D5.3 – TeamMate Car Demonstrator after 2^{nd} Cycle

Project Number:	690705
Classification **Deliverable No.:**	Public D5.3
Work Package(s): **Document Version:** **Issue Date:** **Document Timescale:** Start of the Document: Final version due:	WP5 Vs. 0.3 Month 23 Project Start Date: September 1, 2016 Month 20 Month 23
Deliverable Overview:	**Main document:** D5.3
Compiled by:	CRF
Authors:	Fabio Tango, CRF
Technical Approval:	Fabio Tango, CRF
Issue Authorisation:	Andreas Lüdtke, OFF

DISTRIBUTION LIST		
Copy type[1]	Company and Location	Recipient
T	AutoMate Consortium	all AutoMate Partners

RECORD OF REVISION		
Date	Status Description	Author
21/06/2018	Structure proposal	F. Tango
28/06/2018	First round of comments	M. Eilers
04/07/2018	Second round of comments	E. Landini
05/07/2018	Final structure sent to all partners	F. Tango
18/07/2018	First contributions from partners	E. Landini, M.C. Rahal, F. Tango
27/07/2018	Final contributions from partners	E. Landini, M.C. Rahal, F. Tango, J. Pichen, F. Yan, M. Graf, E. Dogan, A. Castellano
27/07/2018	Document ready for internal review	F. Tango
30/07/2018	Integration of reviewers'	F. Tango

[1] Copy types: E=Email, C=Controlled copy (paper), D=electronic copy on Disk or other medium, T=Team site (Sharepoint)

	comments and feedbacks from partners’	
31/07/2018	Deliverable final version (ready for submission)	F. Tango

Table of Contents

List of Figures

List of Tables

1 Executive Summary

This deliverable describes the integration of the SW/HW components for the TeamMate technologies, as derived by WP2/3/4, into the six project demonstrators in the current project cycle 2, within the current system architecture. These demonstrators are represented by the three driving simulators (from REL, VED and ULM partners) and by the three prototype-vehicles (from VED, ULM and CRF partners). We have also described the set-up tests between the components and the subsystem, in order to ensure proper functionality before demonstrators are ready for the evaluation (in WP6). In addition, the status of each demo has been presented, together with a plan for those enablers and components not yet integrated (to be done in the 3rd Cycle).

2 Introduction

This document describes the integration of the SW components for the TeamMate technologies from WP2/3/4 into the three project demonstrator vehicles (VED, ULM and CRF), in the current project cycle 2, within the system architecture and defined in D5.1. In addition, this deliverable illustrates also the set-up tests between the components and the subsystem, in order to ensure proper functionality before demonstrators are ready for the evaluation (in WP6). This document includes also both the description of the simulator and vehicle demonstrators.

In the next sections, the demonstrators and the related status of their integration are described and detailed, considering both the driving simulators (REL, VED and ULM partners) and the real-prototype vehicles (CRF, VED and ULM partners). Of course, not all demonstrators have the same level of development: for some of them, in particular the ULM driving simulator and the CRF prototype-vehicle, the complete integration details will be available in the next cycle, where dedicated experiments will be carried out.

All in all, we have almost completed the integration of the enablers into the simulators, while we are still working on the integration into the vehicles (as planned in the DOW). In particular, for the vehicles, we will provide a plan to complete the integration in the 3rd cycle (M29-31), where it will be described in D5.6 "TeamMate Car Demonstrator after 3rd Cycle" (leaded by CRF).

3 Common Architecture Framework

In the following we give a brief description of the currently used common architecture. A detailed description can be found in D5.1. Figure 1 shows the currently used approach to integrate the automate enablers together with a given platform, i.e., vehicle or simulator.

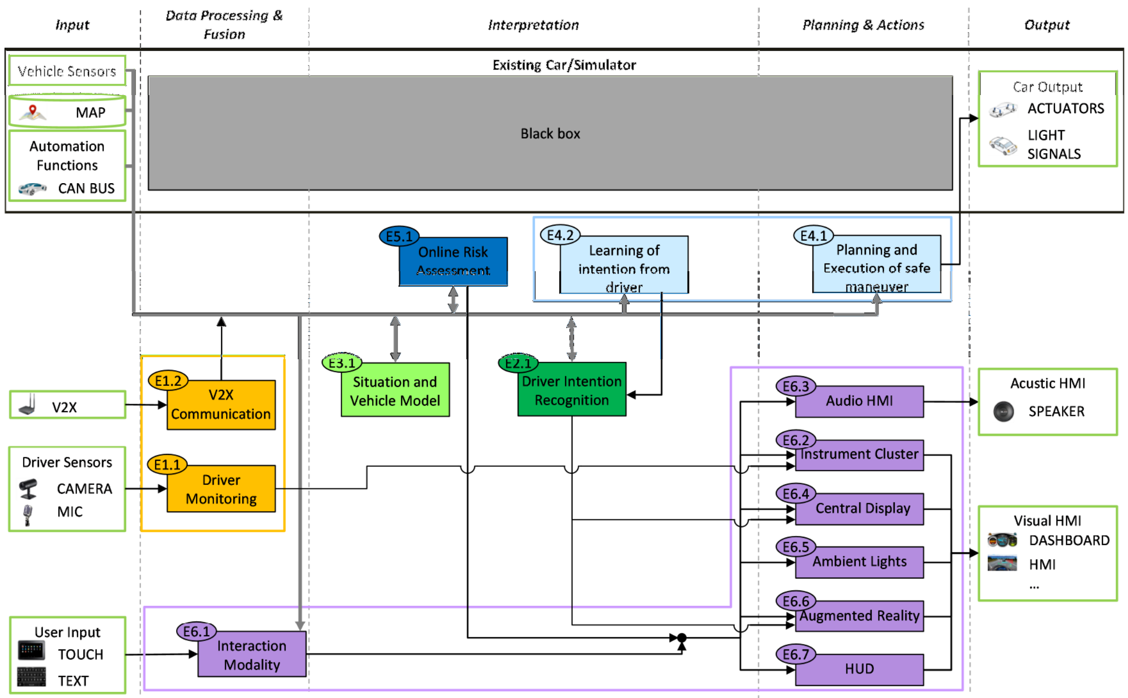

Figure 1: Current Common Architecture.

The enablers support different functional steps: "data processing & fusion", "Interpretation", and "planning & actions". The existing platform might already have modules which also perform one or more of these steps. The TeamMate system does not need to know how these internal modules of the existing platform function or interact. Thus, most parts of the existing platform are considered as a black box. However, the TeamMate system requires the simulator or the vehicle to provide interfaces for certain input and output data.

Input data from automation functions, from maps, and from vehicle sensor are expected to be provided by the existing vehicle or simulator. Further inputs introduced by the Teammate architecture are V2X data, driver sensor data, and user input via touch or text interfaces. Each enabler is represented by a software **component dependent on their concerns**. A **message bus oriented data exchange between** the components is implied to support a communication via one or more channels. The TeamMate system delivers its outputs via acoustic and visual human-machine interfaces to the driver. Further the existing platform is required to provide output interfaces to car actuators and light signals.

4 REL Simulator Demonstrator

REL driving simulator is based on SCANeR Cabin, a vehicle mock-up including real commands and different input modalities. All the components installed and connected to the pc communicates via UDP with the simulation software. The messages communicated with the simulator are the following:

- Acceleration
- Braking
- Steering
- Hand-braking
- Ignition
- Indicators
- Gear

In the next paragraphs the enablers integrated until this cycle and the specific system architecture will be described.

4.1 Enablers and system Architecture

List of enablers that have been integrated into the demonstrator:

ID	Enabler	Demonstrator
E1.1	Driver monitoring system with driver state model for distraction and drowsiness	REL simulator

E6.2 - 3 -4	TeamMate HMI (Cluster + audio, Central stack display, HUD)	REL simulator

Table 1: list of Enablers for REL driving simulator.

In the following figure, the schema of the architecture is presented:

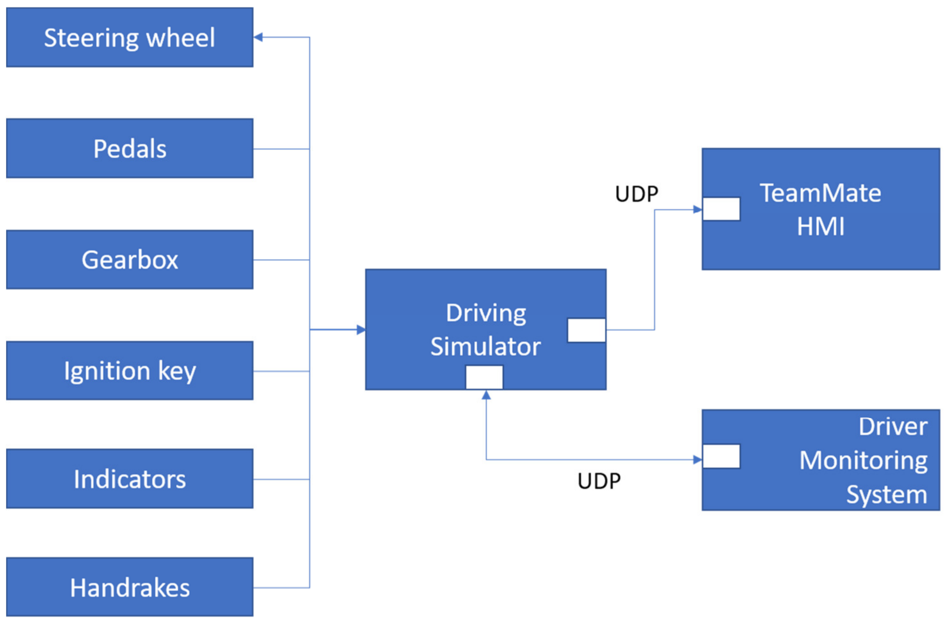

Figure 2: architectural scheme for driving simulator integration.

This architecture is an instantiation of the common scheme as provided by HMT and described in chapter 3.

4.2 Results of Set-up Tests

This section includes check list for functional tests and data communication (as described in T5.5) that show that the integration has been completed (i.e. enablers communicate with demonstrators).

4.2.1 HMI Integration

The HMI software, developed in QT, has been installed on the simulator PC. A communication protocol has been implemented in order to allow the sharing of information between the HMI and the driving simulator.

The communication has been implemented through a client-server system; it was tested with a range of data simulated inside the server in order to verify the correct data packet flow.

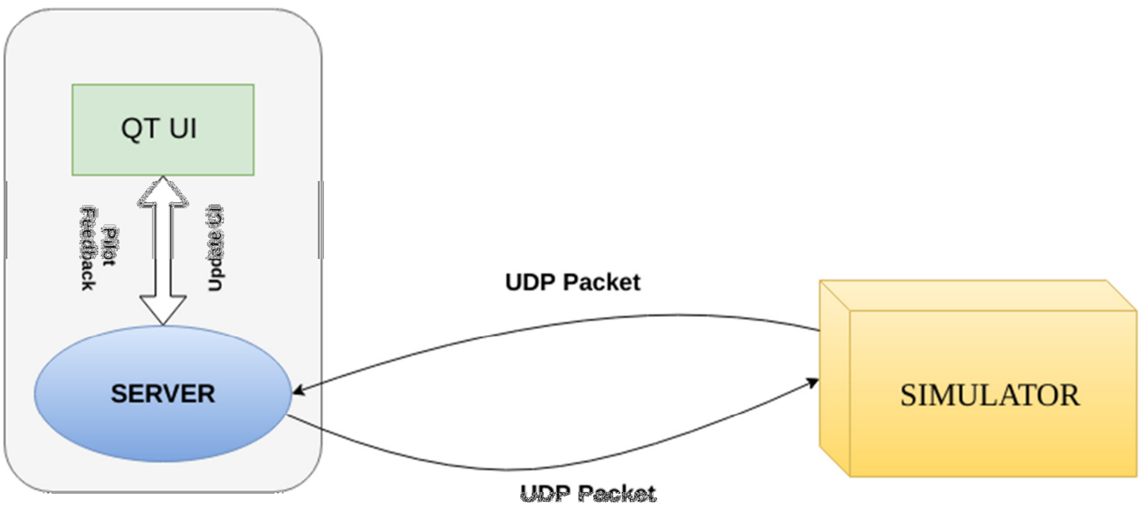

Figure 3: protocol communication and scheme for the HMI integration.

Then, the SCANeR APIs have been integrated in order to read in real-time the data from the simulation. The APIs and the client have been built as part of an integrated module, imported inside the driving simulator. All the information sent by the simulator to the HMI have been tested, comparing them to the Driving Simulator's CONTROLPAD.

The figure describes the procedure and the set-up tests for the HMI integration:

ID	Step and test	Accomplished
1	Install the QT UI software in the simulator PC	Yes
2	Develop and test the communication protocol between the simulator and the HMI	Yes
3	Integrate the SCANeR APIs and the client into the driving simulator building a standalone module and test with random data	Yes
4	Test the data communication and the consistency of all the information (e.g. speed in the real scenario)	Yes
5	Test that the data is consistent with the "state machine" modes (e.g. automated/manual mode)	Yes

Table 2: Set up tests for HMI integration.

The HMI modes have been named and tested by handling them from the script, i.e. the simulator's functionality allowing to force a state or a message.

4.2.2 DMS integration

The first step of the integration was to find the best place for the Driver Monitoring Camera. In fact, the position of the camera has to respect the working range of the face tracker and the face of the driver needs to be in the centre of the image. Different positions were tested in order to keep the position which doesn't hide the HMI and which respects all the recommendations of the Driver Monitoring System.

The second step was to create the 3D world of REL simulator. In order to do this, an application based on the Kinect was used. A chessboard was positioned in the cockpit as a reference. This application allows to select the different objects of the simulator (Windshield, rear mirror, instrument cluster, etc.) in a merged image between the colour image and the depth image and to explain all the coordinates of these objects in the cockpit chessboard world. It generates a world file that can be used by the Driver monitoring application.

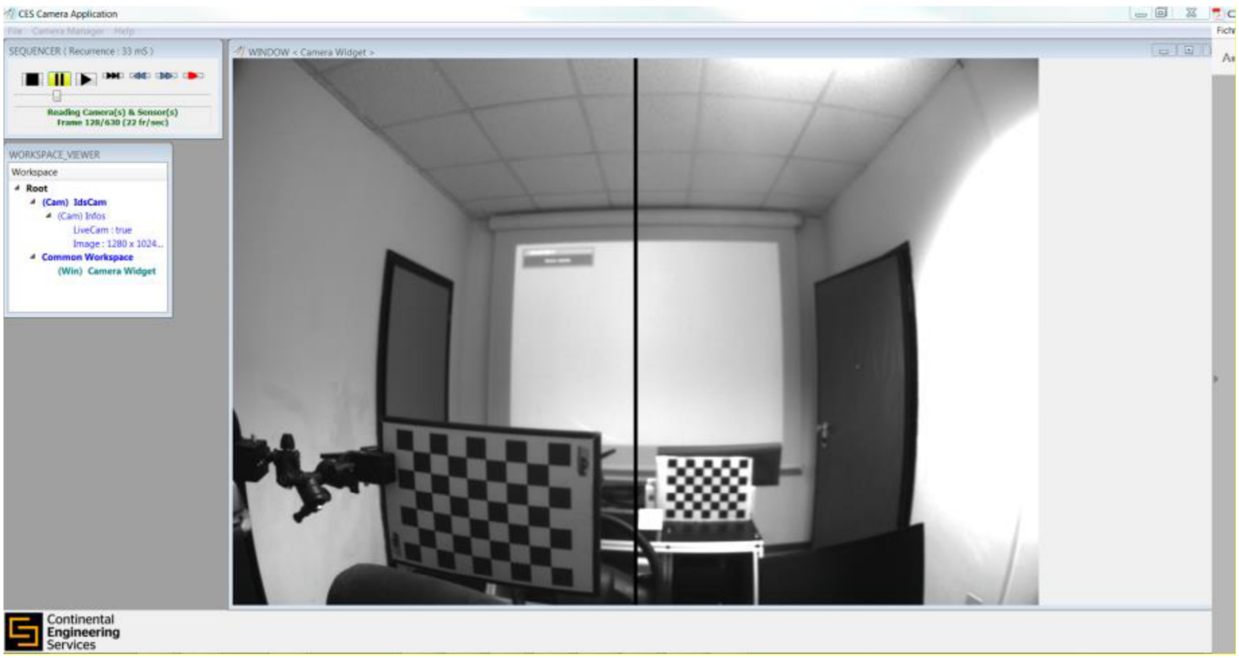

Figure 4: REL driving simulator during DMS installation/calibration

In a third phase, the DMS camera was calibrated with a Continental calibration application. A second chessboard was placed in a potential driver head position with no roll, yaw and pitch. In this position the chessboard has to be entirely visible by the driver monitoring camera. It defined the world coordinate system (WCS).

The last calibration task was to integrate the simulator world in the WCS. We used an auxiliary camera to realize this step. We choose the position of the camera in order to visualize both chessboards. The same application used in the third phase was used to process the transition matrix from the cockpit chessboard to the WCS chessboard. At the end of this step, all the face tracker data (head position, gaze direction, etc.) and the simulator world can be expressed in the WCS.

Figure 5: SW framework for the DMS integration.

In order to send the data on the Simulator PC by UDP, a local network was configured between the DM laptop and the Simulator PC. The UDP communication was validated thanks to a UDP Test application which receives the data sent by the ICP Demo Driver Monitoring application and plots them on the terminal console.

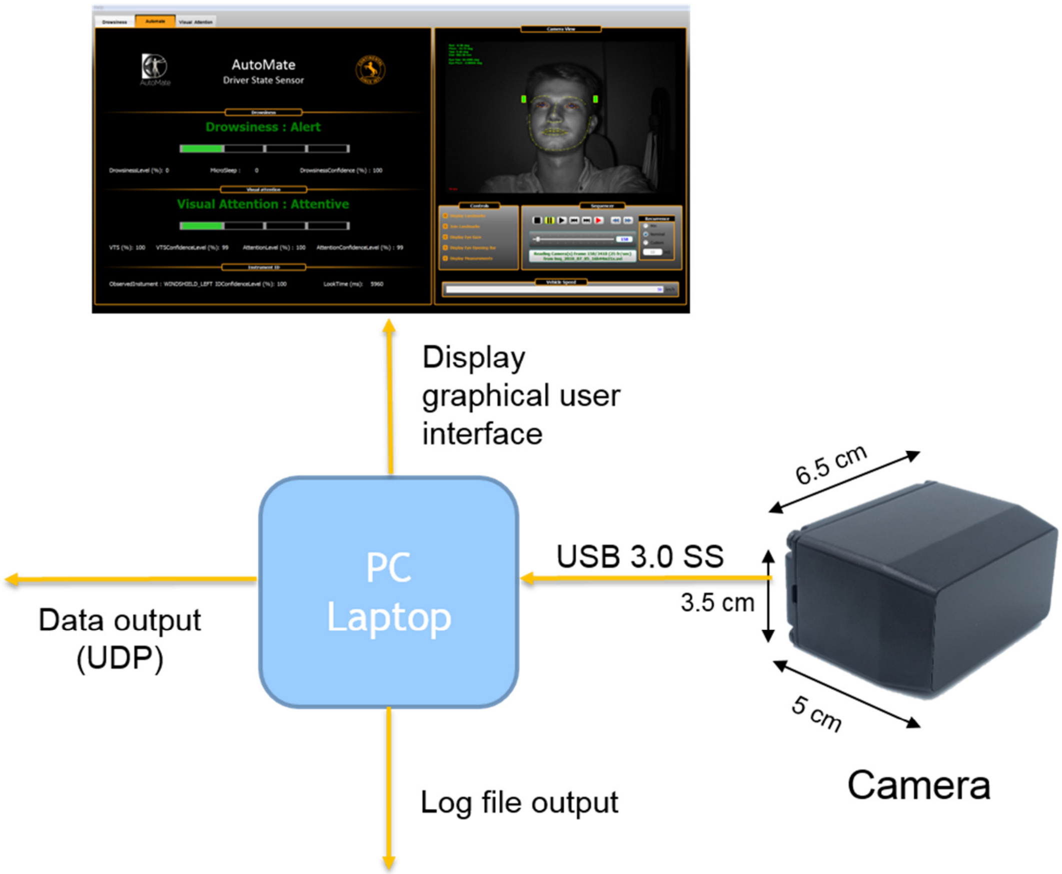

Figure 6: DMS architecture

The figure above schematically shows the steps and the procedure for the DMS integration and testing:

ID	Step and test	Accomplished
1	Place the camera in the most suitable position	Yes
2	Create the driving simulator's 3D world	Yes
3	Calibrate the Camera	Yes
4	Integrate the simulator world in the world coordinate system	Yes

5	Connect with a local network the DMS app and the simulator PC	Yes
6	Create and validate the UDP protocol for data communication	Yes
7	Integrate the classified data into the simulation software	Yes

Table 3: Set-up tests for DMS integration

Finally, the data received by the Simulator PC was integrated into SCANeR studio, the simulation software, in order to use it in the simulation (e.g. to change the HMI state or to consider it into the interaction strategy according to the attention state).

4.3 Demonstrator

Besides the baseline, the TeamMate demonstrator includes a 15.6" screen located behind the steering wheel, in the common instrument cluster position. The screen is connected to the simulator PC through a DPort cable. The software with the HMI is installed on the simulator PC; the information of the simulation software is sent to the HMI software and displayed; the HMI modes (i.e. the "states" of the state machine) are managed by the simulator's script.

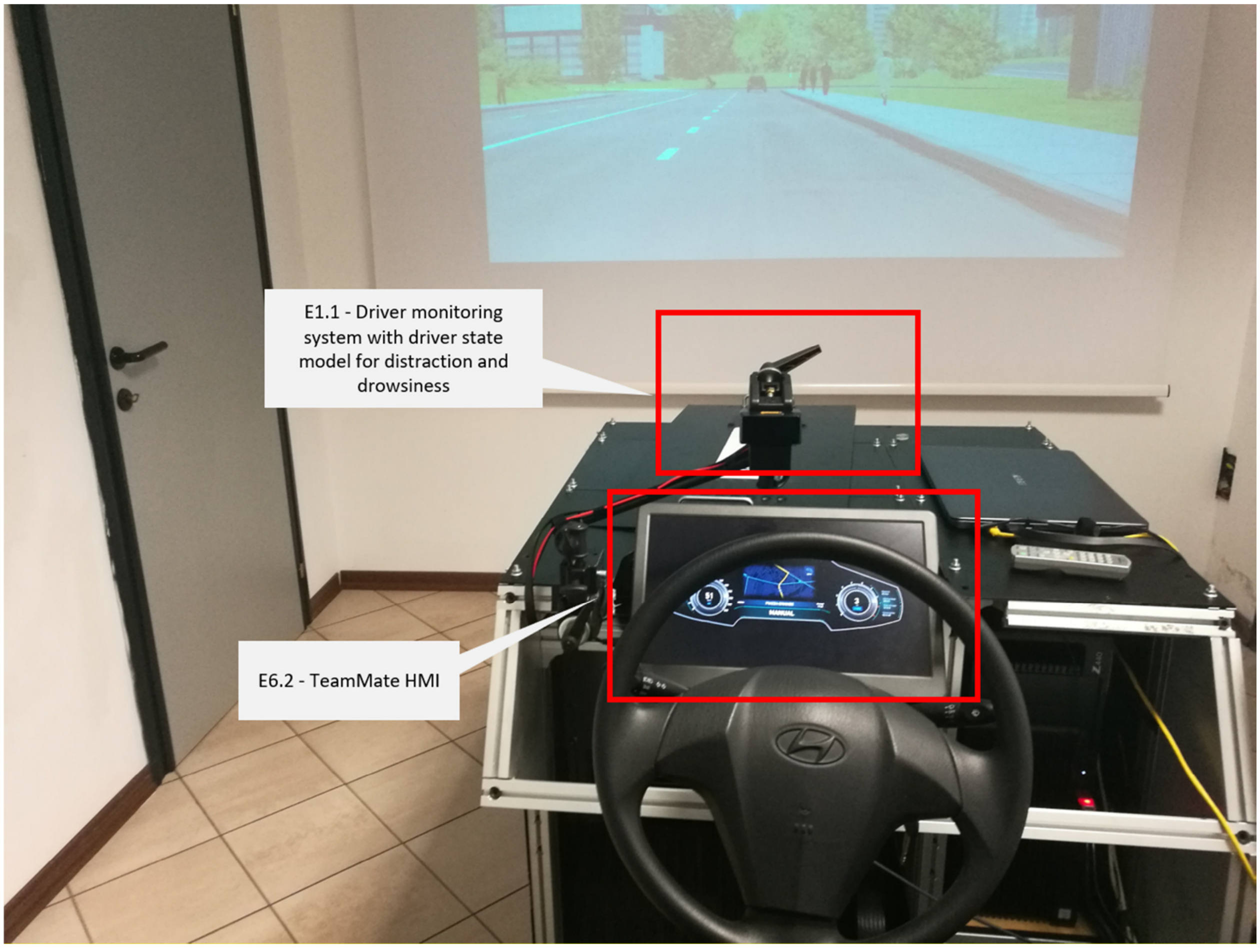

Figure 7 - REL driving simulator

The DMS is instead installed behind the instrument cluster in order to avoid the occlusion of the HMI. Since the scenario of the simulation is shown to the driver through a projector, the camera doesn't obstruct the view of the driver. The camera has been mounted with a mechanical support; it is handled by its own PC and is connected to a local network with the simulator system.

4.4 Plan for 3rd cycle integration

In this section, we show which additional enablers are planned to be integrated in the 3rd cycle:

ID	Enabler	Enabler owner	Notes
E1.1	**Driver monitoring system with driver state model for distraction and drowsiness**	CAF	Already integrated in the 2nd cycle
E2.1	Driver intention recognition	OFF	
E4.1	Planning and execution of safe manoeuvre	TBD	
E4.2	Learning of intention from the driver	OFF	
E5.1	Online risk assessment	OFF	
E6.1	**Interaction modality**	ULM	Already integrated in the 2nd cycle
E6.2 - 3 -4	**TeamMate HMI (Cluster + audio, Central stack display, HUD)**	REL	Already integrated in the 2nd cycle

Table 4: enablers to be integrated in the 3rd cycle in REL driving simulator.

As listed in the table, the integration in REL driving simulator in the first two cycles focused on the integration of the hardware components, such as the DMS, and the HMI, since the use case to be tested was focused on the driver state and the interaction. These tools allowed to measure the indicators to evaluate the most relevant AutoMate features, including the cooperation between the driver and the vehicle.

The 3rd cycle of integration will instead focus on the integration of the enablers allowing the adaptive automation functionalities. In order to integrate the E2.1 ("Driver Intention Recognition") and the E4.2 ("Learning of intention from the driver"), REL provided the enabler owners (i.e. OFF) with the part of the scenario in the simulation in which the intention should be recognized and learned, i.e. the roundabout. The part of the scenario that will be modelled specifically for EVA's scenario and then integrated into REL driving simulator will be the "intention to enter the roundabout", in order to adapt the vehicle behaviour according to the specific drivers' needs.

5 VED Simulator Demonstrator

In this section, the VED simulator demonstrator is described.

5.1 Enablers and system Architecture

List of enablers that have been integrated into the demonstrator:

ID	Enabler	Demonstrator
E1.1	Driver monitoring system with driver state model for distraction and drowsiness	DMS : Yes, • Drowsiness : Yes • Distraction : Yes This module has been successfully installed and tested by CAF in VED simulator.
E1.2	V2X communication	Simulated
E3.1	Situation and vehicle model	From simulator (idealistic perception and ego-pose/state)
E4.1	Planning and execution of safe manoeuvre	From simulator (Scanner autonomous driving module)

E6.1	Interaction modality	No
E6.2 - 3 -4	TeamMate HMI (Cluster + audio, Central stack display, HUD)	Cluster : Yes, provided by REL, installed by VED. Audio : No, the audio messages were in English. In the third cycle VED will provide a French version of the messages. Bip messages where implemented by VED

Table 5: list of Enablers for VED driving simulator.

5.2 Results of Set-up Tests

This section includes functional tests and data communication that show that the integration has been completed (i.e. enablers communicate with demonstrators).

5.2.1 DMS

Acyclic test procedure has been done to install the DMS enabler (E1.1). The enabler E1.1 software has been installed with a SmartEye eye tracker system. It takes the outputs of the SmartEye system and produces as an output the state of the driver.

In the beginning the E1.1 crashed after two minutes of run, the UDP packets exchanged between the software of the E1.1 and the smartEye system have been checked and recorded for post-processing in order to refine the tuning of the application and the configuration of the system. To ease this, CAF has developed an intermediate module (a gateway) between the enabler E.1.1 and SmartEye.
The enabler E1.1 is now used in the simulator without any problem. It will be integrated in the second use case of the Martha scenario dealing with the distraction of Martha on the highway.

The calibration and testing of the enabler E1.1 on VED simulator followed the same schema of integration than REL-simulator since REL and VED has the same simulators and a smartEye system for eye tracking (for more details see, section **Errore. L'origine riferimento non è stata trovata.**, page **Errore. Il segnalibro non è definito.**):

ID	Step and test	Accomplished
1	Place the camera in the most suitable position	Yes
2	Create the driving simulator's 3D world	Yes
3	Calibrate the Camera	Yes
4	Integrate the simulator world in the world coordinate system	Yes
5	Connect with a local network the DMS app and the simulator PC	Yes

6	Create and validate the UDP protocol for data communication	Yes
7	Integrate the classified data into the simulation software	Yes

Table 6: Set-up tests for DMS integration in VED simulator.

5.2.2 HMI

The definition of the different HMI states that were needed for each use-case was made. An executable of the HMI has been coded by REL in order to test if the HMI included all needed states.

All the displayed messages were translated in French by VED and integrated in the code of the HMI provided by REL. For the vocal messages, VED decided to remove them as they were in English and needed to be translated in French in order to be included in the HMI, this part will be done in the third cycle of the project.

Many tests of the exchange of information between the HMI compiled code and the driving simulator were made by running different scenarios in the driving simulator and the check of the update of the state of the HMI and the variables displayed in the HMI.

Two HMI sizes were tested in order to determine the one that fit the most VEDECOM display screen. The size allowing readability of written pieces of information was chosen.

REL sent the code allowing VEDECOM to modify the written messages for each HMI State and to add vocal messages in French.

5.3 Demonstrator

VED Simulator Platform is a static driving simulator with 2 places (one driver seat and one front passenger seat). It allows to display the driving scene on 4 screens of 32” with a total field of view of 120°. There are 3 integrated rear-view mirrors and an instrumental board of 10".

This baseline platform is also equipped with:

- 3 pedals (with a possibility to only use two pedals)
- Gearbox 7 or Automatic Option
- Peugeot Steering Wheel & Commodo Functions
- Force feedback system
- Sound system with 5 speakers & subwoofer
- 4 infrared Cameras

Figure 8: VED Simulator.

This simulator runs on Oktal SCANeR driving simulation software.

5.4 Plan for 3rd cycle integration

All the needed enablers for the Martha scenario on VED simulator has been integrated and tested.

Audio: In the third cycle VED will provide a French version of the messages and REL will provide the commented code in order to integrate the French messages in the audio part of the HMI.

6 ULM Simulator Demonstrator

In this section, the ULM simulator demonstrator is described.

6.1 Enablers and system Architecture

List of enablers that have been integrated into the demonstrator:

ID	Enabler	Demonstrator
E1.1	Driver monitoring system with driver state model for distraction and drowsiness	DMS : Yes, • Drowsiness : Yes • Distraction : Yes This module has been successfully installed by CAF and tested in the ULM simulator.
E3.1	Situation and vehicle model	From simulator
E3.2	Driving task model - DriveGOMS	Yes, first experiment was conducted to analyze the PETER scenario
E4.1	Planning and execution of safe manoeuvre	From simulator

E6.1	Interaction modality	Different interaction modalities implemented and tested
E6.5	Augmented reality	Yes

Table 7: list of Enablers for ULM driving simulator.

6.2 Results of Set-up Tests

The implemented Enablers were tested separately and will be integrated and tested all together in the next experiment.

6.3 Demonstrator

ULM driving simulator is equipped with the SILAB driving simulation engine. The simulator is a mock-up that represents a real car (as shown in Figure 5) with a driver and a passenger seat. Additionally, there are several features in the driving simulator (see D1.3):

- steering wheel (force-feedback)
- pedals
- indicators
- central touch panel
- displayed rear mirrors (central, left, right)
- Smart-eye camera (static eye tracking system)

Figure 9: ULM driving simulator inside projection screen.

The driving simulator also includes three high definition beamers that project the simulated environment onto a projection screen in front of the driver to create an immersive driving environment (see D1.3).

6.4 Plan for 3rd cycle integration

In this section, we show which additional enablers are planned to be integrated in the 3rd cycle:

ID	Enabler	Demonstrator
E2.1	Driver intention recognition	Not yet, but in contact with OFF to implement it.

E4.2	Learning of intention from the driver	Not yet, but in contact with HMT to implement it.
E5.1	Online risk assessment	Not yet, but in contact with OFF to implement it.
E6.2 - 3 -4	TeamMate HMI (Cluster + audio, Central stack display, HUD)	Delivered but not implemented yet

Table 8: enablers to be integrated in the 3rd cycle in ULM driving simulator.

During the integration workshop, data format representing enablers status and outputs have been defined. In particular, OFF and HMT enablers will be provided as a DPU (SILAB), together with a list of needed input, while ULM will provide the test track modules from SILAB.

All enablers will be integrated and tested during the next experiment in the 3rd Cycle.

7 CRF Vehicle Demonstrator

In this section, the CRF vehicle demonstrator is described.

7.1 Enablers and system Architecture

List of enablers that have been integrated into the demonstrator:

ID	Enabler	Demonstrator
E1.1	Driver monitoring system with driver state model for distraction and drowsiness	Yes, done in a dedicated workshop (July 2018).
E3.1	Situation and vehicle model	Yes, as developed by CRF.
E6.5	Augmented reality	Not compliant with CRF demo-vehicle.

Table 9: list of Enablers for CRF demonstrator car.

In case the enabler is not installed yet, a possible date of installation is given in Section 7.5.

7.2 Results of Set-up Tests

This section includes check list for functional tests and data communication (as described in T5.5) to show that the integration has been completed (i.e. enablers communicate with demonstrators). In respect to the CRF demonstrator, the vehicle preparation and implementation are a little bit in a delay; therefore, we focus specifically on the integration of the Driver Monitoring System (DMS) as provided by CAF partner (since this activity is very similar to the integration already performed in REL driving simulator, some parts will be in common, but duplicated for sake of readability).

Like in the integration in the REL driving simulator, also here the first step was to find the best place for the Driver Monitoring Camera. In fact, the position of the camera has to respect the working range of the face tracker and the face of the driver needs to be in the centre of the image. Different positions were tested in order to keep the position which doesn't hide completely the on-board HMI of the vehicle (e.g. the panel instrument) and which respects all the recommendations of the DMS.

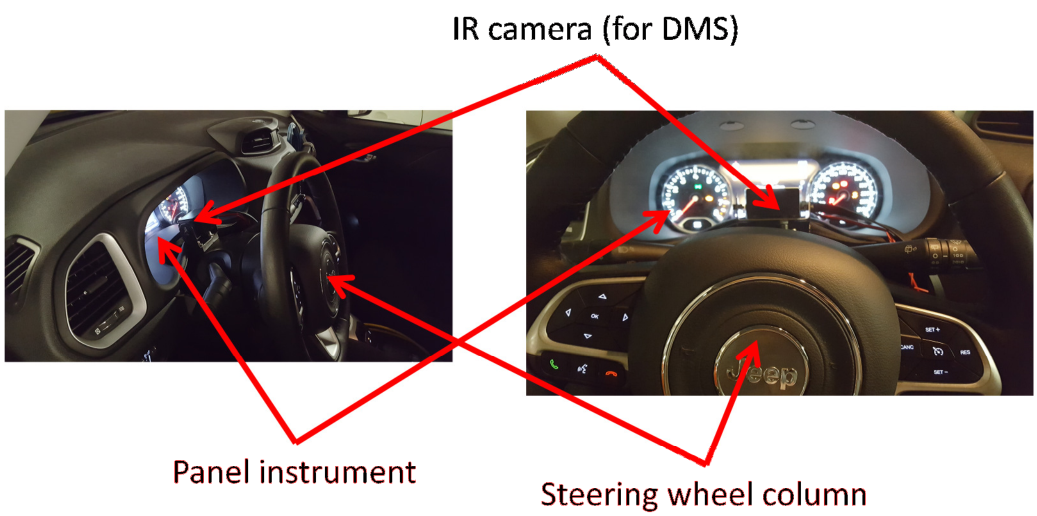

Figure 10: CRF demo vehicle during DMS installation/calibration.

The second step was to create the 3D world of CRF prototype-vehicle. In order to do this, an application based on the Kinect was used. A chessboard was positioned in the cockpit as a reference. This application allows to select the different objects of the simulator (Windshield, rear mirror, instrument cluster, etc.) in a merged image between the colour image and the depth image and to explain all the coordinates of these objects in the cockpit chessboard world. It generates a world file that can be used by the Driver monitoring application. This can be visualized and detailed also in similar section 4.4.2 (for REL driving simulator).

In the third phase, the DMS camera was calibrated with the Continental calibration application. To achieve that, a second chessboard was placed in a potential driver head position with no roll, yaw and pitch. In this position the chessboard has to be entirely visible by the driver monitoring camera. It defined the world coordinate system (WCS).

The last calibration task was to integrate the simulator world in the WCS, using an auxiliary camera. The position of the camera has been chosen to visualize both chessboards, using the same application of the third phase, in order to process the transition matrix from the cockpit chessboard to the WCS chessboard (as also described in the same procedure for the integration of DMS in REL simulator). Finally, all the face tracker data (head position, gaze direction, etc.) and the simulator world can be expressed in the WCS.

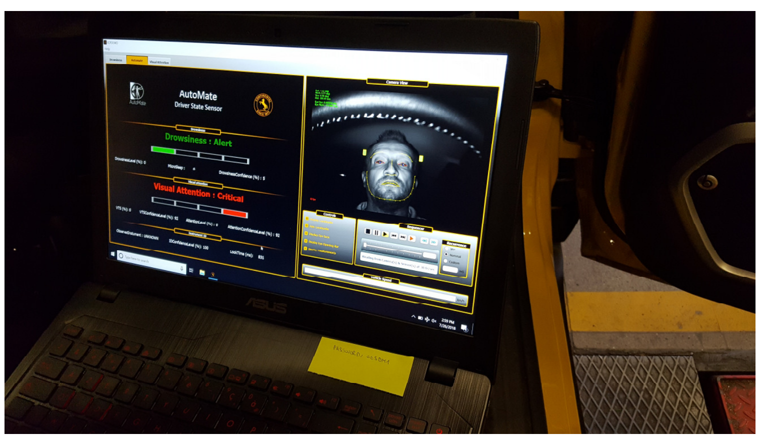

Figure 11: SW framework for the DMS integration.

In order to send the data on the Simulator PC by UDP, a local network was configured between the DM laptop and the Simulator PC. The UDP communication was validated thanks to a UDP Test application which receives the data sent by the ICP Demo Driver Monitoring application and plots them on the terminal console.

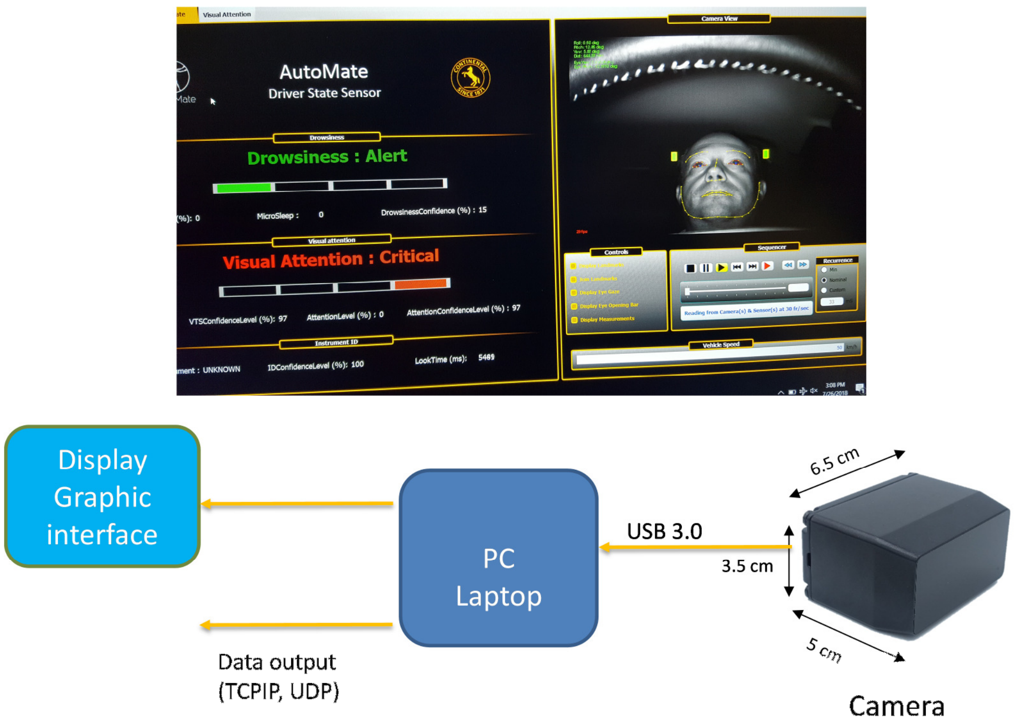

Figure 12: DMS architecture for CRF case.

The following table schematically shows the steps and the procedure for the DMS integration and testing:

ID	Step and test	Accomplished
1	Place the camera in the most suitable position	Yes
2	Create the driving simulator's 3D world	Yes
3	Calibrate the Camera	Yes

4	Integrate the simulator world in the world coordinate system	Yes
5	Connect with a local network the DMS app and the simulator PC	Yes
6	Create and validate the UDP protocol for data communication	Yes
7	Integrate the classified data into the simulation software	Yes

Table 10: Set-up tests for DMS integration in CRF vehicle.

Finally, the data received by the Simulator PC was integrated into the vehicle system architecture, in order to use it in the demonstrator (e.g. to change the HMI state or to consider it into the interaction strategy according to the attention state).

Hereafter, we report the main characteristics of the DMS:

- Camera module (All in One unit) dimensions: 6.5cm x 5cm x 3.5cm
- 3 m USB3 cable + 3m extension
- Camera power supply: 12V DC 2A supply
- PC Laptop dimension: 25.5cm (D) x 38.1cm (W) x 3cm (H), with 120W max.

The main available features are listed as following:

- Face detection/tracking
- Drowsiness estimation
- Visual distraction
- Face recognition

- Eye opening, closed eyes
- Eye/head gaze depending
- Observed areas (display, instruments, etc.)
- Active exposure control
- Head Eye Detector/Tracker
- Camera Blockage
- Camera world calibration

More details can be found in the deliverables D2.4 and D2.6.

7.3 Demonstrator

The CRF demonstrator is based on a Jeep Renegade 1.4 MultiAir 140HP DDCT, as illustrated in the figure:

Figure 13: CRF demonstrator vehicle.

In the following figures, the location on the vehicle of the sensors we used, is sketched:

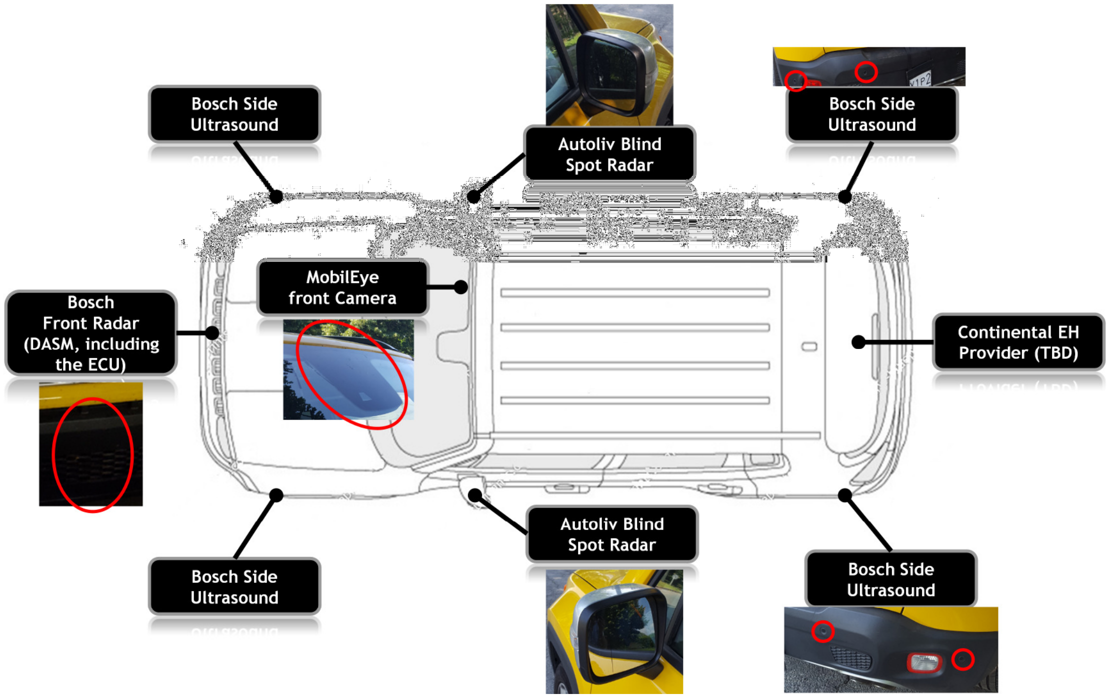

Figure 14: location of sensors in CRF prototype vehicle.

The frontal part is covered by two sensors: a medium range radar and a mono-camera, which is able to provide several features, such as object detection and classification, lane recognition, traffic-sign detection (speed limits, roundabout, etc.) and so on. This sensorial platform is used to feed the enablers E3.1, E4.1 and E5.1.

At the moment, the electronic horizon has still to be integrated and it will be done in the 3rd cycle.

The following deployment diagram shows where the different SW modules are implemented, in terms of HW:

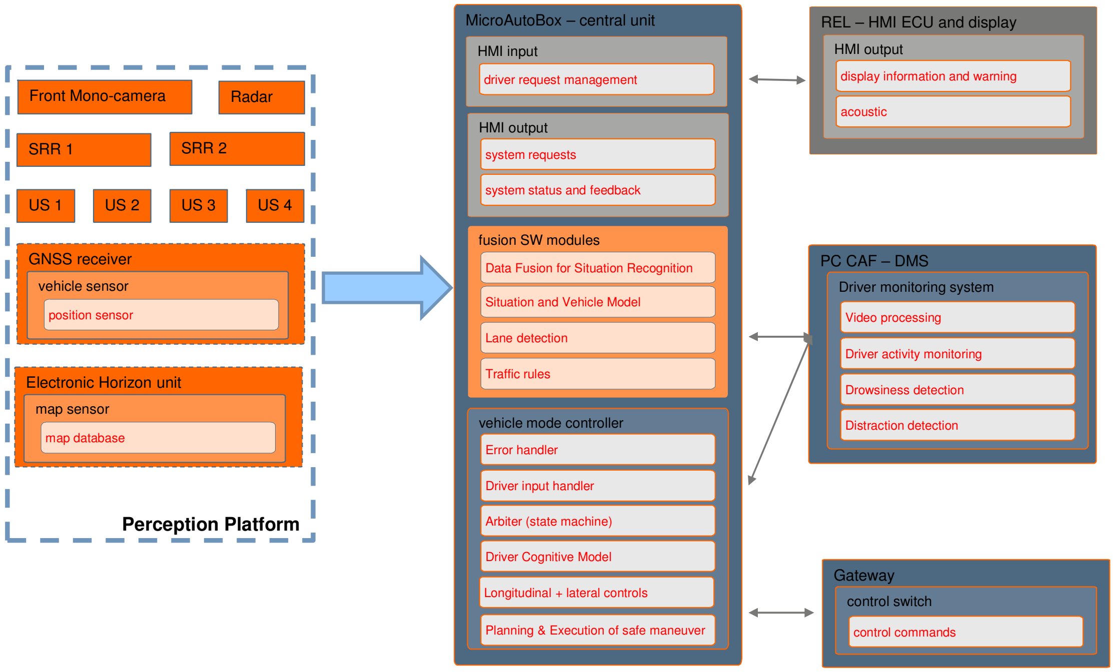

Figure 15: deployment diagram for SW modules in specific HW components, in CRF prototype vehicle.

From the deployment diagram, it is possible to see that the most relevant enablers (namely, the corresponding SW modules) are implemented[2] in the *dSpace MicroAutoBox*, which is the system central unit. Another ECU is used for the implementation and integration of the HMI. The PC is used for the image processing and the monitoring of driver's status. In particular, the enablers are/will be installed in the car as follows:

- The enablers E3.1, E4.1 and E5.1 will be embedded in the *MicroAutoBox* Central Unit. For the enabler E4.1, it is still under investigation of a specific control PC is necessary.
- The enabler E1.1 is embedded on a dedicated hardware (PC connected with the IR camera from CAF partner).

[2] Some enablers will be implemented in the 3rd cycle.

The enabler E2.1 is at the moment under evaluation, with OFF colleagues, to understand if its integration on real-vehicle is feasible.

All the enablers related to HMI will be installed in the 3rd cycle, in a dedicated workshop with REL colleagues; a dedicated ECU is currently foreseen, as depicted in the figure, which can be also a gateway to convert data from UDP to CAN protocols.

The perception platform (see left part in the figure) was already present before AutoMate and not developed for this project (only adaptation has been necessary).

The electrical system is stored in the car trunk, which is illustrated in the next figure.

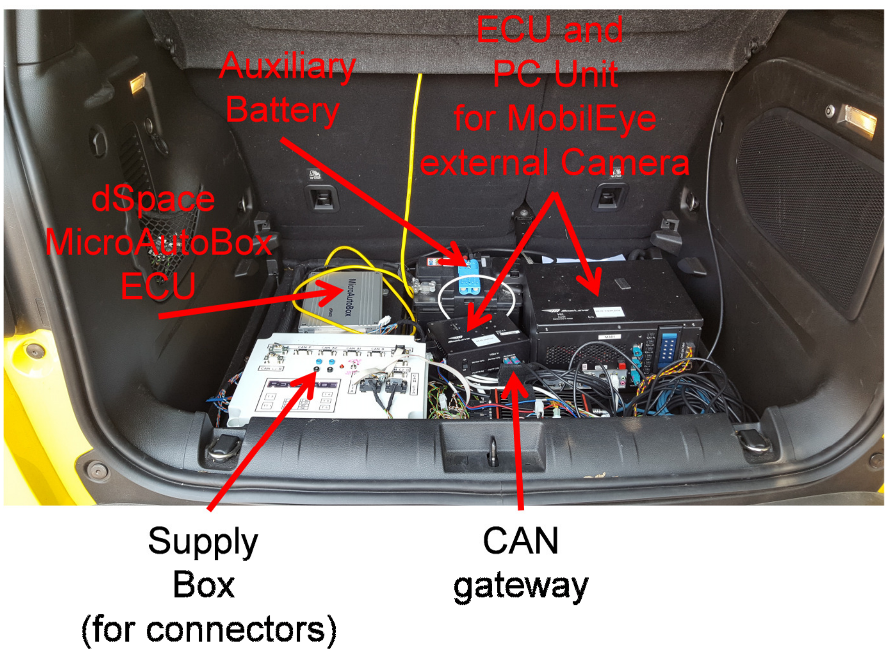

Figure 16: the car trunk, with the hardware and electronic part of the CRF vehicle.

With reference to the figure, the SW modules corresponding to the enablers at the moment implemented in *dSpace* ECU. The ECU and the PC unit are used for data processing and debugging purposes, respectively, of the front camera, the most relevant source of information to detect and recognize the external scene. In the car trunk, we can access to the vehicle data and to the actuators via the primary and the secondary CAN networks, as well as the private network (by means of the dedicated CAN gateway).

The following messages are exchanged between modules:

Name	Unit	Range	Type	Meaning
ACC_rel_speed	m/s	0 --70	Float32	Main object relative speed
ACC_object_category	enum	0--255	Uint8	Type of object (CAR, TRUCK)
ACC_lateral_speed	m/s	0-70	Float32	Main object lateral relative speed
ACC_target_id	enum	0--255	Uint8	ID object
ACC_confidence	enum	0--1	Boolean	Valid object status
ACC_distance	m	0-200	Uint8	Main object relative distance
ACC_angle	degree	-60--+60	Int16	Main object heading angle
RoadSign_Type	enum	0--255	Uint8	Road Sign Type (i.e: speed limit, RoundBound signals)
RoadSupplementSign_Type	enum	0-255	Uint8	Supplementary info of road sign type (i.e: fog, rain added info)
Sign_Position_X	m	0--200	Uint8	X sign position from camera vehicle mounting position
Sign_Position_Y	m	-60--+60	Int16	Y sign from camera vehicle mounting position

Name	Unit	Range	Type	Meaning
LeftLine_Type	enum	0 --15	Uint8	0x0 - Undecided 0x1 - Solid 0x2 - RoadEdge 0x3 - Dashed 0x4 - DoubleLane 0x5 - Bott'sDots 0x6 - Barrier 0xF - Invalid
LeftLine_Quality	enum	0--3	Uint8	0x0 - Low 0x1 - Low 0x2 - Medium 0x3 - High
LeftLinePosition	m	-128--+127	Int8	Physical distance between left lane mark and camera on the lateral position.
LeftLineCurvature	1/m	-0.032--+0.032	Float32	Left line curvature parameter
LeftHeading_Angle	rad	-0.357--+0.357	Float32	Left line heading angle
RightLine_Type	enum	0 --15	Uint8	0x0 - Undecided 0x1 - Solid 0x2 - RoadEdge 0x3 - Dashed 0x4 - DoubleLane 0x5 - Bott'sDots 0x6 - Barrier 0xF - Invalid
RightLine_Quality	enum	0--3	Uint8	0x0 - Low 0x1 - Low 0x2 - Medium 0x3 - High
RightLinePosition	m	-128--+127	Int8	Physical distance betweenright lane mark and camera on the lateral position.
RightLineCurvature	1/m	-0.032--+0.032	Float32	Right line curvature parameter
RightHeading_Angle	rad	-0.357--+0.357	Float32	Right line heading angle

Table 11: AutoMate signals used in CRF car, for longitudinal control and for lateral control, respectively.

All these messages are defined in such a way to be compliant with the signals defined in the openAPI framework for AutoMATE (see Deliverable D5.1) and are used in the CRF demonstrator as message format between the integrated enablers.

7.4 Plan for 3rd cycle integration

Additional enablers are planned to be integrated in the 3rd cycle, as described in the following table:

ID	Enabler	Demonstrator
E2.1	Driver intention recognition	Discussed during the integration workshop in Paris (June 2018). It will be verified in October 2018 with OFF partner.
E3.1	Situation and vehicle model	Further integration to be discussed with DLR partner.
E4.1	Planning and execution of safe manoeuvre	Discussed during the integration workshop in Paris (June 2018). It

		will be verified in October 2018, with ULM partner.
E5.1	Online risk assessment	Discussed during the integration workshop in Paris (June 2018). It will be verified in October 2018.
E6.2 - 3 -4	TeamMate HMI (Cluster + audio, Central stack display, HUD)	Discussed during the integration workshop in Paris (June 2018). It will be installed and verified in October 2018, with REL partner.

Table 12: enablers to be integrated in the 3rd cycle in CRF demonstrator car.

The integration in CRF real-vehicle has been a little bit in delay in the first two cycles, focusing mainly on the preparation and adaptation of sensor data-fusion (SDF), that is the Enabler E3.1 "Situation and vehicle model" and on the integration of the DMS (including hardware and software components). These tools allowed the preparation of the baseline cars for the Eva scenario (see also deliverables D5.1 and D1.1).

The 3rd cycle will focus on the integration of the remaining enablers (as described above), in particular the HMI components, allowing the adaptive automation functionalities. For some modules, the discussion with owner partners is still open, in order to understand if this is feasible (such as for E2.1 “Driver Intention Recognition” and for E4.1 “Planning and execution of safe manoeuvre”, where the adaptation to recognize the “intention to enter the roundabout”, as well as the planning of a safe trajectory in Eva’s scenarios, are necessary).

8 VED Vehicle Demonstrator

In this section, the VED vehicle demonstrator is described.

8.1 Enablers and system Architecture

In this section we detail the installed and nearly installed enablers in the VED demo-car:

ID	Enabler	Demonstrator
E1.2	V2X communication	Yes for the receiver. The protocol of communication is under discussion with BIT, to perform V2X tests.

Table 13: list of Enablers for VED demonstrator car.

In case the enabler is not installed an approximate date of installation is given in the last sub-section.

8.2 Results of Set-up Tests

The VED demo-car has not yet integrated all the enablers of the project. We installed only a V2X on-board unit, which will be only behave as a receiver in the case of Martha scenario that receives the roadworks area related information from the infrastructure (road side unit).

Our receiver has been tested with different standardized ETSI messages. We receive all kinds of ITS messages, CAM, DENM, SPAT, etc.

In order to fulfil the Martha scenario (A2H in perception) VED has asked BIT for sample data from their V2X hardware. VED receiver was not be able to understand the Wireshark capture file containing the sample data. Further cooperation is required to align the transmitted data.

8.3 Demonstrator

This section describes the current status of VED demonstrator. We will describe the different installed hardware that will welcome the different enablers and ease the integration process.

Let us remember (as described in the D5.1) the different sensors and PCs embed in the demonstrator.

VED's demonstrator is an automated vehicle which is under construction from a Citroen C4 Picasso platform. This vehicle has two driving modes:

- Standard manual driving and
- Level 4 + self-driving mode.

In manual driving mode, the performance of the vehicle remains compatible with a standard vehicle, with added access to the connectivity services of Cooperative ITSs (standardized V2X messaging). Access to these services is guaranteed by the on-board platform which includes several communication media: 802.11n standard WiFi, 802.11p and 3G/4G cellular network, which is compatible with the Martha scenario.

Following the architecture of the figure bellow the VED demo car will be equipped with:

- 6 **Valeo Scala LIDARs** ensuring 360° of coverage, with a minimization of blind spots.
- 2 **Radars Continental** ARS 408, one at the front and one in the rear of the vehicle.

- 2 **cameras** at the front and 2 in the rear allowing both mono and stereo vision for ground detection and lane marking detection.
- A **Velodyne VLP 16** for hd Maps building.
- **Atlans IMU** for accurate positioning.
- **Spetentrio RTK (real time kinematics) GPS**.

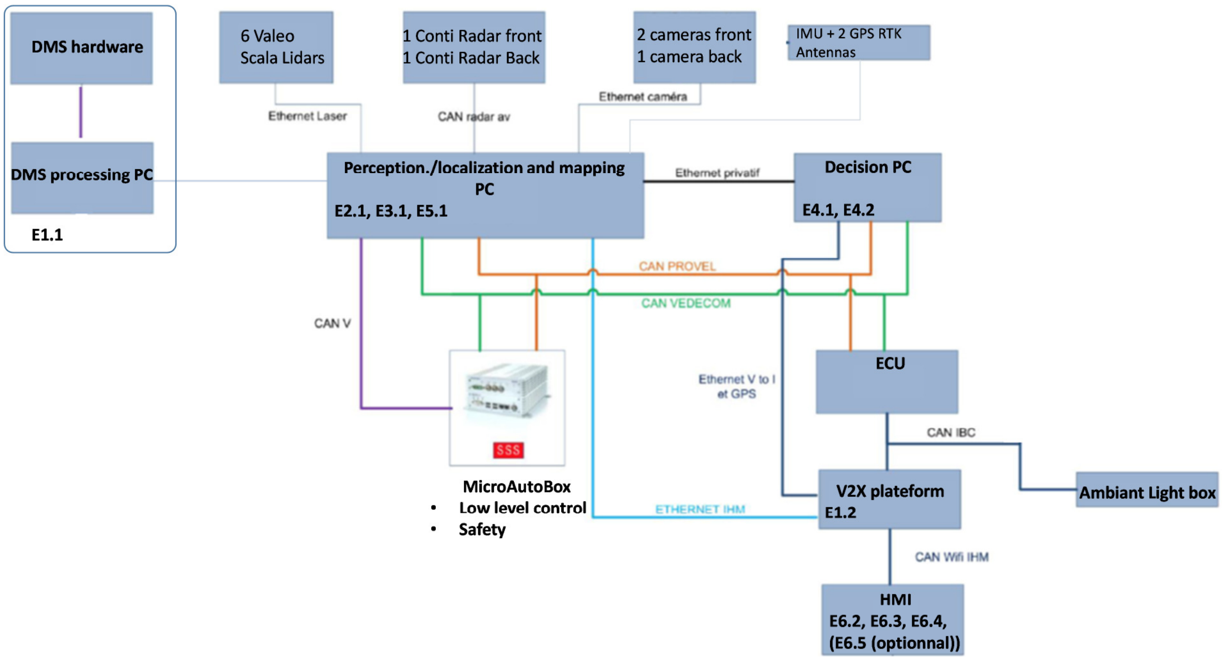

Figure 17: HW architecture and deployment diagram of VED-Demo car.

The picture bellow shows the perception/localization capabilities of the car in order to ease the installation of the perception layer (situation and vehicle model, online risk assessment and path planning).

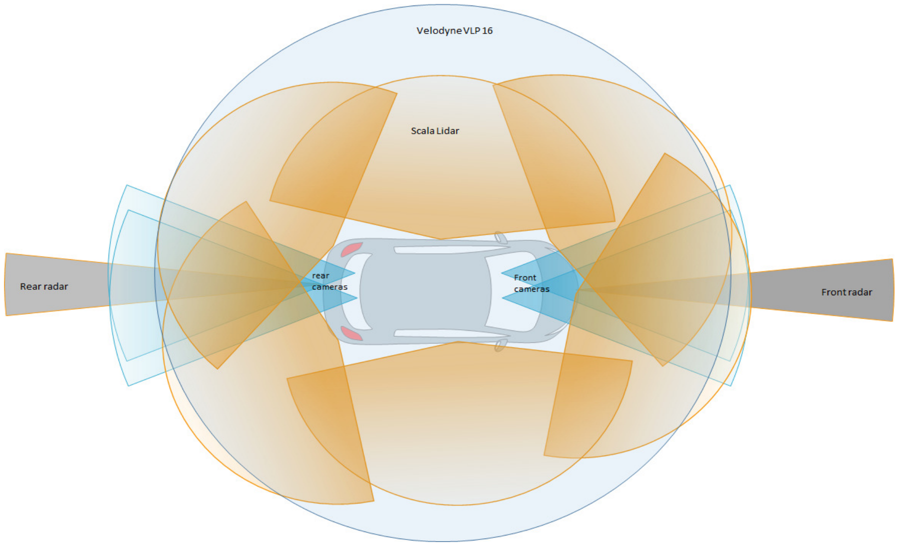

Figure 18 : overview of the localization and perception sensors capabilities.

Figure 19: Global overview

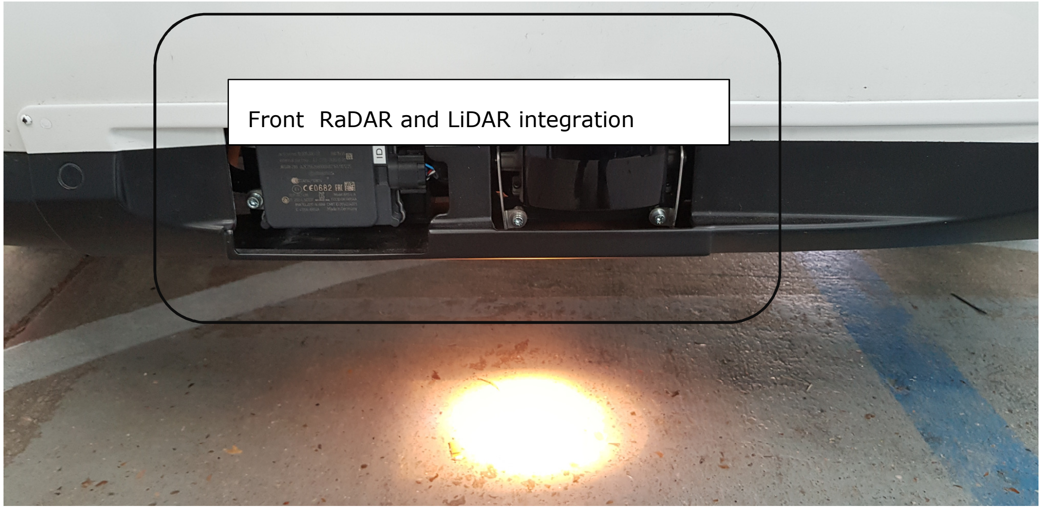

Figure 20: RaDAR, LiDAR and odemeter (light on the ground) installation.

The pictures bellow show the different hard ware integrations in the vehicle. In particular, the figure shows an example of the integration of the front/lateral LiDAR and a front RaDAR in addition to the optic odometer (light on the ground). The LiDAR and the radar are used for front obstacles detection and will feed the enablers E.5.1 and E.3.1. On the other hand the odometer will an accurate entry to the IMU in order to refine the position of the ego-vehicle which is a very important input to the enablers E3.1, E.4.1 and E.5.1.

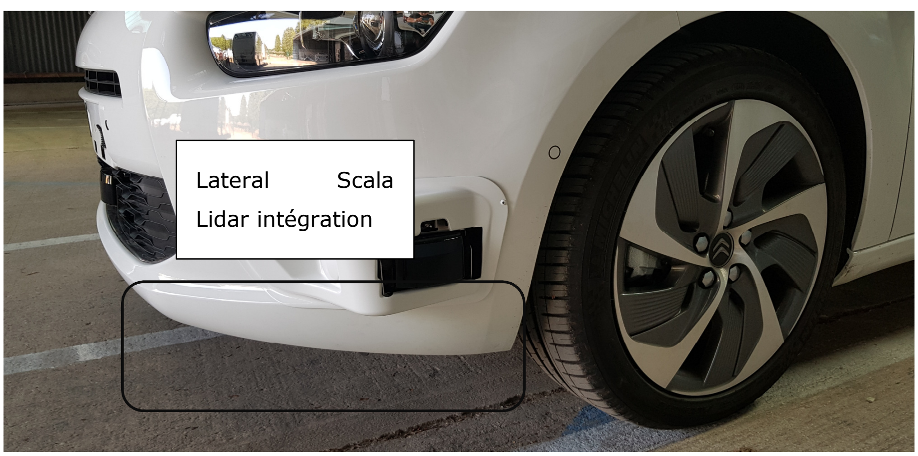

Figure 21: Lateral LiDAR integration.

The installations on the rough of the car are shown in the following figure:

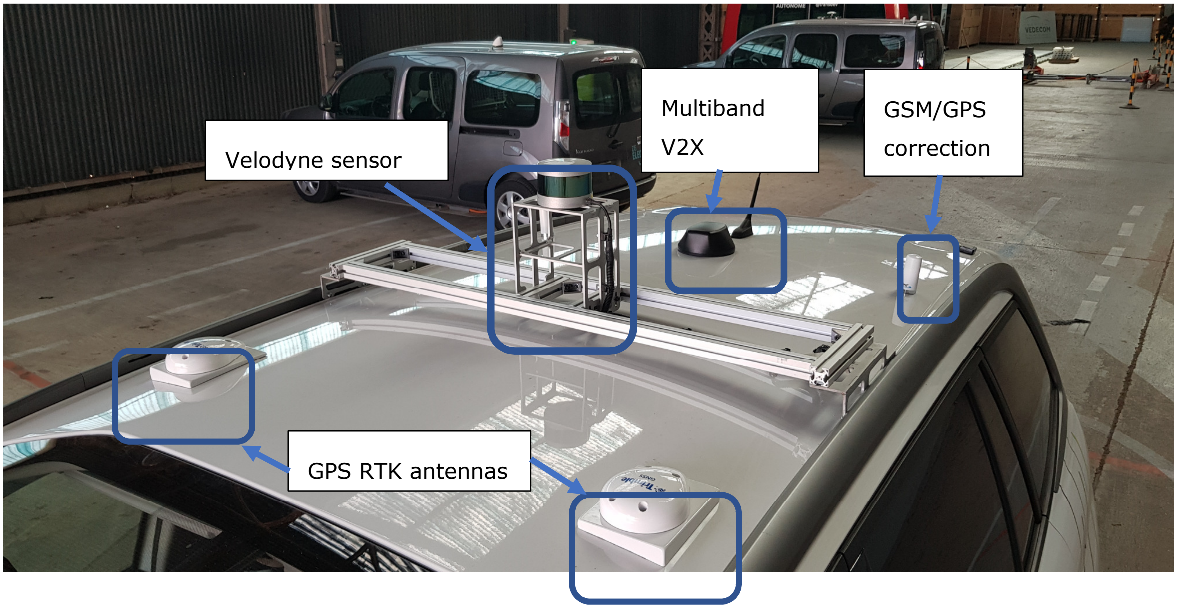

Figure 22 : Integrations on the rough of the VED demo car.

We can see on this figure the following sensors and antennas:

- 2 RTK GPS antennas for the positioning
- 1 multi-band V2X antennas, this platform includes several communication media: 802.11n standard WiFi, 802.11p and 3G/4G cellular network, which is compatible with the Martha scenario, this part will be used by the enabler E1.2
- A Velodyne VLP 16 for hd Maps building, ground and obstacles detection.
- GSM (2G/3G) antenna which can connect to a correction server in order to filter out and decrease the uncertainty of the position received by the GPS.

The electrical system is stored in the car trunk, the next figure gives a panorama of the different PCs switches that are installed. In the car trunk, we can see that we have access to the vehicle data and to the actuators via the primary and the secondary CAN networks, this variables will enrich the inputs of the enablers E.3.1, E.4.1 and E.5.1.

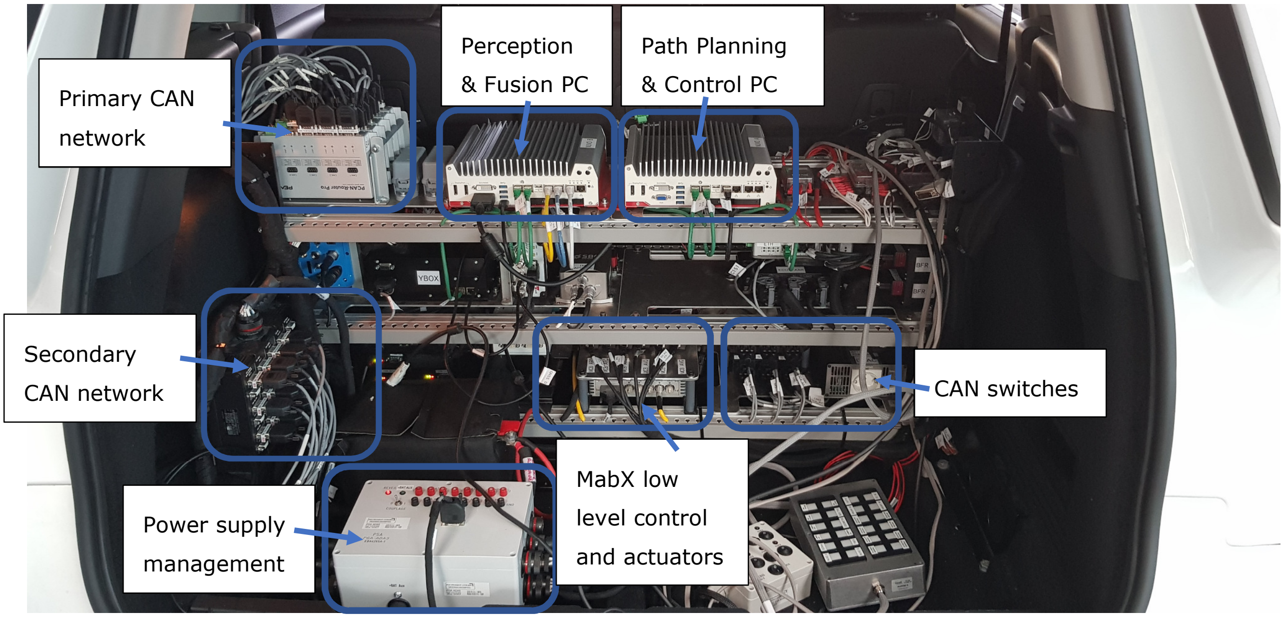

Figure 23: The car trunk and the hardware and electronic part of the vehicle

The enablers will be installed in the car as follows:

- The enablers E2.1, E3.1 and E4.2 and E5.1 will be embedded in the Perception and fusion PC.
- The enabler E4.1 will be installed in the path planning and control PC.
- Since the enabler E1.1 is a hardware sensor connected to a dedicated mini-PC which is used as a gateway to send the data to the Perception PC.
- The enabler 1.2 is already installed, the antenna is on the rough and the onbord unit is installed in the car trunk.

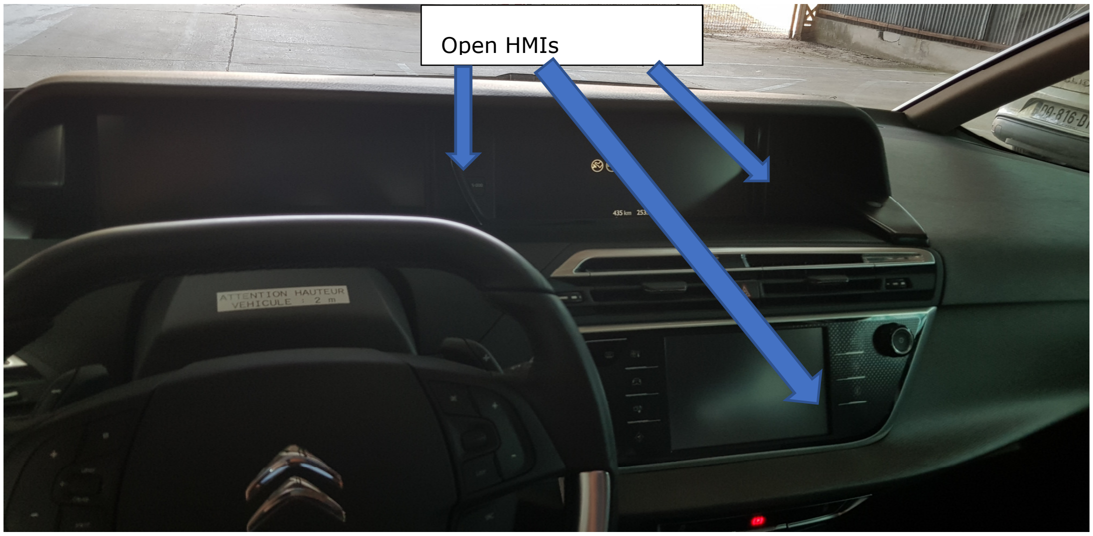

Figure 24: HMI emplacements

Since the car is also equipped with an open dashboard and cluster, there is possibility to install the HMI enablers (E.6.1, E6.2, E6.3 and E6.4) in the different surfaces, as shown in the figure**Errore. L'origine riferimento non è stata trovata.**.

8.4 Plan for 3rd cycle integration

In this section we details the plans for the third Cycle concerning the installation of the enablers in the VED demo car. An integration workshop took place in Paris in month 20, in this workshop the integration of the enablers has been discussed according to the hardware architecture of the VED CAR and of course by respecting the requirements and the common architecture of the TeamMate car. During this integration workshop, data format representing the inputs and the outputs of the enablers have been discussed and defined.

All exchanged information, mandated for the AutoMate open API definition have been discussed and validate by all partners.

Below is a remember of all data format specifications resulting of these discussions and already described in D5.1.

8.4.1 Map

The MAP will be only ONE road without intersection, defined by waypoints as follow:

Symbol	Variable	Format	Units
ID	An ID for the current record	**int**	$\in \mathbb{N}$
NbLanes	The number of available lanes	**int**	$\in [1, 2, 3]$
Right_LaneInfo	Information of Rightest Lane	**LaneInfo**	-
Middle_LaneInfo	Information of Center Lane	**LaneInfo**	-
Left_LaneInfo	Information of Leftest Lane	**LaneInfo**	-

Table 14: road structure definition.

and X_LaneInfo, which refers to to a specific Map is defined by the following variables:

Symbol	Variable	Format	Units	Comments
a	Availability	**bool**	**true**/**false**	Is the lane exist ?
{ x, y }'	Position	**float** (2x)	**m**	UTM Coordinates of lane center
hw	Helf Width	**float**	**m**	-
msl	Mandatory Speed Limit	**int**	**km.h-1**	-
{ rm, lm }	Lane Marking (right and left)	**int** (x2)	-	0: dashed; 1: continuous
h	Orientation/Heading (Optional)	**float**	-	from **North**, clockwise

Table 15: lane structure definition.

8.4.2 Ego Vehicle

At each time step, the ego-vehicle status contains all these information:

Symbol	Variables	Format	Units	Comments
t_ms	TimeStamp	**long**	**ms**	from 01/01/1970-00:00:00
{ x, y }'	Position of rear axle center	**float** (x2)	**m**	UTM Coordinates
{ σ_x, σ_y }'	Standard Deviation of Position	**float** (x2)	**m**	-
{ w, h }'	Dimension of the Vehicle	**float** (x2)	**m**	-
θ	Heading	**float**	**degree**	from **North**, clockwise
σ_h	Standard Deviation of Heading	**float**	**degree**	from **North**, clockwise
{ x', y', | $\vec{v}$ | }	Speed	**float** (x3)	**m.s-1**	-
{ x'', y'', | $\vec{a}$ | }	Acceleration	**float** (x3)	**m.s-2**	-
yaw	YawRate	**float**	**degree.s-1**	-
steering	Steering Angle	**float**	**degree**	-

Table 16: ego-vehicle structure definition.

8.4.3 Obstacles

Contrary to the previous information, the Obstacles is a dynamic format, with a specified pattern but that could has different size depending of the result of perception

<ID>, <TimeStamp>, <NbObject>, <Object1>, ..., <Objecti>, ..., <ObjectN>

Each message is composed by a header that includes : an unique identifier, the timestamp of the perception results and the number of detected objects. An object definition is defined in the following table.

Symbol	Variables	Format	Units	Comments
ID		**int**	-	-
{ x, y }'	Relative Position	**float** (x2)	**m**	in Vehicle Referential, center of object BBox
σ_{pos}	Relative Position Uncertainty	**float** (x4)	**m**	-
{ x', y' }'	Relative Speed	**float** (x2)	**m.s-1**	in Vehicle Referential (Relative Velocity from Vehicle)
σ_{vel}	Relative Speed Uncertainty	**float** (x4)	**m.s-1**	-
{ X, Y }'	Absolute Position	**float** (x2)	**m**	in World Referential, center of object BBox
σ_{POS}	Absolute Position Uncertainty	**float** (x4)	**m**	-
{ X', Y' }'	Absolute Speed	**float** (x2)	**m.s-1**	in World Referential (Absolute Velocity)
σ_{VEL}	Absolute Speed Uncertainty	**float** (x4)	-	-
yaw	Orientation	**float**	**degree**	in Vehicle Referential
σ_{yaw}	Orientation Standard Deviation	**float**	**degree**	-
{ w, h }	Dimensions	**float** (x2)	**m**	-
$\{\sigma_w, \sigma_h\}$	Dimensions Uncertainty	**float** (x4)	**m**	-
label	Classification	**int**	-	Same as IBEO
s_label	Classification Score	**float**	$\in [0,1]$	-
p_exist	Existence Probability	**float**	$\in [0,1]$	Not sure we can get it

Table 17: obstacles structure definition.

8.4.4 Trajectories

For the Enabler 5 « Online Risk Assessment », we also have to define the last kind of message requested for communication between enablers. As for the Obstacles definition, trajectories are dynamic information and the exhancge format must take into account this aspect. So, as for the Obstacles message definition, a header is requested in the message format to include the number of points defining the trajectory (**N**), with the global timestamp for realisation (**global_timestamp)**.

Symbol	Variables	Format	Units	Comments
N	Number of WayPoint	**long**	$\in \mathbb{N}$	-
global_timestamp	Global Expected TimeToReach	**long**	[T_ms, T_ms + δ]	expected time to reach the 1^{st} way point (incremental in the list of way point) since 1970
-	-	-	-	-
t_i	Expected TimeToReach	**long**	[0, ...]	expected time to reach the way point (incremental in the list of way point, and expected to start from 0, see global_timestamp)
$\{x_i, y_i\}$	Position	**float** (x2)	**m**	defined from rear axis-Center
$\{X_i, Y_i\}$	Global Position	**float** (x2)	**UTM**	defined in the World Referential (UTM Coordinates)

Table 18: trajectories structure definition.

All these definitions correspond to messages defined in the openAPI framework for AutoMATE (see Deliverable T5.1) and are used in the VED demonstrator as message format between the integrated enablers.

The openAPI provide a solution to easily use this formalism by providing a Google ProtoBuf definition of all messages, and the associated source code for different development languages.

The AutoMATE API also specify the transport layer, which rely on the standard DDS – Data Distribution Service – and is compliant with the open source objectives of an European project.

However the VED demonstrator relies on a third party application (RtMAPS) which already includes helpers to exchange information between modules. It provides solution for primitive arrays, strings and other kinds of simple messages…

To keep VED Demonstrator compliant with the AutoMATE API, a set of new messages, respecting the formalism of AutoMATE have been defined to guarantee the compatibility of the demonstrator with the AutoMATE framework, but which are not directly linked to the Google ProtoBuf interface provided in the AutoMATE OpenAPI.

Moreover, as the employed third party application also provides mechanisms to exchange data, the VED demonstrator does not rely on DDS but on a proprietary solution, which will not interfere with the philosophy of an AutoMATE demonstrator as all Enablers will be integrated in the global VED software solution.

If during the remaining time of AutoMATE, a requested is made to integrate a non-mandatory enabler in the demonstrator, as for other partner module, a by-pass solution could be defined (and discussed), or the DDS protocol could be integrated.

8.5 Summarizing next Steps

In this section, we show which additional enablers are planned to be integrated in the 3rd cycle:

ID	Enabler	Demonstrator
E1.1	Driver monitoring system with driver state model for distraction and drowsiness	Planned for end September 2018
E1.2	V2X communication	BIT will provide standardized DENM message containing roadworks related information regarding the ETSI standards[3].
E2.1	Driver intention recognition	Discussed the integration workshop in Paris (June 2018). Will be installed and verified in October 2018

[3] ETSI EN 302 637-3 (V1.2.1), ETSI TS 101 539-1 (V1.1.1)

E3.1	Situation and vehicle model	Discussed the integration workshop in Paris (June 2018). Will be installed and verified in October 2018
E4.1	Planning and execution of safe manoeuvre	Discussed the integration workshop in Paris (June 2018). Will be installed and verified in October 2018
E4.2	Learning of intention from the driver	Not yet, Will be installed and verified in October 2018
E5.1	Online risk assessment	Discussed the integration workshop in Paris (June 2018). Will be installed and verified in October 2018
E6.2 - 3 -4	TeamMate HMI (Cluster + audio, Central stack display, HUD)	An integration workshop will be scheduled for October 2018. And the integration in November

		2018. This task come after the installation of all low level enablers.
E6.5	Augmented reality	Will not be installed, is not compliant with VED demo-car.

Table 19: enablers to be integrated in the 3rd cycle in VED demonstrator car.

9 ULM Vehicle Demonstrator

In this section, the ULM vehicle demonstrator is described.

9.1 Enablers and system Architecture

List of enablers that have been integrated into the demonstrator:

ID	Enabler	Demonstrator
E3.1	Situation and vehicle model	YES
E4.1	Planning and execution of safe manoeuvre	YES
E6.1	Interaction modality	YES
E6.2 - 3 -4	TeamMate HMI (Cluster + audio, Central stack display, HUD)	YES

Table 20: list of Enablers for ULM demonstrator car

9.2 Results of Set-up Tests

Ulm has two computers integrated in the car. There is a perception and an application PC. The communication takes place by using the UDP common protocol. The sensor setup is connected to the vehicle via can bus, while the ADMA module uses Ethernet. The tests were successful and the communication is running already. To make sure that the communication works watchdog signals are used to detect timeouts.

9.3 Demonstrator

The Ulm demonstrator vehicle is based on a Mercedes-Benz E-class T-model and depicted in the following figure:

Figure 25: ULM demonstrator vehicle.

The demonstrator contains a two PCs, with Ubuntu 14.04 installed. The first one is the perception PC with an intel i7 processor processor, 32GB RAM and a Nvidida Gforce gtx1070 graphics card. The second one is the application PC with an intel i7 processor octacore processor, 32GB RAM and a Nvidida Gforce gtx1070 graphics card. Also a dspace microautobox is built in on which the security system, as well as the vehicle controller are running. The following figure shows the cars luggage space where the PCs are built in:

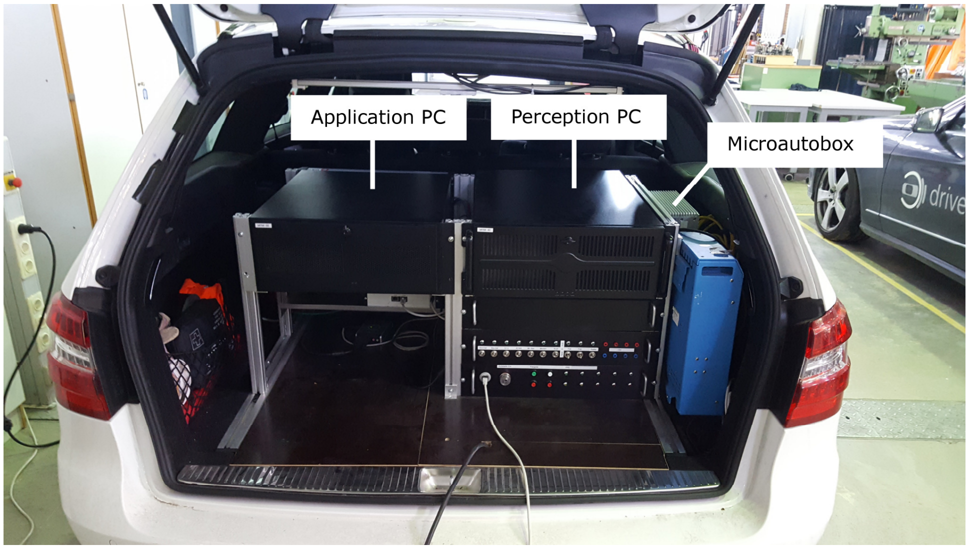

Figure 26: cars luggage space with the illustrated PCs and ECUs.

To perceive the environment, the demonstrator uses a Lidar, an long-range radar and 4 short range radars as to see in the following figure:

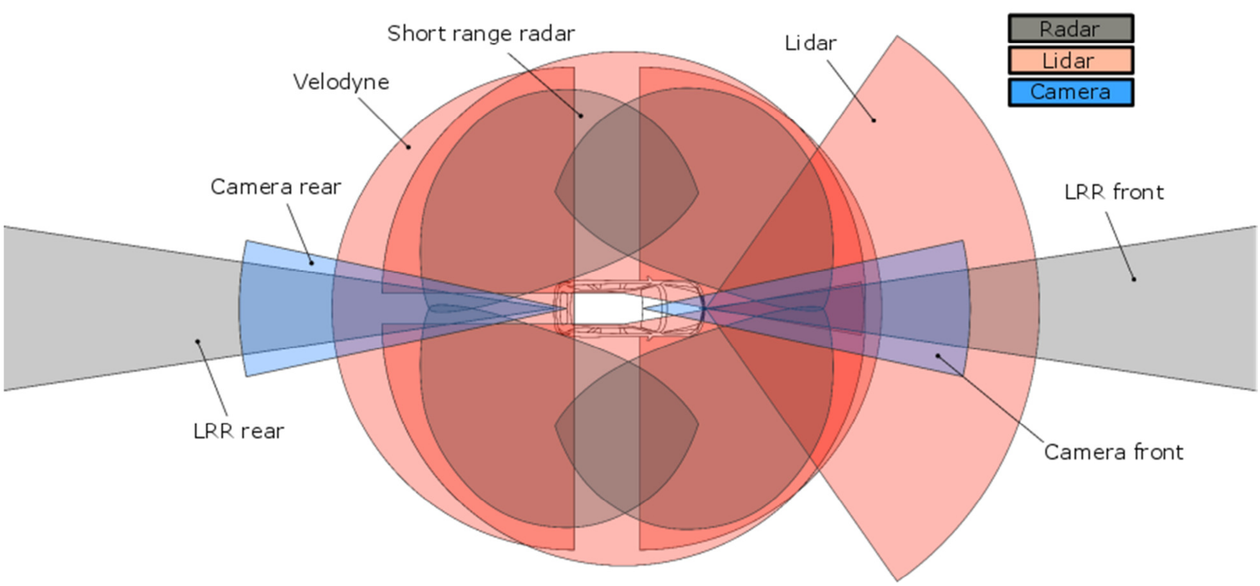

Figure 27: sensorial platform and related field of view.

The status of software component integration is as follows. The situation and vehicle model (E3.1). is already integrated in the demonstrator and provides information about the ego-motion state of the vehicle, as well as positions velocities etc. about other traffic participants. This enabler works directly on the pre-processed sensor data. To enable autonomous driving a planning and execution of safe manoeuvre (E4.1) has already been integrated. This enabler uses the situation and vehicle model, as well as some further information to plan trajectories guiding the vehicle safe through the environment. The following figure visualizes the planned architecture.

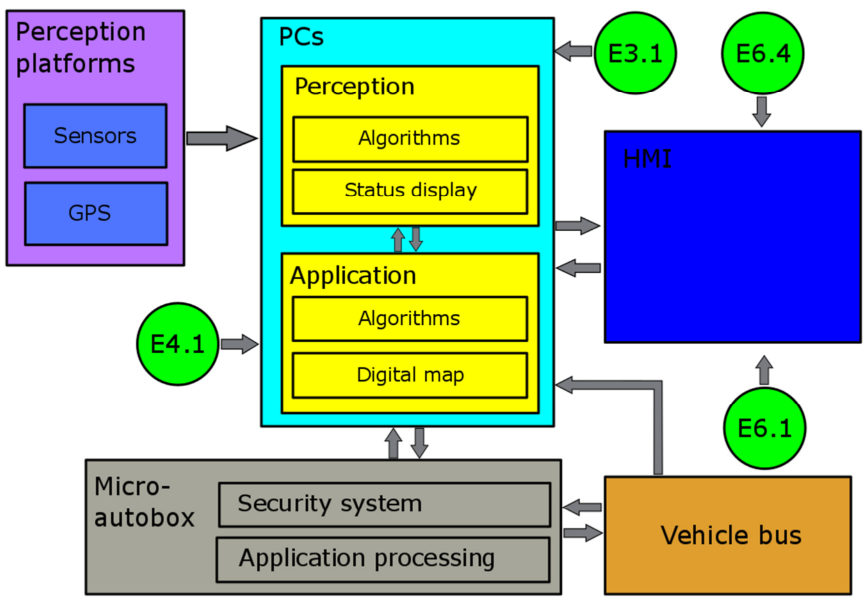

Figure 28: system architecture and deployment diagram in ULM demonstrator vehicle.

To enable permanent interaction with the driver, the teammate HMI (E6.5) will be integrated within the cars cockpit in the form of a HUD and audio system. Via this system, the driver is able to supply the vehicle with information and the automation can assist the driver.

9.4 Plan for 3rd cycle integration

No additional updates, see Table 20, in Section 9.1.

10 Conclusions

This deliverable has described the integration of the SW/HW components for the TeamMate technologies from WP2/3/4 into the six project demonstrators in the current project cycle 2, within the current system architecture (sketched in Chapter 3 and detailed in D5.1). By the way, input data from automation functions, from maps and from vehicle sensors are provided by the existing vehicle or simulator. Each enabler is represented by a software component dependent on its own concern. A message bus oriented data exchange between the components is implied to support a communication via one or more channels. The TeamMate system delivers its outputs via acoustic and visual human-machine interfaces to the driver.

All in all, these demonstrators are three driving simulators (from REL, VED and ULM partners) and three vehicles (from VED, ULM and CRF partners). In addition, this document has illustrated the set-up tests between the components and the subsystem, in order to ensure proper functionality before demonstrators are ready for the evaluation (in WP6). Together with these aspects, we have also presented the status of each demo and a plan for those enablers and components not yet integrated (but that will be done in the 3rd Cycle).

The integration of the different modules and sub-systems into the driving simulators has been almost completed, while we are still working on the integration into the vehicles (as planned in the DOW and detailed in the document). In particular, for the vehicles, the full integration in the 3rd cycle (M29-31) will be described in D5.6 “TeamMate Car Demonstrator after 3rd Cycle” (leaded by CRF).