Deliverable Number D3.05.

Intelligent truck

WP 3 – Integration of novel intelligent harvesting systems operating in mountain areas

Task 3.5 - Intelligent transport truck

Revision: Final

Authors: Juan De Dios Díaz

Dissemination level	PU (Public)
Contributor(s)	Juan de Dios Díaz (ITENE)
Reviewer(s)	Daniele Magliocchetti (GRAPHIETECH) Martin Kühmaier (BOKU) Gerhard Pichler (BOKU)
Editor(s)	Raffaele De Amicis (GraphiTech)
Partner in charge(s)	ITENE
Due date	31-December-2015
Submission Date	02-February-2016

REVISION HISTORY AND STATEMENT OF ORIGINALITY

Revision History

Revision	Date	Author	Organisation	Description
1	31-December-2015	Juan de Dios Díaz	ITENE	First version
2	8-January-2016	Martin Kühmaier Gerhard Pichler	BOKU	Comments on different issues
3	5-January-2016	Daniele Magliocchetti	GRAPHITECH	Comments on different issues
4	12-January-2016	Juan de Dios Díaz	ITENE	second version
5	2-feb-2016	Juan de Dios Díaz	ITENE	Final version

Statement of originality

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both.

Index

1 Introduction 6
2 Use case and general description 7
3 Hardware Description 9
3.1 Raspberry Pi 2 Model B 9
3.2 GlobalSat BU-353-S4 USB GPS 10
3.3 Huawei E3256 USB GPRS 10
3.4 Belkin F8T013-1 11
3.5 Edimax ew-7811Un 11
3.6 CAEN R1240I - qID 12
3.7 Digitus 4-port USB 2.0 Hub 12
4 Software Description 13
4.1 Python libraries 13
4.2 Main classes. 13
4.3 Slope local database. 14
4.4 Communication with SLOPE cloud database. 16
5 User manual 17
5.1 Accessing the user interface 17
5.2 Main Screen (MS) 17
5.3 Test Screen (TS) 18
5.4 Configuration Screen (CS) 19
5.5 Manual RFID Reader Screen (MRS) 20
6 Troubleshooting 23
6.1 No GPS is found 23
6.2 No RFID Reader is found 23
6.3 No database is found in postgress 24
6.4 No GPRS connection is found 24

List of figures

Figure 1 : Use case....8

Figure 2 : Hardware elements of the SLOPE intelligent truck system. 9

Figure 3 : Raspberry Pi 2 Model B 9

Figure 4 : GlobalSat BU-353-S4 USB GPS. 10

Figure 5 : Huawei E3256 USB GPRS....10

Figure 6 : Belkin F8T013-1. 11

Figure 7 : Edimax ew-7811Un. 11

Figure 8 : CAEN R1240I – qID. 12

Figure 9 : Digitus 7-port USB 2.0 Hub. 12

Figure 10 : Main table of the system. 15

Figure 11 : Main Screen (MS). 17

Figure 12 : Test Screen (TS). 18

Figure 13 : Configuration Screen (CS). 20

Figure 14: Manual RFID Reader Screen (MRS). 21

Acronyms

CA	Consortium Agreement
DM	Data Manager
DBH	Diameter at breast height
GA	Grant Agreement
GA	General Assembly
OM	Operational Manager
PC	Project Coordinator
QRM	Quality and Risk Manager
SB	Stakeholders Board
TB	Technical Board
TL	Task Leader
WPL	Work Package Leader
RFID	Radio Frequency Identification
HF	High Frequency
UHF	Ultra-High Frequency
NFC	Near Field Communication
RPI2	Raspberry pi 2 model B
VPN	Virtual Private Network
USB	Universal Serial Bus

1 Introduction

This task describes the work performed under the task 3.5 of the SLOPE project.

In this task, an intelligent truck solution was developed in order to trace the logs during transportation from the storage point (forest, landings) to the mills where the logs are processed. This tracing is done using the RFID technology for identifying the logs, GPS to position the truck and GPRS to communicate with the SLOPE server in a real time solution.

In a typical operation, the driver of the truck first loads the truck, then identifies the logs, make the trip to the mill, and then read the logs again. This is explained in Chapter 2.

Chapter 3 describes which hardware is used in the solution. There are 6 main components: the main CPU, the Wi-Fi adapter, the GPS adapter, the GPRS adapter, the Bluetooth adapter and the portable RFID reader.

Chapter 4 describes the software used in the solution. In this chapter the Python libraries used in the project are presented, and the classes created in the solution are presented. Also, the structure of the local database is shown, and the interaction with the SLOPE cloud database introduced.

Chapter 5 presents the user manual. It explains the four main screen of the truck software: main screen, test screen, configuration screen and portable RFID reader screen.

Finally, chapter 6 is left for Troubleshooting description.

2 Use case and general description

Task 3.5 activities start when the trees had been processed (usually debranching, topping and crosscutting) and the logs are placed in a storage area (usually in the forest at the roadside or at the landing). Each of the logs has an RFID tag usually stapled to the log crosscut. This tag was integrated in the marking tree task or in the processor head activity. Those tags are linked in the database with an identification number, which relates to real data of the log, i.e. harvesting area, tree characteristics (e.g. wood quality, volume, size, DBH, etc.). More information about the RFID integration into the log, reading performance and tag survival can be found in Deliverable D3.01[1], D3.02[2], and D3.06[3].

The first activity of the truck driver is to load the logs on the truck. Once the truck is ready to depart, the driver activates the RFID reader and first of all reads the truck RFID tag, which includes a truck id. This action indicates the system that all the tags that are going to be read next need to be assigned to the truck. The truck tag is meant to be placed in the truck, i.e. in the dashboard of the vehicle.

After the truck tag has been read, the driver proceeds to read all loaded RFID logs. The list of the read tags can be checked if the Truck system is connected to a screen.

Then, the driver reads one more time the truck RFID tag. The system recognises this action as a signal to indicate that all logs have been completed, and that the route can start. All unique ids received are stored at that moment in a local database.

Next, the driver starts driving towards the mill. The systems start measuring the GPS position, date and time, and stores them in a local database with a predefined time interval. Therefore, a number of points are recorded during the route.

Once the driver arrives to the mill, the procedure is repeated to assure that all logs arrived correctly. First the truck tag is read, then the log tags, and finally truck tag again. This finalises the operation.

Finally, during the whole procedure of the activity, the system is trying to send all information in the local database to the SLOPE database. To achieve this, a GPRS with a SIM card is included in the solution. The information is stored directly in the SLOPE database with its web services.

The whole process chain of log identification and tracking is shown in Figure 1.

[1] SLOPE Deliverable D3.01. Portable RFID tag reader

[2] SLOPE Deliverable D3.02. RFID tag test

[3] SLOPE Deliverable D3.06. RFID tag survival test along the supply chain

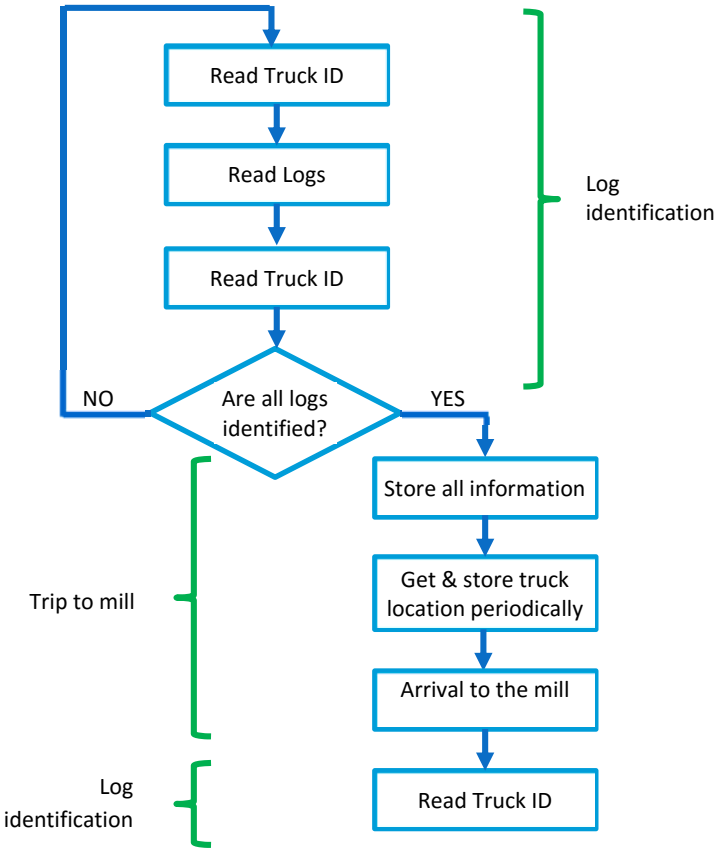

Figure 1 : Use case

3 Hardware Description

The hardware components of the system are composed of a low-cost multipurpose computer, a GPS receiver, a USB Wireless cellular modem, a Bluetooth device, a USB adapter and a portable UHF RFID reader. All the components are described in the next subchapters.

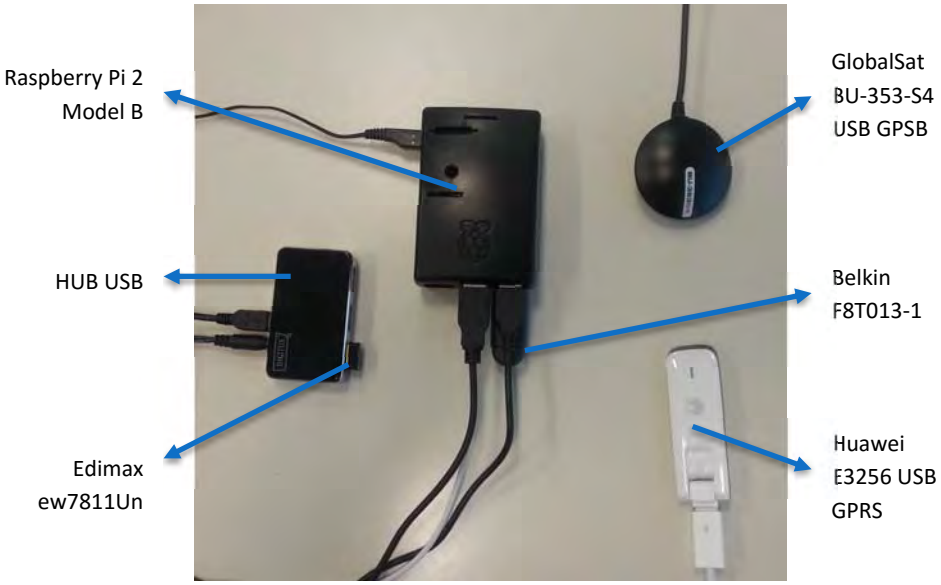

Figure 2 : Hardware elements of the SLOPE intelligent truck system.

3.1 Raspberry Pi 2 Model B

Figure 3 : Raspberry Pi 2 Model B

The raspberry pi 2 model B (RPI2) is a low-cost multipurpose computer with a 900MHz Quad core ARM V7 processor architecture, and 1GB of RAM. Several Linux distributions have been developed for it, allowing highly customizable applications.

It is manufactured by the Raspberry Pi Foundation (https://www.raspberrypi.org/).

The RPI2 will be used as the Intelligent Truck Core system. I will receive GPS information (via USB adapter) and RFID information (via Bluetooth adapter), store it in a local database, and send it to the SLOPE cloud server (via GPRS adapter). Also, the computer deploys a Wi-Fi hotspot (via Wi-F USB adapter) which allows to other systems to connect to the system using a VPN connection (Virtual Private Network).

3.2 GlobalSat BU-353-S4 USB GPS

Figure 4 : GlobalSat BU-353-S4 USB GPS.

The BU-353-S4 is a USB magnet mount GPS receiver, with low power consumption and good performance in canyons and dense foliage regions. It is connected to the RPI2 with a USB connector.

It is manufactured by USGlobalSat Inc. (http://usglobalsat.com/).

The BU-353-S4 will receive the latitude, longitude, altitude and time directly from the satellite connection. This information is received by the RPI2 and used to define current location.

3.3 Huawei E3256 USB GPRS

Figure 5 : Huawei E3256 USB GPRS.

The E3256 is a Huawei USB Wireless cellular modem, allowing for communication in GSM, GPRS, UMTS, EDGE, HSPA+ and DC-HSPA+. It allows up to 43.2 Mbps, and is connected to the rpi with a USB 2.0 connector.

It is manufactured by Huawei (http://consumer.huawei.com/en/).

The E3256 will allow access to internet to the system. To accomplish this, a SIM card from a mobile operator has to be installed in the device.

3.4 Belkin F8T013-1

Figure 6 : Belkin F8T013-1.

The Belkin adapter F8T013-1 allows to use wireless Bluetooth v2.0 capabilities. It is connected to the RPI2 using USB 2.0.

The adapter is manufactured by Belkin (https://www.belkin.com/).

The Belkin adapter will allow to connect to the RFID manual reader. The information sent by the reader will be received and integrated in the Truck system.

3.5 Edimax ew-7811Un

Figure 7 : Edimax ew-7811Un.

The Edimax ew-7811Un is a Wireless 802.11b/g/n USB adapter. It supports WPS, WPA2, and 802.1x.

The adapter is manufactured by Edimax (http://www.edimax.es/edimax/).

The Edimax adapter will allow the RPI2 to deploy a Wi-Fi Hotspot. Using this hotspot with a VPN connection, external systems like smartphones or laptops can connect to the RPI2 and the software running inside.

3.6 CAEN R1240I - qID

Figure 8 : CAEN R1240I – qID.

The R1250I is a portable UHF RFID reader, fully compatible with the EPC gen 2 RFID standard. The reader has USB and Bluetooth communication, integrated dual linear polarized antenna, ergonomic form factor and it is battery powered. The reader communicates with the rpi2 using Bluetooth communication.

The reader is manufactured by CAEN.

The reader allows to receive the information included in the memory section of a RFID tag which is inside the reading field range.

3.7 Digitus 4-port USB 2.0 Hub

Figure 9 : Digitus 7-port USB 2.0 Hub.

The Digitus USB 2.0 Hub has four 2.0 compliant USB ports, dimensions of 7cm x 4cm x 1.45cm and a power supply of 5V and 2A.

This USB hub is manufactured by ASSMANN Electronic GmbH.

This USB hub is needed to correctly supply the Wi-Fi adapter. Since the truck system already uses 3 other USB ports (Bluetooth, GPS and GPRS), the remaining power is not enough to correctly power up the Wi-Fi dongle. This was solved including an external USB hub with its own power source.

4 Software Description

The iTruck solution uses the following software components

- Raspbian Linux distribution as an operative system, a free operating system based on Debian and optimized for the rpi hardware.
- PostgreSQL as a database software. PostgreSQL is cross platform, open source and free to use.
- Python 3 as programming language, a general-purpose, object-oriented, high-level programming language defined for code readability. Python is free and open source.

4.1 Python libraries

The following python libraries are the main libraries used in the solution:

- **Tkinter**. It is the default python GUI (Graphic User Interface) package.
- **ElementTree** XML API. The ElementTree class can be used to wrap an element structure, and convert it from and to XML.
- **Psycopg2.** It is a postgres adapter for the Python language.
- **Gpsd and gps3.** Allow receiving GPS data and managing it with python.
- **Bluetooth.** Allows for Bluetooth usage and connectivity.
- **Requests:** Allows for http communication.
- **Rfid_manual_reader:** Custom library to add extra functionalities for the portable RFID reader activity.
- **Other packages:** time, threading, datetime, socket, json.

4.2 Main classes.

There are 8 classes implemented in the iTruck solution, which are described in Table 1.

Table 1. Classes description.

Gui	Main class. It generates the main Tkinter window and provide functionality to the buttons and secondary windows. The execution line of the thread is stopped in this instance when the program is running. A method “updateMe” is executed every 500ms to update the received values and status. The class creates the main window, test window, configuration window. It allows to store data in xml and retrieves it, and stores data in the local database.

	The GPS and RFID data is stored in queues, where the external workers introduce the received data. The queues are processed on the "updateMe" method.
cloudUploader	An instance of this class connects to the local database, looks for rows not uploaded to the SLOPE database, upload the information in the cloud, and if the result is correct it updates the local database. A specific column in the local database, "is_Sent" has been set to keep track of this.
databaseEntry	Used to create database objects. Each object is created as a structure, and includes id, truckId, time, longitude, latitude, altitude, speed, EPCData, isSent, and comments.
gpsPoint	Used to create GPS objects. Each object includes time, longitude, latitude, altitude, and speed.
gpsWorkerThread	Thread defined exclusively to check if new GPS data arrived, retrieve it and introduce it in a specific buffer inside the main Gui instance.
rfidWorkerThread	Thread defined exclusively to check if new RFID data from the fixed RFID reader arrived. It retrieves it and introduces it in a queue inside the main Gui instance.
gprsWorkerThread	Thread defined exclusively to check if there is communication to the online SLOPE.
manualRfidReader	An instance of this class is used to manage the RFID portable reader. It creates the bluetooth connection, and if it is successful, launches the RFID portable reader window. All information generated through this is stored in the queue of the main Gui instance.

4.3 Slope local database.

The truck local database is a postgress database. All of the information stored in the database is placed in only one table, with the following structure.

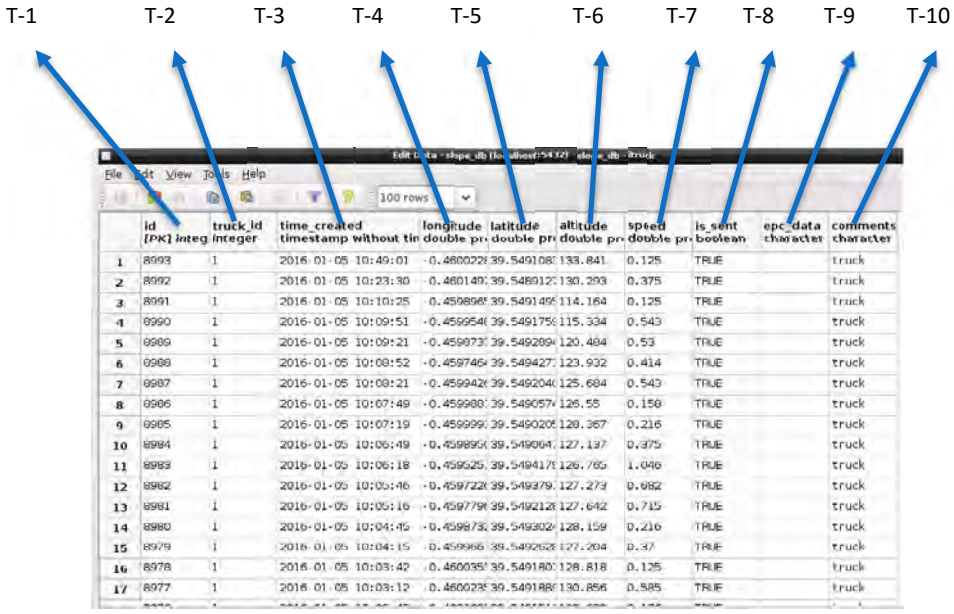

Figure 10 : Main table of the system.

- **T-1: Id (integer).** Used as the key of the table.
- **T-2: truck Id (integer).** Numeric value that is used to identify a truck.
- **T-3: time_created (timestamp).** Date and time value of the moment in which the data is taken. This value comes directly from the GPS device.
- **T-4: longitude (double).** This value comes directly from the GPS device.
- **T-5: latitude (double).** This value comes directly from the GPS device.
- **T-6: altitude (double).** This value comes directly from the GPS device.
- **T-7: speed. (double).** This value comes directly from the GPS device.
- **T-8: is_sent (boolean).** This value is used to keep track of which values have already been stored in the SLOPE database. The truck system has a routine to send all datasets which have this value as FALSE. If the point is stored successfully, the value is changed to TRUE.
- **T-9: epcData (character).** Value included in the RFID tag memory. 10 RFID tags read at the same time will produce 10 rows, each with a different epcData. If the dataset is a truck position (and not an RFID entry), the epcData field is empty.
- **T-10: comments (character).** Used to specify type of db entry. For truck position, the defined value is “truck”. For RFID entry with the manual reader, the value stored is “man”. For a RFID entry with a fixed reader, the value stored is “fixed”.

4.4 Communication with SLOPE cloud database.

The information stored locally in the database needs to be sent to the main SLOPE cloud server. The SLOPE cloud server has available webservices to allow reading and writing into the database, which are documented in the deliverable D5.01[4].

SLOPE FIS Web Service implements HTTP Basic authentication over SSL which means that every **connection is forced to use HTTPS connection**. Prototype's server uses self-signed certificate.

Therefore, the information to be send by the truck system is firstly converted in JSON format, and sent to the database via an http request. The Python library "**Requests"** is used for this action.

[4] SLOPE Deliverable D5.01. Inventory module of the FIS.

5 User manual

There are 4 main screens in the iTruck program. Although the system is designed to work without any supervision, a user interface was created to check the status, configure the solution and analyse the existing data.

5.1 Accessing the user interface.

There are several options to access the user interface:

- Direct connection of a monitor, keyboard and mouse with the DVI and USB port on the RPI2.
- Connection through Ethernet. The RPI2 has a VNC server running at: 192.168.0.2:0. Password: "p@ssw0rd".
- Connection though Wi-Fi. The RPI2 has also a Wi-Fi hotspot, with net name: "*slopeitruck*" and password "*slopeitruck*". Once it has been connected, the VNC server can be accessed with 192.168.42.1:0. Password: "p@ssw0rd".

Once the RPI2 has been accessed, the python code can be started with the IDE Genie, and the starting file is "itruck.py"

5.2 Main Screen (MS)

The main screen appears when the iTruck software is executed. It is designed with the idea to show a quick status of the current situation of the system.

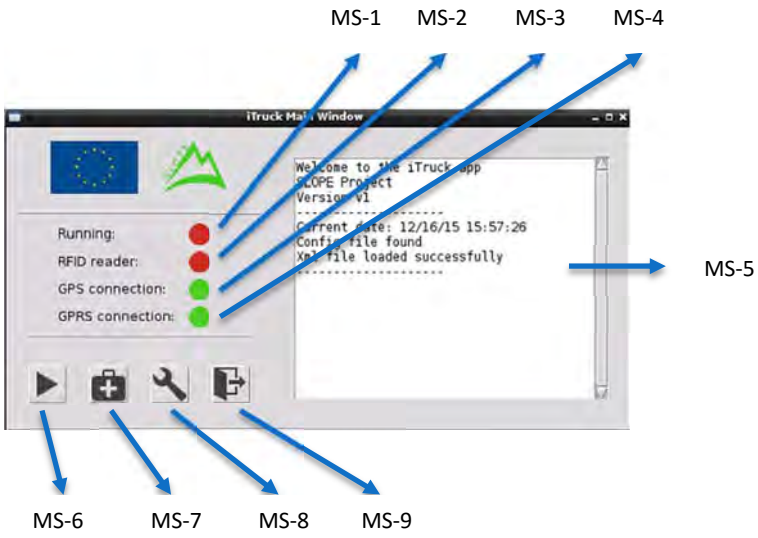

Figure 11 : Main Screen (MS).

- MS-1: Indicator to show if the automatic storage of the truck location and data sending to cloud is turned on or off.
- MS-2: Indicator to show if an RFID reader has been connected.
- MS-3: Indicator to show if GPS data is being received.
- MS-4: Indicator to show if the SLOPE database is accessible.
- MS-5: Log windows where different information is being shown.
- MS-6: Toggle button which allows to put the solution in run mode or stop mode. In run mode, the data is sent periodically to the SLOPE database.
- MS-7: Button to open the test window.
- MS-8: Button to open the configuration window.
- MS-9: Button to exit the program.

5.3 Test Screen (TS)

This screen is designed to test several options of the solution as a stand-alone module. The screen is shown when the MS-8 button is pressed.

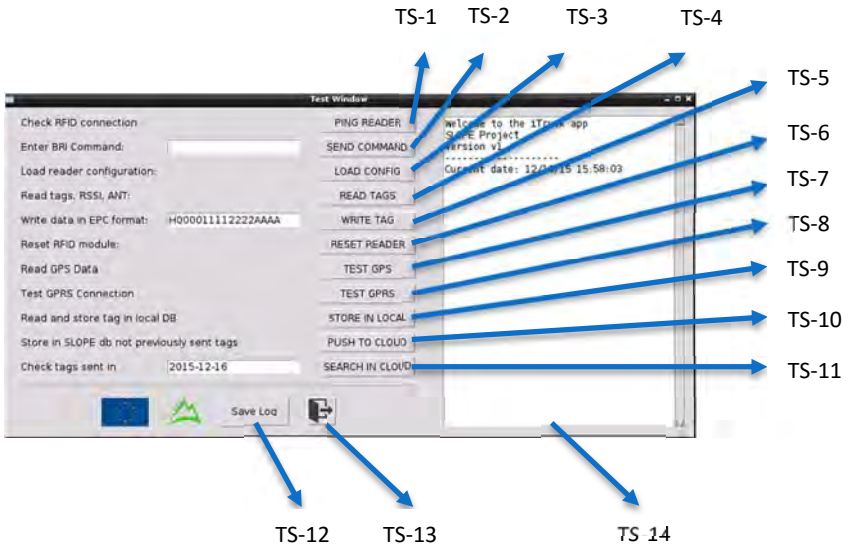

Figure 12 : Test Screen (TS).

- **TS-1: Ping Reader.** It is used to send a ping command to the RFID fixed reader. If the reader is available, the response is shown in the log window.
- **TS-2: Send command.** The RFID reader answers to BRI commands, an ITERMEC communication protocol. Here the commands can be typed,

and sent to the reader when the button is pressed. The response is shown in the log window.

- **TS-3: Load Config.** The button sends the current reader configuration, stored in the configuration window, to the RFID reader. The result is shown in the configuration screen.
- **TS-4: Read tags.** A Read command is sent to the fixed RFID reader. The response is shown in the log screen.
- **TS-5: Write tags.** A Write command is sent to the fixed RFID reader. The EPC code introduce in the text field is stored inside all tags encountered by the reader. The result is shown in the log screen.
- **TS-6: Reset Reader.** This button forces the RFID reader to shut down the RFID module, and start it again.
- **TS-7: Test GPS.** The last information received from the GPS is shown in the log screen.
- **TS-8: Test GPRS.** A simple access is done to the SLOPE. The Http response is showed in the log screen. An http response of 200 means connection successful.
- **TS-9: Store in Local.** A Read command is sent to the RFID reader, and the detected tags are stored in the local database. The log screen shows the operation result.
- **TS-10: Push to cloud.** This button is used to check the local database. Those entries that still have not been updated to the SLOPE database are sent to the cloud. To keep track of this, a specific field in the local database was created (field “isSent”, yes or no). Once the information is correctly updated in the cloud, the local database is updated.
- **TS-11: Search in cloud.** This allows to check all RFID entries introduced in the database on a specific date.
- **TS-12: Save log.** It stores all information that currently exists in the log screen in a text file.
- **TS-13: Exit.** Closes the Test Screen.
- **TS-14: Log screen.** Displays all information about the different controls.

5.4 Configuration Screen (CS)

This screen is designed to configure different options and variables in the solution. The screen is shown when the MS-8 button is pressed.

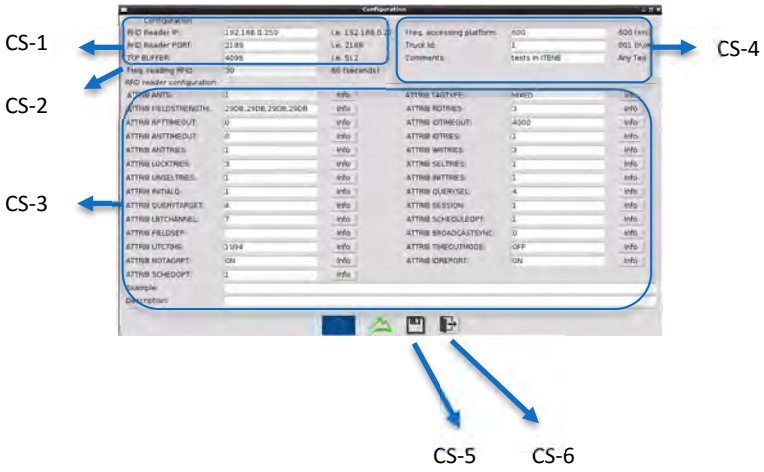

Figure 13 : Configuration Screen (CS).

- **CS-1: Fixed RFID reader communication**. This section id is used to configure the variables related to the fixed RFID reader. This includes The IP address, the communication port, and the buffer size.
- **CS-2: Reading frequency of RFID reader configuration.** Here the frequency in which the reader reads the logs along the transportation are defined in seconds.
- **CS-3: Fixed RFID reader configuration.** This section is used to configure the RFID reader. Variables like number of antennas, field strength, and tag type are used.
- **CS-4: Database related information.** In this section three values are defined: (i) frequency in which the local stored data is sent to the database; (ii) The truck ID in which the device is integrated; and (iii) comments text that will be included in the fixed RFID entry.
- **CS-5: Save button.** This button stores all information in the configuration xml file. The file is loaded when the software starts, so all the changes are kept.
- **CS-6: Exit button.** Closes the configuration screen.

5.5 Manual RFID Reader Screen (MRS)

This screen appears automatically when the RFID portable reader is detected via Bluetooth. It is used to monitor and control all activity created by the portable reader.

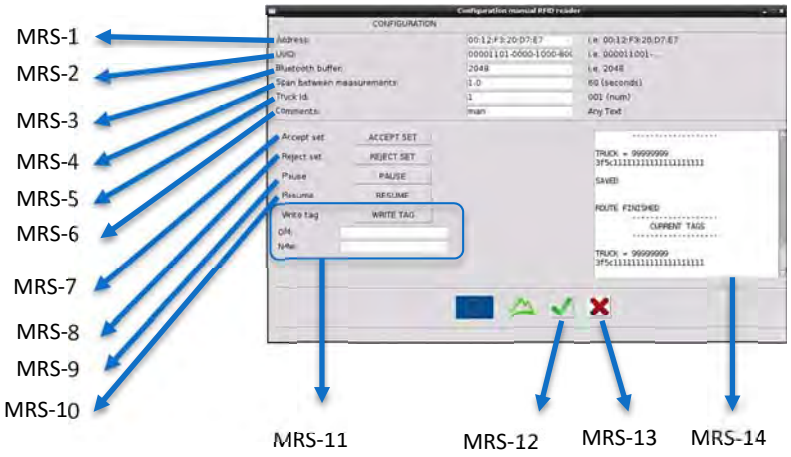

Figure 14: Manual RFID Reader Screen (MRS).

- **MRS-1: Manual reader MAC address.** Used to define the MAC address of the reader.
- **MRS-2: UUID.** The Universally unique identifier is used to identify a particular service provided by a Bluetooth device, in this case the RFID reader service.
- **MRS-3: Bluetooth buffer.** The buffer used for the Bluetooth communication.
- **MRS-4: Span between measurements.** This defines the frequency in which the portable RFID reader reads tags when active.
- **MRS-5: Truck Id.** Allows to introduce the ID of the truck which will be assigned to the tags read with the portable reader.
- **MRS-6: Comments.** Allow to introduce text that will be included in the table field on the local database.
- **MRS-7: Accept test.** This is used for debug. When the logs have been read and identified, this allows to accept the batch and store it in the local database.
- **MRS-8: Reject test.** This is used for debug. When the logs have been read and identified, this allows to reject the batch in case some logs are missing or some not-desired logs were also read.
- **MRS-9: Pause.** This is used for debug. Allows to leave the reader on pause.
- **MRS-10: Resume.** This is used for debug. Allows to exit from the pause mode and resume normal reading activity.
- **MRS-11: Write section.** This section is used to write tags. First the old and new EPC value needs to be introduced, and pressing the button allows to change the EPC information.
- **MRS-12: OK button.** The same as MRS-7, used to accept a batch.

- **MRS-13: EXIT button.** Closes the RFID activity and the Manual RFID reader screen.

6 Troubleshooting

This section describes the most common possible errors and how to solve them.

6.1 No GPS is found

1. Check if it works

Open the Linux Terminal	To be able to send shell commands
cgps -s	Shows a screen with gps data

2. Check if the GPS is detected

lsusb	Here appears a Profilic Technology device

3. Check if the serial device USB exists

ls /dev/ttyUSB*	This shows a list with USB devices. The ttyUSB0 is used for the GPS dongle.

4. Reset GPS daemon

sudo killall gpsd	Eliminate the gpsd daaemen
sudo gpsd /dev/ttyUSB0 -F /var/run/gpsd.sock	Restart the gpsd daemon

6.2 No RFID Reader is found

1. Check for correct ip

Open the Linux Terminal	To be able to send shell commands
ifconfig	Shows all network interfaces. Eth0 should have an ip in the range of 192.168.0.xx, since the reader is 192.168.0.250
Ifconfig eth0 192.168.0.252	Sets the ip of eth0 to 192.168.0.252

6.3 No database is found in postgress

This can happen after a hard reset.

1. **Create again the database connection**

Open pgAdmin III	Software to manage postgress db
Click on add connection to a server	Here we create a new connection
Name: slope_db Host: localhost Port: 5432 Username: postgres Password: p@ssw0rd	Values of SLOPE itruck local database

6.4 No GPRS connection is found

1. **Check GPRS modem status**

Open Midora web browser	The app is in the desktop
Connect to ip: http://192.168.1.1/	This connects to the GPRS modem control page
Check the current status, to see if there is any connection problem.	The information shown should be discussed with the phone service provider to identify why the connections is not working.