LIFE+10 ENV/IT/000389

INTEGREEN

Action 3: System design

D.3.2.1

On-Board Telematic Unit

Project Coordinating Beneficiary	Municipality of Bolzano
Project Associated Beneficiary n.2	TIS innovation park (TIS)
Project Associated Beneficiary n.3	Austrian Institute of Technology (AIT)

Document history

Date	Document Author(s)	Document Contribution
31/03/2014	Reinhard Kloibhofer, Franco Fresolone, Wolfgang Ponweiser (AIT), Roberto Cavaliere, Paolo Valleri, Stefano Seppi (TIS)	Document finalization

Dissemination level: PU[1]

Delivery month: M32

Status: public version

Submitted to EC: Confidential version CO[1]

[1] **PU** = Public; **CO** = Confidential (for members of the consortium only, including the European Commission); **RE** = Restricted, i.e. only a selected target of stakeholders defined by project consortium and validated by European Commission can access to the document. The final dissemination level will be defined in cooperation between the INTEGREEN consortium and the European Commission.

Table of Contents

Table of Figures

Table of Tables

1 Introduction

The System Design action aims primarily at producing the technical specifications for the Supervisory Centre and for the Mobile Systems. It follows directly after the Requirements phase which is the main input to this Action as it can be seen in Figure 1 here below.

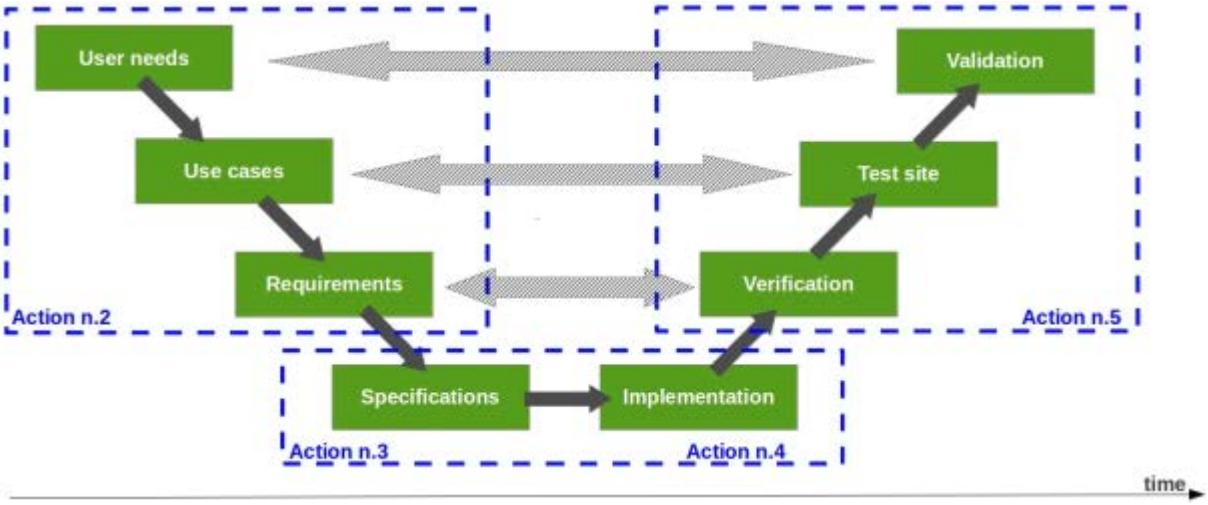

Figure 1: The V-model approach applied in the INTEGREEN project

1.1 Purpose of the document

This document Deliverable D.3.2.1 is the first of the two AIT deliverables of Action 3: System Design.

AIT is the responsible beneficiary for the activities in Action 3 and directly responsible for the execution of Task 3.2 Mobile System Design, while the execution of Task 3.1 Supervisory Centre design is under the responsibility of TIS.

Task 3.2 is divided in three subsystems as follows:

- On-board telematic unit
- On-board traffic monitoring unit
- On-board environmental monitoring unit

The three subsystems are clearly visible in Figure 2 here below.

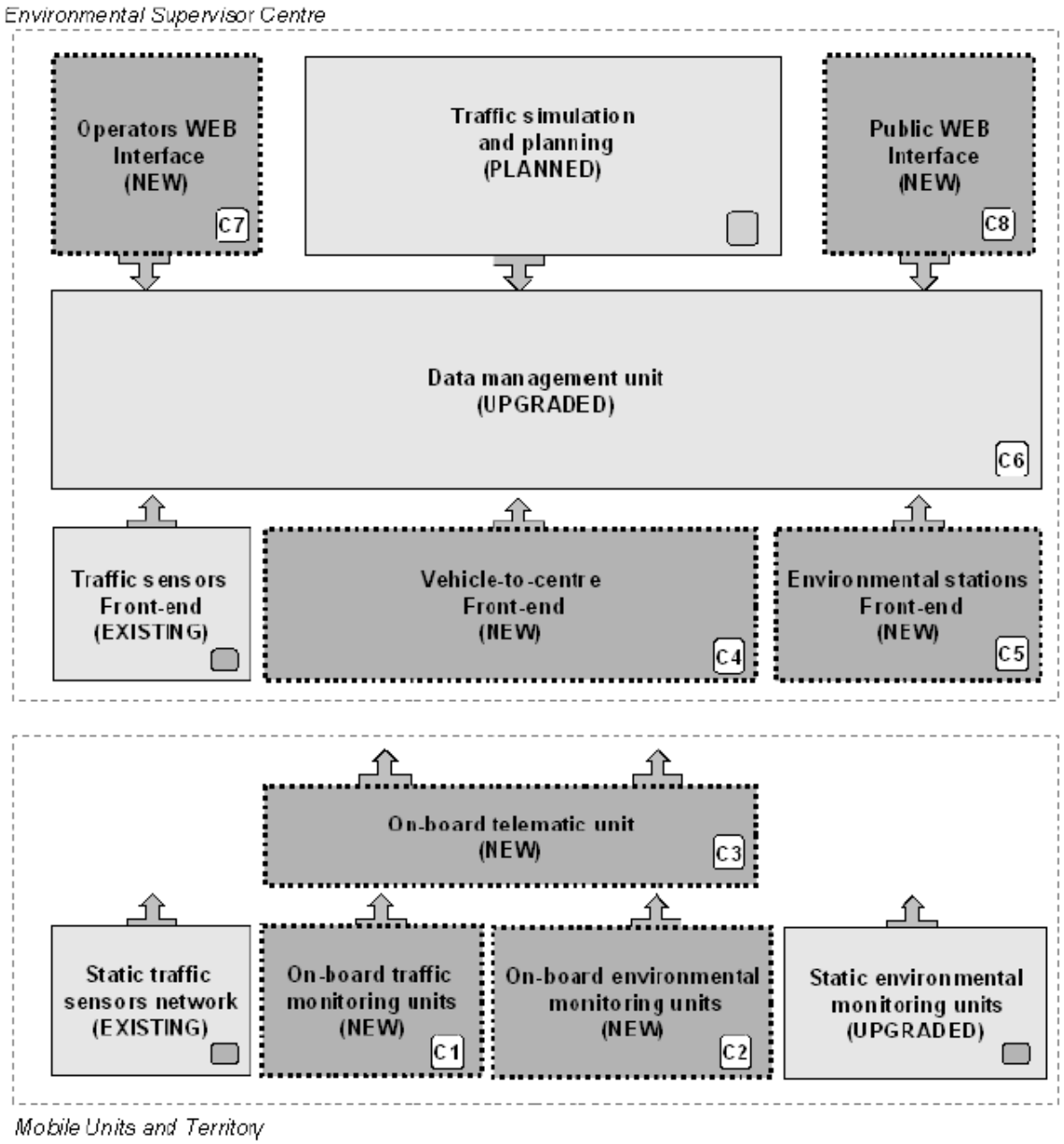

Figure 2: INTEGREEN system architecture

The design activities of the On-board telematic unit are described in this deliverable D.3.2.1 while the design activities of the On-board traffic monitoring unit and of the On-board environmental monitoring unit are described in the deliverable document D.3.2.2.

The overall design objective has been to achieve a solution based on selecting a processing platform already present on the market and at the same time suitable for automotive applications. On the selected platform the INTEGREEN specific interfaces and functionalities have been designed ad-hoc.

Basic research has not been considered as the final prototype system will have to be suitable for commercialisation after INTEGREEN.

In Task 3.2 a complete mobile system suitable for automotive application has been designed.

The system designed will be the input for the next Action to implement and build a prototype with full functionalities which will allow vehicles to have traffic and environmental detection capabilities, as well as real-time communication functionalities, in particular between the On-board telematic unit and the Vehicle front-end and subsequently to the Supervisor Center system.

This document deliverable contains the first output of Task 3.2 as the On-board telematic unit has been designed ad-hoc for the INTEGREEN solution. The design of this unit is based on a processing platform and software components. For the selection of a suitable off-the-shelf embedded platform, the latest automotive standards and technological advancements have been taken into account.

1.2 Specification definition methodology

The methodology that has been applied in the design activity is illustrated in Figure 3. It is composed by sequential and parallel activities which aim at (i) to assess and evaluate the critical requirements that most influence the High-level architecture, (ii) to design a modular system where single units can be easily reused in the future and (iii) to produce a set of design specifications that are ready for commercialisation after INTEGREEN.

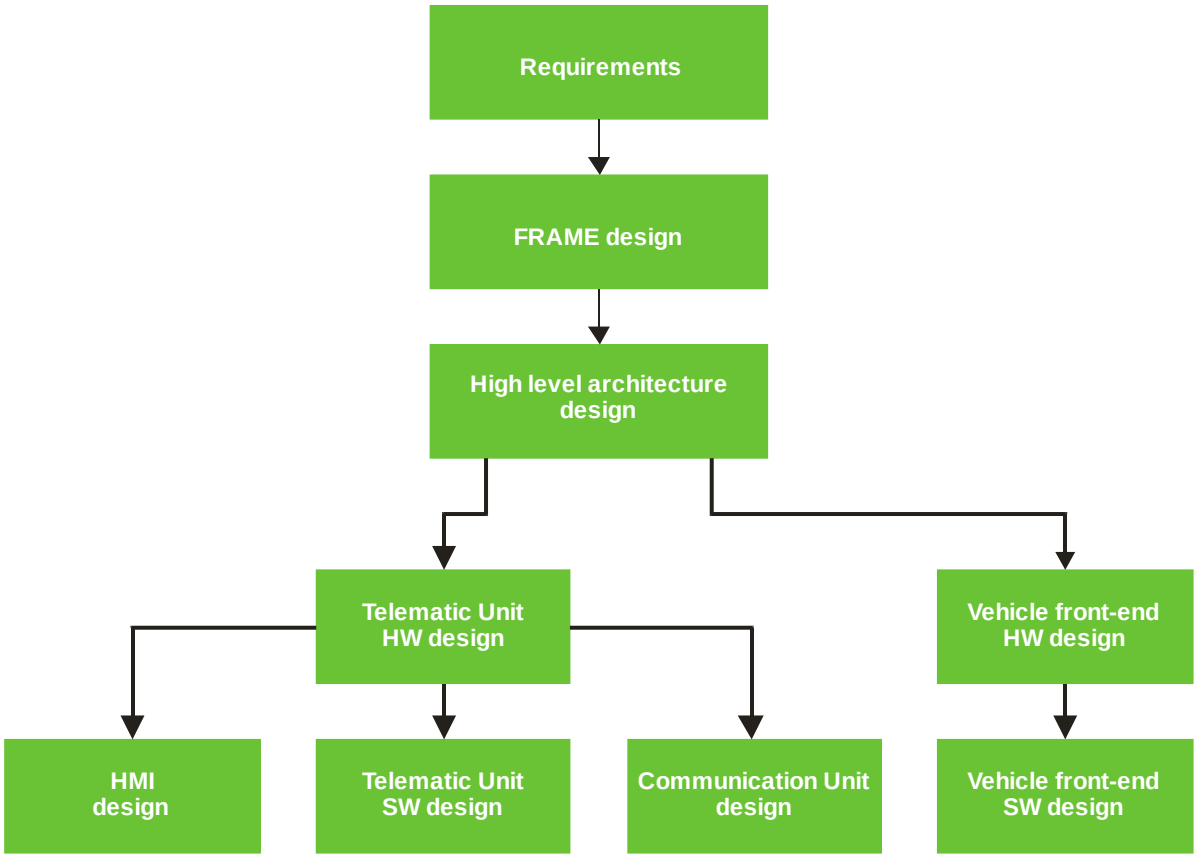

Figure 3: Methodology for the design of the On-board Telematic unit and associated functions

1.2.1 Requirements

The output from Action 2 defined and documented in deliverable D.2.2.1 have been analysed, assessed and prioritised.

1.2.2 Frame design

The INTEGREEN system has been modelled in a FRAME ITS architecture. This topic is placed in deliverable D.3.1.1 (Data management unit and environmental stations front-end design).

1.2.3 High level architecture design

Analysis of different modules and options that will fulfil the requirements.

1.2.4 On-board telematic unit HW design

The architecture of the On-board telematic unit is divided on one main HW unit supporting three subunits respectively for the in-vehicle human interface, for the processing of the data and for the communication with the external world. The main HW unit has been selected and engineered to provide the physical interfaces and the processing power to fulfil the requirements.

1.2.5 HMI design

The HMI has been designed to fulfil the HMI requirements of providing information and operation functionality for the whole Mobile system.

1.2.6 Telematic unit SW design

The SW architecture of the Telematic unit is an ad-hoc design but based on standard off-the-shelf SW components.

1.2.7 Communication unit design

The communication unit has been designed to provide a suitable bidirectional communication channel based on public 3G and 3.5G wireless networks

1.2.8 Vehicle front-end HW and SW design

The Vehicle front-end HW and SW design is done in two steps. As first step a standard HW Windows environment computing platform has been chosen. As second step an ad-hoc SW module is designed to provide a two-way data transfer between one or more vehicles on one side and the Supervisor Center on the other side.

2 Requirements

2.1 Mobile System Requirements from previous Action

The complete set of mobile system requirements taken from deliverable D.2.2.1 is listed here below in Table 1. It is worth noting that no requirements are defined for the vehicle front-end, as its functionality is dependent by the requirements defined for the Vehicle Data-Source described in deliverable D2.1.1 Supervisor Centre components requirements.

Components	ID	Name	Type	Priority
On-board telematic unit	OBU_1	Computing capacity	F	M
	OBU_2	Storage capacity	P	M
	OBU_3	Storage capacity (optional)	P	C
Communication unit	CU_1	Communication technology	NF	M
	CU_2	Communication protocols	NF	M
	CU_3	Communication load	P	M
HMI	HMI_1	HMI - Information content	F	M
	HMI_2	GUI - Information content	F	C
On-board traffic monitoring unit	OBTU_1	Kinematic sensors	F	M
	OBTU_2	Kinematic sensors plus	F	M
	OBTU_3	Kinematic sensor quality	F	S
On-board environmental monitoring unit	OBEU_1	Environmental sensors	F	M
	OBEU_2	Meteorological sensors	F	M
	OBEU_3	Environmental sensors plus	F	C
	OBEU_4	Environmental sensor quality	F	S

Table 1: Mobile system requirements list.

The requirements for the On-board traffic monitoring unit and for the On-board environmental monitoring unit are related to the design activity of the On-board traffic and environmental monitoring unit and consequently are described in Deliverable D.3.2.2 (main design work). On the other hand the requirements for the On-board telematic unit (OBTU), for the Communication unit (CU) and for the HMI are descripted in this deliverable document.

2.1.1 On-board telematic unit requirements

The following tables summarise the specific requirements of the On-board telematic unit.

ID	OBU_1
Name	Computing capacity
Description	The CPU of the on-board telematic unit requires enough computing capacity to accomplish the following tasks: • the control of the attached units; • the basic map matching and filtering of the data stream from the traffic and environmental units; • the processing of the communication protocols; • the preparation of the data to be displayed as well as the interaction with the

	HMI.
Rationale	System design – central processing and control of all components of the mobile probe system
Type	Functional
Priority	Must

Table 2: Requirement OBU_1 (computing capacity).

ID	OBU_2
Name	Storage capacity
Description	The on-board telematic unit must be equipped with enough memory to backup the recorded data of the traffic and environmental unit for at least 15 minutes.
Rationale	System design – enable to bridge temporal failures of the contact to the stationary system
Type	Performance
Priority	Must

Table 3: Requirement OBU_2 (storage capacity).

ID	OBU_3
Name	Storage capacity (optional)
Description	Further memory allocation must be considered in the case the on-board telematic unit needs to store basic map representations.
Rationale	System design – possible extension of the on-board data representation modalities.
Type	Performance
Priority	Could

Table 4: Requirement OBU_3 (storage capacity (optional)).

The above tables don't include HW platform requirements as the selection of an adequate processing unit is a result of these requirements.

2.1.2 Communication unit requirements

The following tables summarise the specific requirements of the Communication unit.

ID	CU_1
Name	Communication technology
Description	The communication unit must apply standard communication technologies such as 2G, 3G, WLAN (802.11) and Bluetooth
Rationale	System design - fast and future-proofed communication technologies.
Type	Non-functional
Priority	Must

Table 5: Requirement CU_1 (communication technology).

ID	CU_2
Name	Communication protocols
Description	The communication unit must apply standard communication protocols, e.g.:

	• XML, GPRS (generic data) • UMTS, CALM (traffic data)
Rationale	System design – well established and standardized communication protocols.
Type	Non-functional
Priority	Must

Table 6: Requirement CU_2 (communication protocols).

ID	CU_3
Name	Communication load
Description	The communication between the mobile probe system and the receiving vehicle front-end has to transfer: • traffic data (position, heading, velocity and acceleration) with an update frequency of 5 [s]; • environmental data (air pollutants and meteorological data) with an update frequency of 60 [s];
Rationale	System design - all the measured data must be communicated to the stationary system
Type	Performance
Priority	Must

Table 7: Requirement CU_3 (communication load).

2.1.3 HMI requirements

The following tables summarise the specific requirements of the Communication unit.

ID	HMI_1
Name	HMI - Information content
Description	The human machine interface has to provide functionalities to display: • the status of the INTEGREEN mobile probe system, including basic analyses and configuration capabilities; • the information provided by the Supervisor Center;
Rationale	Usability – testing of the equipment as well as continuous information transfer to the driver from the Supervisor Center.
Type	Functional
Priority	Must

Table 8: Requirement HMI_1 (HMI – information content).

ID	HMI_2
Name	GUI - Information content
Description	The graphical user interface of the mobile subsystem has to provide functionalities to display elementary map representations including the current position.
Rationale	Usability – continuous geo-spatial information transfer to the user
Type	Functional
Priority	Could

Table 9: Requirement HMI_1 (GUI – information content).

2.2 Automotive considerations

The On-board telematic unit, the communication unit and the HMI should be mounted in test vehicle. Therefore different considerations should be considered:

Physical dimension:

The On-board telematic unit and the communication system should be mounted in the vehicle in a way such that they should not be visible to the driver. This means small physical size is important. If necessary also the communication antenna should be “non-visible” to the driver.

Start of the system:

The system should start fully automatically and autonomously connect to the monitoring units and to the Supervisor center (via the Vehicle front-end).

Lost data:

In case of communication problems between the vehicle and the stationary system data should not be lost. Communication disruption can happen at any time also during a data exchange due radio link disturbance (small scale fading, large scale fading, radio interference …). After a communication disruption the system tries to reconnect in an automatic way and retransmits the data that couldn’t be delivered within the planned time. In a worst case scenario, data older than a certain period (for example 15 minutes) is not any more meaningful for the Supervision center and can be dropped. In the case of test and validation of the system data should never be lost.

Power supply:

The electrical power supply for the INTEGREEN modules comes from the vehicle power supply (+12V for cars, +24V for busses and trucks). In cars two types of power supply can be used:

- +12 V ignition
- +12V continuous

As the data collection, elaboration and transmission should be working with running vehicle engine the main part of the power should be come from “+12 V ignition” supply. Only for data saving issues “+12 V continuous” can be used.

Temperature range:

The temperature in a vehicle can be varying in a wide range. For use in European countries a possible temperature range of -20 °C to 70 °C should be considered.

Vibration:

The vibration in vehicles can damage electronic or mechanical parts. So in the design care should be taken to avoid larger electronic and mechanical components or fix them in a adequate way.

2.3 Considerations for the stationary part

The Vehicle front-end is a stationary system and therefore the automotive considerations are not valid. However also for this part special considerations have to be followed:

System always powered:

The Vehicle front-end has to be powered continuously. After a power-fail the system has to restart automatically.

System always connected:

The Vehicle front-end is continuously connected to the Internet. Therefore it is always able to communicate with the Supervisor Center and to any (new) vehicle with the INTEGREEN mobile system on board.

It can be always controlled and reconfigured via remote connection after a security log-in.

3 High level Architecture of the mobile system

The INTEGREEN mobile system includes all functions and modules of the mobile probe and the stationary server part up to the Vehicle front-end interface to the Supervisor Center (see Figure).

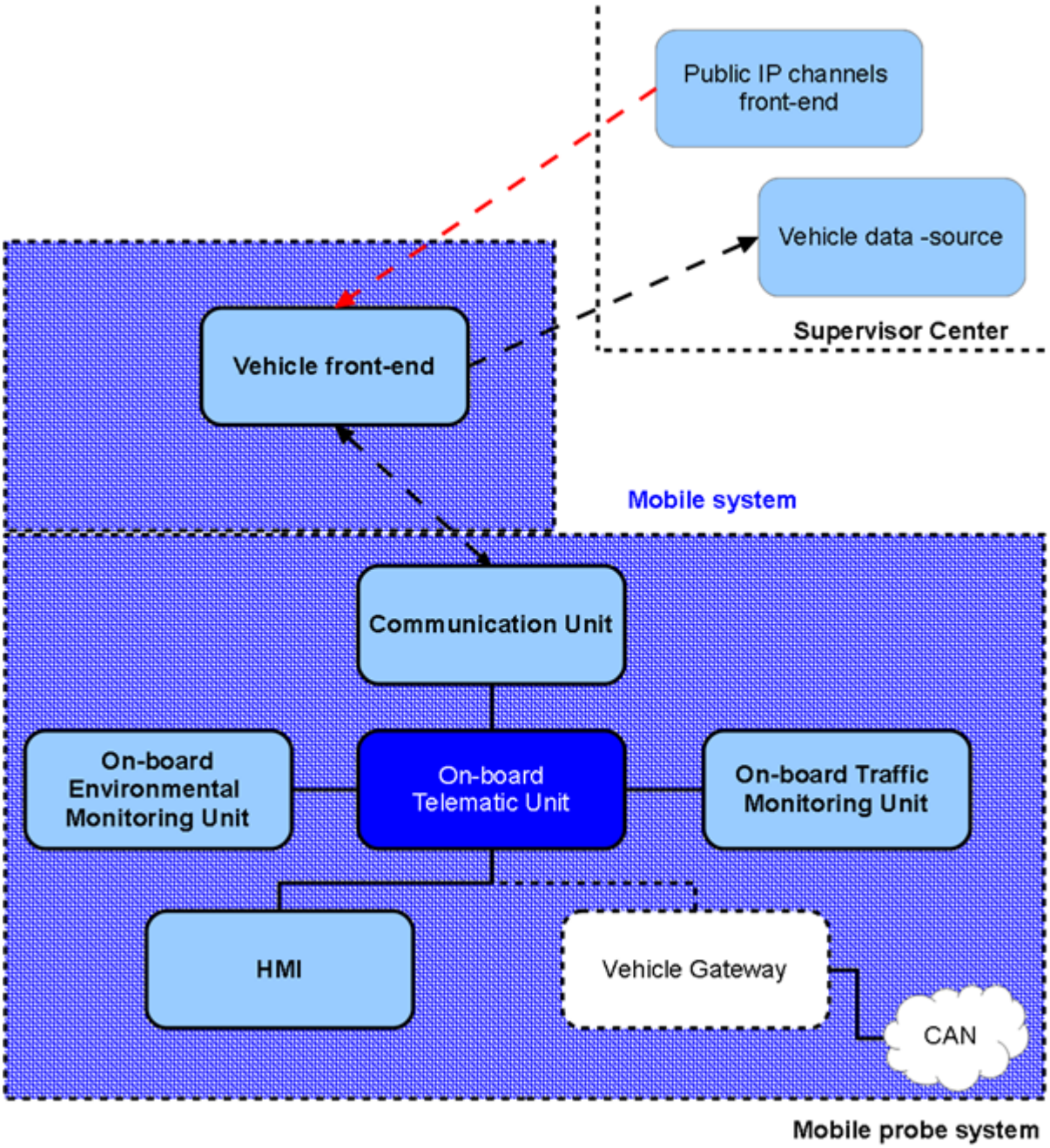

Figure: 4: The reference architecture of the INTEGREEN mobile system

The mobile probe system can be mounted on board of vehicles like cars, busses or trucks and is fully supplied with electrical power from the vehicle itself. Data communication is only possible via radio communication.

3.1 Functional blocks of the mobile system

The main functional blocks of the mobile probe system are:

- **On-board Environmental monitoring unit**: it collects environmental data particularly air pollution data and meteorological data as well as position data. This data is transmitted to the On-board Telematic unit. The detailed design of this unit will be described in the deliverable D.3.2.2.

- **On board Traffic monitoring unit**: it collects traffic related data, in particular vehicle speed, kinematic data of the vehicle and position data. Also this data is transmitted to the On-board Telematic unit. The detailed design of this unit will be described in the deliverable D.3.2.2.

- **On-board Telematic unit**: it receives the data from the On-board environmental monitoring unit and from the On-board traffic monitoring unit. This data will be elaborated and then transferred to the communication unit. To avoid a continuous transmission on the air data will be transmitted in blocks (one minute of data blocks)and therefore the Telematic unit must be able to save data. The detailed design of this unit will be described in this document.

- **Communication unit**: this unit communicate fully automatic via an air interface with the vehicle front-end in a bidirectional way. It transmits environmental, traffic and vehicle specific data to the vehicle front end and receives information data for the driver. The detailed design of this unit will be described in this document.

- **HMI**: the Human-Machine Interface is an interface for the driver of the vehicle. The information from the Supervision Center will be illustrated. The detailed design of this unit will be described in this document.

- **CAN-bus access**: data from the vehicle like speed, acceleration can be collected directly from the On-board telematic unit or via the On-board traffic monitoring unit. The detailed design of this unit will be described in the deliverable D.3.2.2.

The functional block of the stationary part of the mobile system:

- **Vehicle front-end:** this unit is connected via an air interface to one or more vehicle(s). The received data from the Mobile probes are collected and transferred via the Vehicle front-end to Supervisor Center interface to the Supervisisor Center. In the other direction it receives information data for the drivers from the Supervision Center and transfers it to the Mobile probe. The detailed design of this unit will be described in this document.

3.2 Interfaces between the functional blocks of the mobile system

The Interfaces between the different functional blocks of the mobile system are described in the chapter 3.2 of deliverable D.3.2.2.

4 Design of the On-board telematic unit

4.1 Hardware Architecture of the On-board telematic unit

The goal for the hardware architecture of the On-board telematics unit was to find an off-the-shelf computing system that fulfils the requirements which can be divided in four categories:

- calculation power
- memory space
- power supply
- mechanical, compactness

As the required function in INTEGREEN is well known, there should also be the possibility to expand the system (more and other sensors, more advanced sensor calculations…) in the future. The system should have an exceeding calculation power and the memory sizes should be expandable.

The decision was to take an embedded processing unit from the German company “Plug in” with expandable main storage and expansion connectors. The unit is shown in Figure 5.

Figure 5: Embedded processing unit

This unit has a size of 260 x 175 x 79 mm (without mounting bars) and is completely passive

cooled (no ventilator inside). It is designed for automotive applications and has already the right power supply (6 – 36V DC). The main memory is a SSD (solid state disc) and can be replaced with another bigger SSD. So there are no moving parts in the enclosure.

Other specifications of the processing unit:

- Mini-PCIe socket for expansion functions
- Standard USB, LAN and monitor connectors

4.2 Software Architecture of the On-board telematic unit

One design goal of the software architecture was to use as many standard software components as possible which implies to have most functionality that is not closely hardware related implemented on the embedded processing unit. The choice for the operating system wars Windows 7 as there are already many available tools that can be used.

The main structure of the SW architecture is illustrated in Figure 6.

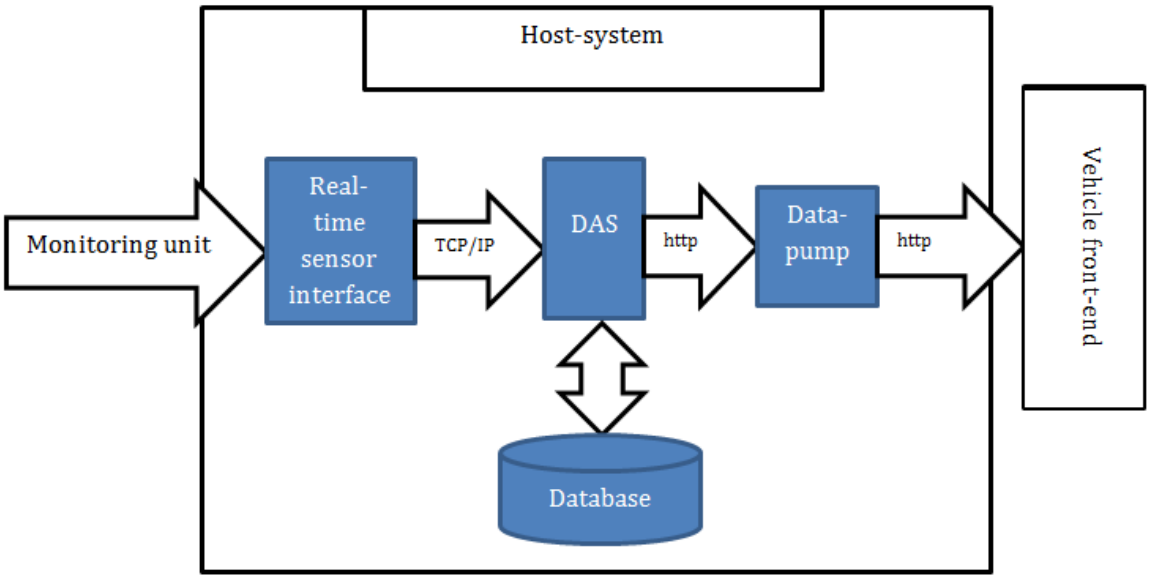

Figure 6: SW architecture of the Telematic unit

It can be divided in the following functions:

- Real-time sensor interface units and filtering
- Data acquisition, database and data pump

The Real-time sensor interface can be considered like a driver for the monitoring units, but it includes much more functions than a simple interface converter. It will be described in chapter 4.2.1.

An important function in the system is the software unit (a process) named DAS (Data acquisition system). This process is implemented using TSAPI (Time Series API), a modular system framework for time series handling and processing. TSAPI was created as a prototype during the EU project InnoSens[2] and vastly refined during the EU project CARBOTRAF[3]. The flexibility of TSAPI comes from the fact that peripheral functionality, like tapping measurement devices or other systems, is widely located in plugins (called "datahandler") that can be written for the purpose. With the same datahandlers written specially for INTEGREEN, TSAPI was also capable of providing the functionality on the mobile unit.

TSAPI is implemented in JAVA and consequently has problems accessing native interfaces like USB. To solve this issue, the UART2Net program was written (see chapter 4.2.1), which reads the data from the physical data interface, elaborates data and sends it further through a TCP/IP interface to any consumer who connects to it.

The DAS is responsible for storing the data until it can be delivered to the Vehicle front-end. This is done in an external database. The DAS mostly follows the architecture role of a "server" which means it is a passive running process and doesn't start any actions by itself.

Another process called "DataPump" actively started by a timer is responsible for conveying measurement data from the DAS to the Vehicle front-end.

4.2.1 Real-time sensor interface

The real-time sensor interface includes the configuration of the physical interface towards the monitoring units, the configuration of the interface towards the database system, the sample rate reduction of high speed sensors configuration of vehicle data and the calibration data for sensors and log-file capacity for tests.

The SW was written in the programming language C++ and generates an executable file (UART2Net.exe) which can be started with different parameters. A UART2Net program can address only one monitoring unit, this means for each monitoring unit a separate UART2Net file is necessary.

Parameter for UART2Net

The parameters for starting UART2Net are:

- Interface from the monitoring unit: for example if the monitoring unit is connected via a COM 15 port, the parameter is "–c 15" (`UART2Net –c 15`)
- Interface to the database system: The interface is defined as a TCP/IP interface. If the local IP-address is used, only the port number has to be defined. For example if TCP/IP Port 18250 is used, the parameter is "–n 18251" (`UART2Net –n 18251`)

[2] GZ N 226/2007
[3] Grant Agreement Number 287867

Sample rate

The sample rate of some sensors can be high, i.e. motion sensors can deliver data in the range of milliseconds.

The data rate to the database system is defined with one sample per second. Therefore data sensors with higher sample rate are down-sampled to one sample per second.

Configuration and calibration

Configuration data and calibration data are saved in an ini-file (UART2Net.ini) which is read automatically at the start of UART2Net.

Fixed parameters defined in the ini-file are:

- VehicleID
- MobileSystemID
- DriverID

Additionally the calibration parameters for the sensors are also saved in the ini-file. For linear sensors offset and slope parameters are defined, for non-linear sensors the conversion curve (raw value to sensor value) is saved.

Log-file capacity

For debugging purpose the data coming from the monitoring unit and the data sent to the database system can be logged. This is very useful for the calibration activity and functional validation of the sensors.

4.2.2 Architecture of the database system and the DAS

The DAS is based on TSAPI which provides the functionality often needed to implement applications that deal with time series handling and processing.

TSAPI datahandler

One of the core features of TSAPI is that it defines a model of how external systems interact with TSAPI based applications. External systems in this case can be measurement devices, other applications, databases etc. This allows defining plugins which deal with special circumstances. A typical datahandler is written for one special purpose and thus provides a limited functionality. Datahandler interaction is completely governed by TSAPI and do not interact directly with others. Consequently there is marginal interdependence between different datahandlers. This makes development and maintenance easy and as a result cost effective. Indeed the typical datahandler is not written with reusability in mind but for the intended special purpose.

TSAPI provides a functional layer that allows controlling datahandler instantiation and

interaction using external configuration (typically a config file). This allows writing applications that are widely reusable and adaptable to different situations. Indeed the OBU was realized using such an existing application.

For typical re-occurring tasks like interfacing to a database or network interlinking TSAPI provides ready-made datahandlers which can be customized for special purposes.

TSAPI allows to route data through special data handlers called “processors”. This allows calculating derived data like aggregates. Other typical use cases are e.g. unit conversions or generating alarms when certain thresholds are violated.

DAS customisation for the OBU

Specialised datahandler

A datahandler had to be written to tap into the adapter program. One instance of this handler is capable of dealing with data from either the On-board traffic unit and/or the On-board environmental monitoring unit, thus two instances of the handler where needed.

Database customization

Looking at the list of parameters provided from the monitoring units (see below) it is possible to identify four groups of parameters, which can be efficiently stored in individual specialised tables.

1. Scalar measurements (e.g. concentrations, temperature, …)
2. Locations (GPS)
3. Inertial related data (accelerations, ...)
4. System information

Thus four specialized tables where created for storing measurement values. They are augmented by a handful of tables which hold data relevant for describing and managing the measurement data (e.g. ISDs of known parameters and their units).

Database selection

Generally, every method that can store data could have been used to temporarily store data on the Telematic unit. The decision for choosing a PostgreSQL database was driven by several factors, where not all of them are purely technical.

Why storage in a persistent store?

It is not guaranteed that data can be transmitted in near real time. It is even not guaranteed that data can be transmitted before a potential power down of the system. Thus using any data store method which is purely RAM-based is out of question for the INTEGREEN solution.

Why a database?

Any storage solution based on files could be an option for this use case. The use of a database has the advantage that the organization of data within the file is easy. One of the main reason for the decision of using a database was that different previous implementations were available in AIT and with minimum adaptation effort, one of them could have been made suitable the for use in INTEGREEN. A new implementation of a database for a specialised file based storage solution would have taken more effort of the available budget.

Why PostGreSQL?

There are many databases on the market. Some of them (e.g. Oracle) come with a significant license fee while many are free-ware. The potential candidates on the free market – just to name a few -- would have been mysql, PostGreSQL, JavaDB also known as BerkeleyDB etc. An important criterion for the database selection is the existence of a JDBC (Java Database Connection) driver; a criterion which most contemporary database implementations fulfil. Nevertheless this excludes some popular products like e.g. Microsoft's JetEngine (also known as "Microsoft Access"), as the OBU is equipped with sufficient hard disk capacity the footprint of the dataset is not a real issue. A design decision was made that a significant amount of data was to be kept locally at least during the system test phase in case data transmission fails. This solution assures no data is lost even when there is a transmission breakdown. The amount of data to be stored is not negligible which in turn excludes some database products designed for smaller amounts of data. Most of the above mentioned database products would have done the job. The real criterion for choosing PostgreSQL was not just technical but also due to the fact that AIT has good experience with running this brand of database and the costs of installing and setting up were thus low.

DAS configuration

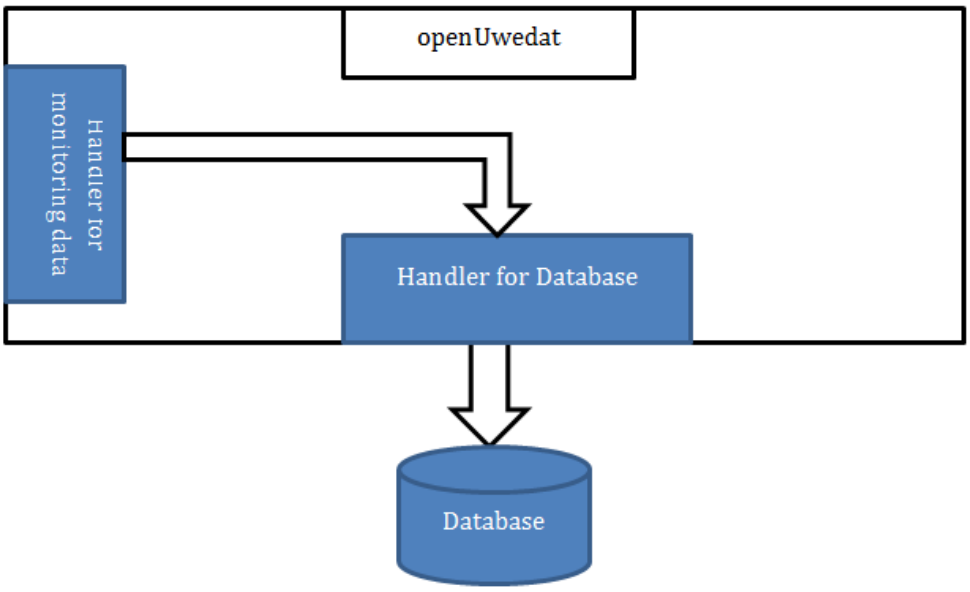

Figure 7: Dataflow during data gathering

The configuration file that describes this setup:

Restricted Information.

For further information please contact the appropriate specialists of AIT Austrian Institute of Technology GmbH.

Data transfer to the Vehicle front-end

The data can stay in the database for an arbitrary amount of time but one of the requirements is to send the data in near real time (NRT) to the Vehicle front-end. Currently this is interpreted as sending the data at least once per minute. Data transfer may fail for a handful of different reasons. Thus data must at least be stored until it is successfully sent to the Vehicle front-end. There are reasons like debugging to keep the data for a longer period in the database, which can easily be configured.

Once per minute the data related to the last complete minute is extracted from the database and sent to the Vehicle front-end. This is done by a separate process named “datapump” which is started periodically.

The “datapump” is another program from the TSAPI suite. Its purpose is to copy data between entities encapsulated by data handlers. Typical use cases are copying data between a set of data files to a server system or – like in this case – between two systems.

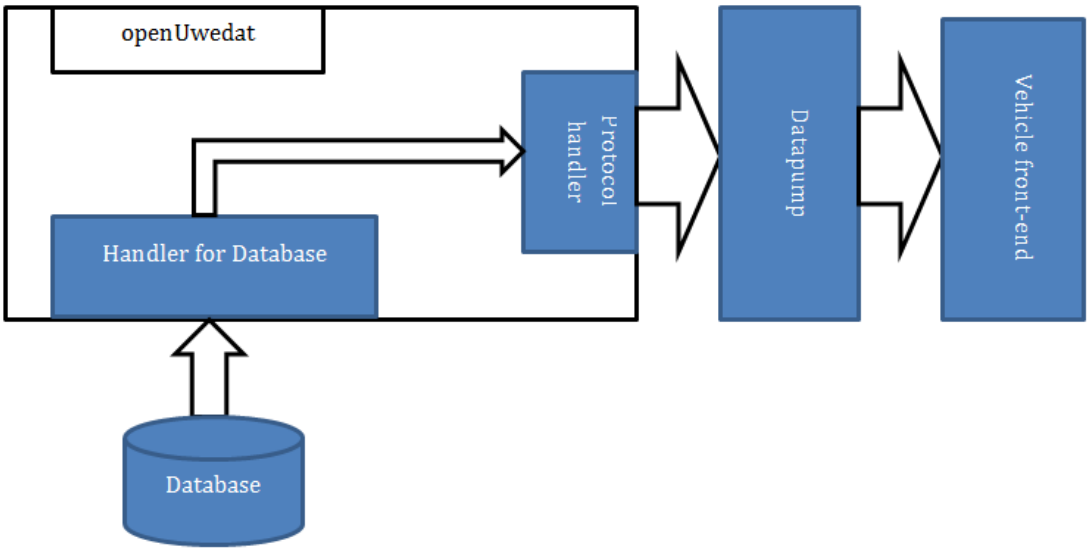

Figure 8: Data transfer to the Vehicle front-end

In the INTEGREEN use case the datapump was configured to use a HTTP protocol that is proprietary to the TSAPI. The sole reason for this was that this protocol was readily available. The use of this protocol is encapsulated in a data handler. Thus implementing another protocol simplifies writing and installing a new data handler and can be done any time when the need arises.

Autostart issues

A task which is easily done on a Unix machine is a real problem on a windows machine: start programs with no user logged in, i.e. in the background. While Unix-based operating systems

allow launching any console based program in the background and start those up during system start-up, windows needs programs being specially designed for this purpose; so called “Services” which must implement a special interface. A typical solution for this problem is to use “service wrapper” which provide the service interface while starting another program as a child process. Several solutions are available which all solve the core problem. AIT uses a wrapper developed by AIT which is augmented with a few utilities which are not otherwise available in this special composition. This program called “srvarcs” is used to start the DAS process at system start and to schedule the datapump to run once a minute.

Potential adaptions

openUwedat offers the ability to insert so called “processors” in any data stream by just editing the configuration file. The processors allow processing of the data. In particular they allow calculating of aggregations like averages, maximum, minimum etc. If special operations are needed the system can be augmented with user defined operators written in java or python that act as plugins.

Processing can be done on incoming data (see Figure 9) as well as on queried data (see Figure 10).

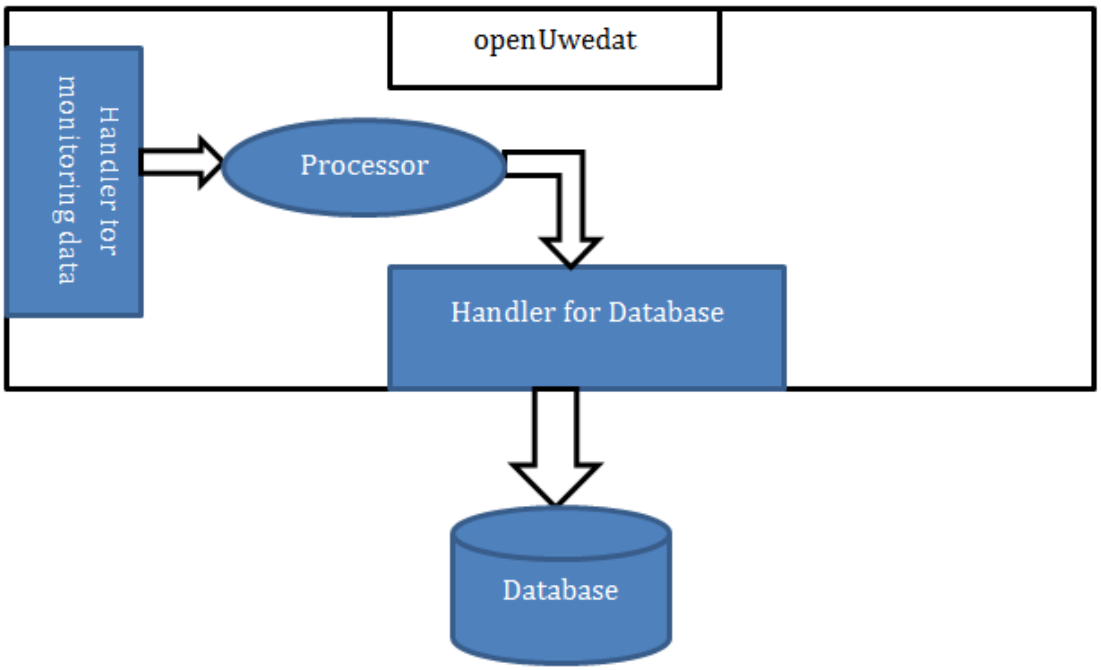

Figure 9: Processor for incoming data

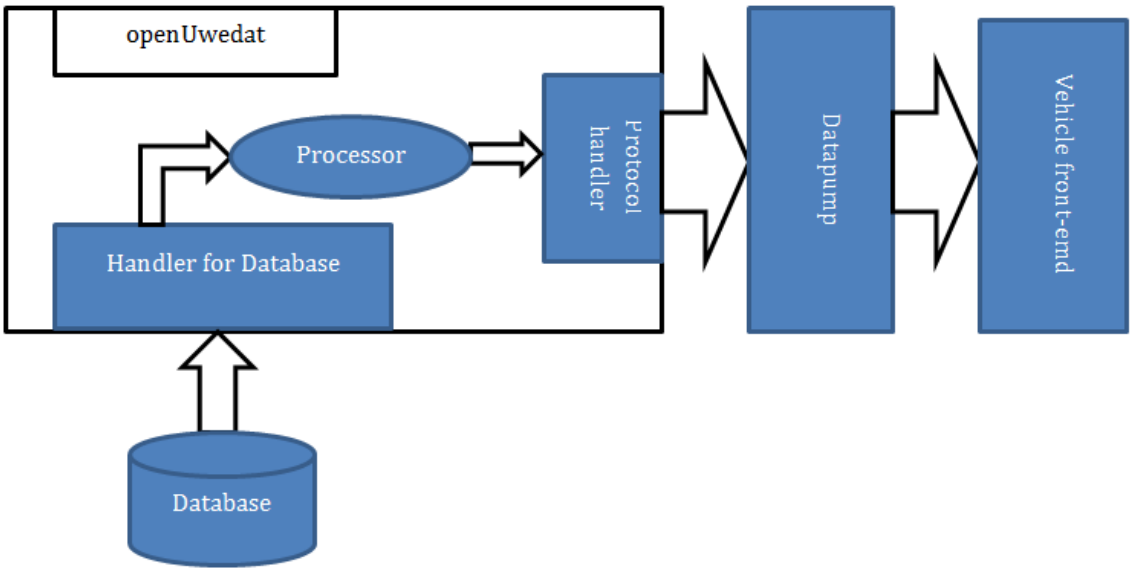

Figure 10: Processor for queried data

5 HMI design

In the INTEGREEN on-board system a HMI for the driver is planned. Two types of interfaces should be distinguished:

- HMI for the end-user
- HMI for testing and evaluating the on-board system

In INTEGREEN two test vehicles are planned: the AIT test vehicle which is more technical orientated and the TIS test vehicle which is more end-users orientated.

The project team decided to have different types of HMI for the two test vehicles:

- TIS test vehicle: a smart phone which is connected to the internet is mounted like a navigation system on the front windscreen or dashboard. With a standard browser connected to the Supervisor Center it displays information for the driver provided from the Supervisor Center.
- AIT test vehicle: a VGA monitor, a keyboard and a mouse which are connected to the Telematic unit will be used. With this configuration the complete on-board system including monitoring units, telematics unit and communication unit can be fully controlled from a front-seat passenger. The monitored sensor values are displayed in real-time and can be validated according to the real traffic situation from the front-seat passenger. The system can also be connected to the Supervisor Center web page with a standard browser to see the information for the general INTEGREEN users. The Telematic unit is already connected to the Internet by the wireless communication unit. The screen mounted in the AIT test vehicle is shown in Figure 11.

Figure 11: On-board Display in the AIT test vehicle

From a software point of view data and control functions of the INTEGREEN on-board system in the AIT test vehicle is displayed by using a standard windows graphical screen. On this screen a program called “GTE” is started. GTE is another member of the openUwedat family of programs and stands for (G)raphical (T)imeseries (E)ditor. The GTE uses the same HTTP-based interface that the “datapump” program uses to obtain data.

The GTE allows the configuration of many functions of the display like which data to display, which time interval, display styles etc. All selected settings can be stored in a so called “profile”.

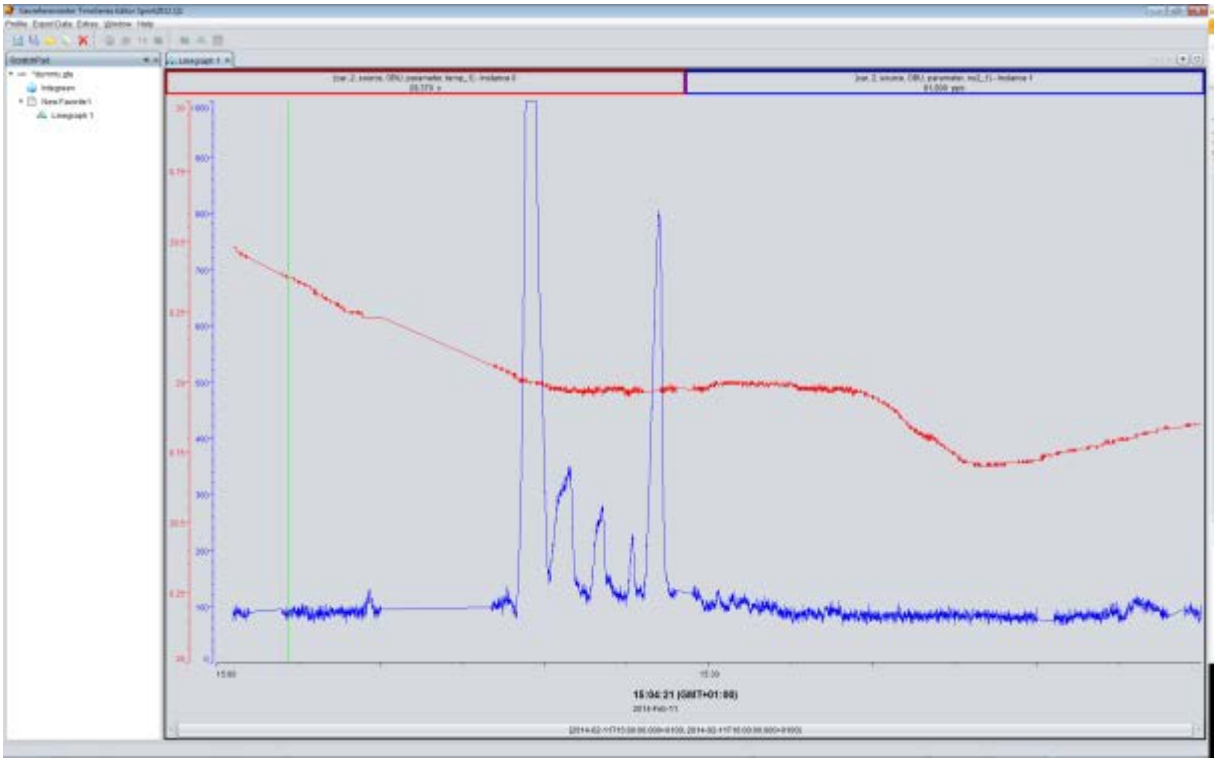

Figure 12: Data sample from test drives Feb 2014

The GTE allows the displaying of live data. To achieve this, the GTE uses a “publish-subscribe” pattern also known as “event listening”. When started the GTE registers itself as listener for the data it currently displays. When new data arrives, an event is sent to the GTE. This event is a container for information about what has changed and in what time range. After receipt of such an event this new data is queried from the DAS process and displayed.

Above it was mentioned that the interface is HTTP based. The data direction of HTTP does not allow sending data to a client by a server initiated action. Data will always and only be sent on queries started by a client. So the HTTP concept would prohibit events as described here below.

To achieve event handling without periodical polling the following method is used:

The GTE sends a query for events to the server. This query will NOT return unless there is a waiting event. As queries can’t be hold by the server forever, then when no event occurs after a defined period (typically about 30 seconds) a dummy response is sent from the server. When the GTE gets a response – either an event or a dummy response – it immediately opens a new query to the server.

By this procedure a fair response time to events is combined with moderate network traffic.

6 Communication Unit design

For the communication unit different solutions were discussed:

- W-LAN connection: it works fine if free W-LAN hotspots are “visible” and the travelling speed is low (no real hand-over from one hotspot to the next). Due the fact that on the INTEGREEN test route there are only a few W-LAN hotspots this solution wars not effective. But this solution can be a good choice if the On-board system is mounted on a bus and on his route there are a good number of W-LAN hotspots.

- WiMAX: for the INTEGREEN test route there is no WiMAX connection

- DSRC (Dedicated Short Range Communication): at the moment there is no DSRC infrastructure on the INTEGREEN test route. This technique should be considered for future developments of the INTEGREEN applications.

- 3G / 3.5G modem: In the city of Bolzano and therefore also on the INTEGREEN test route there are different wireless internet providers for connection via 3G / 3.5G modem. This was at the end the selected solution for the INTEGREEN Mobile probe.

On the selected Telematic unit (see chapter 4.1) there are different internal and external interfaces available. To have a compact solution an internal 3.5 G GSM/UMTS modem for a miniPCIe slot was selected. Additionally a SIM-card connector is already present on the mother board. The very small miniPCIe module didn’t have an integrated antenna; consequently an external antenna directly mounted on the SMA antenna connector at the rear of the Telematic unit had to be used.

The external antenna can be a small monopole antenna (see Figure 13) if the signal level reaching the Telematic box is not attenuated by the vehicle itself depending on the mounting position inside the vehicle. If the signal level is too low, then an external antenna mounted on the roof of the vehicle is the preferable alternative.

Figure 13: External GSM/UMTS antenna

The antenna solution as shown here above in Figure 13 has been successfully used during all measurement campaigns conducted with the AIT test vehicles.

7 Design of the Vehicle front-end

7.1 Hardware design of the Vehicle front-end

The Vehicle front-end hardware is an off-the-shelf PC. It is implemented using virtualisation so no new real hardware was necessary.

7.2 Data transmission from Vehicle front-end

The specification of the data package to be transmitted to the Supervisor Centre is defined in D.3.1.2.

7.3 Software design of the Vehicle front-end

The Vehicle front-end computer uses Windows 7 as operating system.

The Vehicle front-end is a virtual twin duplication of the OBUs.

7.3.1 The Vehicle front-end process

Most of the core functionality of the Vehicle front-end – accept data input from the monitoring units – is implemented in a process called “Central” which runs permanently. The central process is bases on TSAPI and differs from the DAS process only in a few details.

Again data is stored in a PostgreSQL database. The table layout is identical to the database on the OBU.

7.3.2 Data transfer to the Vehicle data-source in the Supervisor Center

Data export from the local system

The main difference between the OBU and the Vehicle front-end is the program that transfers data to the Vehicle data-source in the Supervisor Center. A special protocol based on JSON was designed to convey data from the Vehicle front-end to the Vehicle data-source. Wrapping this protocol in a datahandler would have added no real benefit as it is only used in one very specialized and otherwise simple application. So a new application based on TSAPI was written that taps into the central process to export data and translate it into the JSON format. JSON form data is then stored in form of files on the hard disk.

Data submission

Another program walks the list of exported files. Ideally only one such file should be present on the hard disk but in case of earlier problems older files may still be present.

Files are then sorted by their age, where older files come first. Each file is uploaded to the Vehicle data-source system.

If the Vehicle data-source sends back a positive response, then the file is deleted from the file system. In all other cases it is kept. In the next transmission cycle another attempt of uploading the file is done without further measures.

Both programs – data export and data transmission – are started as one job and one schedule once a minute. The execution as one job determines the sequence of their execution.

8 Improvements of components in case of connection interruption

In a first approach of the OBU, the data transfer to the Vehicle front-end and the data transfer to the Supersior Center runs unsynchronised.

Due to different sources of jitter the temporal relation between these actions is not constant.

Sources for such jitter are:

- Differences between system clocks can differ significantly. In praxis today system clocks are usually synchronized with the “network time” using time servers so this effect is small.
- The accuracy of scheduling by the service wrappers. This is the main source of jitter as the scheduling mechanism used has accuracy not better than one minute.
- Data transfer is not taking place in an infinitesimal small amount of time. Depending on bandwidth and transmission speed available it may take several seconds to convey the data.

Network problems that delay the submission:

An additional problem is that the system of the Vehicle data-source at the Supervisor centre does not allow retransmission of data sets and their interpretation of the term “data set” is the complete set of data for one vehicle. Thus it must be made sure that the data for one vehicle is completely available at the time of data export.

During the first field test a simple scheme was used to solve this problem through the addition of temporal safety margins. The time delay from data retrieval in real-time at the OBU up to data transmission may be up to 90 seconds when both involved clocks are in sync. To add more safety transmission to the Vehicle data-source was delayed for 3 minutes. To avoid multiple re-transmissions to the Vehicle data-source only the last complete 60 seconds were exported.

No bookkeeping was used to keep track of data not yet transmitted.

This approach turned out to be very simple but unfortunately it was not too efficient. It failed in case of interruptions of the transmission (i.e. loss of 3G connection) from the OBU to the Vehicle front-end as the data not transmitted in the first attempt was never re-transmitted. (It was still not lost for later evaluation as it was still in the local database).

The following measures will be taken to come around these shortcomings:

a) Instead of a direct system-to-system transmission from the OBU to the Vehicle front-end the following algorithm will be used:

I. Export the data to be transmitted into a file and store it locally

II. Transmit all files to the Vehicle front-end.

III. Only on success of step 2: delete the files locally.

IV. Import those files into the Vehicle front-end system.

V. This procedure introduces an implicit book keeping of data already sent and it allows an easy retransmissions in case of system failure.

b) The server of the Vehicle data-source will get the method to query the latest available time stamp from the vehicle. Data export on the Vehicle front-end will be done for all data older than the obtained time stamps and will include all data up to now.

If data is not available yet, this will not be available in the data files and thus will not modify the “latest available time stamp” in the Vehicle data-source. Thus the next export attempt will again try to re-export data for any newly arrived data without any extra effort.

Conclusions

The overall design objective has been to achieve a solution based on selecting a processing platform already present on the market and at the same time suitable for automotive applications.

The architecture of the On-board telematic unit is divided on one main HW unit supporting three subunits respectively for the in-vehicle human interface, for the processing of the data and for the communication with the external world. The main HW unit has been selected and engineered to provide the physical interfaces and the processing power to fulfil the INTEGREEN requirements. The INTEGREEN specific interfaces and functionalities have been designed and engineered ad-hoc to optimise the performance for mobile applications.

On the selected Telematic unit (see chapter 4.1) there are different internal and external interfaces available. To provide the communication from the mobile system to the supervisor centre and at the same time to have a compact solution, an internal 3.5 G GSM/UMTS modem for a miniPCIe slot has been selected.

In the INTEGREEN on-board system a HMI for the driver has been implemented which provides the functionality for testing and evaluating of the system. Additionally for the final version a HMI for the end-user will also be implemented.

For the first field tests a simple data transmission scheme was used to send the data to supervisor centre where the data was not retransmitted in case of transmission interruptions. In the final version the data transmission method will be improved to cope with all events.

Bibliography

[1] Comune Bolzano, Assessorato Mobilità, “Piano Urbano della Mobilità 2020,” Dicembre 2009.

[2] Provincia di Bolzano - Agenzia Provinciale per l'Ambiente, “Situazione dell'aria in Provincia di Bolzano,” [Online]. Available: http://www.provinz.bz.it/umweltagentur/2908/luftsituation/index_i.asp.

[3] Provincia Autonoma di Bolzano - Agenzia Provinciale per l'Ambiente, “Valutazione della qualità dell'aria 2005-2015 - Allegato 2,” Bolzano, 2010.

[4] CISMA srl, “Valutazione dell'impatto sulla qualità dell'aria nella città di Bolzano,” Bolzano, 2010.

[5] EMISIA SA, “COPERT IV,” [Online]. Available: http://www.emisia.com/copert/.

[6] L. Ntziachristos and Z. Samaras, “COPERT III Computer programme to calculate emissions from road transport - Methodology and emission factors,” 2000.

[7] International Institute for Applied Systems Analysis, “GAINS Europe,” [Online]. Available: http://gains.iiasa.ac.at/gains/docu.EUR/index.menu.

[8] J. Scire, D. Strimaitis and R. Yamartino, “A User'sGuide for the CALPUFF Dispersion Model,” 2000.

[9] W. Sparber, R. Fedrizzi, S. Avesani, D. Exner and H. Mahlknecht, “Calcolo e valutazione delle emissioni di CO2 e definizione di scenari di riduzione per la città di Bolzano,” 2010.

[10] IPCC, “2006 IPCC Guidelines for National Greenhouse Gas Inventories,” 2006.

[11] ECO Speed, “ECO Speed,” [Online]. Available: http://eco5.ecospeed.ch/ecospeedhome/index.html?sc=0.

[12] Heads of Compulsory Third Party Insurance in Australia and New Zealand, “The role of tyre pressure in vehicle safety,” INJURY and ENVIRONMENT, 24 April 2007.

[13] ENI, “La guida pratica per l'efficienza energetica,” Maggio 2007.

[14] M. Forcolin, “EURIDICE project factsheet on CORDIS,” 30 May 2012. [Online]. Available: http://cordis.europa.eu/fetch?ACTION=D&CALLER=OFFR_TM_EN&RCN=8697.

[15] M. Weilenmann, R. Alvarez and M. Keller, “Fuel Consumption and CO2/Pollutant Emissions of Mobile Air Conditioning at Fleet Level - New Data and Model Comparison,” Environmental Science & Technology, 2010.

[16] INTEGREEN project, “INTEGREEN proposal: technical forms,” 2011.

[17] “Serielle Bussysteme im Automobil,” auto & electronic, 6 2006.

[18] O. J. Woodman, “An introduction to inertial navigation,” Technical Report Nr. 696, University of Cambridge.

[19] I. Sonoma Technolgy, “Characterization of Low-Cost NO2 Sensors,” September 2010.

[20] “seventh framework programme D1.1: Report on: sensor selection, calibration and testing; EveryAware platform; smartphone applications,” 2012.

[21] M. Gerboles, “Developments and Applications of Sensor Technolgies for Ambient Air Monitoring,” in Workshop 'Current and Future Air Quality Monitoring', Barcelona, Spain, April 25-26, 2012.

Appendix A: Acronyms and Definitions

DAS: Data Acquisition System

FW: Firmware

HW: Hardware

JDBC: Java Database Connection

NO: nitrogen oxide

NO_2: nitrogen dioxide

NO_X: nitrogen oxides, which include NO and NO_2

O_3: ozone

OBU: on-board unit

ppb: parts per billion

ppm: parts per million

ppt: parts per trillion

SW: Software

TLA: Three Letter Acronym