THE PROTOTYPE PARTLY REPRESENT THE USE CASE FROM MUSHFIQ, WHICH HE PRESENTED IN SEPTEMBER OSLO MEETING

The goal of the use case was:
- access to infrastructure and thereby information
- interoperable security when different and diverse parties are involved

This prototype addresses the first goal. The aim of this prototype implementation is to securely integrate on-train installed sensors with the Telenor Shepherd Platform by ensuring SPD requirements established in Dxx. The prototype follows the ETSI TS 102.690 specifications. The implementation contains the micro node and power node. Instead of reinventing the wheel from scratch, we utilize the available sensor and embedded hardware for this prototype implementation.

Micro Node – Sun SPOT Sensor Platform
The implementation uses the Sun SPOT sensor platform as micro node. Sun SPOT is a useful platform for developing and prototyping application for sensor network and embedded system. Sun SPOT is suitable for application areas such as robotics, surveillance and tracking. The main units are Sunspot devices with embedded sensors and base station. Each Sunspot has a so-called eSPOT with battery, while the base station is not equipped with battery and must be powered from the host computer via an USB cable. The Sunspot does not need to run any operating system, it needs only JVM that runs on bare metal, and executes directly out of ash memory.
Stack-boards composed of specific sensors and actuators such as accelerometers, light sensor and temperature sensor. Fig. 1 shows the hardware components of a processor board:

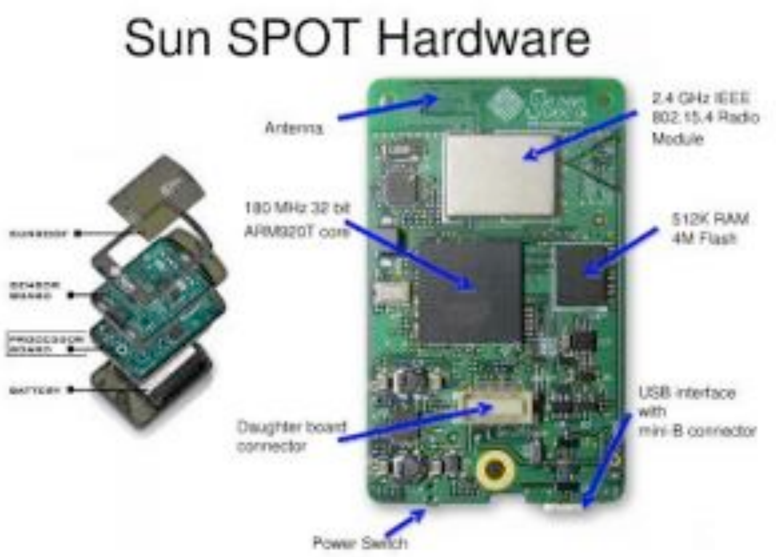
Sun SPOT Hardware
Antenna
2.4 GHz IEEE 802.15.4 Radio Module
180 MHz 32 bit ARM920T core
512K RAM 4M Flash
Daughter board connector
USB interface with mini-B connector
Power Switch

Figure 1. Micro Node - Sun SPOT hardware

The components are:
• 180MHz for 32-bit ARM920T core processor with 512K RAM and 4M Flash, runs on Squawk
• 2,4GHz based IEEE 802.15.4 radio (radio ChipCon TI CC2420) which is integrated in the antenna
• USB interface for connecting to a host computer
• 3.7V battery (720 mAh)
• Sleep mode (32 uA)

Fig. 2 shows the hardware components of a sensor board:

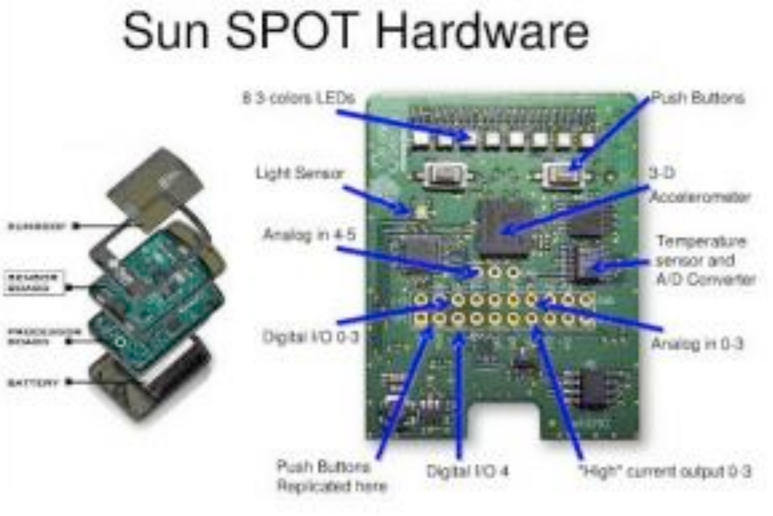

Figure 2. Hardware component of a sensor board

The components are as follows:
• 3 axis accelerometer
• 8 tri-color LEDs
• 2 push-buttons control switches
• 5 digital I/O pins
• 6 analog inputs
• 4 digital outputs

The sensor board has the following sensors integrated into it:
• **Temperature Sensor:** Chip-type is ADT7411 sensor that measures temperature with ADC. ADC is integrated into eDemo, and can measure temperatures between -40°C to +125°C.
• **Accelerometer sensor:** 3-axis accelerometer of the type LIS3L02AQ, designed by ST Micro Systems and is in eDemo Board. This sensor can measure the x-axis, y-axis and z-axis in the direction up and down with the value either ±2G or ±6G. When the Sunspot is at rest, it measures x = y = 0 and z = 1G.
• **Light sensor:** It is of the type TPS851, designed by Toshiba. The sensor can measure the voltage between 0.1V (dark) - 4.3V (light), and converts the voltage to the brightness of Luminance (lx) 3.

The supports sun SPOT Java Virtual Machine (JVM) which is called Squawk. Squawk is open source and has been written in the Java programming language. It is a virtual machine, and is a highly portable Java VM. Figure 3 below shows the architecture of Squawk. The advantage of Squawk is that it can run on bare metal instead of being run on top of the operating system. This means that applications can be isolated and be treated as application objects. This allows multiple applications running on the same virtual machine. Squawk also supports CLDC 1.1 that facilitates connectivity to mobile phones.

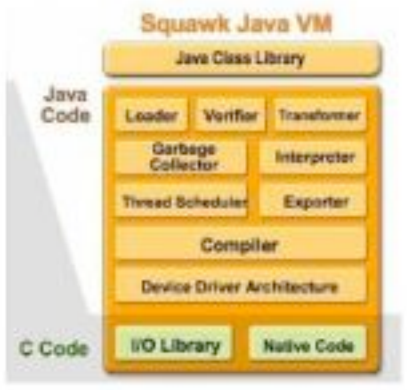

Figure 3. Squawk Java Virtual Machine.

Personal Node – VIA Embedded Board

The VIA EPIA N700 is a compact, low heat, power-efficient Nano-ITX board (fig. 4), ideal for compact industrial PCs and embedded automation devices. The board is integrated with the VIA VX800 media system processor, an all-in-one chipset solution that provides an extensive feature set while using less real state, helps to make the VIA EPIA N700 a superb choice for compact systems.

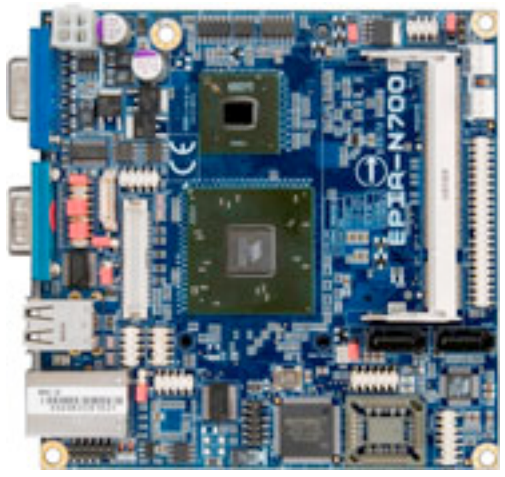

Figure 4. The Nano-ITX board.

It is based on Nano-ITX form factor (12cm x 12cm. VGA, USB, COM, Compact Flash (CF) and Gigabit network ports are provided on the board to help reduce system foot-print size and eradicate cluttered cabling for improved air-flow and enhanced stability in always-on systems.

The VIA VX800 offers an integrated DirectX9 graphics core and excellent hardware accelerated video playback for MPEG-2, WMV9, VC1 video formats. An on-board VGA port is provided along with support for DVI and a multi-configuration 24-bit, dual channel LVDS transmitter, enabling display connection to embedded panels. The VIA EPIA N700 is equipped with a power-efficient 1.5GHz VIA C7, supports up to 2GB of DDR2 system memory and includes two onboard S-ATA connectors, USB 2.0, COM and Gigabit LAN ports. Expansion includes a Mini-PCI slot with an IDE port, additional COM and USB ports and PS/2 support available through pin-headers.

The VIA EPIA N700 offers total system stability at extreme temperatures ranging from -20°C to 70°C, an ideal solution for our Norwegian rail use case to meet the extreme weather condition of Norway.

The implementation on uses Ubuntu Linux Kernel 2.6.32-24-generic and Java runtime environment (JRE) 1.6 for development.

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 (COs) [32]. This means that any the pluggable component can be connected, and be integrated in Shepherd® platform as a Connected object (CO). Fig. 5 depicts the overview of Shepherd® platform.

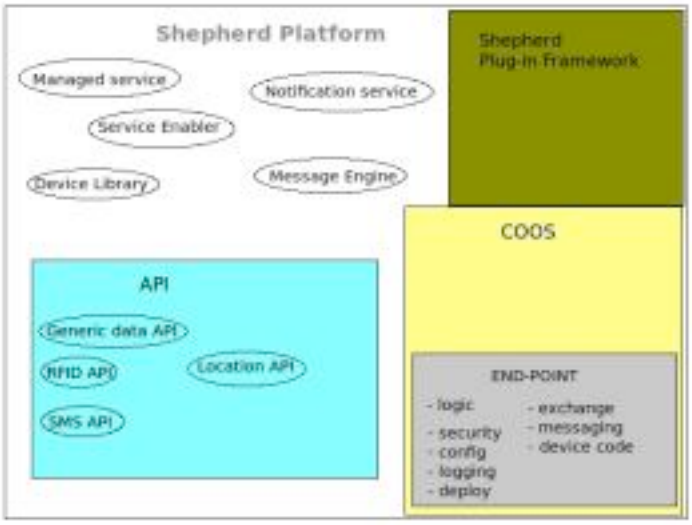

Figure 5. Overview of the shepherd platform.

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

Implementation of the connectivity with Shepherd

Shepherd® offers two methods for establishing connection. These include:

1. **HTTP Connection API** - This mechanism establishes a direct connection to the Shepherd by using the HTTPS protocol. With this method, it requires the development of the HTTP API of

the object. Shepherd accepts both methods POST and GET. When the connection is established, the Shepherd sends a response code back to that object as a confirmation of success or failure of reception. To be able to connect to the Shepherd, the "device object" is identified with an Application ID and an Object ID.

2. **Connected Objects Operating System (COOS)** - is an open source and has been written in Java. When using the COOS instance, the applications can connect to Shepherd in a secure, reliable and stable manner. In particular, it is important in this respect that eavesdropping by third parties is not possible when using COOS. Reliable in the sense that it is an M2M network, and communication between objects with COOS instance and the Shepherd will not be interrupted or delayed more than necessary. Thereby, ensuring a stable environment for the users and the applications. It requires therefore, developing an application using COOS, so this can apply to device so it can communicate with the Shepherd. From a programming perspective, "Connected Objects Operating System (COOS) is an application distributed in a container so that it can enable data exchange between the object and the Shepherd. In COOS concept, every component that is integrated, and can be pluggable is called "Connected Object (CO). This means that a COOS instance can have multiple Connected Object, and each COOS instance carries its own distinctive character. The Fig. 6 illustrates the relationship between COOS instance and CO.

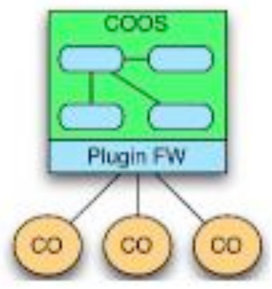

Figure 6. Relationship between COOS and CO.

The Fig. 7 presents the system overview. It shows the establishment of an intended two-way communication between Sun SPOT sensors and its base station, and also two-way communication between the embedded Linux system and the Shepherd Platform.

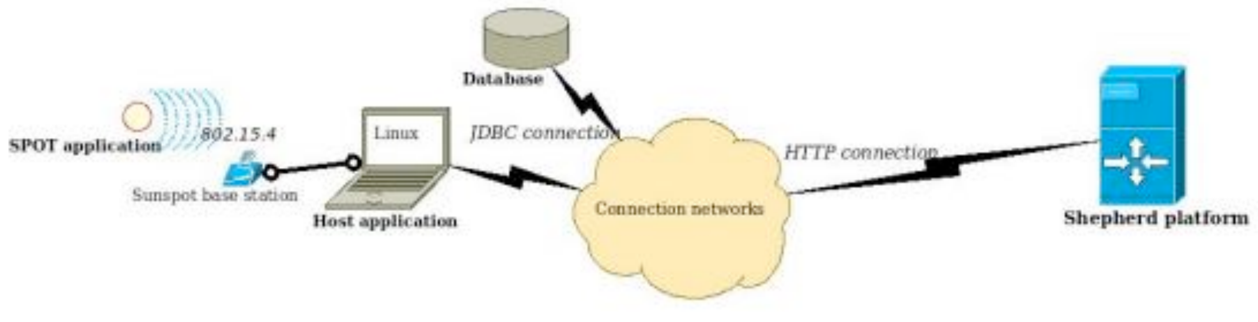

Figure 7. Overview of the implementation.

A Host application has been developed, that performs broadcasts every fifteen seconds. While, the spot application will detect the broadcasts every thirty seconds. But, it does not transmit the values to the base station after one minute has passed since the last envoy. When the values arrive, these will also be stored in cache. At the same time, the Host application sends out a request to Shepherd for receiving the values. The connection is opened until the application has received confirmation from the Shepherd of receipt. However, the values to be sent to Shepherd, only happens in every five minutes. The SPOT application is also designed to detect spot's battery level prior it using the wireless communication. If spot battery is either lower than -32 or greater than 32, then the MIDlet will be destroyed, terminated and a notification will be send to the concerned actors.