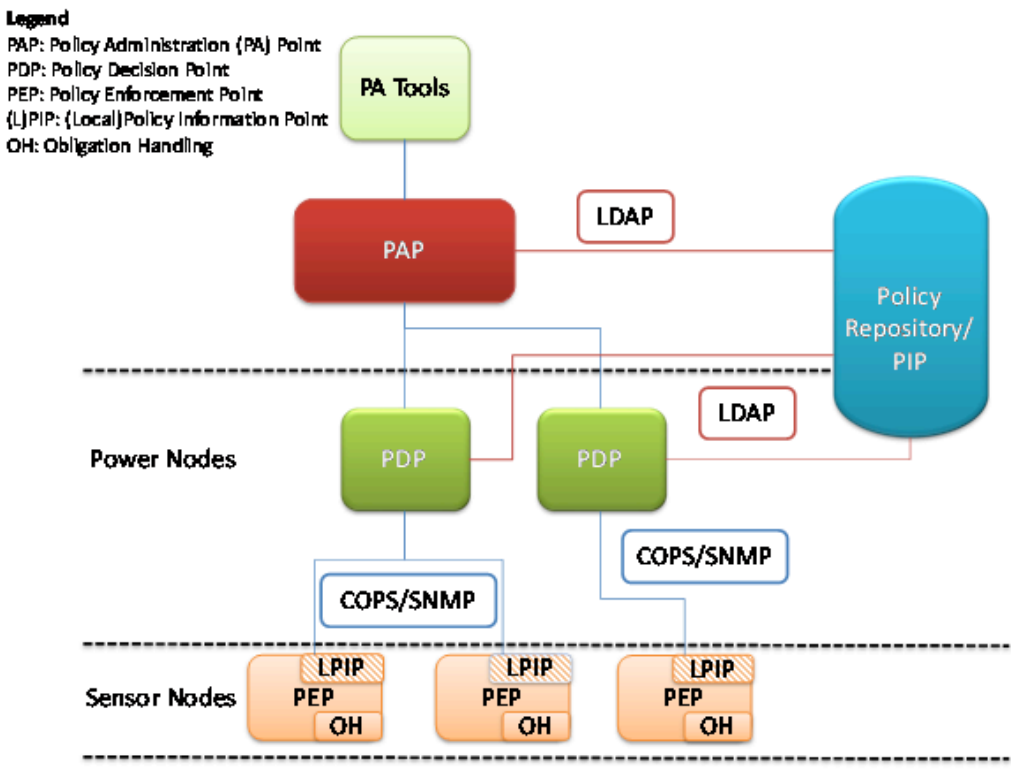


Figure 13 PBM Mapping

A group of PDPs can access the repository of policies, (i.e., PIP) in order to retrieve needed polices for evaluation. This is done through LDAP that is a protocol suited for lightweight read-intensive operations allowing for directory access from different platforms and locations. The policy repository is managed solely by the policy administrator point (PAP). Also, PAP is responsible for providing policy authoring tools besides management and control capabilities. These could include creation, termination, activation, listing, amending and synchronizing policies

Concerning pSHIELD's main scenario where a monitoring and access control system is put in place to oversee rail-transported hazardous materials, the above PBM is considered suitable. Locking and access control mechanism in addition to installed sensors can be seen as PEPs where the central control unit in the train carriage can be seen as a PDP with local access to PIP. Moreover, the central command centre overseeing the operation of the monitoring system is seen as a PAP with policy administration tools and repository support. The PIP is expected to be distributed which allows a given PDP to access it locally where a PAP can manage such a distributed PIP through LDAP.

This analysis will not be integrated in the final demonstrator due to the limited resources dedicated to this activity and the adaptation effort needed to integrate a policy based management in the OSGI Framework (integration is possible on a technological perspective, but requires time and resources: for that reason it will be one of the objective of the nSHIELD project).

7.4 Hybrid Automata approach

The last delivered prototype is the formalization, by means of Hybrid Automata Theory, of some control laws that are supposed to drive the SPD composition.

This concept is simple but effective: the Common Criteria approach defines a standard methodology to compose elements with precise quantification of their SPD level. Since the solution of the composition problem is not always unique, we can enrich this composition by setting further rules that allows to discriminate from one configuration to the other. This can be done by creating a dynamic model of the system and verifying, with respect to pre-defined objective functions, the most convenient configuration.

Two different approaches have been demonstrated to validate this theory, both supported by numerical simulations (that constitute the final output).

7.4.1 Prototype a – Static Approach with Simple Optimization

The first, simple, approach, is based on the following steps.

At first the system "state" is identified, i.e. the set of active components (node, protocols or applications). A state is a screenshot of the system in a specific condition (for example with the node E switched on) and with the dynamics associated to this condition (for example the evolution of the node's power consumption).

The selected dynamics considered for this model constitutes the so-called context information: since the SPD is controlled via the common criteria approach, we need to insert into the model variables that could be significant to control (optimize) the evolution of the system. They could be, for example, the power consumption, the computational resources utilization, the bandwidth utilization, and so on.

The state identified in this step is depicted in Figure 14.

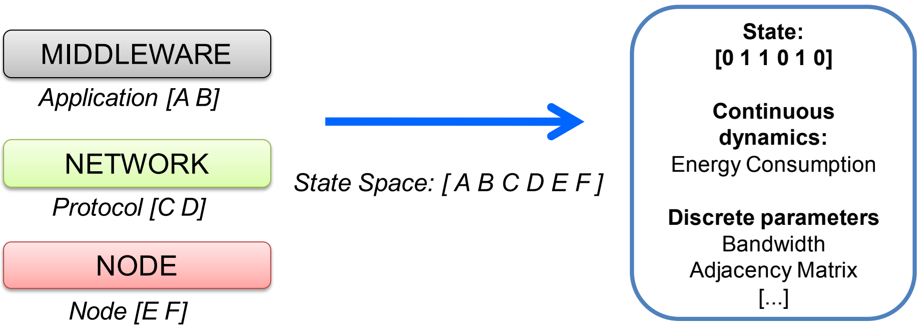


Figure 14 Single State representation

Secondly, different states are concatenated to obtain the universe of all the possible condition of the system: this is an enumeration of configurations. For example in a system with two nodes, two network protocols and two middleware services with 8 states (at least one component must be active).

Q = {[101010], [101001], [100110], [100101], [011010], [011001],[010110], [010101]}.

The result is depicted in Figure 15.

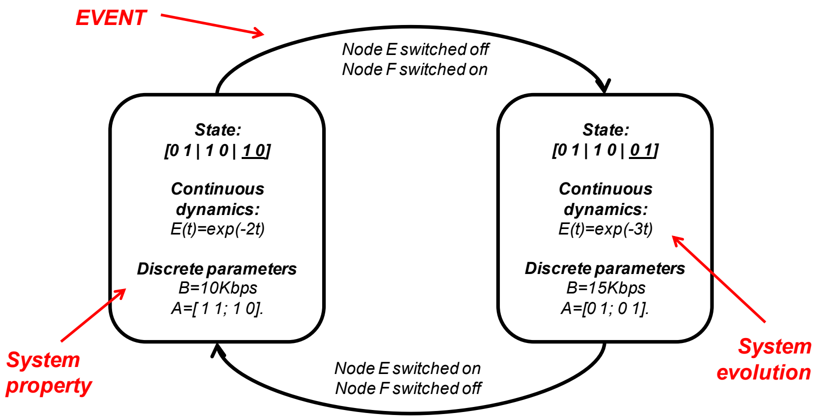


Figure 15 Hybrid Automata to describe all the possible configurations

The transition can be voluntary and expected (control action) or not (due to fault) but in any case each event is captured and in every moment it is possible to check the status (and evolution) of the system:

D = {switch configuration$_1$, fault$_1$, …, switch configuration$_n$, fault$_n$}.

The third step is the identification of the internal variables (and dynamics) to control. For the pilot project a simple case is considered where:

- the relevant dynamic is the power consumption of the system in a specific configuration and
- the amount of bandwidth provided by the network layer.

These variables have opposite behaviours (higher bandwidth, higher power consumption) so the purpose of the control algorithm is to choose the configuration that optimizes one of them.

This scenario has been implemented in Matlab-Simulink (see Figure 16) and is composed by two nodes with two different dynamics for the power consumption and for bandwidth utilization. It is important to notice that both these configurations should be valid SPD configurations (see CC approach).

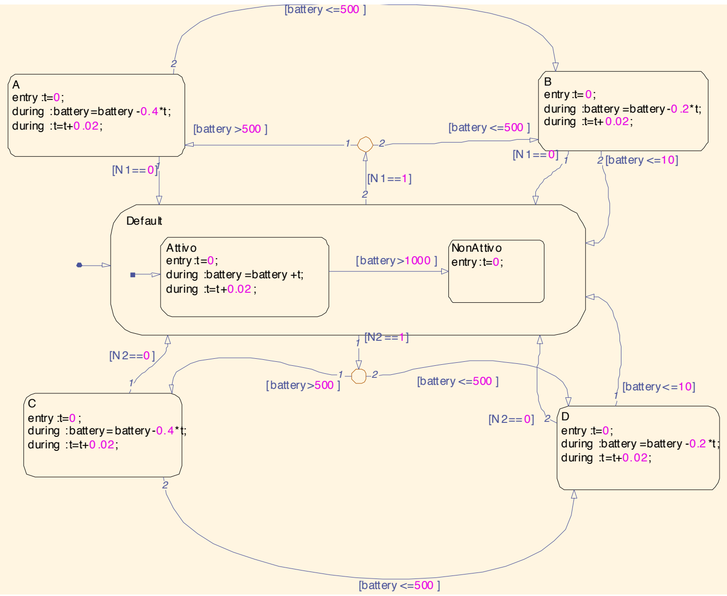


Figure 16 Hybrid automata Matlab Prototype

7.4.2 Prototype b – Operating conditions approach with MPC Control

The second prototype aims at being more efficient and flexible to cope with the scalability issues that in a complex system may arise. This has been obtained by clustering the representation of the configurations in amore restricted environment: the operating conditions. Given an Embedded System (pSHIELD Node) it is possible to identify a set con elements (battery, buffers, CPU) that can be associated to an operating conditions: a buffer can be saturated, full or empty; a CPU can be idle, working or overloaded; a battery can be full or empty. All these components can also be broken. The combination and aggregation of these conditions allows to create an exhaustive model of a pSHIELD node, as depicted in Figure 17. The aggregation is possible, since some behaviours of the components have the same effect of the system (if the CPU or the Buffer is full, the result is always the impossibility of processing data).

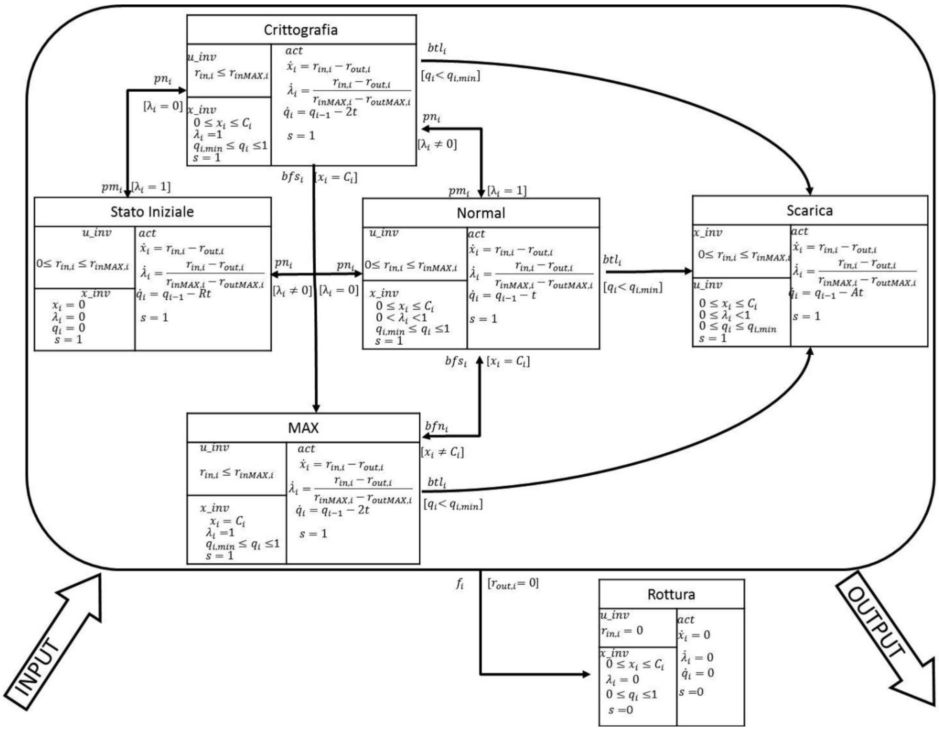

Crittografia
u_inv
$r_{in,i} \leq r_{inMAX,i}$
x_inv
$0 \leq x_i \leq C_i$
$\lambda_i = 1$
$q_{i,min} \leq q_i \leq 1$
$s = 1$
act
$\dot{x}_i = r_{in,i} - r_{out,i}$
$\dot{\lambda}_i = \frac{r_{in,i} - r_{out,i}}{r_{inMAX,i} - r_{outMAX,i}}$
$\dot{q}_i = q_{i-1} - 2t$
$s = 1$
btl_i
$[q_i < q_{i,min}]$
pn_i
$[\lambda_i = 0]$
pn_i
$[\lambda_i \neq 0]$
bfs_i
$[x_i = C_i]$
pm_i
$[\lambda_i = 1]$
pm_i
$[\lambda_i = 1]$
Stato Iniziale
u_inv
$0 \leq r_{in,i} \leq r_{inMAX,i}$
x_inv
$x_i = 0$
$\lambda_i = 0$
$q_i = 0$
$s = 1$
act
$\dot{x}_i = r_{in,i} - r_{out,i}$
$\dot{\lambda}_i = \frac{r_{in,i} - r_{out,i}}{r_{inMAX,i} - r_{outMAX,i}}$
$\dot{q}_i = q_{i-1} - Rt$
$s = 1$
pn_i
$[\lambda_i \neq 0]$
pn_i
$[\lambda_i = 0]$
Normal
u_inv
$0 \leq r_{in,i} \leq r_{inMAX,i}$
x_inv
$0 \leq x_i \leq C_i$
$0 < \lambda_i < 1$
$q_{i,min} \leq q_i \leq 1$
$s = 1$
act
$\dot{x}_i = r_{in,i} - r_{out,i}$
$\dot{\lambda}_i = \frac{r_{in,i} - r_{out,i}}{r_{inMAX,i} - r_{outMAX,i}}$
$\dot{q}_i = q_{i-1} - t$
$s = 1$
btl_i
$[q_i < q_{i,min}]$
Scarica
x_inv
$0 \leq r_{in,i} \leq r_{inMAX,i}$
u_inv
$0 \leq x_i \leq C_i$
$0 \leq \lambda_i < 1$
$0 \leq q_i \leq q_{i,min}$
$s = 1$
act
$\dot{x}_i = r_{in,i} - r_{out,i}$
$\dot{\lambda}_i = \frac{r_{in,i} - r_{out,i}}{r_{inMAX,i} - r_{outMAX,i}}$
$\dot{q}_i = q_{i-1} - At$
$s = 1$
bfs_i
$[x_i = C_i]$
MAX
u_inv
$r_{in,i} \leq r_{inMAX,i}$
x_inv
$x_i = C_i$
$\lambda_i = 1$
$q_{i,min} \leq q_i \leq 1$
$s = 1$
act
$\dot{x}_i = r_{in,i} - r_{out,i}$
$\dot{\lambda}_i = \frac{r_{in,i} - r_{out,i}}{r_{inMAX,i} - r_{outMAX,i}}$
$\dot{q}_i = q_{i-1} - 2t$
$s = 1$
bfn_i
$[x_i \neq C_i]$
btl_i
$[q_i < q_{i,min}]$
f_i
$[r_{out,i} = 0]$
Rottura
u_inv
$r_{in,i} = 0$
x_inv
$0 \leq x_i \leq C_i$
$\lambda_i = 0$
$0 \leq q_i \leq 1$
$s = 0$
act
$\dot{x}_i = 0$
$\dot{\lambda}_i = 0$
$\dot{q}_i = 0$
$s = 0$
INPUT
OUTPUT

Figure 17 Hybrid Automata representing the pSHIELD node

At this point the problem of scalability of composition is solved, since the introduction of a new node in the system doesn't imply an exponential increase in the model size, but a linear growth (6 states for each additional node and 4 states for each additional network layer).

Last, but not least, interesting control algorithms can be applied to the system model due to its formulation by means of these operating conditions (see for example the work of Bemporad [8] and [9]). In particular for the pSHIELD purposes the framework developed in [9], based on Model Predictive Control (MPC), has been considered to verify the effectiveness of the Hybrid Automata approach.

For the simulations it has been used the Matlab Toolbox for Hybrid System with the default configuration (standard MPC problem). The Objective of the control algorithm has been to maximize the amount of data processed by the node while preserving the battery and leaving a certain amount of "reserved" resources for potential emergency tasks.

The Hybrid Automata prototypes will not be integrated in the final demonstrator because their role is mainly to validate the control law and not to "implement" the control law in the OSGI environment. However further studies in nSHIELD project will lead to the translation of the Matlab simulations file into a C++ or Java language to perform the same task directly in some software routines at middleware level.

As for the Policy based Management, the "potential" integration is assured by the software abstraction and definition of proper interfaces (even if it will be carried out in nSHIELD).

8 Security integration across heterogeneous platforms

The recent year's developments in the sensor technology produces diversified sensor nodes in terms of CPU, memory, communication interfaces, operating system and programming environment. These sensor nodes were being involved for solving small-scale specialized problems in the industrial and scientific domain. The introduction of sensors into mobile devices opens new doors for the society for the enablement of mobile-as-a-sensor node. Currently, these sensor-enabled mobile devices are in the hand of massive people, which bring opportunity for service providers to design innovative and diverse context-aware applications and services. Now-a-days, mobile manufacturers are assembling mobile devices with different sensor functionalities. For instance, GPRS, NFC and Bluetooth/ZigBee have been adapted into mobile devices and computing devices. These sensor-enabled mobile devices, sensor nodes and other computing devices collectively can be considered as heterogeneous platforms that bring not only the implementation challenges for application and services but also proliferates the security integration challenges. The reason of the security integration across heterogeneous platform to be considered as a challenging task is due to the support of different security protocols by different sensor nodes. For instance, SPIN [1] protocol has been developed for TinyOS supported sensors nodes only and TinyPEDS [2] act as a middleware for providing a secure persistent for the collected data from the sensors.

A limited research work on security integration across heterogeneous platforms is carried out because most of the work has been done on the flat and homogenous platforms where either all the sensor nodes have the same capabilities or they come under same sensor platform. For the pSHIELD demonstration, the security goals have been identified with respect to SPD functionalities and the goals are to establish secure communication, incorporate proper authentication and access control mechanism regardless of sensor devices supported security protocols. The Interoperable Rail Information System is selected for the use case and the Telenor Shepherd platform is selected for the secure integration of connected- sensor devices.

Figure 18a - JBV measurement locomotive Roger

In the pSHIELD project the various nodes are studied and thus categorized into nano, micro/personal and power node-The more detail can be found in D3.1 and D3.2.

8.1 Use Case description

The use case for the demonstration is to continuously monitoring of trains and railway infrastructure. The purpose is twofold (i) detecting any unusual condition such as high temperature, strange sounds and unexpected movement, and (ii) transferring such information to different actors (i.e., train operator, train infrastructure owner and consumer) involved in the rail system both automatically and in a request/response demand-based passive mode. The train is equipped with several heterogeneous computing devices such as sensors, actuators, GPS receiver, and gateway embedded computer for detection of each conditions. These devices interact using the heterogeneous protocols for sensing the information in their vicinity and sending it to the gateway. As an intelligence device, the gateway figures out any irregularity, and it sends the details to all actors including smart train operator, infrastructure owner and consumer.

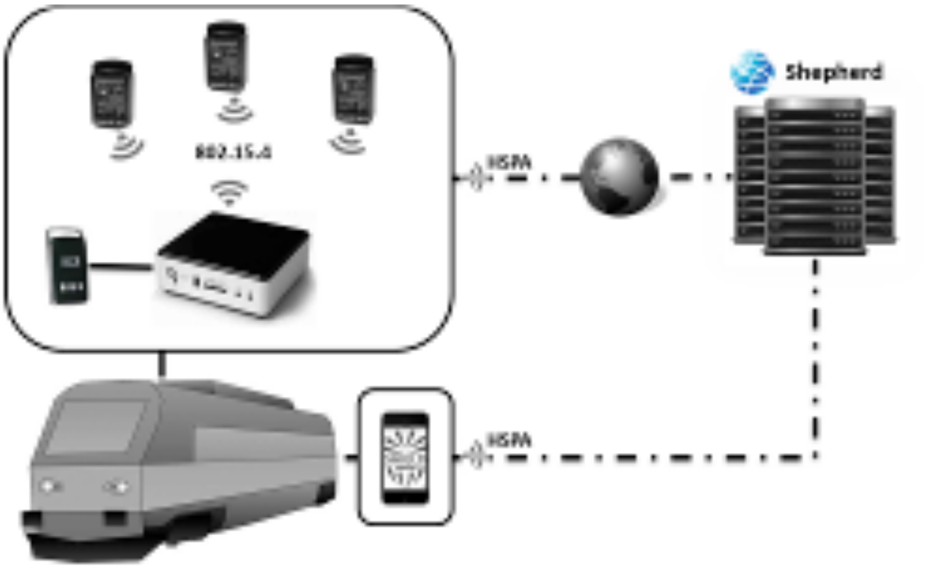


Figure 18b - Use case scenario infrastructure architecture

8.2 SPD Features across heterogeneous platforms

The following SPD features are integrated across the platform.

1. In case of power failure all the heterogeneous computing devices will be automatically start up.
2. In case of any software failure all the heterogeneous computing devices will be reconfigured and synchronized with other devices.
3. Interoperable with Telenor Shepherd platform in a secure fashion. The communication channel is secured from heterogeneous computing devices to the Telenor object through HTTPS protocol.
4. Linking sensor data while preserving privacy. Only legitimate user of Telenor Shepherd platform can access to sensor data. Telenor Shepherd platform denies the sensor data request if an illegitimate user request arrives.

Figure 18c - Installation on board of Roger, with nano, micro, personal and power platform

8.3 Secure Integration with Telenor Shepherd platform

Telenor, Norway have introduced a platform (named as Shepherd®) for interoperability and integration that supports communication between connected devices (nano and micro nodes) and makes them accessible from anywhere at anytime.

The Shepherd® is a platform for Connected Objects meaning that the pluggable component can be connected, and be integrated in Shepherd® platform as a Connected object (CO). Figure 19 depicts the overview of Shepherd® platform.

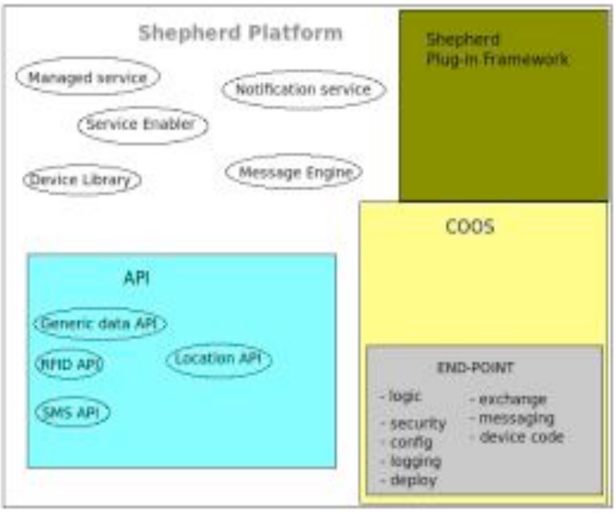
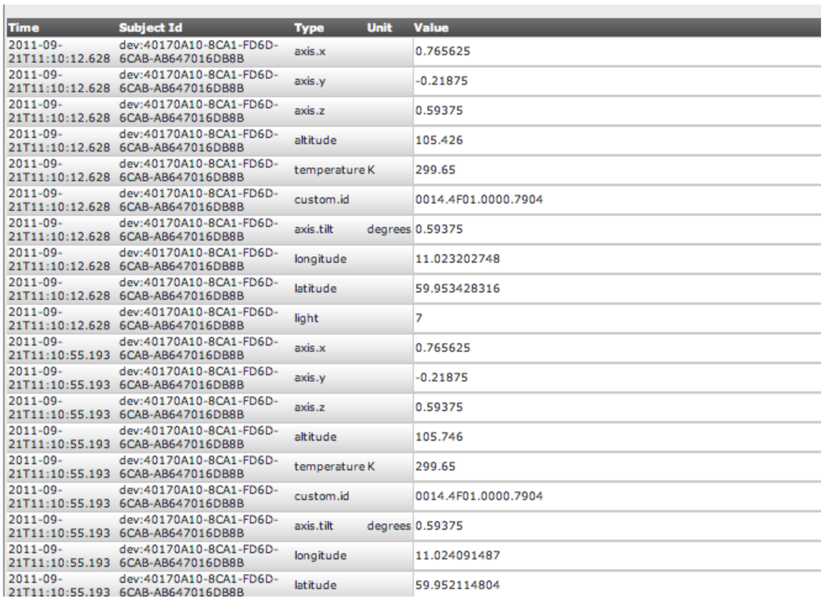


Figure 19a - Details of Telenor's Shepherd platform

It offers variety of services, which includes:

1. Service Management for monitoring, device configuration, SLAs, and supporting.
2. Service Enabler has a specific API that allows further access to other modules.
3. Message Engine handles and secures the process of message flow, including capturing, processing, routing and storage of data in an environment.
4. Notification services that inform about the status of devices and applications.
5. Device library consists of interfaces for tools and services recognition.

The different categories of pSHIELD nodes support different security protocols based on their capabilities. This heterogeneity of devices leads toward heterogeneity of security. For instance in the context of IRIS use case the nano nodes do not support any security protocol. Those nodes are connected with wire so they are considered to be somehow secured as compared to wireless nodes, which are more prone to attacks. The micro nodes only support a number of security protocols and algorithms such as TLS and AES. Whereas the personal nodes support a multiple number of security protocols ranging from different asymmetric and symmetric security protocols to accomplish different security operations. Thus such heterogeneity of security



Time	Subject Id	Type	Unit	Value
2011-09-21T11:10:12.628	dev:40170A10-8CA1-FD6D-6CAB-AB647016DB8B	axis.x		0.765625
2011-09-21T11:10:12.628	dev:40170A10-8CA1-FD6D-6CAB-AB647016DB8B	axis.y		-0.21875
2011-09-21T11:10:12.628	dev:40170A10-8CA1-FD6D-6CAB-AB647016DB8B	axis.z		0.59375
2011-09-21T11:10:12.628	dev:40170A10-8CA1-FD6D-6CAB-AB647016DB8B	altitude		105.426
2011-09-21T11:10:12.628	dev:40170A10-8CA1-FD6D-6CAB-AB647016DB8B	temperature	K	299.65
2011-09-21T11:10:12.628	dev:40170A10-8CA1-FD6D-6CAB-AB647016DB8B	custom.id		0014.4F01.0000.7904
2011-09-21T11:10:12.628	dev:40170A10-8CA1-FD6D-6CAB-AB647016DB8B	axis.tilt	degrees	0.59375
2011-09-21T11:10:12.628	dev:40170A10-8CA1-FD6D-6CAB-AB647016DB8B	longitude		11.023202748
2011-09-21T11:10:12.628	dev:40170A10-8CA1-FD6D-6CAB-AB647016DB8B	latitude		59.953428316
2011-09-21T11:10:12.628	dev:40170A10-8CA1-FD6D-6CAB-AB647016DB8B	light		7
2011-09-21T11:10:55.193	dev:40170A10-8CA1-FD6D-6CAB-AB647016DB8B	axis.x		0.765625
2011-09-21T11:10:55.193	dev:40170A10-8CA1-FD6D-6CAB-AB647016DB8B	axis.y		-0.21875
2011-09-21T11:10:55.193	dev:40170A10-8CA1-FD6D-6CAB-AB647016DB8B	axis.z		0.59375
2011-09-21T11:10:55.193	dev:40170A10-8CA1-FD6D-6CAB-AB647016DB8B	altitude		105.746
2011-09-21T11:10:55.193	dev:40170A10-8CA1-FD6D-6CAB-AB647016DB8B	temperature	K	299.65
2011-09-21T11:10:55.193	dev:40170A10-8CA1-FD6D-6CAB-AB647016DB8B	custom.id		0014.4F01.0000.7904
2011-09-21T11:10:55.193	dev:40170A10-8CA1-FD6D-6CAB-AB647016DB8B	axis.tilt	degrees	0.59375
2011-09-21T11:10:55.193	dev:40170A10-8CA1-FD6D-6CAB-AB647016DB8B	longitude		11.024091487
2011-09-21T11:10:55.193	dev:40170A10-8CA1-FD6D-6CAB-AB647016DB8B	latitude		59.952114804

Figure 19b - Log of pSHIELD prototype data in the Shepherd platform

protocols among different types of pSHIELD nodes demands an interoperable solution that bridges all security operations. This is achieved by abstracting the security operations on personal nodes in the context of IRIS use case by employing the security proxy approach [3] and integrating with Telenor Shepherd platform in order to enforce fine grained access control regardless of underlying authentication and access control mechanism.

9 Hardware prototypical implementation of specific layers

The proposed system helps in addressing the monitoring and protection of fixed or mobile assets, like freight cars, against both natural and intentional threats. It requires a central control room in which a security management system is able to remotely collect alarms about some parameters (e.g. temperature, humidity and vibrations) and monitor the status of assets. The provided solution combines the advantages of using self-powered wireless devices, with advanced intelligent capabilities.

Regarding the specific case of freight train monitoring, the main requirements to be fulfilled by the autonomous protection system, are the following:

- Secure handling of the critical information of the transported material;
- Secure and dependable monitoring of the transport.

In this considered scenario, both natural and malicious faults can have an impact on system availability and indirectly on safety. In particular, some critical points are listed below:

- when hazardous material is mismanaged it has the potential to pollute the environment and threaten human health;
- integration of distributed smart-sensors in order to monitor car integrity and warn about possible leaking of hazardous material.

In Figure 20 is represented an architectural scheme for the remote monitoring of unpowered freight cars possibly used to transport valuable and/or hazardous materials, based on the following devices:

- **GW:** Gateway node for the wireless sensor network;
- **S:** wireless smart-sensors (possibly integrating a GPS) measuring parameters like: temperature, vibration, humidity, light level, etc.

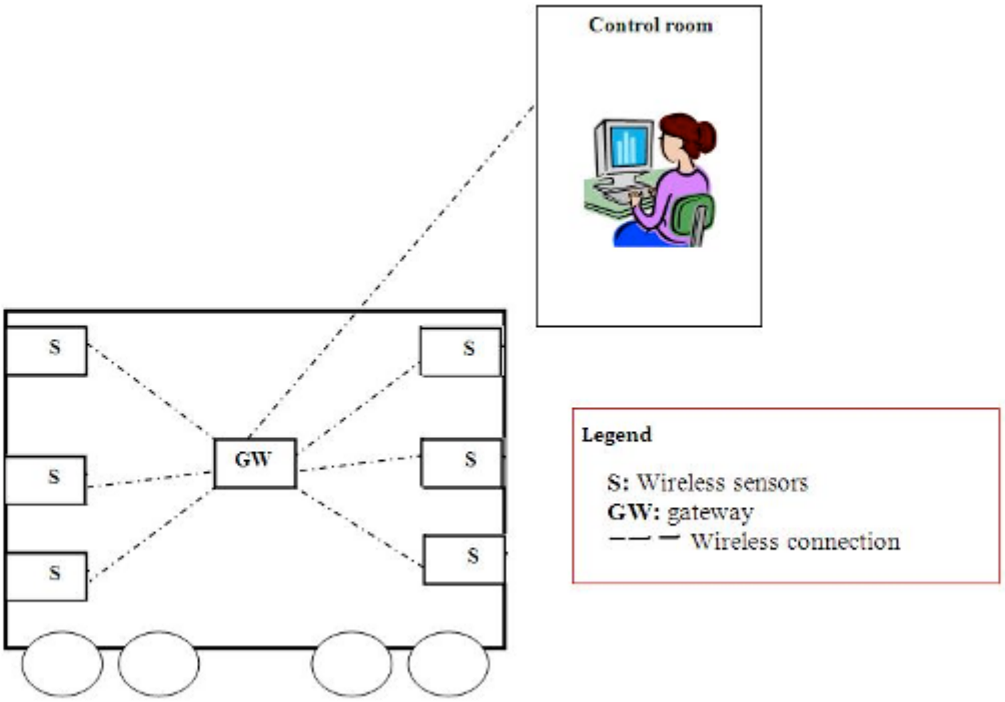


Figure 20 The wagon platform

As mentioned above, the monitoring is aimed to the detection of abnormal operating environmental conditions on board of vehicles as well as threats of burglary. So, the basic working logic is the following: whenever an abnormal event (e.g. very high temperature or out of range vibrations) is detected by sensors, its transmission unit is activated and data is received by GW. If the anomaly is confirmed, the gateway sends an appropriate warning message to the control center and the control room asks the current position (GPS sensor on board).

Obviously, the self-powered smart-sensors send data to the gateway, through a secure connection (the information between sensors and gateway are encrypted). The gateway transmits data to the control room, by means of a secure wireless connection (e.g. HTTPS, SSL).

The objective is to simplify anomaly and fault detection (considering sensors measuring the same parameters in the same area), to improve the overall detection reliability and to make possible complex threats detection (considering distributed heterogeneous sensors).

9.1 Preliminary lab approximation

To ensure secure and dependable monitoring of rail cars transporting hazardous materials, will provide resiliency against both random and malicious threats.

The experimentation in divided in two phases:

1. Provide SPD functionalities to off-the-shelf smart-sensors (i.e. WSN motes) measuring environmental parameters like temperature, vibrations, etc. and test them in the laboratory (this section);
2. Develop a monitoring application detecting abnormal operating conditions and test the overall system in a real-environment for SPD functionalities like node authentication, checksum, cryptography, etc. also by simulating SPD threats (Section 9.2).

A typical monitoring system is made of different sensor networks that can be heterogeneous in the technology aspects, in the data formats, in synchronization and localization standards, but also in security requirements. They can be connected in different ways and their data should be elaborated by the same application to enrich the knowledge of observed complex phenomena. From a high-level point of view, a complex monitoring infrastructure composed of several heterogeneous sensor networks can be considered as structured into two main layers, namely the *sensor network layer* and the *distributed application layer*, as shown in Figure 21.

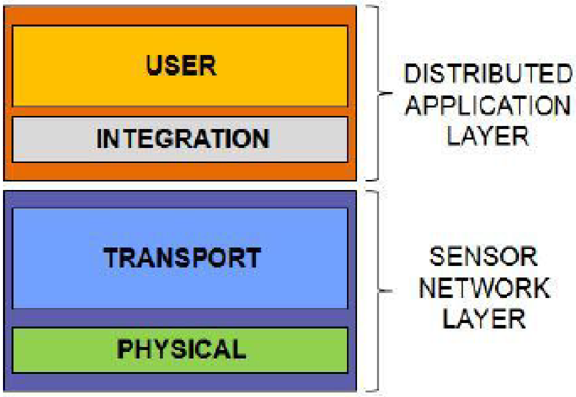


Figure 21 Architecture of monitoring system

The *sensor network layer* can be further divided into two levels:

- **Physical level:** is responsible of the processing of the locally generated data at the node level.

- **Transport level:** controls the communication between the nodes of each network.

The *application layer* deals with the fusion and high level management of the data sensed by different heterogeneous networks; it can be considered as structured into two levels:

- **Integration level:** is responsible of the integration of data belonging to different sensor networks; it typically enforces a translation in a common data model.

- **User level:** executes the user distributed applications, which typically query the underlying networks and sensor features and manipulate the retrieved results for aggregation and decision purposes.

With respect to this architecture, security issues can arise at different levels: at the *application layer*, data retrieved from the different networks are typically processed in a distributed manner, thus raising well-known issues dealing with secure network communication and access control; as this kind of processing is usually done by PC-class devices, the application layer does not suffer of the problems related to the limited resources of sensor nodes, and the well-known security protocols can be directly applied, as for the *sensor network layer*: At the *transport level* it is necessary to secure data exchanged between nodes, and this can be achieved with the adoption of proper security protocols and mechanisms that take in consideration the limited resources; at the *physical level*, it is necessary to provide mechanisms for protecting nodes against physical tampering and DOS and jamming attacks.

The reference architecture is built upon SeNsIM (Sensor Networks Integration and Management) [5], a framework that was designed for integration of heterogeneous sensor networks based on the wrapper-mediator paradigm, described in the deliverable D6.1. It provides a unified interface by which users can easily execute queries on the system to retrieve network information and elaborate sensor data. In SeNsIM each different network of the system is managed by a dedicated *wrapper* that is able to communicate with the specific underlying technology and acts as a connector for the *mediator* component; the mediator is responsible to properly format user requests and forward them to the different wrappers, this translates the incoming queries and injects them into the underlying networks, retrieves the results and passes them back to the mediator.

In the figure 22 is represented the SeNsIM Architecture.

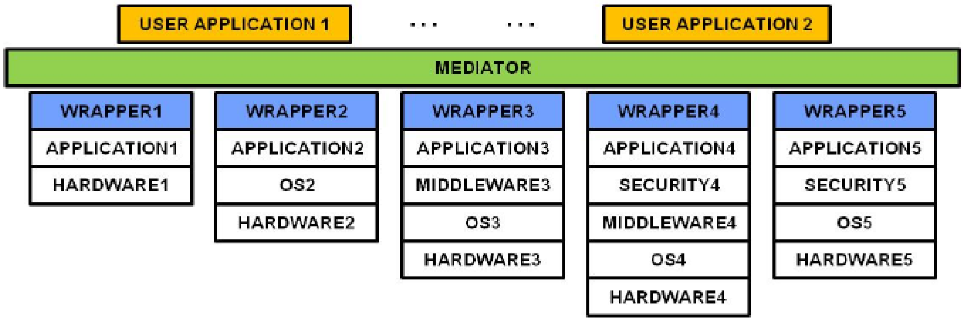


Figure 22 SeNsIM Architecture

9.1.1 The demonstrator's architecture

The demonstrator architecture, based on the above SeNsIM, is shown in Figure 23:

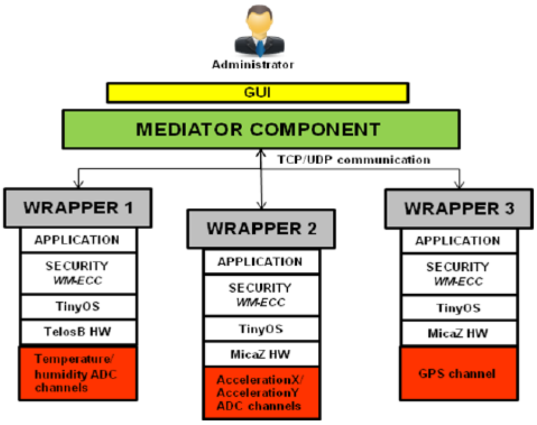


Figure 23 Demonstrator's Architecture

It is composed of a mediator component, accessible by an end-user via a GUI interface, and of 3 different wrappers, each managing a different WSN:

- ***Wrapper 1*****:** it manages a simple network composed of 3 TelosB nodes, based on a 4.15 MHz MSP430 microcontroller and a CC2420 radio chip and having a 10 kBytes internal RAM and a 48 kBytes program Flash memory; the “sink” or “master” node is directly connected to a PC device running the wrapper software and is responsible of forwarding the queries coming from the wrapper to the two motes, and to send back the result samples. The handled network is used to measure temperature and humidity values.

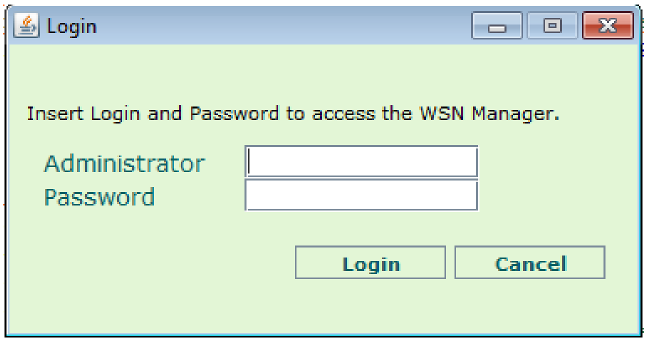
- ***Wrapper 2*****:** it manages a simple network composed of 3 MicaZ nodes, based on an ATmega128L low-power microcontroller and having a 512 kBytes program Flash memory; the “sink” or “master” node is directly connected to a PC device running the wrapper software and is responsible of forwarding the queries coming from the wrapper to the two motes, and to send back the result samples. The handled network is used to measure the acceleration on both axes.

- ***Wrapper 3***: it manages a simple network composed of 2 MicaZ nodes, a master connected to the wrapper and a mote able to capture the GPS signal for localization purposes.

The 3 networks are all based on the TinyOS Operating System and implement a security cryptosystem [4,5] in order to enforce confidentiality and integrity requirements at the node communication level and to ensure the authentication of the master node (See Section 5.4 of D6.2).

9.1.2 The prototype application

At application start, the panel in figure 24 will appear:



Login

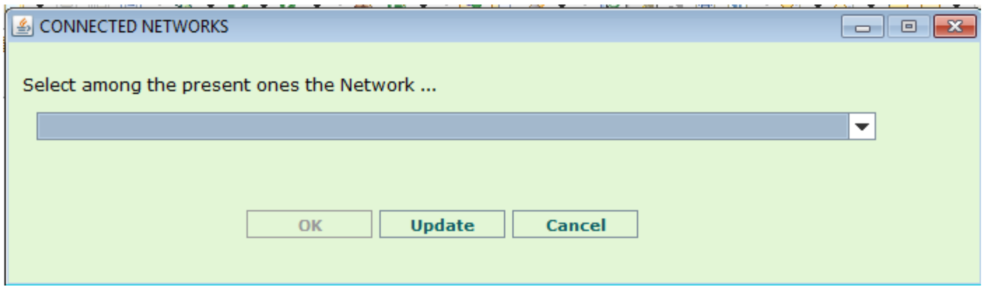
Insert Login and Password to access the WSN Manager.

Administrator

Password

Figure 24 Login Panel

After the login, the mediator starts listening for incoming connections, which will arrive on a UDP Socket bound to the 7000 port (this information, along with the IP address of the mediator machine is specified in a configuration file which is read by the wrapper component at its startup). The panel in figure 25 appears which shows the list of connected networks.

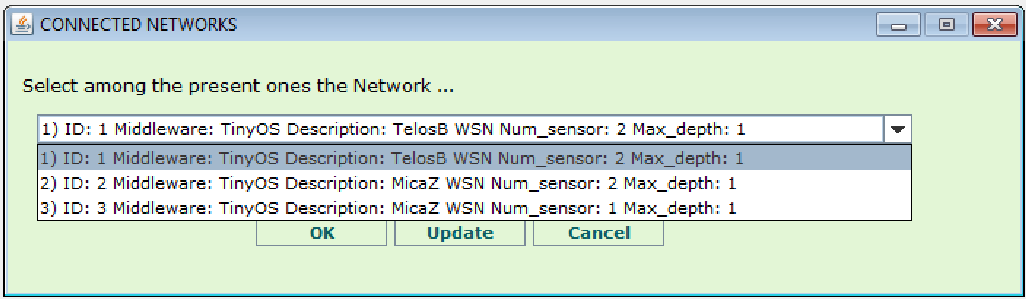


CONNECTED NETWORKS

Select among the present ones the Network ...

Figure 25 Connected Networks

After the registration of the 3 networks the Network List Panel will look like in figure 26 this:



CONNECTED NETWORKS

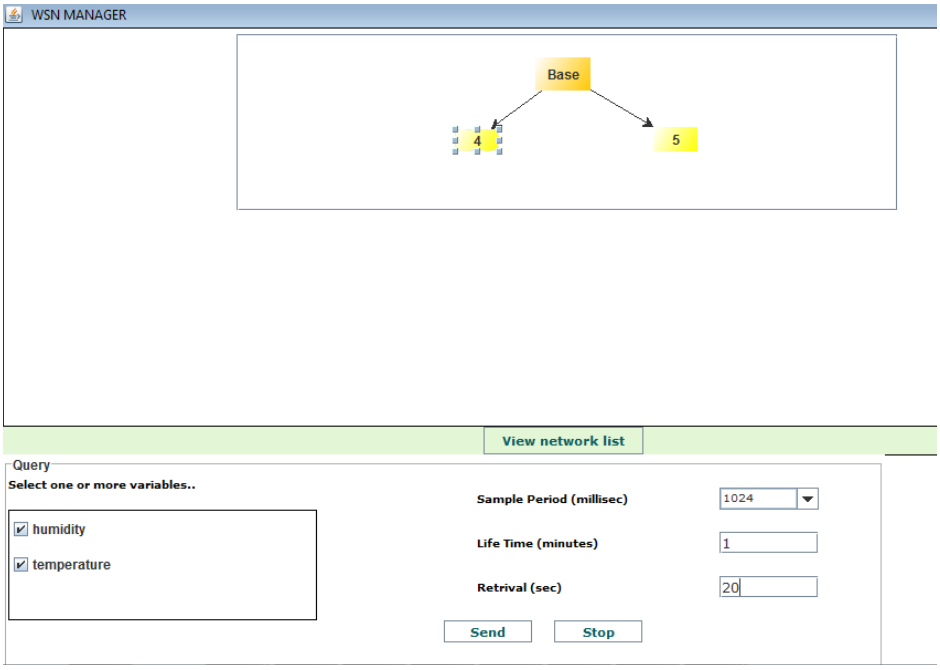
Select among the present ones the Network ...

1) ID: 1 Middleware: TinyOS Description: TelosB WSN Num_sensor: 2 Max_depth: 1

1) ID: 1 Middleware: TinyOS Description: TelosB WSN Num_sensor: 2 Max_depth: 1
2) ID: 2 Middleware: TinyOS Description: MicaZ WSN Num_sensor: 2 Max_depth: 1
3) ID: 3 Middleware: TinyOS Description: MicaZ WSN Num_sensor: 1 Max_depth: 1

Figure 26 Connected Networks after registration

From the Network List Panel it is possible to select each of the connected networks in order to start a query process. By selecting a network from the list and pushing the *OK* button, the *WSN Manager Panel* will appear (Figure 27): it shows the graphical structure of the network and contains a section which becomes visible when selecting a node from the graph. This section allows the definition of a query by specifying the predicates to retrieve, the query duration (lifetime in the panel), the sample period and the desired retrieval interval.



Select one or more variables..

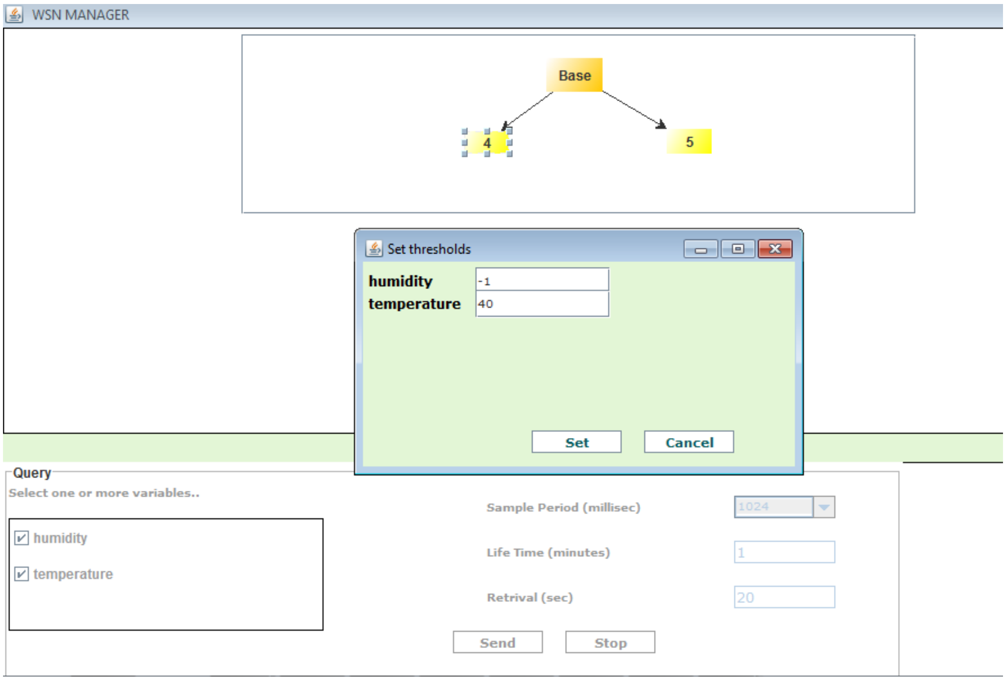
☑ humidity
☑ temperature

Sample Period (millisec) 1024
Life Time (minutes) 1
Retrival (sec) 20

Figure 27 Manager Panel

When pushing the *Send* button, the user will be prompted to select thresholds for each of the selected predicates (see figure). If the user does not want to specify a threshold for some predicates, it is sufficient to keep the default value (-1).

After setting the thresholds, the query will start and several tables will appear, as many as the number of query predicates, showing the retrieved samples for each node of the network (figure 28). The tables will update by themselves at the arrival of each result.xml file, displaying in red the values which exceed the defined thresholds (Figure 29).



humidity -1

temperature 40

Query

Select one or more variables..

☑ humidity

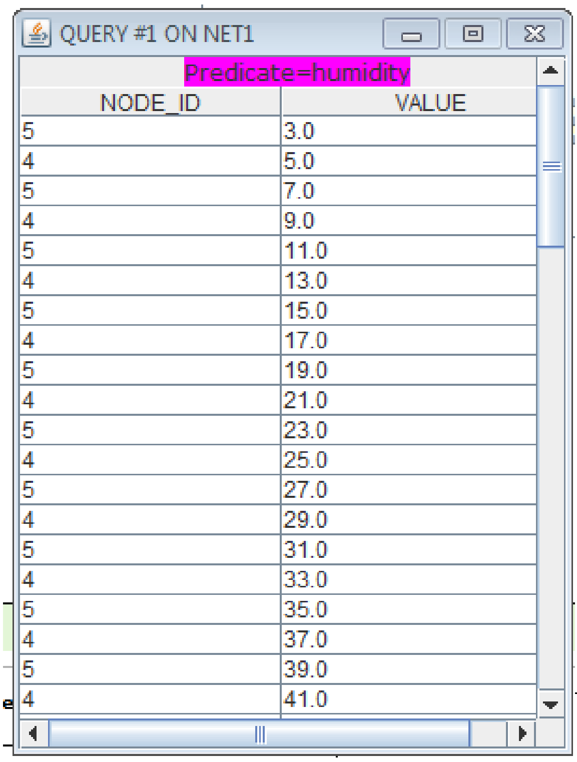
☑ temperature

Sample Period (millisec) 1024

Life Time (minutes) 1

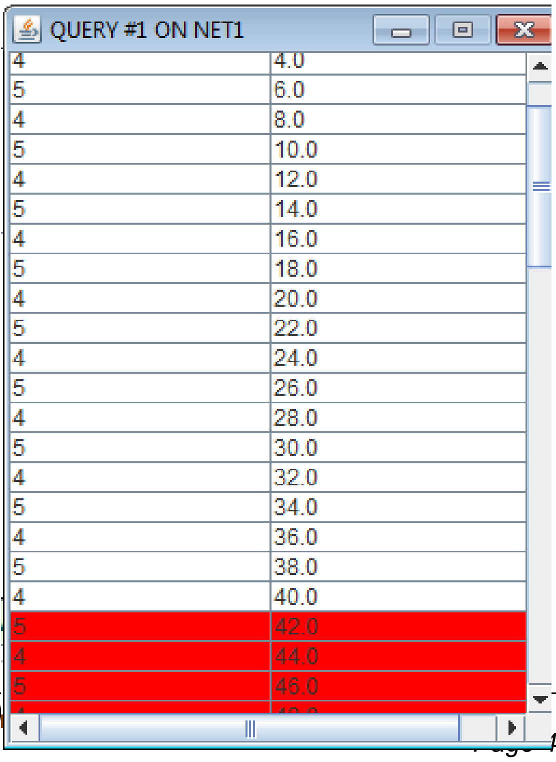
Retrival (sec) 20

Figure 28 Panel for thresholds



Predicate=humidity

NODE_ID	VALUE
5	3.0
4	5.0
5	7.0
4	9.0
5	11.0
4	13.0
5	15.0
4	17.0
5	19.0
4	21.0
5	23.0
4	25.0
5	27.0
4	29.0
5	31.0
4	33.0
5	35.0
4	37.0
5	39.0
4	41.0



4	4.0
5	6.0
4	8.0
5	10.0
4	12.0
5	14.0
4	16.0
5	18.0
4	20.0
5	22.0
4	24.0
5	26.0
4	28.0
5	30.0
4	32.0
5	34.0
4	36.0
5	38.0
4	40.0
5	42.0
4	44.0
5	46.0

Figure 29 Query Results

9.2 Description of the testing environment

The platform described in the previous section is installed on a real freight car made available by the Italian Railway Authority (RFI/Trenitalia) at Roma Smistamento in Via di Villa Spada, Roma. Some picture of environment are in figure 30 and 31.

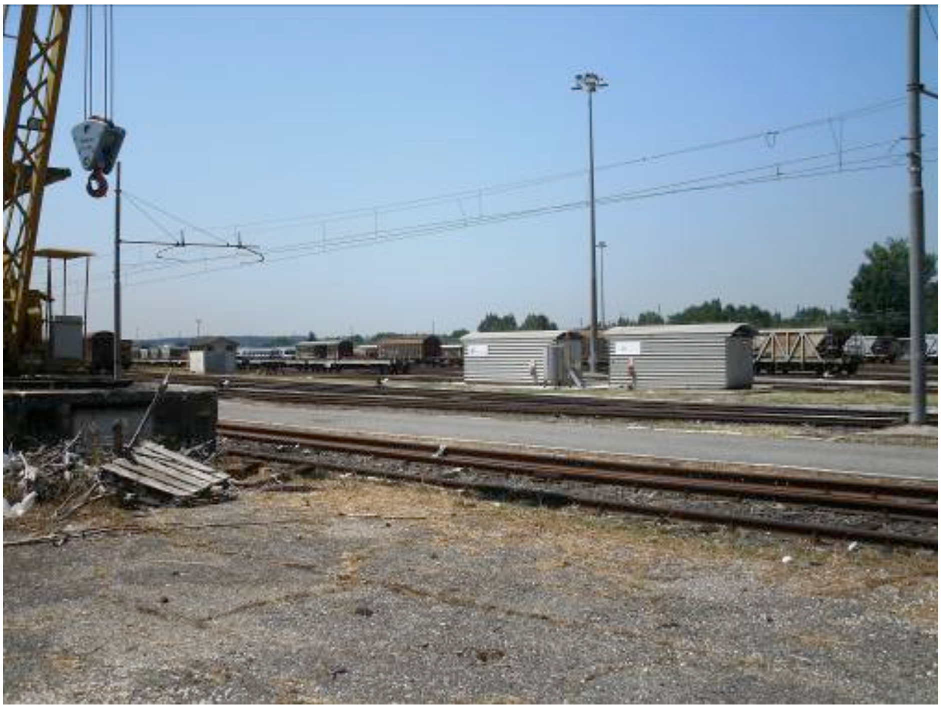


Figure 30 Roma Smistamento

Figure 31 Wagons at Roma Smistamento

The sensors are installed on freight car, at a distance of 30 meters there is a location as a control room.

On the car there are 8 sensors, grouped by 2 networks of 2 sensors each and 1 network of 1 sensors (GPS network): one measures temperature and humidity, one measures acceleration and the last measures GPS coordinates. On Car there is also the gateway of each network that are linked to a pc on a car. This PC communicates via wireless with a PC in a control room.

9.2.1 Installation of sensors on real car

In order to test the parameters, the three networks were installed on car in this way:

1. The networks that measures humidity and temperature, was installed in the inside of cars as showed in the figures 32-33. This network is composed by the sensors Telosb.

2. The second network that measure the acceleration and composed by sensors Micaz, was installed outside of car, as showed in the figure 35. further the motes are in the outside they are equipped with a box in order to protect them from bad weather figure 34. The master of this network was installed inside of car as showed in figure 36.

3. The third network that measure GPS coordination was installed outside with GPS receiver. Figures 37,38,39.

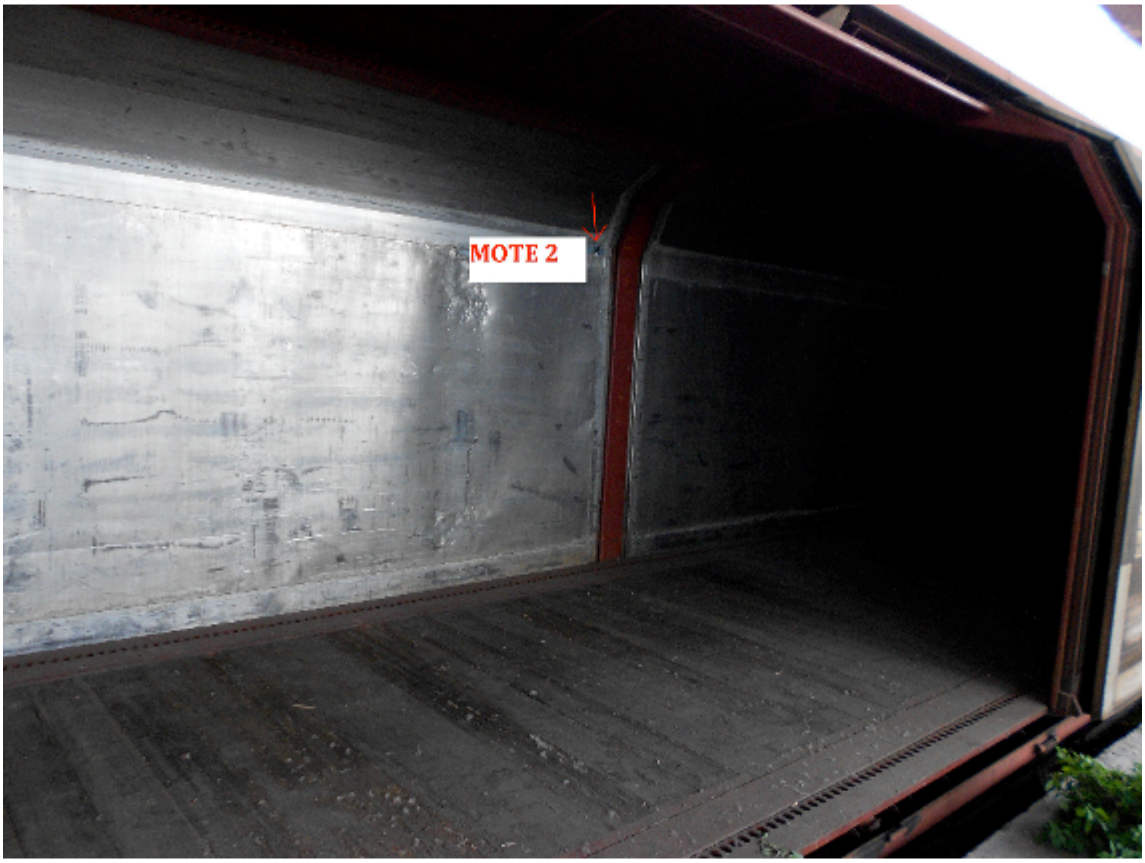


Figure 32 Inside of car mote 2

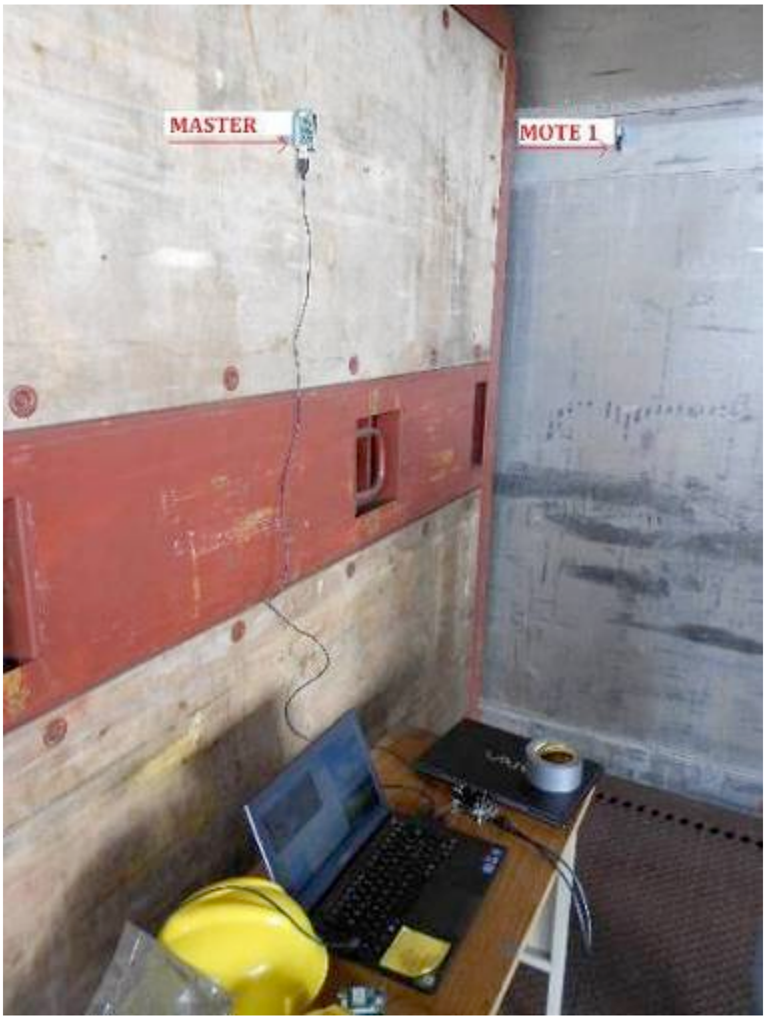


Figure 33 inside car master and node 1

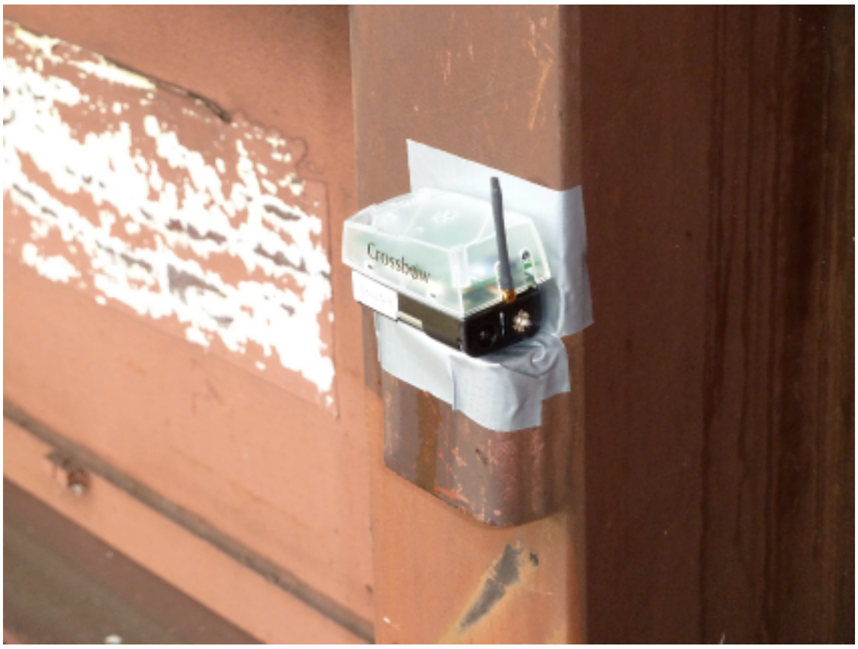


Figure 34 Box for sensor's protection

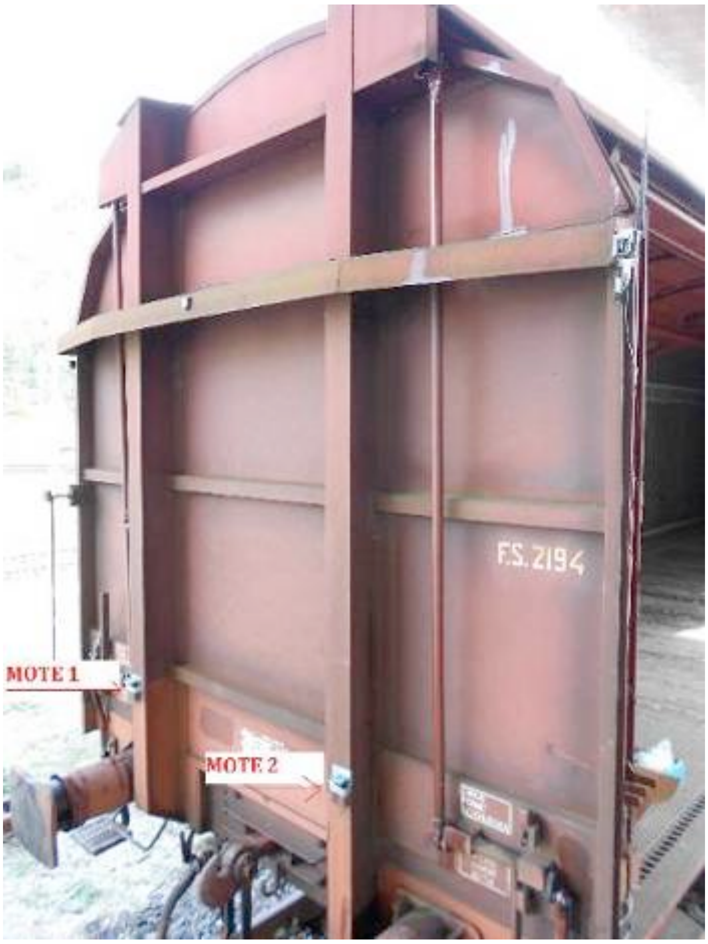


Figure 35 motes Networks 2 (MicaZ)

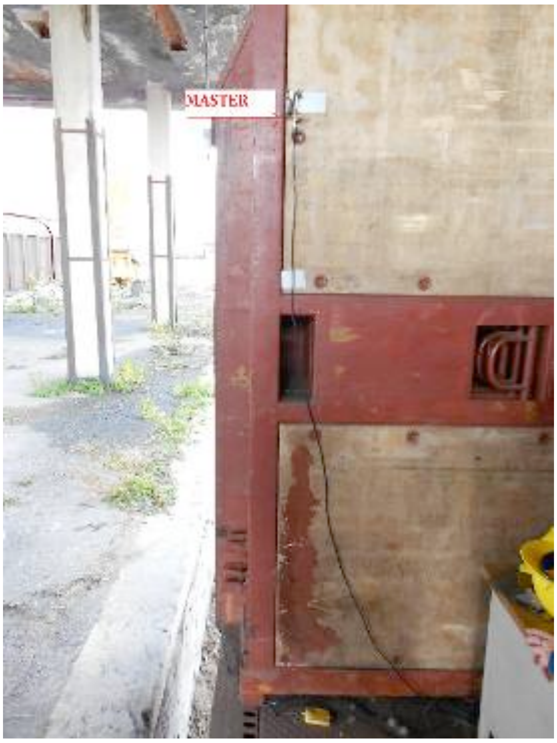


Figure 36 Master's MicaZ network

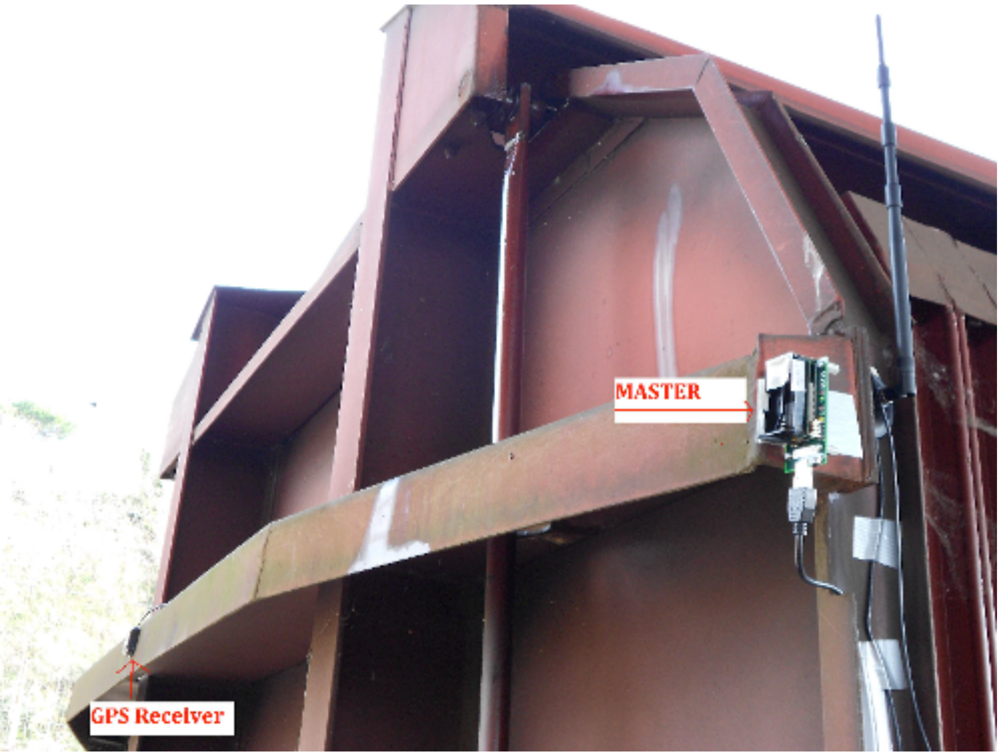


Figure 37 Third Network - GPS and Master

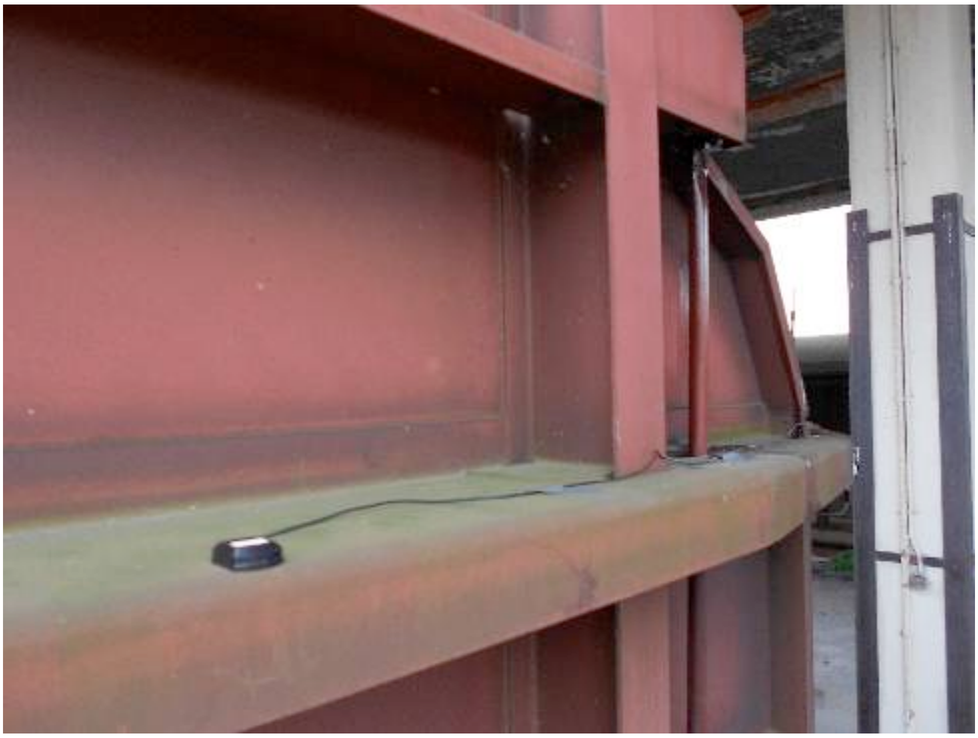


Figure 38 GPS Receiver

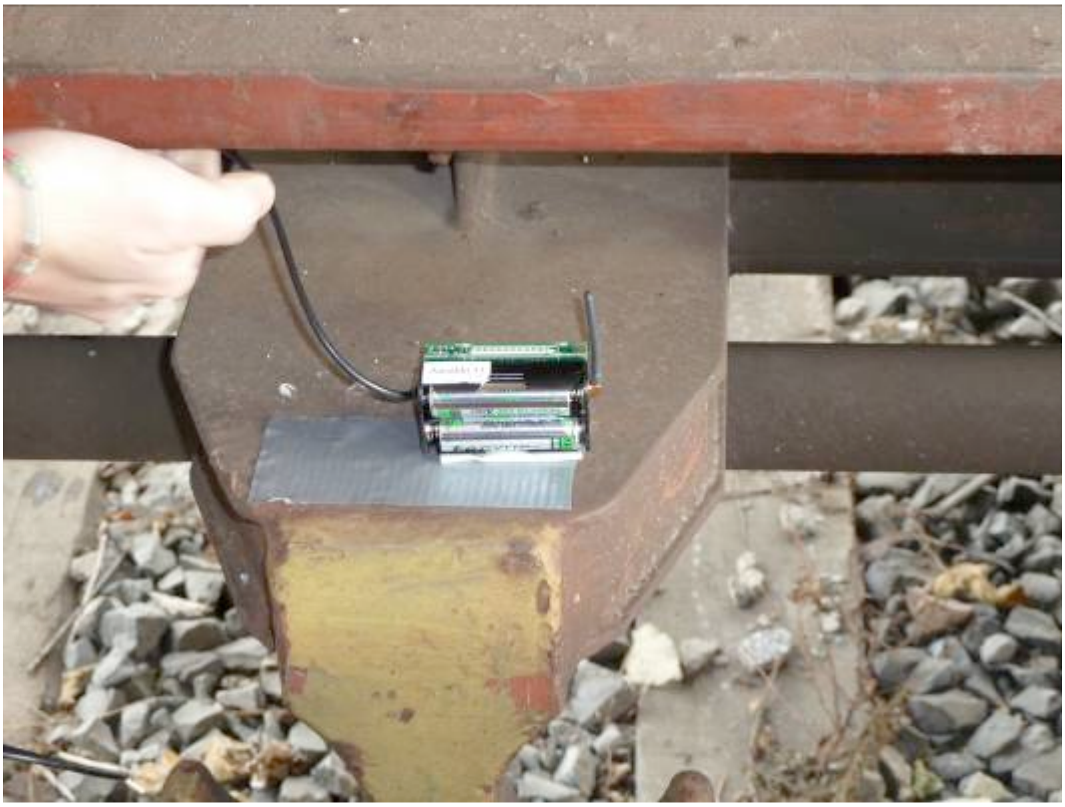


Figure 39 Mote GPS

For the wireless connection between the two PC one on board and one in the control room, is installed an Ethernet Wireless with a Signal Amplifier on a car, figures 40 and 41. It was configure a private network wireless between the two PC.

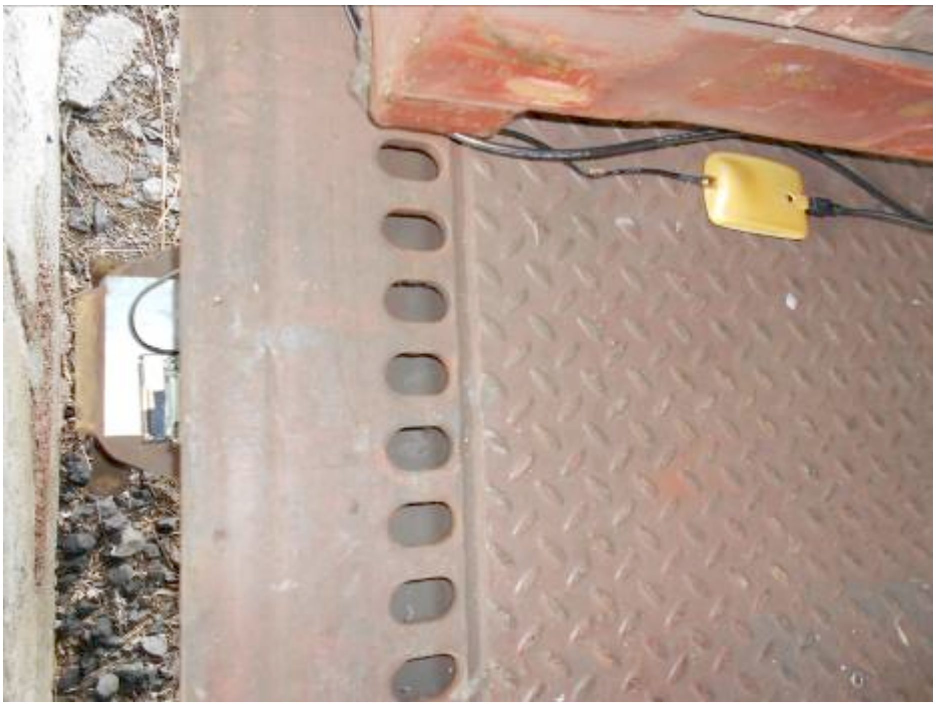


Figure 40 Ethernet Wireless

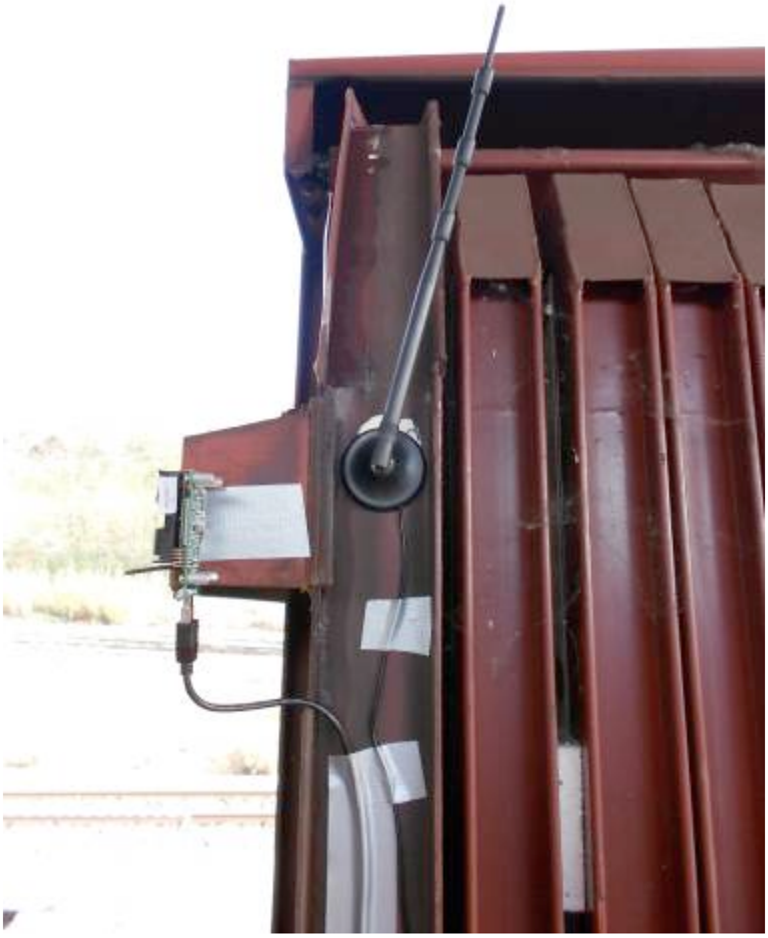


Figure 41 The antenna

9.2.2 Test cases

The test cases for the experimentation are follow:

1) The first test is when the car is stopped, this in order to verify the connect between nodes and PCs.

2) The second test is always when car is stopped, but to test the exceeding temperature threshold.

3) The third test is in car movement and test the exceeding acceleration threshold.

4) The fourth is a security test, it verifies that an intruder node is unable to participate at session.

9.2.2.1 Test for node connection

In the figure 42 is showed the screen of the networks, the net 1 that measures temperature and humidity and the network 2 that measures the acceleration. In the figures 43, 44 and 45 is the verification that all nodes measure the parameters and communicate with the master.

In figures 46 and 47, are showed the result of GPS coordinates (from the result file) and the corresponding coordinates on the map.

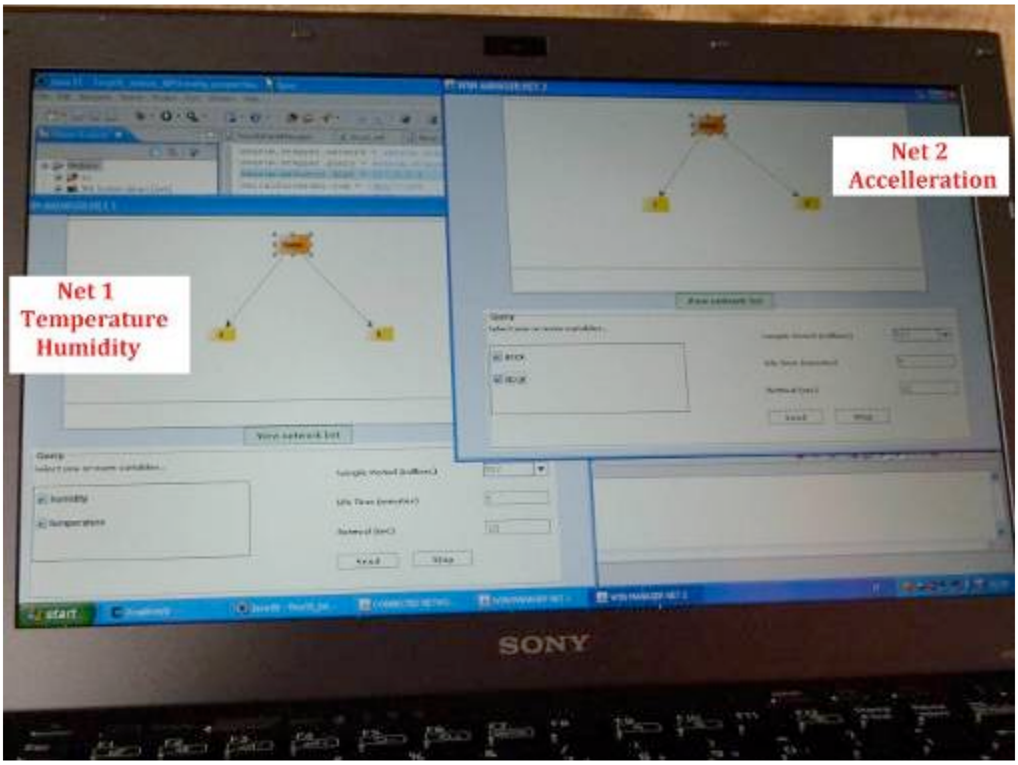


Figure 42 The nets

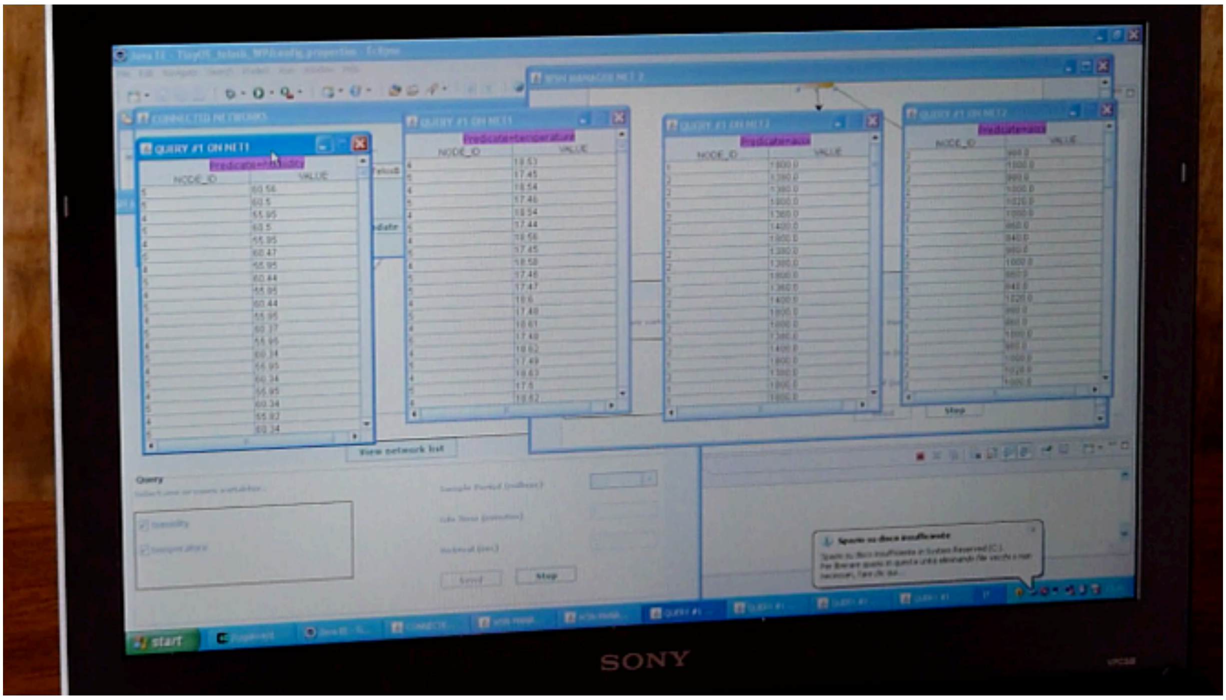
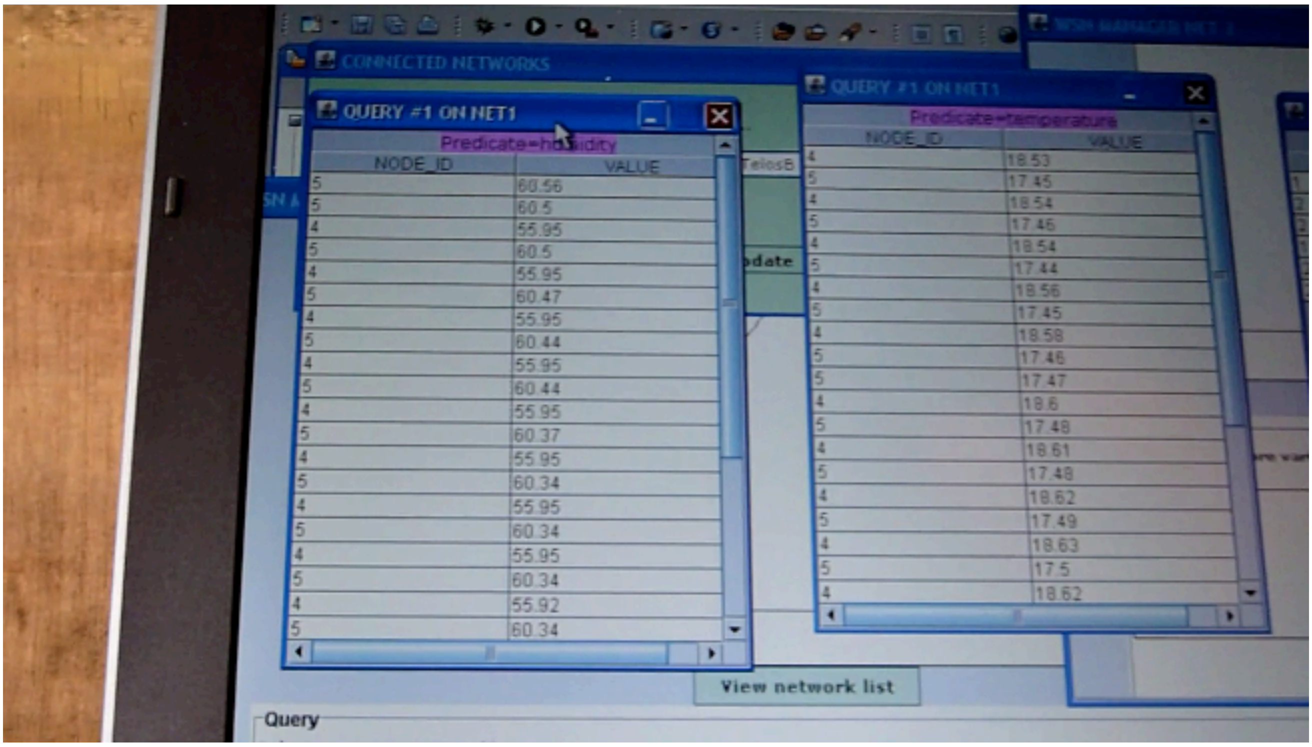


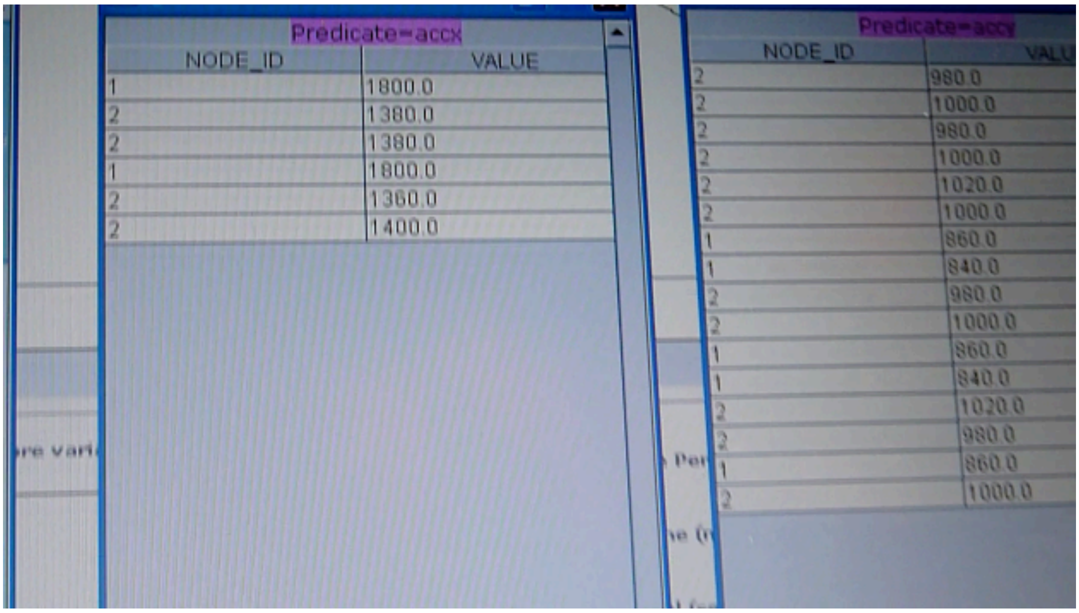
Figure 43 The parameters



Predicate=humidity	
NODE_ID	VALUE
5	60.56
5	60.5
4	55.95
5	60.5
4	55.95
5	60.47
4	55.95
5	60.44
4	55.95
5	60.44
4	55.95
5	60.37
4	55.95
5	60.34
4	55.95
5	60.34
4	55.95
5	60.34
4	55.92
5	60.34

Predicate=temperature	
NODE_ID	VALUE
4	18.53
5	17.45
4	18.54
5	17.46
4	18.54
5	17.44
4	18.56
5	17.45
4	18.58
5	17.46
5	17.47
4	18.6
5	17.48
4	18.61
5	17.48
4	18.62
5	17.49
4	18.63
5	17.5
4	18.62

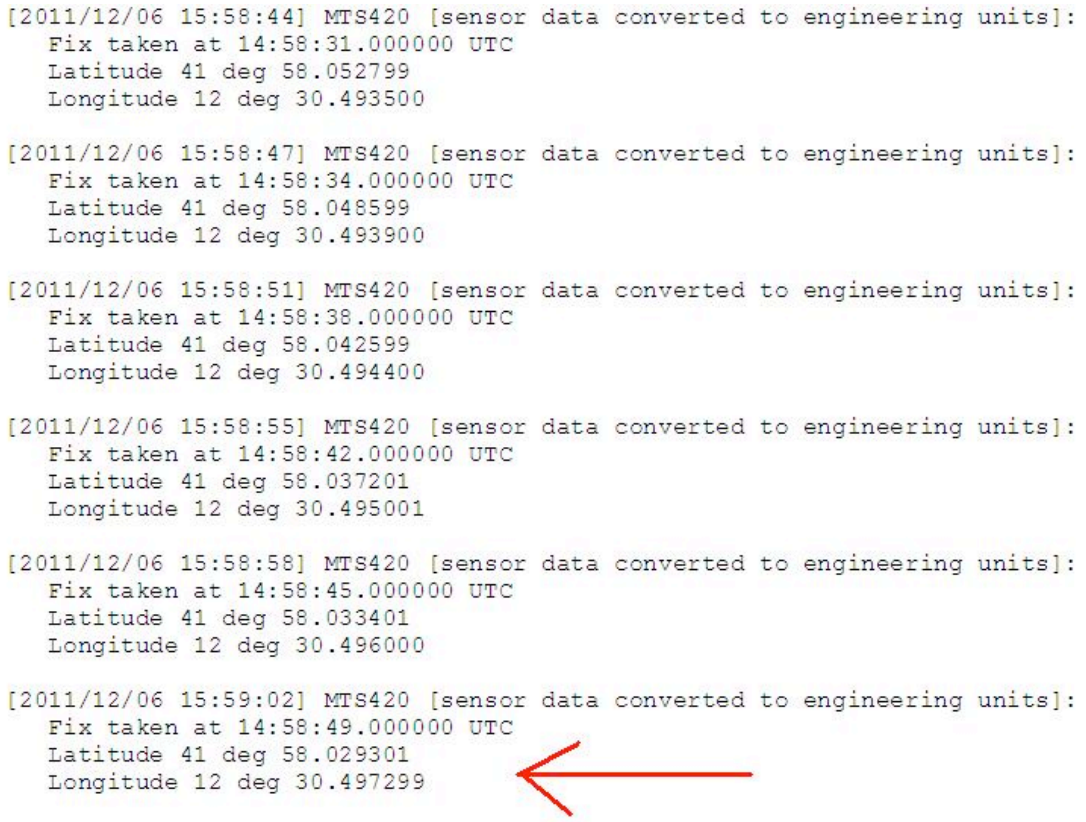
Figure 44 Humidity and Temperature



Predicate=accx	
NODE_ID	VALUE
1	1800.0
2	1380.0
2	1380.0
1	1800.0
2	1360.0
2	1400.0

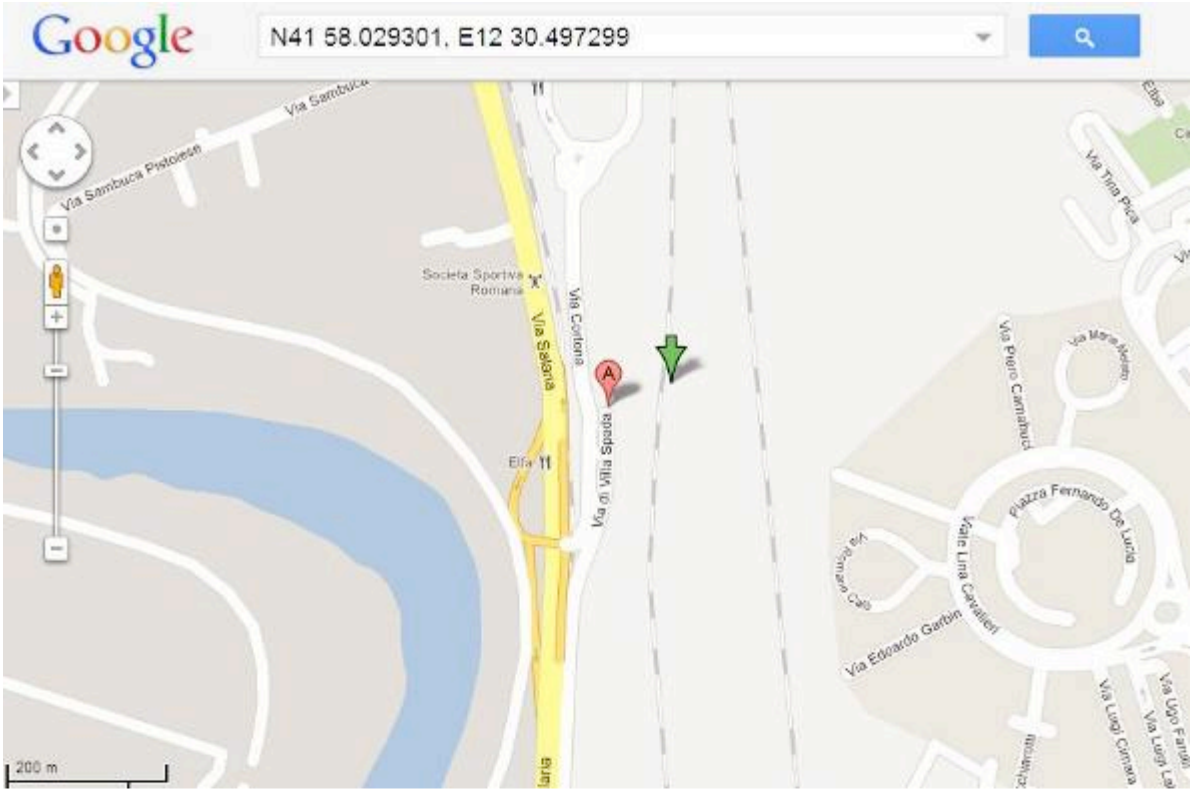
Predicate=accy	
NODE_ID	VALU
2	980.0
2	1000.0
2	980.0
2	1000.0
2	1020.0
2	1000.0
1	860.0
1	840.0
2	980.0
2	1000.0
1	860.0
1	840.0
2	1020.0
2	980.0
1	860.0
2	1000.0

Figure 45 Acceleration



```
[2011/12/06 15:58:44] MTS420 [sensor data converted to engineering units]:
   Fix taken at 14:58:31.000000 UTC
   Latitude 41 deg 58.052799
   Longitude 12 deg 30.493500

[2011/12/06 15:58:47] MTS420 [sensor data converted to engineering units]:
   Fix taken at 14:58:34.000000 UTC
   Latitude 41 deg 58.048599
   Longitude 12 deg 30.493900

[2011/12/06 15:58:51] MTS420 [sensor data converted to engineering units]:
   Fix taken at 14:58:38.000000 UTC
   Latitude 41 deg 58.042599
   Longitude 12 deg 30.494400

[2011/12/06 15:58:55] MTS420 [sensor data converted to engineering units]:
   Fix taken at 14:58:42.000000 UTC
   Latitude 41 deg 58.037201
   Longitude 12 deg 30.495001

[2011/12/06 15:58:58] MTS420 [sensor data converted to engineering units]:
   Fix taken at 14:58:45.000000 UTC
   Latitude 41 deg 58.033401
   Longitude 12 deg 30.496000

[2011/12/06 15:59:02] MTS420 [sensor data converted to engineering units]:
   Fix taken at 14:58:49.000000 UTC
   Latitude 41 deg 58.029301
   Longitude 12 deg 30.497299
```

Figure 46 GPS coordination from result file

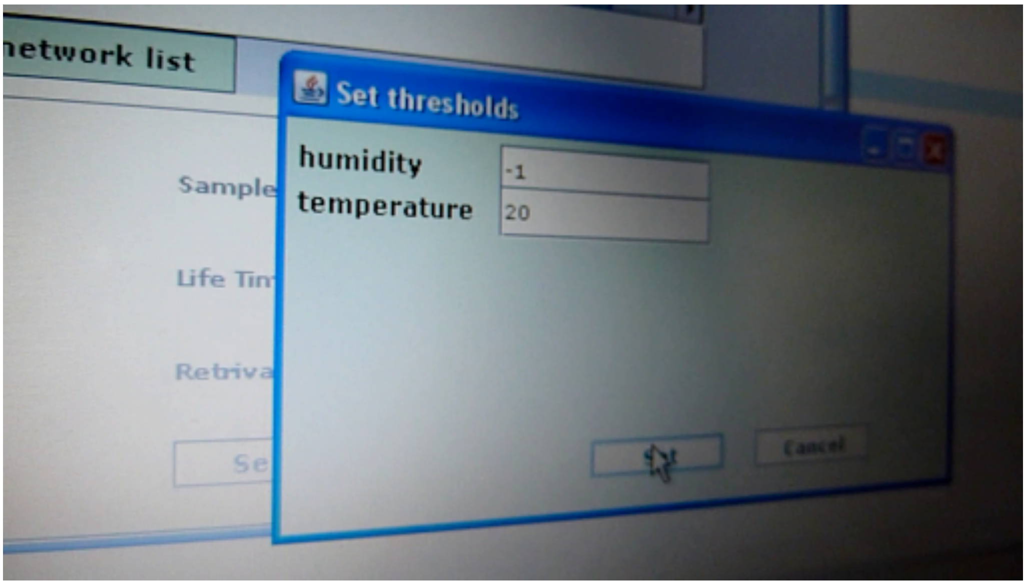


N41 58.029301, E12 30.497299

Figure 47 Corresponding coordinates on map

9.2.2.2 Test exceeding temperature threshold

In the figure 48 is showed the screen to set the threshold. In the figure 44 is possible to see that the temperature is about 18 C, so is configured a threshold about 20 C. With a stress of a temperature sensor (figure 49), the threshold is exceeded and this is signalled by the application (figure 50).



Set thresholds

humidity -1

temperature 20

Figure 48 Set Threshold

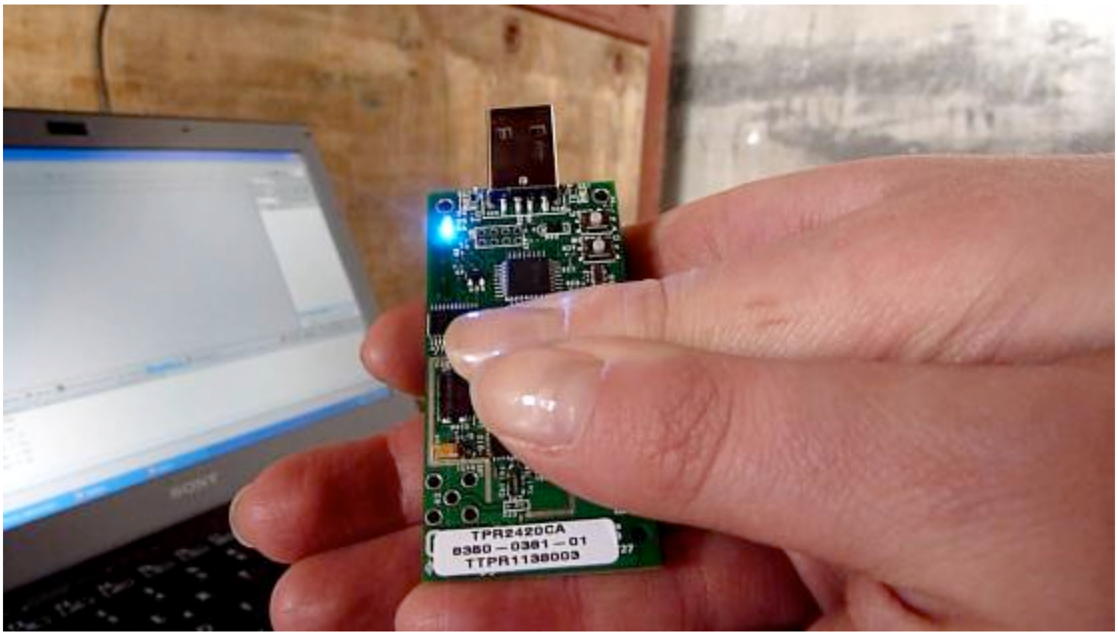
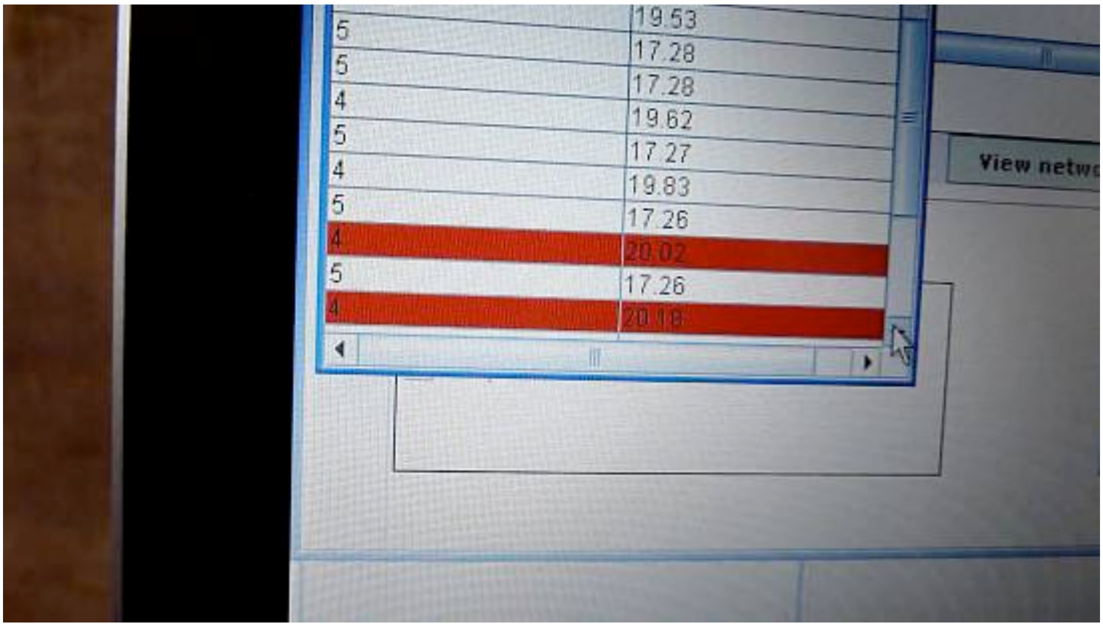


Figure 49 Sensor stress

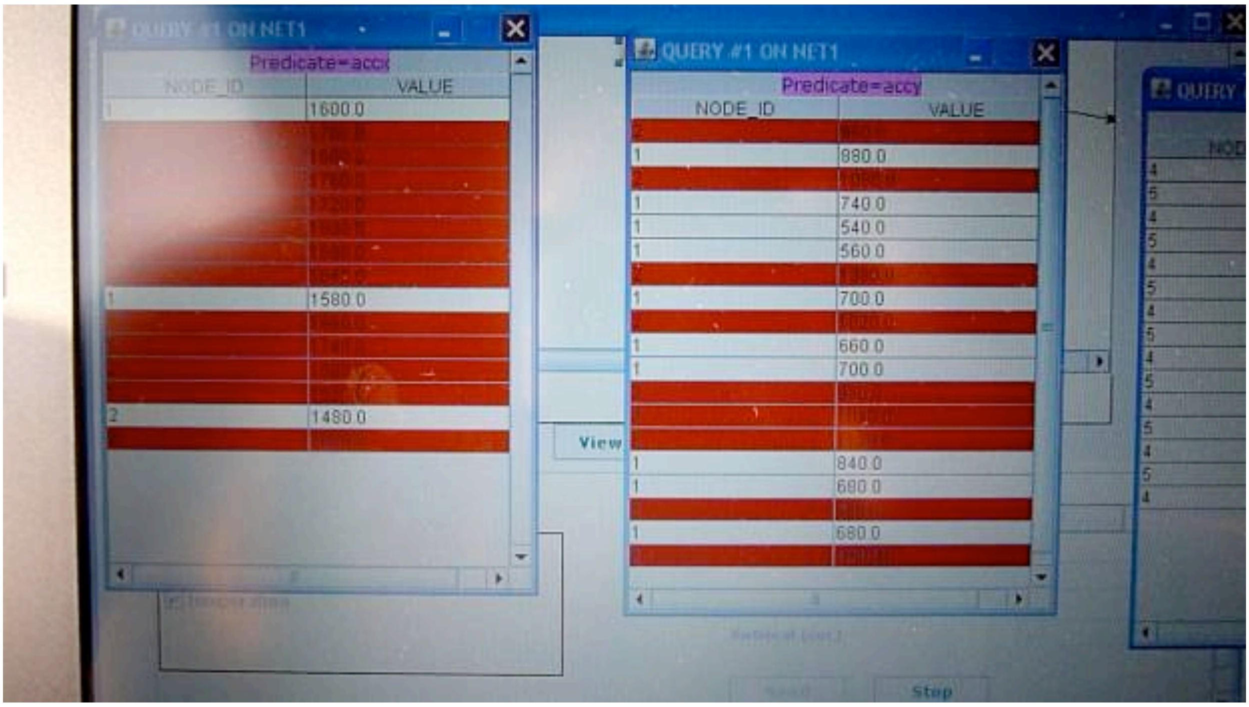


	19.53
5	17.28
5	17.28
4	19.62
5	17.27
4	19.83
5	17.26
4	20.02
5	17.26
4	20.18

Figure 50 Exceeding threshold

9.2.2.3 Test exceeding acceleration threshold during car movement

In this test is verified the exceeding acceleration during car movement. The setting of parameter for accelX and accelY according to the values registered during the experimentation with stopped car are: accelX=1600, accelY=900. In the figure 51 is showed the exceeding.



Predicate=accx

NODE_ID	VALUE
1	1600.0
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
1	1580.0
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
2	1480.0
[illegible]	[illegible]

Predicate=accy

NODE_ID	VALUE
2	[illegible]
1	880.0
2	[illegible]
1	740.0
1	540.0
1	560.0
2	[illegible]
1	700.0
[illegible]	[illegible]
1	660.0
1	700.0
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
1	840.0
1	690.0
[illegible]	[illegible]
1	680.0
[illegible]	[illegible]

Figure 51 Exceeding acceleration

9.2.2.4 Test intruder node

Before starting application the master and motes start the ECDH protocol in order to established a secret key. This test aims to prove the robustness of this protocol.

Infact at the beginning the master knows the numbers of nodes that will participate to the protocol and their ID Number (this is established at the deployment of the system). If master becomes aware that there is an intruder node, it toggle a red led and stops the communication with all nodes. In this example there are nodes with ID 1 and 2, the trusted node (Figure 52), an untrusted node with ID 10 (Figure 53) and the master (Figure 54).

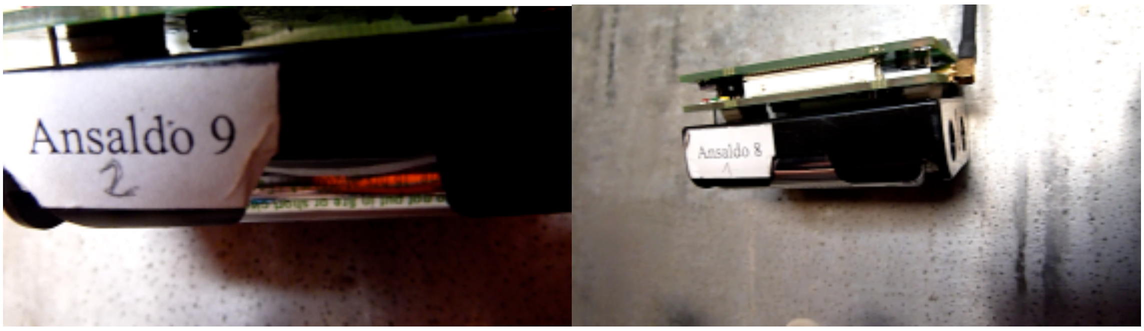


Figure 52 Node 1 and 2

Figure 53 Node untrusted

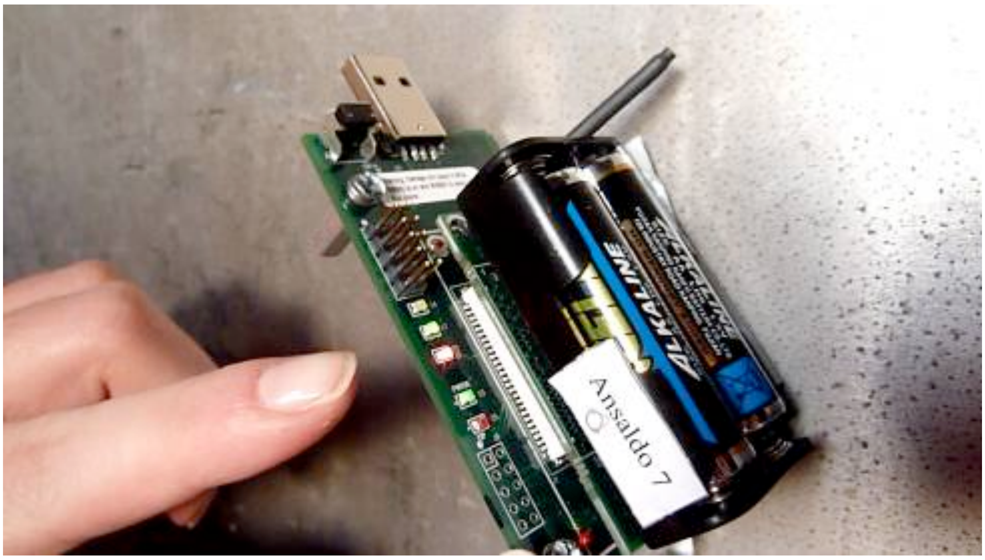


Figure 54 Master node

When the master was rebooted, it starts the protocol. At the end, all motes have the green led figure 55 and 56 , but the master toggle the red led instead green, figure 57. This notify the presence of intruder.

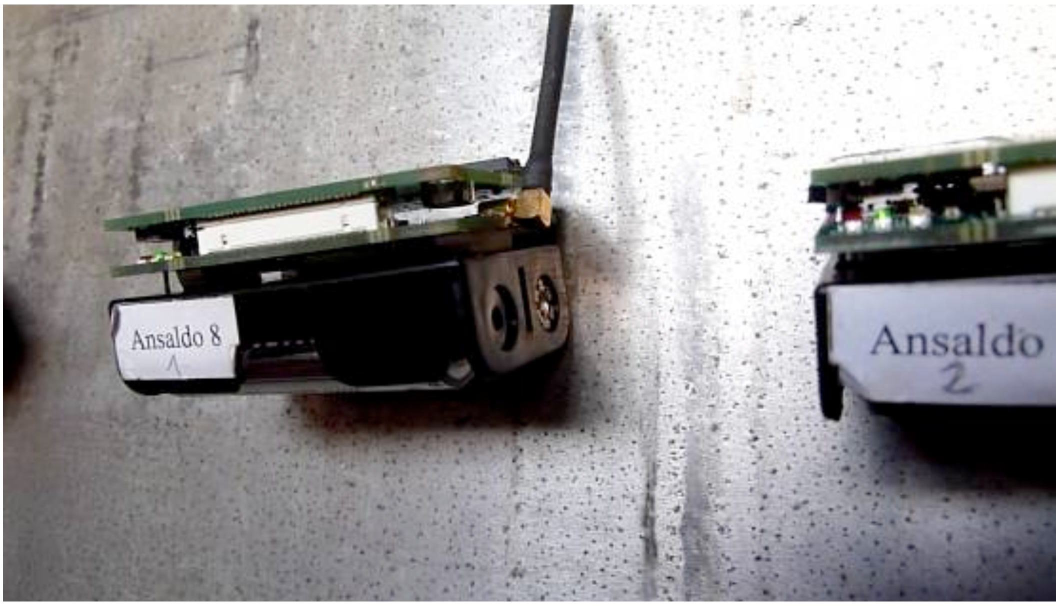


Figure 55 led green after protocol

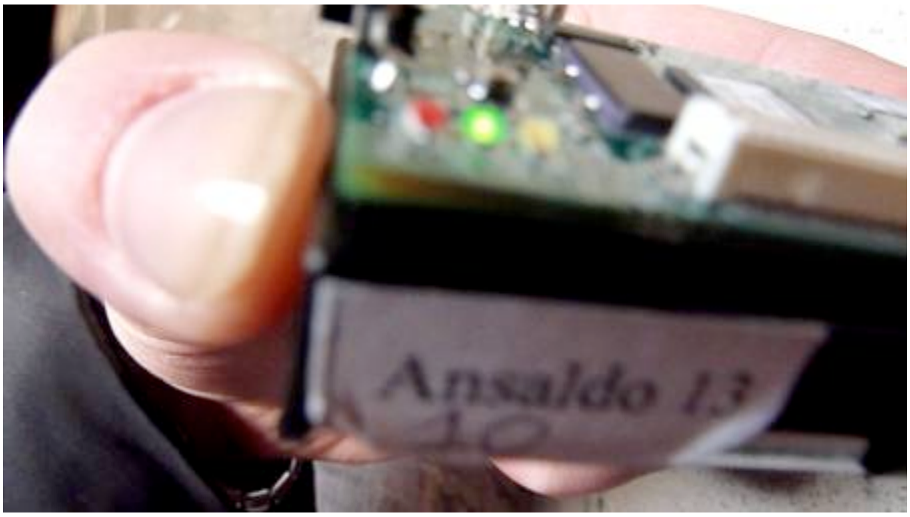


Figure 56 Led green intruder

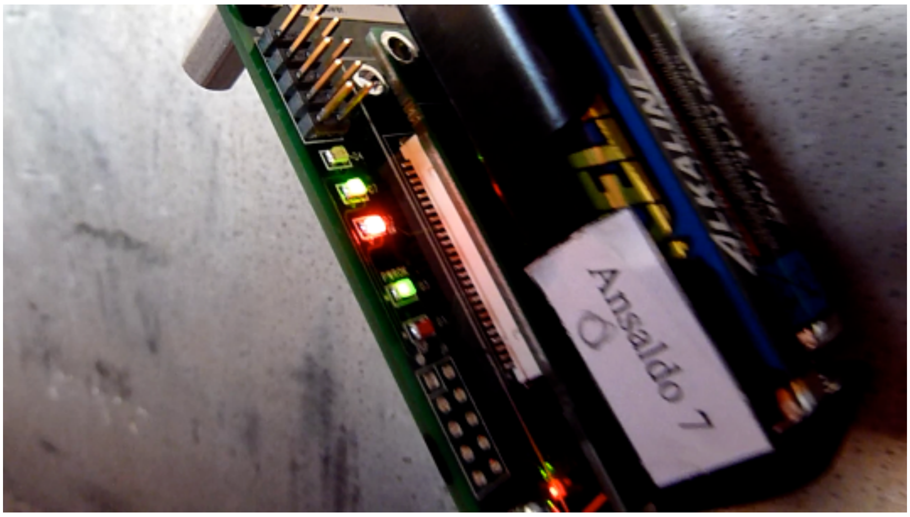


Figure 57 Master notifies the intruder

10.3 Integration with other prototypes

This platform for heterogeneous wireless sensors network is very flexible and it is possible to interfacing with the pSHIELD middleware. As matter of fact the mediator generates two file XML one for the topologies and information of each network, one for the query results. The pSHIELD middleware will be able to read this file, for its composition, and derive the informations for take a decision.

10 Conclusions

The document provides the pilot demonstrators developed during the pSHIELD project, for several level of the system: power node, radio, middleware and implementation of a monitoring platform. The aim is to describe the different technologies, for each there was a description and in some cases an experimental demonstration. Each of this will be the input for the next nSHIELD project.

References

[1] A. Perrig, R. Szewczyk, V. Wen, D. Culler, J. D. Tygar, *SPINS: Security Protocols for Sensor Networks*, 7th Annual International Conference on Mobile Computing and Networks, July 2001

[2] J.Girao, D. Westhoff, E. Mykletun , T.Araki, *TinyPEDS: Tiny persistent encrypted data storage in asynchronous wireless sensor networks*, Ad Hoc Networks, Volume 5, Issue 7, September 2007, Pages 1073-1089, ISSN 1570-8705

[3] S.Alam, *Security Proxy for User-centric Internet of Things,* Wireless World Research Forum (WWRF) #25, Kingston, UK, 16.-18. November 2010

[4] H.Wang, B. Sheng, C.C. Tan and Qun Li, *WM-ECC: an Elliptic Curve Cryptograph Suite on Sensor Motes*, Technical report, Oct. 30, 2007

[5] V. Casola, A. De Benedictis, A. Mazzeo and N. Mazzocca, *SeNsIM-SEC: security in heterogeneous sensor networks, May 2011, SARSSI2011*

[6] Kirk Martinez, Jane K. Hart, and Royan Ong. Environmental sensor networks. *Computer*, 37(8):50–56, 2004.

[7] B. Warneke, M. Last, B. Liebowitz, and K.S.J. Pister. Smart dust: communicating with a cubic-millimeter computer. *Computer*, 34(1):44–51, Jan 2001.

[8] Bemporad A. and Di Cairano, S., "Optimal Control of Discrete Hybrid Stochastic Automata", *Proceedigns of ACM International Conference on Hybrid Systems: Computation and Control (HSCC05)*, pp. 151-167, Zurich, Switzerland, 9-11 March, 2005

[9] Bemporad A. Di Cairano S. and Giorgetti N., "Model Predictive Control of Hybrid Systems with Applications to Supply Chain Management", *Proceedings of 49th Convegno Nazionale ANIPLA*, Naples, Italy, Nov, 2005