Mediator

Development of integrated lab prototypes

Deliverable D2.9 – WP2 – Public

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 814735.

Development of integrated lab prototypes

Work package 2, Deliverable D2.9

Please refer to this report as follows:
Bakker, B., Knauss, A., Rodrigues de Campos, G., Tanov, N., Andersson, S., Mano, D., Athmer, C., Enhuber, S.M., Beggiato, M., Spaan, M. (2022). Development of integrated lab prototypes, Deliverable D2.9 of the H2020 project MEDIATOR.

Project details:	
Project start date: **Duration:** **Project name:**	01/05/2019 48 months MEDIATOR – MEdiating between Driver and Intelligent Automated Transport systems on Our Roads
Coordinator:	Prof. Dr. Nicole van Nes | SWOV – Institute for Road Safety Research Bezuidenhoutseweg 62, 2594 AW, The Hague, The Netherlands
	This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 814735.

Deliverable details:	
Version: **Dissemination level:** **Due date:** **Submission date:**	Final Public 30/11/2022 16/01/2023

Lead contractor for this deliverable:
Bram Bakker – Cygnify

Report Author(s):	Bakker, B. – Cygnify (CYG), The Netherlands Knauss A., Rodrigues de Campos, G., Tanov N. – Zenseact (ZA), Sweden Andersson S., Mano D. – Veoneer (VEO), Sweden Athmer, C., Spaan, M. – Delft University of Technology (TUD), The Netherlands Enhuber, S.M., Beggiato, M. – Chemnitz University of Technology (TUC), Germany

Revision history

Date	Version	Reviewer	Description
05/12/2022	Individual chapters	Bram Bakker – Cygnify, The Netherlands	Integration / editing
14/12/2022	Draft for QA	Diane Cleij – SWOV Institute for Road Safety, The Netherlands	Internal review
27/12/2022	Final draft	Ernst Verschragen (WP2 leader & QA officer) – SWOV Institute for Road Safety Research, The Netherlands	Final revisions and checks
16/01/2023	Final deliverable	Michiel Christoph (Technical project coordinator) – SWOV Institute for Road Safety Research, The Netherlands	

Legal Disclaimer

Table of contents

List of Figures

About MEDIATOR

MEDIATOR, a 4-year project led by SWOV, started on May 1, 2019. MEDIATOR will develop a mediating system for drivers in semi-automated and highly automated vehicles, resulting in safe, real-time switching between the human driver and automated system based on who is most fit to drive. MEDIATOR pursues a paradigm shift away from a view that prioritises either the driver or the automation, instead integrating the best of both.

Vision

Automated transport technology is developing rapidly for all transport modes, with huge safety potential. The transition to full automation, however, brings new risks, such as mode confusion, overreliance, reduced situational awareness and misuse. The driving task changes to a more supervisory role, reducing the task load and potentially leading to degraded human performance. Similarly, the automated system may not (yet) function in all situations. The objective of the mediator system is to intelligently assess the strengths and weaknesses of both the driver and the automation and mediate between them, while also taking into account the driving context.

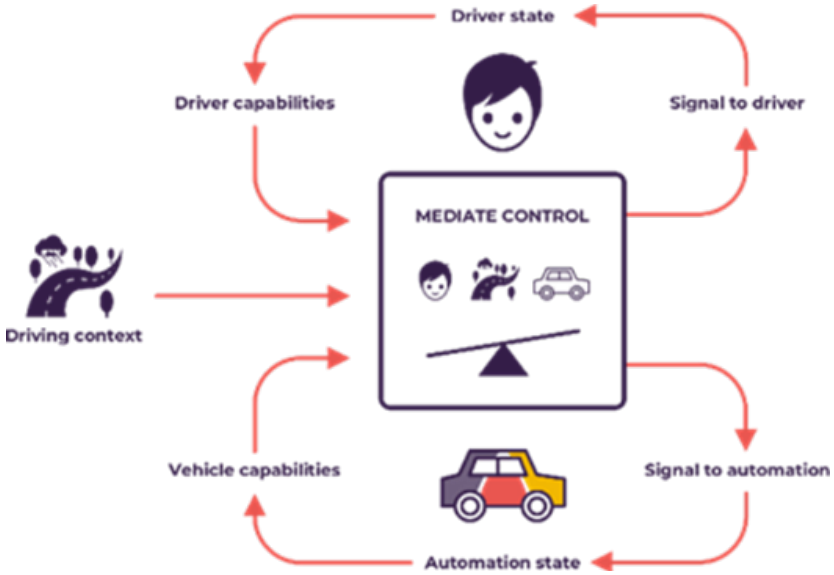

Figure 1 The MEDIATOR system will constantly weigh driving context, driver state and vehicle automation status, while personalising its technology to the drivers' general competence, characteristics, and preferences

MEDIATOR will optimise the safety potential of vehicle automation during the transition to full (level 5) automation. It will reduce risks, such as those caused by driver fatigue or inattention, or on the automation side imperfect automated driving technology. MEDIATOR will facilitate market exploitation by actively involving the automotive industry during the development process.

To accomplish the development of this support system MEDIATOR will integrate and enhance existing knowledge of human factors and Human Machine Interfaces (HMIs), taking advantage of the of expertise in other transport modes (aviation, rail and maritime). It will develop and adapt

available technologies for real-time data collection, storage and analysis and incorporate the latest artificial intelligence techniques, such as deep learning.

Partners

MEDIATOR is carried out by a consortium of highly qualified research and industry experts, representing a balanced mix of top universities and research organisations as well as several OEMs and suppliers. The consortium, supported by an international Industrial Advisory Board and a Scientific Advisory Board, will also represent all transport modes, maximising input from, and transferring results to, aviation, maritime and rail (with mode-specific adaptations).

Executive summary

The MEDIATOR project has built several prototype (demonstrator) systems that can mediate, in real time, between the automated functions of a vehicle and the driver/operator, ensuring that the one that is most fit for the task at hand is in control. The Mediator concept aims to reduce the risks related to the transition towards full automation, a phase that still relies on the human driver for taking over when the automation does not yet function at a sufficiently reliable level or not for all situations and in all circumstances.

The aim of the project has been to develop a Mediator prototype for SAE levels 2 - 4 (In Mediator terms: Continuous Mediation ~ L2 ; Standby ~ L3 ; Time to Sleep ~ L4), and have them tested in a number of relevant traffic scenarios. Each of these levels of automation provide different requirements on the system. Whereas SAE level 2 automation requires drivers to be 'in-the-loop' all the time, the higher SAE levels allow them to be 'out-the-loop' for shorter or longer periods of time. MEDIATOR has focused in particular on how the transitions between these levels should be managed both on a technical and on an HMI level, how the transitions of control between automated driving and manual driving should be managed in those SAE levels 2 – 4, and how the Mediator system can work in terms of obtaining the required situation awareness on driver, automation, and driving context, and how intelligent decisions can be made that make use of that situation awareness.

Aim and scope of this deliverable

The current report describes the design and development of the main Mediator integrated lab prototypes produced within MEDIATOR. They constitute a set of integrated computer lab systems, each consisting of multiple components (which, in turn, were individually developed in component development tasks) *working together*. The aim is to allow testing of their *joint* functionality, and demonstration of the feasibility of the core concepts when put together at the whole system level. Several different lab prototype versions were used, with variations between them depending on the specific use case, the stage of development, and the evaluation focus.

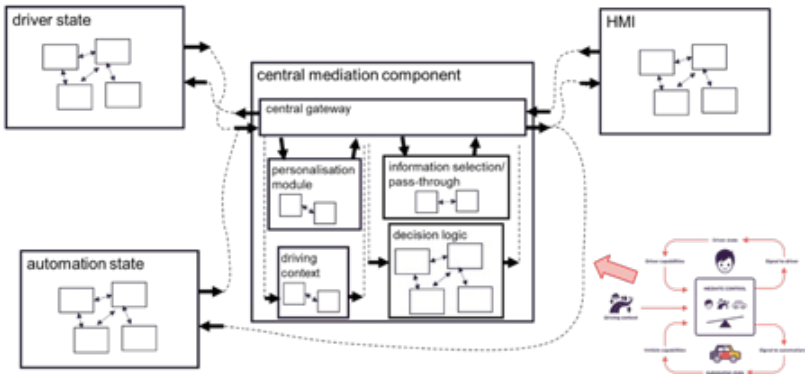

Figure 2 The focus of the work described in this report has been on the realisation and testing of the integrated system in a lab context, consisting of the main components working together (each designed and developed in individual component tasks), to allow testing of their joint functionality, and demonstration of the feasibility of the core concepts.

Note that the main part of this deliverable were the **actual, technical Demonstrator lab prototypes:** algorithms + software running, in simulation or real operation mode, in several computer lab hardware configurations, which were used in MEDIATOR Work Packages WP2 and WP3 testing, demonstration, and evaluation tasks. This document is an accompanying, on purpose relatively compact report, next to these technical deliverables—describing these Lab Demonstrators' goals, design choices, and functioning.

Contents of this report

This report describes:

- The conceptual approach used in developing and testing the main lab prototypes;
- Each of the components used in the lab prototypes;
- The hardware and software set-up of the lab prototypes (different versions were used, with variations depending on the specific use case, development step, and evaluation focus);
- The main scenarios and investigations that were performed with the lab prototypes, using recorded and/or real-time produced data;
- An overall summary of the work and some general conclusions.

1. Introduction

1.1. Objective and approach of the Integrated Lab Prototypes work

One of the two main objectives of the Integrated Lab Prototypes work, performed in one of the project's Design and Development Tasks, namely task T2.7, has been the integration of most of the main Mediator components into a real-time *simulated laboratory demonstrator Mediator system* (with a number of subvariants). The idea is to realise final development and demonstrations of the core Mediator components compiled into an integrated system, *in silico*, i.e. in a mostly computer-simulated lab setting. Put differently, the idea is that before the main components are put together and tested together on actual prototype vehicles (which was done in another task), they are put together and tested together in a partially or completely *virtual* environment, allowing integration testing that is easier and safer to do in such a virtual set-up than on real vehicles. This is a fairly standard approach in automotive complex system design, development, and testing.

Thus, to a significant extent the work of task T2.7 has been a necessary and effective *precursor* to the final vehicle prototype implementations done in another Mediator Task (see also Public Deliverable D2.10). This was done in particular in preparation for the Veoneer Mediator prototype vehicle, a.k.a. "Technical Integration" or *TI* vehicle prototype, which was the main vehicle prototype for the most software technology-advanced integrated implementation of a fully functional (i.e. not simulated) Mediator system; i.e. with working driver state, automation state, driving context, decision logic, and HMI put together, in a vehicle with actual automation capabilities, suitable for real-world driving.
(The second, "Human Factors" or *HF* vehicle prototype, led by partner FCA, in contrast, is more focused on testing interaction with naïve participants, using largely simulated, preprogrammed Mediator scenarios and interactions, but with real driver state measurement equipment and a real, complete and full Mediator Human Machine Interface. The realisation of that was another important development focus of Work Package 2, and the work and results of that are more fully described in deliverables Public Deliverables D2.10 and D2.11.)

The used approach in the MEDIATOR Project, and in task T2.7 in particular, involved the ability to either use the *real*, full hardware + software component that was developed, or a *simulated* version thereof, which often was based on real data collected earlier. For example, for much of the T2.7 lab prototype development and testing, "simulated" Automation State data was produced and used, by "replaying" real Automation State data that was logged earlier, in preliminary runs of that component in action (but running in the background) on real-world test drives. This allows testing of, e.g., the Decision Logic component in combination with realistic Automation State data. This is otherwise very difficult to simulate in a lab setting, because it would involve simulating realistically a more or less complete, partially automated vehicle. For the Driver State component, similarly, logged data from earlier test drives was used--but also, for some testing runs, the actual (camera-based) real-time Driver State system was used as obtained in the lab—to test certain aspects of real-time functioning and a degraded driver in combination with the other components.

A *second* important part of task T2.7 has been the development of new technology and software for the *driving simulators*, in particular the TUC driving simulator, where also several of the Mediator components come together in its own specific lab constellation. A separate chapter is

dedicated to this second part. (The BGU driving simulator, which had its own particular development focus and which was realised to a large extent within the context of the evaluation work package, MEDIATOR Work Package 3, is described in the public deliverable D3.3.)

The MEDIATOR testing and evaluation strategy in WP3 consequently took into account the specific features of the different WP3 evaluation methods (computer simulation study, driving simulator studies, on-road studies). Thus, on the one hand a complementary approach was followed, taking advantage of the specific methodological assets in combination with corresponding Use Cases and Mediator components/prototypes. On the other hand, comparability between all WP3 studies was maximised by using the same Use Cases, the same components (e.g. HMI) and, whenever possible, the same measurements such as questionnaires. These two basic principles were agreed in the consortium and lead to a specific organisation of the work on the lab prototypes: the work plan foresees different Mediator prototypes for the driving simulators and for the vehicles, according to the specifics of each evaluation platform. It allows for exploiting the maximum benefit of each evaluation platform/method. Thus, each study was planned by each study team with close connections to the WP2-teams working on the technical realisation of Mediator components and integrated prototypes. Dedicated “platform subteams” were formed with representatives from WP2 and WP3 to best align technical developments with experimental evaluation designs for each study.

In the lab prototype development task, no extensive, exhaustive accuracy evaluations were done; the focus was on testing as criteria for inclusion in the final (esp. vehicle) Mediator prototypes. Thus, in other words, the focus was very much on readying final integrated software and hardware prototypes. More extensive and exhaustive accuracy evaluation is supplied in public MEDIATOR deliverables D3.3 and D3.4, as part of MEDIATOR’s evaluation tasks. Two types of tests were performed, integration and accuracy testing:

- Integration testing focused mostly on the software implementation level. For integration testing a "test passed" outcome was required, indicating that the software test passed the tests evaluating correct joint functioning based on inputs injected, messages being correctly and timely, etc.
- Accuracy testing focused mostly on the logical, algorithmic level. For accuracy testing, calculated values were compared with ground truth values determined during data collection, to determine sufficient quality algorithm functioning to pass the criteria for inclusion in the final (vehicle and driving simulator) Mediator prototypes. More extensive and exhaustive accuracy evaluation is supplied in public MEDIATOR deliverables D3.3 and D3.4, as part of MEDIATOR’s evaluation tasks.

In the original MEDIATOR plans, the plan was for initial results from Evaluation Work Package 3 to feed back into the design and development work, resulting in one update to the (lab and vehicle) Mediator prototypes. In practice, however, in part due to some of the delays incurred during the project, and in part due to some of the final technical development and the prototype evaluations occuring partly simultaneously and in parallel, this feedback loop was much more *continuous* and *incremental*. That is, early data collection and evaluation rounds fed back into the development process, impacting the final development process in a type of “agile” development process, with incremental improvements being done along the way.

1.2. Description of this document

Note that the main products of task T2.7 were the **actual Demonstrator lab prototypes:** algorithms + software running, in simulation or real operation mode, in a variety of computer lab

hardware configurations. This document is an accompanying, compact report describing those lab Demonstrators' goals, design choices, and functioning.

The remainder of this document is structured as follows:

- Chapter 2 describes the main lab prototypes developed in preparation for the Mediator TI prototype vehicle. This includes sections on:
    a. the conceptual approach used in developing and testing;
    b. each of the components used in these virtual lab prototypes;
    c. the hardware and software set-up of these lab prototypes;
    d. the main scenarios and investigations that were performed with these lab prototypes, using recorded and/or real-time produced data.
- Chapter 3 describes the lab prototype developed in preparation for the TUC driver simulator work. This includes (similarly) sections on:
    a. the conceptual background and approach used in developing and testing;
    b. each of the components used in this driver simulator-based prototype;
    c. the hardware and software set-up;
    d. the incremental integration and testing procedure.
- Chapter 4, finally, presents a brief overall summary of the work, some general observations, and the next steps following the development of these lab prototypes.

2. Lab prototype in preparation for TI vehicle integration

2.1. Core focus and approach

Whereas the focus in several Mediator component development tasks was on development and testing of *individual* components, in lab integration task T2.7 the components come together, providing the opportunity to test and tweak their combined functioning (see Figure 3). In the virtual, lab setting of T2.7, such testing and tweaking is much easier to do than directly (and only) using implementation of all systems into the real prototype vehicles. Thus, to a significant extent the work of T2.7 has been a necessary and effective *precursor* to the final vehicle prototype implementations, in particular the TI vehicle prototype.

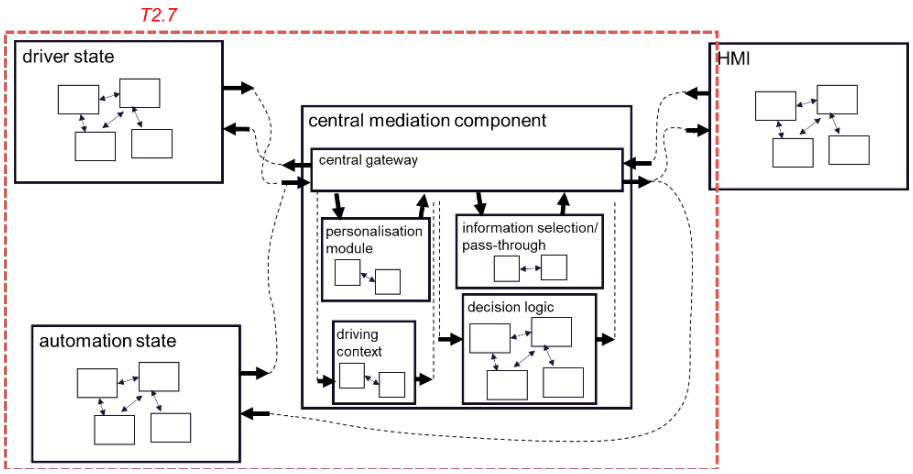

Figure 3 The focus of T2.7 has been on the realisation and testing of the integrated system, consisting of the main components working together, to allow testing of their joint functionality and demonstration of the feasibility of the core concepts.

The core Mediator concepts that these T2.7 lab prototypes focus on comprise the (real, not simulated) central mediating system, including the core component Decision Logic simultaneously monitoring automation, the driver, and driving context in real time (see Figure 3); making informed, proactive decisions about switching between driver and automation, correcting the driver's degraded performance when necessary and possible, and providing the driver with relevant information such as available time budgets for automation modes and other information designed to provide useful transparency about decisions and automation.

I.e., this demonstrator has a focus in particular on these technological core concepts of the MEDIATOR Project. Because of this, the focus in this particular lab prototype has *not* been on the sophisticated HMI that has also been developed in the project. The HMI has been of focus in the TUC prototype (see Chapter 3, and deliverables D2.10 and D2.11), as its specific audio-visual and tactile outputs and implementation are not affecting the main Mediator technological core concepts (explained above) that we focus on in T2.7.

This focus is in line with the focus of the technical Mediator concepts-oriented "Technical Integration" or *TI* prototype vehicle of Mediator task T2.8. This work can, therefore, be viewed in part as preparation for that TI prototype vehicle T2.8 work, where the same components work together for real and in real time (i.e. in a real-world, non-simulated setting).

2.2. Flexibly and incrementally combining real and simulated components

The way that this virtual lab approach of T2.7 is realised is by mixing and matching *real*, real-time operating sensor and software components (coming from Mediator component development tasks T2.2 through T2.4) with *simulated* replacements. Those simulated replacements are realised, in particular, by *replaying*, using a custom data replaying system, *logged* data from the actual (temporarily replaced) components. That logged data was typically obtained in preliminary experiments with the prototype vehicle(s), in particular the TI vehicle.

E.g. in preliminary experiments, the Automation State component (see later sections for further explanation) ran '*in the background'* in the TI prototype vehicle, recording data and giving data outputs as if it were used, but not actually influencing any real decision logic or HMI, and all its outputs were logged together with the timestamps. In the T2.7 lab experimentation, this logged Automation State was replayed using the data replayer, thus mimicking the Automation State component's behavior as though it happens in real time. Similar logging and replaying was done with the Driver State data and the Driving Context context data using data collected in early data recording sessions; thus allowing realistic replaying of all three major input streams for Central Mediation and its constituent Decision Logic component.

For the Driver State component, besides using similarly logged data from earlier real-world test drives, for some lab testing also the actual (camera-based) real-time Driver State system was used, to test its real-time functioning in combination with the other components (see later sections for examples).

Another important and useful approach that was used was *incremental integration and testing*. This means that developing and testing of integrated component functioning was done by adding individual main components one by one; which has the important advantage that not all complexity is addressed all at once but only step by step

2.3. Focus of the lab prototype components

As explained above, the focus of this subtask within T2.7 is in line with the focus of the technical Mediator concepts-oriented TI prototype vehicle of task T2.8, and corresponds to the focus depicted in Figure 3. This means that particular attention is given to the components for Automation State, Driving Context, and Decision Logic; with a (in terms of hardware) relatively simplified Driver State component, and with the HMI modelled as a simple concept-level software component mimicking the API.

2.4. Automation State component

The purpose of the Automation State component is to estimate the current and near future automation performance. The core idea of the automation state component is a set of algorithms which estimate the current and near future automation *fitness score*, a quantification of how good

the automation performance is. The estimated fitness scores, for which “optimistic”, “pragmatic”, and “pessimistic” values are computed, are then compared to cut-off threshold values which allows the component to determine if the automation is fit or unfit to drive.

Automation fitness is, in the final steps of the component, operationalised in terms of *Time To Automation Fitness (TTAF) and Time To Automation Unfitness (TTAU),* in accordance with the overall Mediator operationalisations and APIs, which were designed in earlier stages of the project. Time To Automation Fitness and Time To Automation Unfitness can be understood as representing estimated “time budgets” until the underlying Automation system will likely become available, or unavailable, respectively (see deliverable D2.11 for more information). Figure 4 provides an overview of the core inputs and outputs, as well as a schematic representation of internal inputs and outputs and subcomponents and messaging.

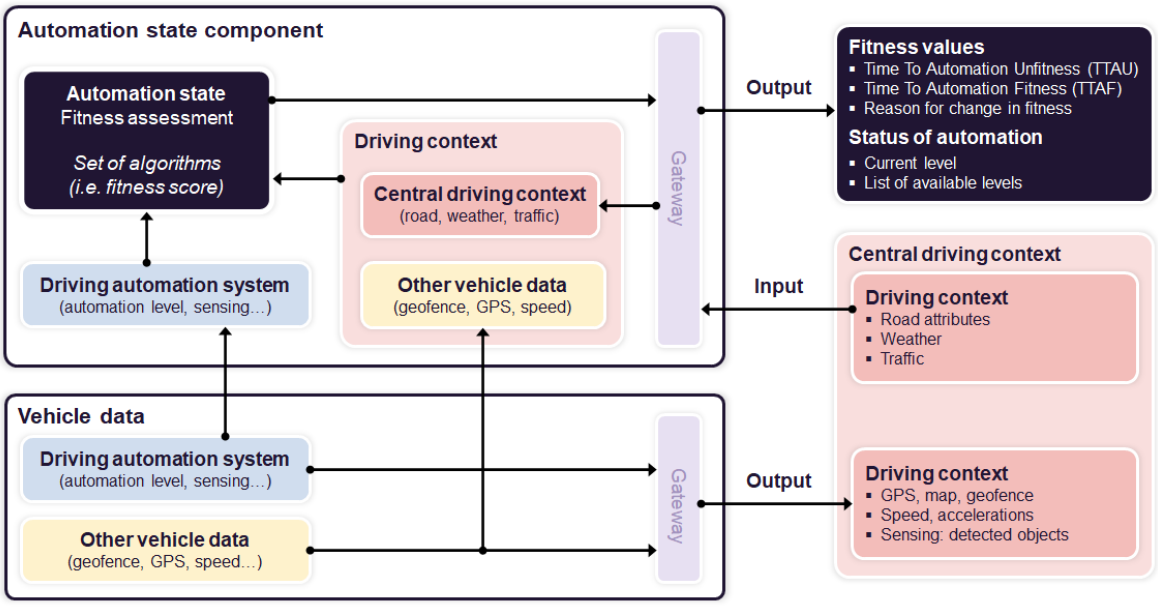

Figure 4 Automation state component design including its inputs and outputs.

The Automation State component used and tested in this lab prototype is, essentially, the real and complete Automation State component developed within the context of Mediator tasks T2.3 and T2.8, with its focus on the Level 2 Pilot Assist/Continuous Mediation (CM) automation system in the T2.8 TI prototype vehicle. However, because for real and real-time operation it requires inputs from real Driving Context data (obtained during driving) and real vehicle automation data (also obtained only during actual driving), for this lab prototype mostly *logged and replayed* data from the component operating in early prototype versions of the TI vehicle was used; as explained (as one of the core approach principles) in section 2.2.

2.5. Driver State component

The Driver State component in this lab prototype is, in terms of total sensing *hardware*, *not* the most complete and most sophisticated, in particular because it lacks Autoliv’s physiological sensors (aimed esp. at fatigue detection), which are on the other hand present in T2.8’s HF vehicle (see deliverables D2.10 and D2.11). However, in terms of real-time *software* the Driver State component in this lab prototype is relatively sophisticated, and essentially equal to what is used in

the VEO vehicle prototype. Similar to the HF vehicle, a multi-camera system (from Cygnify) is used. In the HF vehicle, that system is configured only for recording, with the objective of full but post-hoc analysis of the recorded video data (combining it also with the physiological data). In this lab prototype, as in T2.8's TI prototype vehicle (and in contrast to the HF vehicle prototype), this multi-camera system is configured for real-time use. The focus is on detecting *driver distraction* in multiple ways (thus leaving *fatigue and drowsiness*, another main degraded driver category investigated in the MEDIATOR Project, to the HF vehicle).

In terms of hardware (computers and cameras), this system was installed in the TI vehicle in Sweden in September 2021, which was relatively early when compared to when the final integrated system was completed and full evaluation was done with test drivers. Thus, since early on, many test drives were done with the Driver State system in initially only *recording* mode; allowing the type of recording and post-hoc data log replaying which is so valuable for the development, tuning, and testing of the integrated lab prototypes (as described in section 2.2).

Figure 5 visualises this lab prototype's (and T2.8 TI prototype's) camera and computer set-up. Two computers are used and connected by USB3 cables and connectors to four (high-end automotive grade, from camera manufacturer FLIR) cameras installed in the (simulated or real) vehicle cabin:

1) Monochrome (normal light *and* Near Infrared or NIR sensitive) driver **face view** camera. For the multiple goals of facial feature extraction for emotion/discomfort and fatigue estimation; gaze estimation of where the driver is looking; outside and inside the cabin, and Eyes-Off-Road/NDRT gaze analysis).
2) Driver upper **body view** camera (important for driver activity recognition, for distraction analysis);
3) Outward-looking (**forward view**) camera (for both driving simulator and in particular for the real-world vehicle driving);
4) Over the shoulder (dashboard/**cabin view**) camera mounted on the internal roof: seeing the whole dashboard/HMI that the driver sees including mirrors. (Used for post-hoc analysis and visualization of how the driver interacted with the HMI and where his gaze was.)

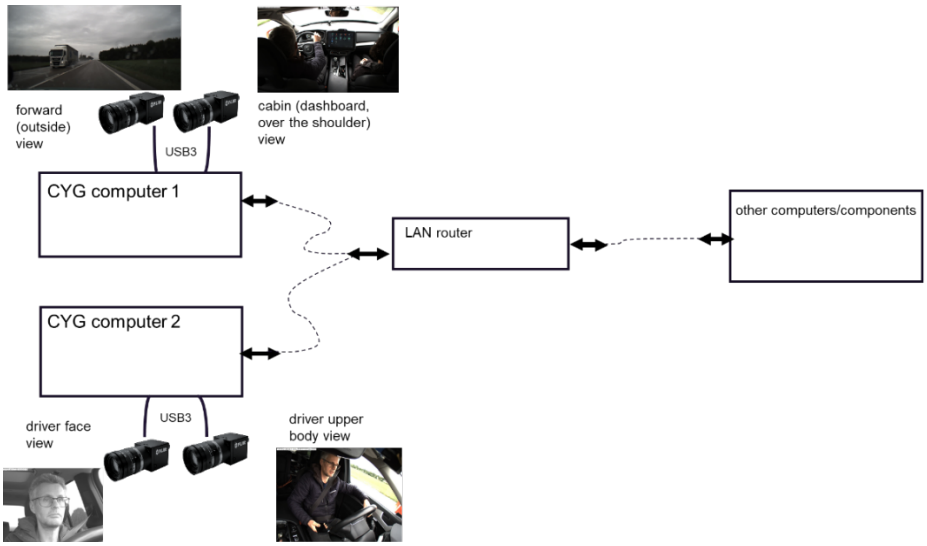

Figure 5 The Driver State component's cameras and computers set-up.

Figure 6 shows the software and messaging design for the complete real-time distraction detection-oriented Driver State system. The subcomponent modules in this Driver State component communicate between themselves using 'private' real-time messaging. The overall Driver State component communicates to the other main components (notably: Decision Logic and Driving Context) using 'public' real-time messaging, as defined by the main API definitions (described in detail in deliverable D2.11).

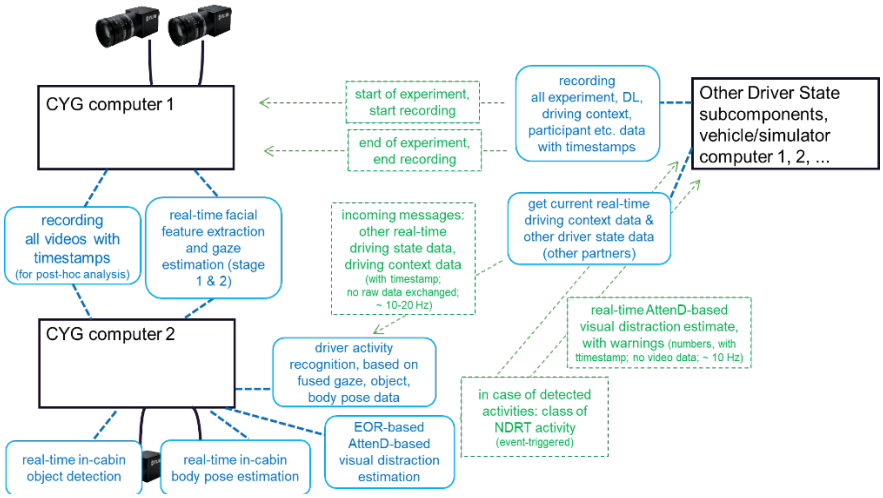

Figure 6 Software set-up and main messaging for the real-time distraction-oriented Driver State system.

Similarly to the Automation State system, driver fitness is operationalised by means of Time To Driver Fitness and Time To Driver Unfitness (TTDF and TTDU), values which can be understood as representing estimated "time budgets" of "driver availability". In other words, these values represent a measure of current and near-future predicted driver fitness by estimating whether the driver is, or would be if needed, available now to drive the vehicle; and if so, for how long (in the order of seconds to many minutes or even hours). And if *not*, how much time (estimated, in the order of a fraction of a second to some number of seconds) it would take to make him/her available again in the near future; given what we know about how long it would typically take the driver to be brought back into the loop by suitable HMI signals from the particular degraded driver state that he/she is in (e.g. a distracted state resulting from reading and/or writing a text message on his/her phone).

For this prototype, like the Automation State component, and as described before, many tests used *logged and replayed* data from the Driver State component that had been operating in the background in the TI vehicle, rather than real-time produced data. (I.e. this was data logged simultaneously for the same, real driving contexts, for both Automation State and Driver State, and also for Decision Logic.) However, in contrast to the Automation State component, for the Driver State component, for some testing also the actual, (camera-based) real-time Driver State system was used (cameras, computers, software), to test its real-time functioning in combination with the other components; since this could not be easily and safely done in the car itself (especially for severe degradation, such as severe distraction with very long glances off-road).

2.6. Driving Context

The Driving Context (DC) component is one of the central Mediation components (see Figure 2), in the final Mediator architecture design. Its role is largely "*supportive*": providing relevant information

on the current route, road, and driving conditions to other components, including Driver State, Automation State, and Decision Logic.

A core ingredient of the DC component is a tabular representation of the (planned or expected) vehicle route, which can be understood as a simplified high-definition (HD) map representation of the route. In Mediator tasks T2.7 and T2.8 and T3.4, the route is known in advance and therefore could be pre-programmed. But it could and would, in future real-world application of Mediator-like systems, come from the vehicle's navigation system; or be estimated online based on current vehicle position and direction and historical drive patterns for the specific driver.

Figure 7 visualizes schematically the Driving Context (DC) component's functioning during driving (*simulated* driving, in this lab prototype), including its main inputs and outputs. The core tabular route information is loaded at start-up and forms the basis for real-time processing. During driving, basic vehicle data comes from the vehicle systems, incl. GPS position, speed, steering wheel, gear, and pedals behavior, etc. In addition, advanced sensor data comes from the vehicle systems: detected objects around the vehicle, weather data & traffic data (sourced externally, detected road geometry, etc. This dynamic data is used to update DC's internal tabular route data structure; and when a significant enough update is done, this update is also sent outwards to other components for which this is relevant.

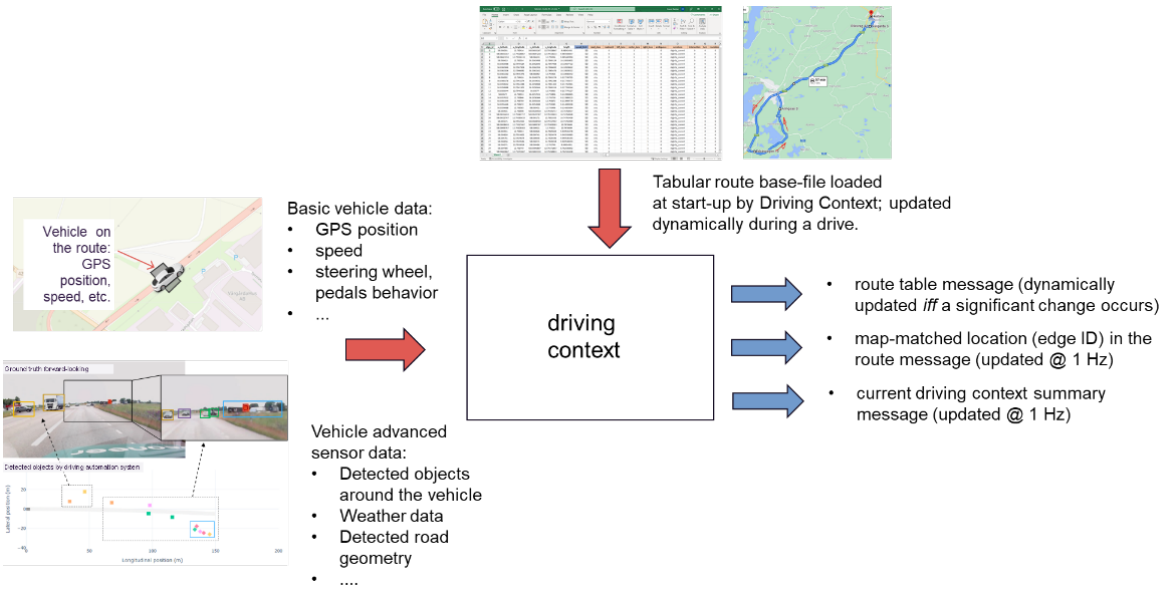

Figure 7 Driving Context Module main inputs (red) and outputs (blue).

For this lab prototype, as with the other components described above, the tests mostly used *logged and replayed* data from an early version prototype Driving Context component operating in the TI vehicle. Other tests used a lab version of the actual real-time Driving Context component—but fed by logged raw vehicle data (such as GPS location and speed). In this way, real-time functioning could be tested, mimicking real-world and real-time behavior in the vehicle.

2.7. Decision Logic

The roles of central Decision Logic (DL) are to:

- integrate the information from Driver State, Automation State, and Driving Context,

- recommend switches between the various available automation levels (with different levels of urgency),
- trigger corrective actions to attempt 'fixing' a degraded driver (based on detection of distraction or fatigue or discomfort);
- enforce emergency actions, such as emergency stops, if needed and possible given the (real or simulated) vehicle;
- provide additional supportive information related to this role to the HMI.

In this lab prototype, all functionalities except the emergency actions are included in the DL component. Emergency actions are excluded because in the follow-up work in the TI prototype vehicle (as part of task T2.8, described in deliverable D2.10, and as evaluated in task T3.4, described in a future deliverable report), emergency actions are not 'relevant' and not available; as that (SAE level 2) TI vehicle does not provide, and allow, proper emergency actions such as fully automatic emergency stops. Furthermore, the final trials are done with "professional" (Veoneer/Zenseact employee) drivers, instructed and trained not to exhibit the type of severe degraded driver behavior that could trigger emergency actions. The DL actions are selected by variations of the decision tree algorithms developed in Mediator task T2.4.

The "Information Selection/Pass-through" subcomponent (see Figure 8), which is technically a separate subcomponent but can be viewed as part of DL in the wider sense, "aides" in the role of DL by providing additional supportive information, especially related to the continuously, frequently updated "time budget" and driving context information, to the HMI.

Figure 8 shows once again the overall Mediator component diagram, with Decision Logic and its partner-subcomponent Information Selection/Pass-through highlighted (in red) and their main "interfaces". The figure shows the main information flows passing to and from them; and in fact the main information flows in the integrated lab prototypes in general.

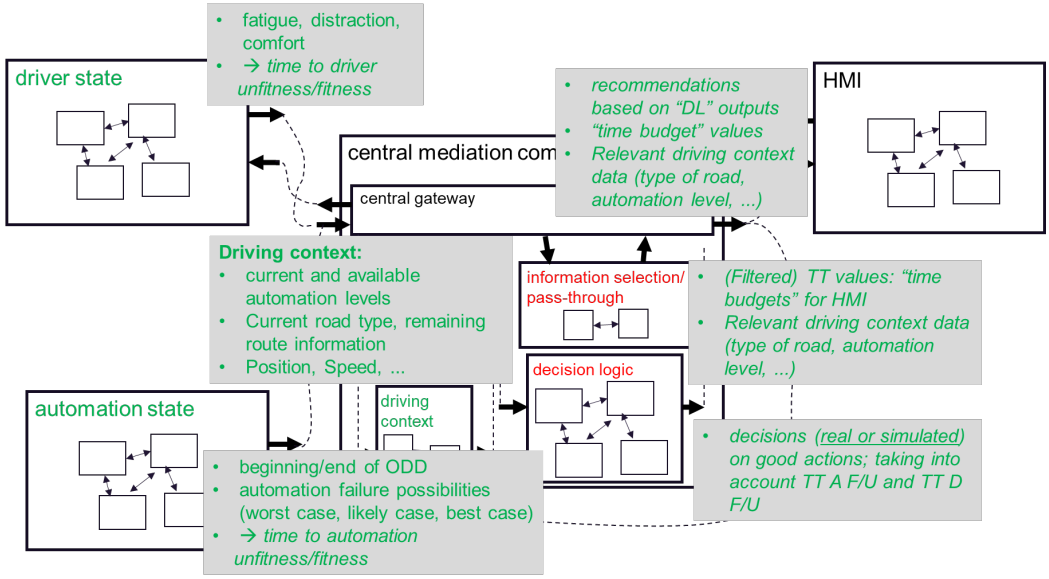

Figure 8 Main information flows passing to and from Decision Logic and its partner-subcomponent Information Selection/Pass-through.

Similar to what happened for some of the Driving Context component lab tests, in this lab prototype some Decision Logic tests used a lab version of the real Decision Logic component actually

running in real-time (i.e. not simulated, 'faked'), fed by *logged data* from the Automation State component, Driver State component, and Driving Context component. In this way, real-time functioning can be tested, mimicking real-world and real-time behavior in the vehicle.

2.8. HMI

In this prototype the HMI component has only been modelled as a simple *abstract, concept-level* software component, i.e. not performing any actual HMI procedures (visuals, sounds, etc.) but only *mimicking* the real HMI's reception of Decision Logic's messages and the HMI's reciprocal messaging towards Decision Logic. In other words, this setup was used to finalise the development and testing of the *API* (Application Programming Interface) interaction of Decision Logic and HMI, in order to test that software logic; without including or testing visual, auditory, or other HMI elements.

DL is the component that integrates information from Automation State (ASM), Driver State (DS), and Driving Context (DC), and that takes care of all interaction with the HMI component. I.e., the DL - HMI interaction API provides the crucial link between the main Mediator overall "logic" and its communication to and interaction with the driver through the HMI. The DL – HMI interaction API was initially conceptually modelled (before final software implementation) by means of *sequence diagrams* that describe the flow of information and the interaction. Subsequently, a simple "*stub*" API component receiving and sending HMI-related messages was used to develop and test the software interface (the API), using the same protobuf messages used in the real DL-HMI API in the final TI vehicle prototype. This interaction is not a 'one-way street' from DL to HMI, but rather an *interactive* process in which HMI informs DL about where it is in its procedure, and DL can interrupt and if needed escalate HMI procedures. More information on the precise interaction and messaging is provided in deliverable D2.11.

2.9. Hardware set-up

This "lab prototype" was not physically located in only one place; instead, different versions were used at different main partner locations (different lab locations)..The different versions were similar in terms of hardware set-up: essentially each corresponds to a set-up of one or typically a few computers, connected in a standard computer LAN network, and optionally with added sensors (as well as auxiliary hardware such as screens for easy visualisation).

The different computers were each responsible for running one or more of the main hardware/software components; and/or they contain data files for replaying logged data, together with custom data replaying applications (see section 2.2 which explains the concepts behind and reasons for this data replaying approach). Depending on the specific investigated scenario, the set-up allowed for replayed data (providing a realistic simulation), or real sensors (in particular cameras), and/or real software components running on the computer(s), making use of either replayed or real-time generated input/sensor data.

Figure 9 and Figure 10 show one such lab prototype hardware set-up, used by, and at, one partner location. Two computers were used in this set-up (which later went into the TI vehicle for T2.8/3.4 after T2.7, as described in deliverable D2.10), as well as four cameras, mirroring the four cameras (of the same type as installed in the TI vehicle) described for Driver State in section 2.5 (including the NIR illuminators). This set-up was used both for purely replayed data including replayed Driver State data, and for real and real-time cameras-based Driver State component tests (shown in Figure 10).

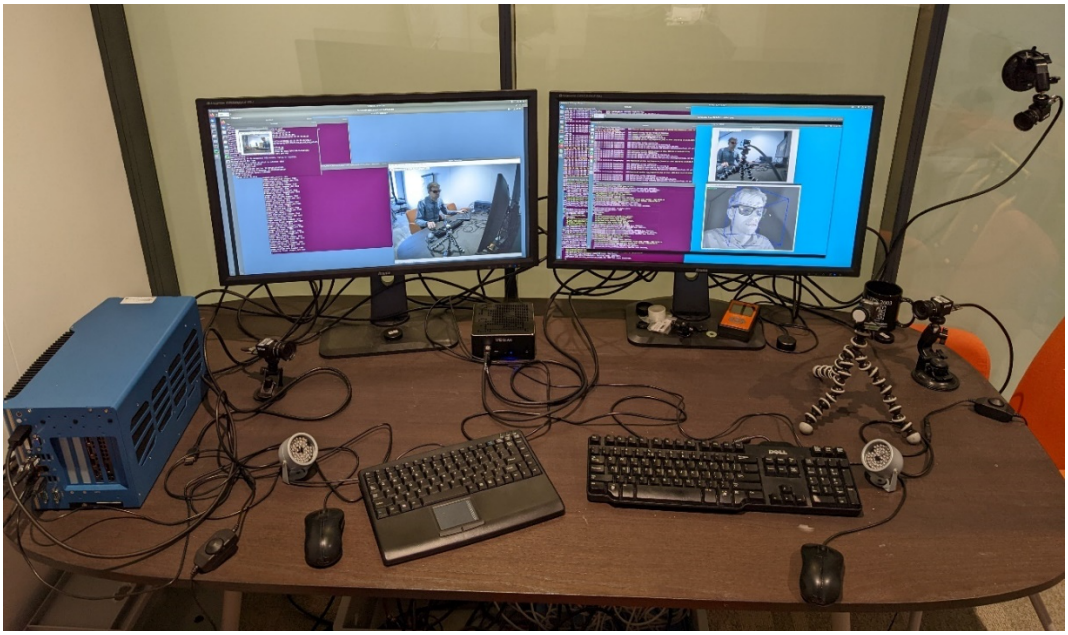

Figure 9 One specific lab prototype hardware set-up used by and a one partner location; showing computers, cameras, NIR illuminators, and auxiliary hardware such as screens and keyboards.

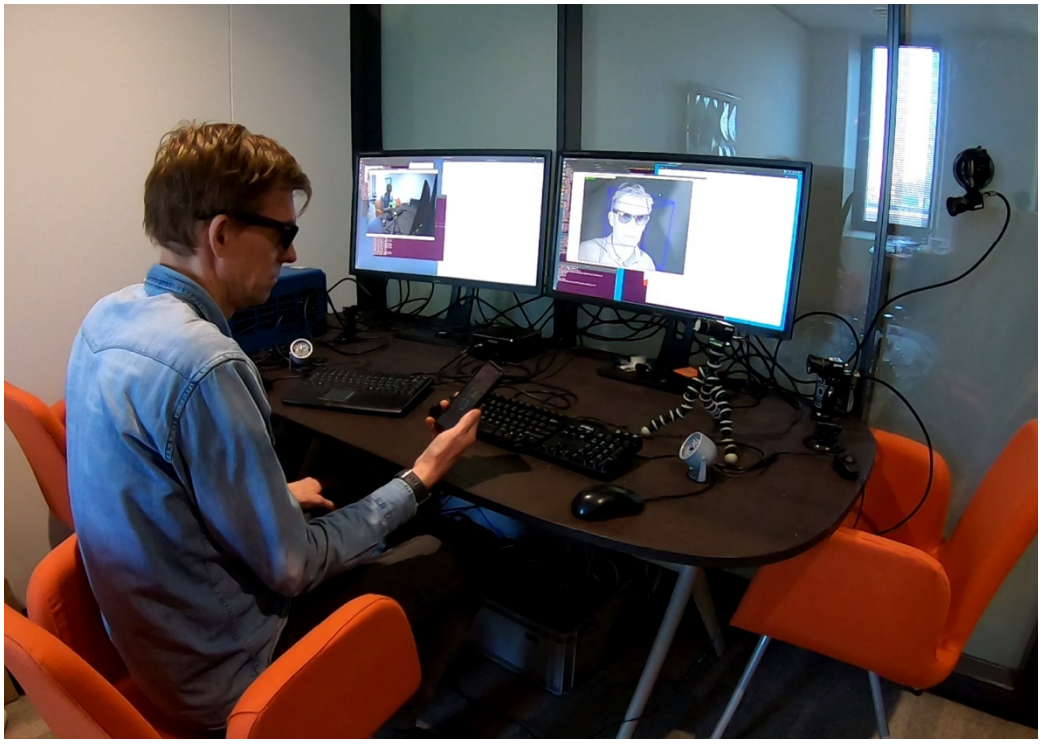

Figure 10 Lab prototype hardware set-up during a real-time cameras-based Driver State component test. In this example, the "driver" wears sunglasses, and the NIR-sensitive camera set-up supported by NIR illuminators set-up which is able to look through sunglasses is tested in conjunction with other components.

2.10. Software communication set-up

The software components used in this document were described in the previous sections. As described before, depending on the investigated scenario a real component could be replaced by a *simulated* component, by *replaying* data logged earlier. This primarily used logged data collected during many TI vehicle test drives performed in the course of tasks T2.2 through T2.7, in 2021 and early 2022.

Exchange of the technical messages realising this replaying and real-time communication between Mediator components was done in the following way, which took into account certain constraints resulting from the prototypes used and the history of the project. The TI vehicle, upon entering the MEDIATOR Project in 2021 (Veoneer being a partner coming aboard a bit later than 'founding' partners), came equipped with an existing extensive real-time messages-based communication and logging technical framework based on MQTT (see https://mqtt.org/). The MEDIATOR Project in general, however, had already decided on (in 2020), and been working with, *another* real-time messages-based communication and logging technical framework, based on ZeroMQ (https://zeromq.org/); a framework which is somewhat similar and comparable, yet also different in the details.

Rather than forcing one or the other set of systems and components to 'switch' to the other framework, it was decided to use both in parallel. Thus, both MQTT-based logging, messaging, and replaying *and* ZeroMQ-based logging, messaging, and replaying are used in this lab demonstrator (and remained so in the remainder of tasks T2.8 and T3.4). Some components 'live' primarily on the MQTT side; others primarily on the ZeroMQ side. A custom *MQTT-ZMQ Interface application* was developed (by partner Cygnify) which provides real-time interfacing between the two sides; such that components on one side (and replayed data) can communicate freely with any component on the other side. This has been made relatively 'easy' by the fact that both Veoneer's MQTT framework and the general Mediator ZeroMQ framework had been using similar "message definitions and formats" based on the widely used, standard Google Protocol Buffer message format and message serialisation.

Figure 11 visualises once again the Mediator system, showing in this figure this set-up of using both MQTT and ZeroMQ (a.k.a. 0MQ), where some components live mainly on the MQTT side, and others on the ZeroMQ side. In the figure, they are separated by the blue dashed line. Communication between the two sides is realized by the custom-built MQTT-ZMQ Interface application, also shown in the figure.

Furthermore, the lab prototype used a variety of analysis and visualisation methods in its tests: from simple console (terminal) windows using printed log textual outputs (e.g. see Figure 14) to visualisations of map-matched road information, sensed road geometry information and forward video data (Figure 12) to real-time visualizations of the functioning of the camera-based Driver State system (Figure 16).

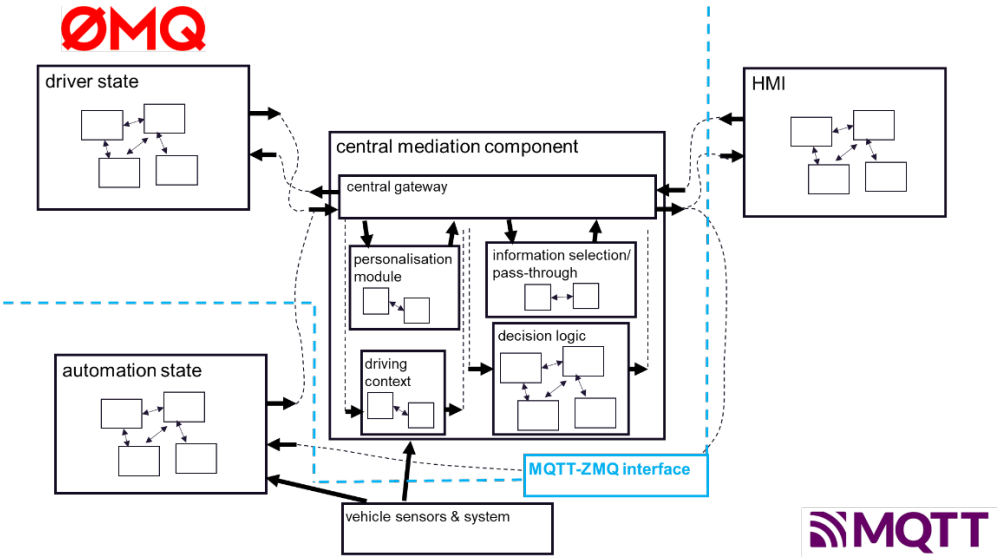

Figure 11 The Mediator system of the lab prototypes using both MQTT and ZMQ as communication and logging framework. Some components 'live' primarily on the MQTT side; others primarily on the ZeroMQ side. A custom MQTT-ZMQ Interface application was developed which provides real-time interfacing between the two sides.

2.11. Incremental integration and testing procedure

A key approach in developing and testing this integrated lab prototype has been to do *incremental* integration and testing. This means that developing and testing of integrated component functioning was done (to a large extent) by adding individual main components (each of which can be conceived as a building block) one by one. This has the important advantage that not all complexity is addressed all at once. Instead, additional complexity was only introduced and tested incrementally.

The tests were very diverse in nature, in terms of type of test and the test-passed criteria. The focus in these tests was on step-wise software system integration in preparation for the real-world prototypes (in particular the TI vehicle of T2.8 and T3.4) and on some (but not exhaustive) component-level accuracy and robustness tests, leaving extensive Mediator component and system-level evaluation to WP3.

Two types of tests were performed, integration and accuracy testing:

- Integration testing focused mostly on the software implementation level. For integration testing a "test passed" outcome was required, indicating that the software test passed the tests evaluating correct joint functioning based on inputs injected, messages being correctly and timely, etc.
- Accuracy testing focused mostly on the logical, algorithmic level. For accuracy testing, calculated values were compared with ground truth values determined during data collection, to determine sufficient quality algorithm functioning to pass the criteria for inclusion in the final (esp. vehicle) Mediator prototypes. More extensive and exhaustive accuracy evaluation is supplied in public MEDIATOR deliverables D3.3 and D3.4, as part of MEDIATOR's evaluation tasks.

2.12. Investigated scenarios and integration tests

Basic Driving Context and Automation State integration

In one set of tests, performed jointly by ZA and CYG, basic Driving Context functionality feeding into Automation State was tested. This involved recording the outputs produced by a basic Driving Context (which did not yet have dynamic route table updating functionality during a drive), fed by logged TI vehicle data (GPS positions, speed, etc.), into a new log file. Subsequently, this DC log file was fed into the basic Automation State component by similarly replaying the log file, and its outputs recorded, and post-hoc evaluated in terms of output accuracy. These tests were still without the more sophisticated elements of dynamic events such as bad weather or bad traffic being included in the whole Driving Context and Automation State algorithm and software chain.

A main part of the test was testing *predicted* TTAF/TTAU values against *realised* TTAF/TTAU (which is possible because data from logged drives could be used when exactly different automation zones were entered). This showed a good match between those predicted and realised values. As described above, in this task no extensive exhaustive accuracy evaluations were done; the focus was on testing as criteria for inclusion in the final (esp. vehicle) Mediator prototypes. More extensive and exhaustive accuracy evaluation is supplied in public MEDIATOR deliverables D3.3 and D3.4, as part of MEDIATOR's evaluation tasks.

In another test predicted and realised TTAF/TTAU values were compared to TUD's purely virtual *simulation model* of the same system and route used for developing and testing DL (see D2.4). Similarly, a good match was found; suggesting both good functioning by the integrated Driving Context-Automation State system and sufficiently realistic simulation by the TUD.

Figure 12 shows one visualisation tool (used by VEO and ZA) for displaying information relevant for Automation State, when fed by logged TI vehicle data and Driving Context information. This tool was used in particular for sanity checking (using the forward video image) in case of unexpected automation unavailability and the like.

More sophisticated Driving Context and Automation State integration

Another set of tests involved more advanced Driving Context functionality, with simulated dynamic events such as bad weather or bad traffic, feeding into and updating Automation State dynamically at multiple points during a simulated drive. Figure 13 shows a screenshot of one visual analysis tool used during lab prototype development and testing for the analysis of automation availability/fitness and static and dynamic driving context information.

Figure 13 shows drives that have been used for analyzing the automation state component's output based on the driving context input. In our case, we had 6 drives that were collected on the route used for this prototype. Since the algorithm was developed based on 20 drives for a different route, we wanted to investigate how accurate it performs on the "new" route. The results (in bottom part of the Figure 13) show that the predictions were only in 0,34% wrong (74,82% true positives, and 24,83% true negatives) for the entire 6 drives; giving confidence in this component's accuracy and its suitability for inclusion into the final (vehicle) prototype

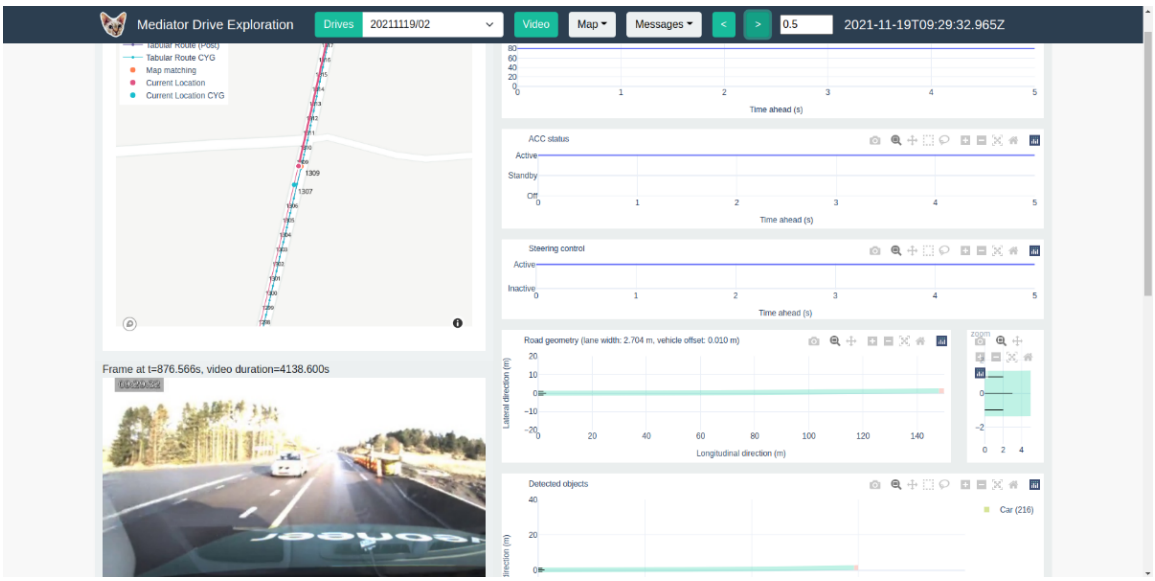

Figure 12 Lab prototype test using visualisations of information relevant for Automation State: map-matched road information from Driving Context and sensed road geometry information. Also shown is forward video data, for sanity checking purposes. All data displayed here was replayed based on log files recorded during earlier TI test drives.

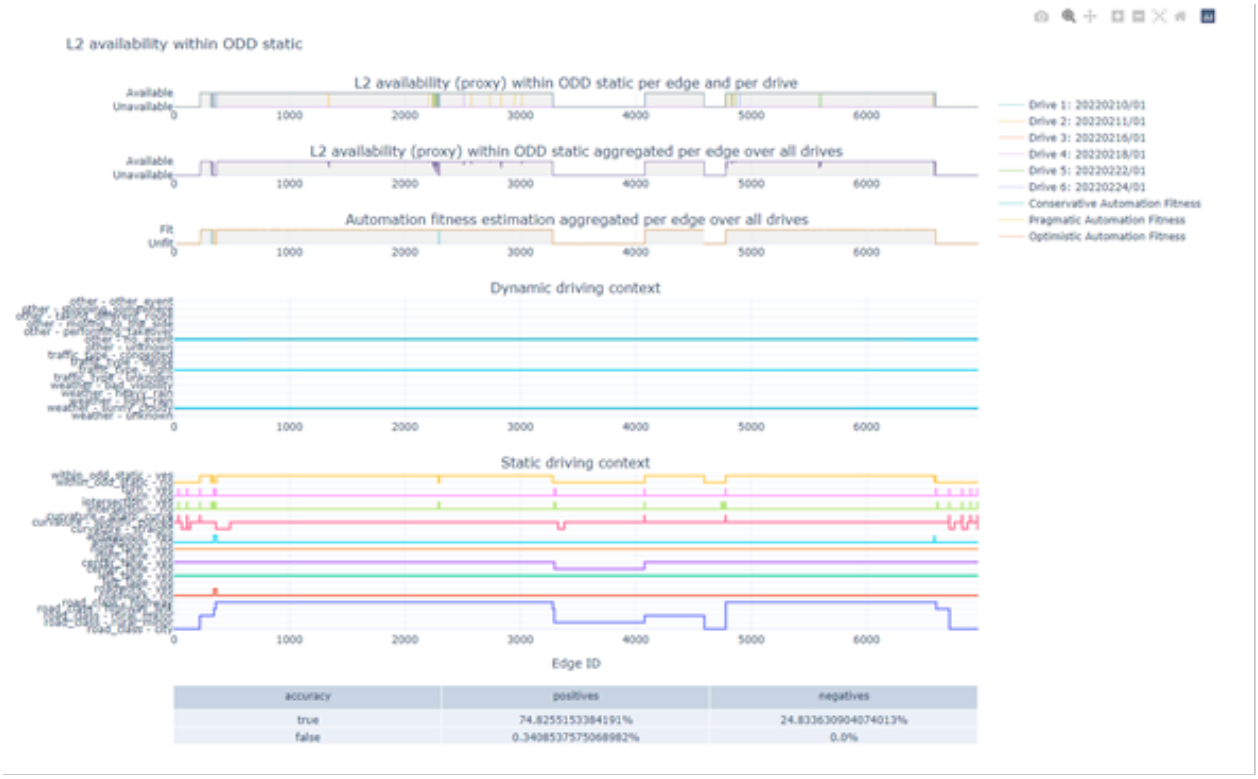

Figure 13 Screenshot of visual analysis tool used during lab prototype development and testing for the analysis of automation availability/fitness and static and dynamic driving context information. In the bottom numerical results are shown.

Driving Context and Automation State feeding into a skeleton Decision Logic component

Another set of tests involved Driving Context and Automation State running simultaneously and feeding into a first, *skeleton* version of integrated Decision Logic, mainly to investigate, tune, and test the input interfaces and messaging. Figure 14 shows a screenshot of the simple text output-based console terminal interface used for this lab prototype test, displaying the behaviours and outputs of the various components involved, all running at the same time (in the different terminals).

The purpose of this test, which focused on software unit and integration testing, and which did not include numerical accuracy analysis, was (in line with what was described above), software tests and sanity checks, testing correct simultaneous and real-time reception and integration of the various input sources for Decision Logic. The result of this test was a simple Test Passed outcome.

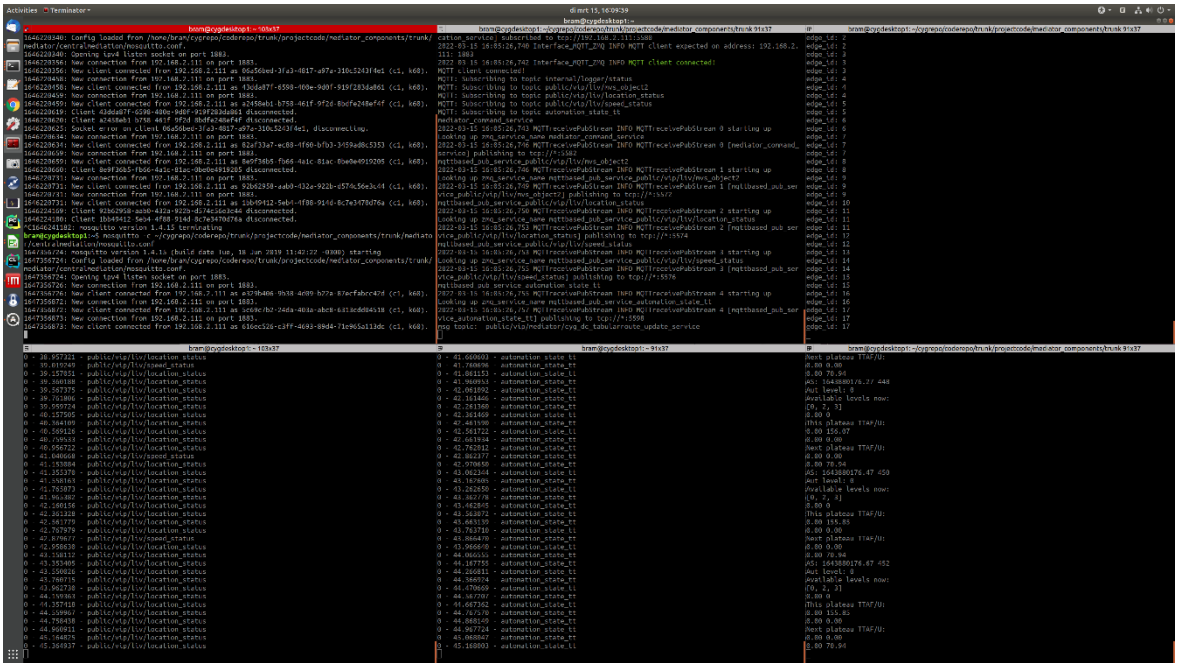

Figure 14 Screenshot of lab prototype's simple console terminal interface, used in testing integration of Driving Context and Automation State running simultaneously and feeding into a first, skeleton version of Decision Logic.

More complete Decision Logic, ignoring Driver State

Another set of tests involved a more complete Decision Logic component (based on Mediator task T2.4 results, i.e. its Decision Tree algorithms). This version was fed by Driving Context and Automation State, but ignoring Driver State for the moment (i.e. assuming no driver degradation). This test, similarly to the one before, had a software integration focus and looked primarily at syntactically correct integration of the inputs and correctness of the Decision Tree implementation, and resulted in a “Test Passed” outcome.

Driver State in combination with Driving Context scenarios using logged data

Another set of tests focused on the inclusion of Driver State into the system, in particular in combination with Driving Context scenarios, all based on recorded (logged) and replayed data from TI vehicle test drives. Figure 15 and Figure 16 show raw, *recorded* and *processed* camera-based Driver State data, respectively. This lab prototype test used logged camera images, replaying them, and feeding into a real-time running Driver State system operational in the lab prototype.

In this test we investigate on the one hand correct integration into Driver State of Driving Context summary information regarding current automation status, basic driving variables (such as gear), road type, and the like (Test Passed); and on the other hand feeding some Driver State information on basic driver activity back to Driving Context (Test Passed).

Figure 15 Example logged camera images from the four cameras installed in the TI vehicle, replayed for Driver State oriented lab prototype tests.

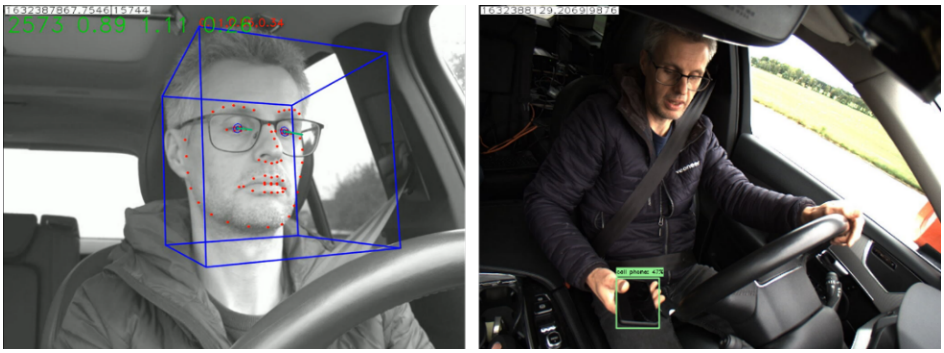

Figure 16 Lab prototype test using logged camera images, replaying them, and feeding into a real-time running Driver State system operational in the lab prototype. In this example, the face view camera is processed on the left, detecting in particular eye closure and gaze as well as facial features; on the right, the body view camera is camera is processed and secondary task activity recognition is performed, using (among other aspects) detection of cell phone use.

Driver State in combination with Driving Context scenarios using live camera data

Another set of tests similarly focused on including Driver State into the whole system, but now using live camera data from actually connected live cameras in the lab set-up (see also Figure 9 and Figure 10), replacing the logged and replayed data from the set of tests described above. In this test, more extreme eyes off road distraction and extensive cell phone use and other types of secondary tasks was tested successfully. Figure 17 shows some of the Driver State component's processed image data from these tests. Again, more extensive and exhaustive accuracy evaluation was not part of these tests; this is done and reported elsewhere (D2.11, D2.11, D3.3, D3.4).

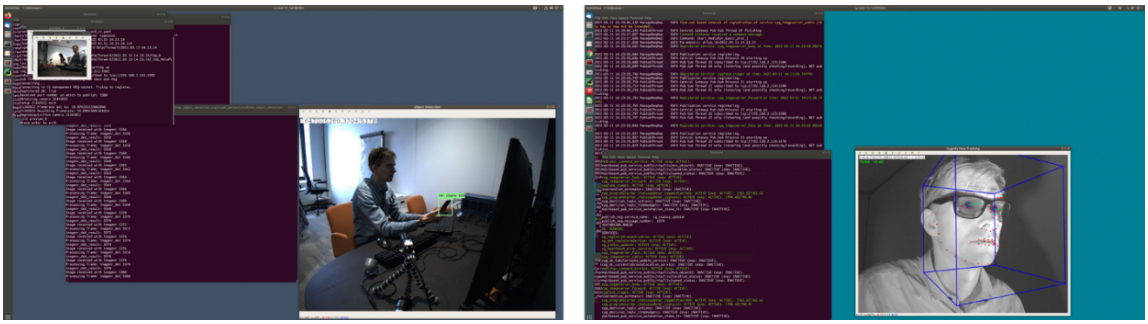

Figure 17 Lab prototype test using the real, real-time camera-based Driver State system, with live cameras running. In this example, the face view camera is processed on the right, detecting eye closure and gaze as well as facial features; on the left, the body view camera imagery is processed and secondary task activity recognition is performed, using (among other aspects) detection of cell phone use.

More complete Decision Logic including Driver State

Another set of tests involved an almost-complete Decision Logic component (based on Mediator DL component development results, i.e. its Decision Tree algorithms), by Driving Context and Automation State and now also fed by Driver State. Decision Logic's interaction with HMI was left out in this test, however. This test (outcome: Test Passed) showed good overall software integration.

Complete lab prototype: Driving Context, Automation State, Driver State, Decision Logic, simulated interaction with HMI

A final set of tests involved the complete lab prototype described in this chapter, consisting of Driving Context, Automation State, Driver State, Decision Logic, and the simulated interaction with the conceptual HMI simulation module. That is, this final complete prototype adds the Decision Logic – HMI interaction testing, based on the conceptual HMI simulation module described in section 2.8. As before, the focus of this test (outcome: Test Passed) was mainly on testing software integration, in preparation for the real-world TI vehicle prototype.

Further fine-tuning and optimization was done in the final stages leading up to final implementation in the vehicle integration task. Further results (including accuracy on the component-level and integrated system-level) were produced after that, using the real system applied in the real world (esp.: the TI vehicle prototype), and reported in public deliverable D3.4.

3. Lab prototype for the TUC driving simulator study

3.1. Core focus and approach

The technical work on the driving simulator lab prototype formed the basis for the evaluation study at TU Chemnitz in Mediator's third Work Package "testing and evaluation" (WP3).

The present chapter presents the lab prototype work for the TUC driving simulator. As the main focus of the WP3 driving simulator study at TUC was on the drivers' perception of the Mediator system and its functionalities, the HMI played an important role. Thus, this lab prototype is mainly focussed on HMI, whereas the other Mediator components such as automation state, driving context and decision logic were implemented in a simulated way to ensure that every participant experience and judge the same Mediator functionality (for details on the experimental setup see deliverable D3.3 on evaluation results of the driving simulator studies).

3.2. Automation State component

The overall approach of having an automation state component that estimates the current and near future automation fitness was also adopted in the driving simulator prototype. However, the driving simulation software (SILAB 7.0 at TUC) does not contain any real sensors but an "automation module" that allows for configuring and manipulating the automation behaviour at any time during a drive. This is also possible in real-time, meaning that e.g. driver inputs can trigger certain automation states such as deactivating automation by pressing the brake pedal or turning the steering wheel. However, the main purpose of the TUC driving simulator study was to let all participants experience the same automation states with corresponding HMI messages in order to perform a standardized user evaluation (details and results in D3.3).

Time To Automation Fitness (TTAF) and Time To Automation Unfitness (TTAU) were also adopted in accordance with the overall Mediator operationalisations. However, these events were linked to certain route meter on the simulated drive, which does not give lower and upper bound estimates to account for uncertainty as in the vehicle prototype. An example is drive 2, where all participants drove in Mediator CM mode and a sudden manual take over was requested within 10 s TTAU after a few kilometres due to increased fog (Figure 18). Another event was reaching the position 1.000 m before a traffic jam, where automation (CM in drive 2, SB in drives 3+4) became available, i.e. start of the Operational Design Domain (ODD). This availability in terms of TTAF was already communicated via the HMI 2.000 m before reaching the traffic jam. A last event was reaching the end of ODD in all drives 2 to 4 because of exiting the highway, where TTAU was used to trigger the HMI behaviour. All further details about the TUC experiment design can be found in D3.3.

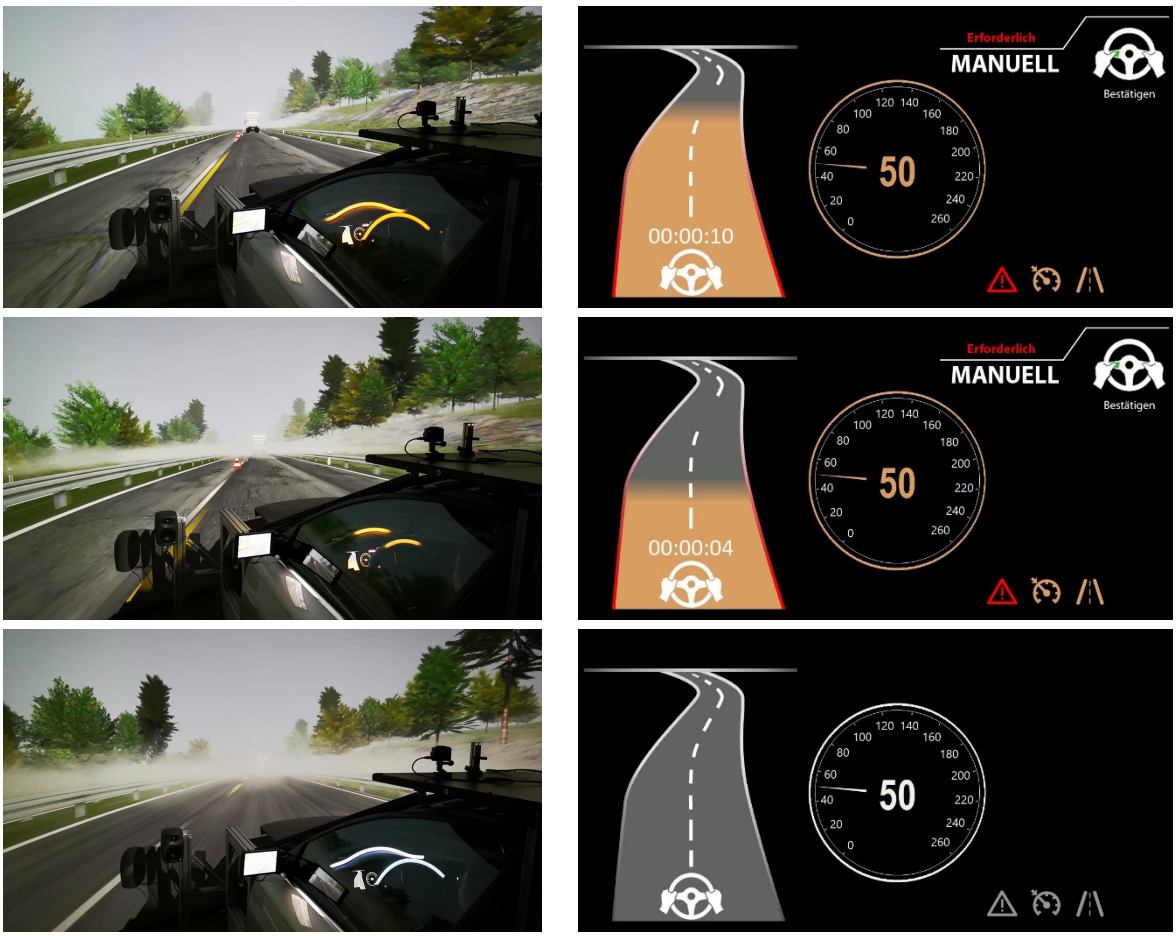

Figure 18 Example of the TUC lab prototype system in Mediator CM mode with outside view including LED strips (left) and corresponding HMI information in the instrument cluster (right). The scenario screenshots were taken during the sudden manual take over request in CM due to increased fog with 10 s TTAU (top), reduced TTAU to 4 s (middle) and after manual take over (bottom).

Another central aim of the TUC driving simulator prototype was to clearly show the current automation mode all the time in order to avoid mode confusion. Thus, every Mediator automation state (manual, CM, SB) was prominently displayed to the user by dedicated colours, symbols and LED bars (Figure 18; Figure 23). All these components connected to the automation state and TTAF/TTAU values were extensively tested in the lab setting before the actual implementation in the driving simulator.

3.3. Driver State component

The main aim of the driver monitoring system for the TUC driving simulator study was to detect whether the participant is focused on the driving task or is distracted due to executing a non-driving related task (NDR-T) based on the glance behaviour. Therefore, participants were instructed to work on a laptop (i.e. answer an E-Mail) in certain sections of the route while driving automated.

The Driver State component's cameras and computers were set up similar to the lab and vehicle prototype described above (see Figure 5) but with an additional TUC computer receiving the image data from the Cygnify computers that accessed the cabin, forward, driver body and face cameras respectively. The communication between the Cygnify and TUC computers was following the general Mediator ZeroMQ framework mentioned above. Figure 19 gives example images of the driver monitoring while the participant is driving manually (top left), is monitoring the automated drive (top right) or is executing a NDR-T, i.e. working on a laptop (bottom).

Figure 19 Example images of the TUC driver monitoring.

The implemented TUC Driver State component consisted of sequential steps. First, in order to estimate the participant's glances, a Bayes classifier trained on features of head tracking, i.e. head position and rotation, has been applied (see details in D3.3). However, as shown in the example images above (bottom right), aside focused glances on the laptop participants also did control glances on the road while executing the NDR-T. Thus, in a second step, hidden markov models (HMM) were trained on sequences of glances to detect characteristic glance patterns and determine if the participant is driving manually/monitoring the driving task (during automated driving) or is distracted by executing a NDR-T.

As the TUC driving simulator study in WP3 focused on use cases without a real-time interaction with the Driver State component to create standardised scenarios over all participants, no live

output was generated during the study. Nevertheless, the performance of the TUC distraction detection was applied on real-time replay of the recorded data set gathered from the TUC driving simulator study and evaluated by measures of signal detection theory, i.e. hit and false alarm rates (see Figure 20). Figure 21 illustrates the results of a leave-one-out cross-validation separated by participant. The hit rate = .84 and false alarm rate = .08 reflect a decent sensitivity index *d'* = 2.4.

	Driving/Monitoring detected	NDR-T detected
NDR-T present	Miss	Hit
Driving/Monitoring present	Correct rejection	False alarm

Figure 20 Hit and false alarm rates from signal detection theory.

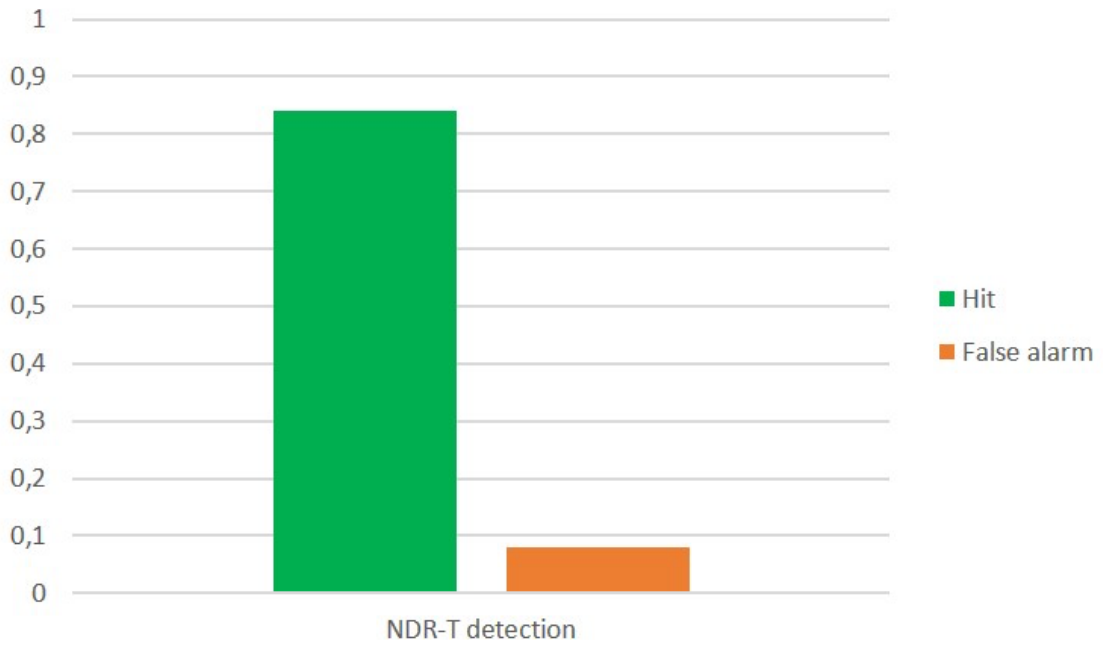

Figure 21 Results of leave-one-out cross validation for the TUC Driver State distraction detection.

3.4. Driving Context

Very similar to the vehicle prototype, the tabular representation of the route including detailed information on road features was also used in the driving simulator prototype. In analogy to the vehicle prototype, the route is known in advance and therefore pre-programmed - with of course muss less variability than on-road. The table for the driving simulator prototype contained in principle the same information as in the vehicle:

- Route meters to determine the current position on the route (instead of map matching as in the vehicle);
- Information on road properties such as road type (onramp/motorway), number of lanes, speed limits, road signs, roadworks, congested traffic, ODD etc., allowing for calculating TTAU/TTAF and HMI message timings.

Figure 22 shows part of the table containing relevant pre-programmed events along the route in the driving simulator. The tabular route information is loaded at start-up of the four study trips and forms the basis for real-time processing. During driving, basic vehicle data comes from the driving simulator software (SILAB), including meter position on the route, automation state, speed, steering wheel position, pedals position, etc.

	A	B	C	D	E	F	G	H	I	J	K
1	trip	trip_condition	meter_start	meter_end	weather	road_type	speed_limit	signs_events			
26	2	CM - Continuous Mediation (L2)	0	5090	fog (visibility approx. 200m)						
27	2	CM - Continuous Mediation (L2)	5090	8000	heavy fog (visibility approx. 50m)						
28	2	CM - Continuous Mediation (L2)	0	216		onramp to motorway					
29	2	CM - Continuous Mediation (L2)	216	4290		2 lane motorway					
30	2	CM - Continuous Mediation (L2)	4290	5850		1 lane motorway (construction zone)					
31	2	CM - Continuous Mediation (L2)	5850	7830		2 lane motorway					
32	2	CM - Continuous Mediation (L2)	7830	8000		motorway exit					
33	2	CM - Continuous Mediation (L2)	0	4100			80				
34	2	CM - Continuous Mediation (L2)	4100	5900			60				
35	2	CM - Continuous Mediation (L2)	5900	8000			80				
36	2	CM - Continuous Mediation (L2)	300	300				sign: speedlimit 80 km/h			
37	2	CM - Continuous Mediation (L2)	3130	3130				Start of ODD for CM (offer by audio message)			
38	2	CM - Continuous Mediation (L2)	3500	3500				sign: reduction to 1 lane in 800m			
39	2	CM - Continuous Mediation (L2)	4100	4100				sign: speedlimit 60 km/h			
40	2	CM - Continuous Mediation (L2)	4200	4200				traffic jam / close approach to truck			
41	2	CM - Continuous Mediation (L2)	4350	4350				sign: speedlimit 60 km/h			
42	2	CM - Continuous Mediation (L2)	5140	5140				Start Takeover ritual CM to manual with 10s time (audio + LED)			
43	2	CM - Continuous Mediation (L2)	5280	5280				End of ODD for CM due to heavy fog			
44	2	CM - Continuous Mediation (L2)	5900	5900				sign: speedlimit 80 km/h			
45	2	CM - Continuous Mediation (L2)	6800	6800				sign: exit in 1000m			
46	2	CM - Continuous Mediation (L2)	7300	7300				sign: exit in 500m			
47	2	CM - Continuous Mediation (L2)	7500	7500				sign: exit in 300m			
48	2	CM - Continuous Mediation (L2)	7600	7600				sign: exit in 200m			
49	2	CM - Continuous Mediation (L2)	7700	7700				sign: exit in 100m			
50	2	CM - Continuous Mediation (L2)	7880	7880				sign: exit "Knoblach"			
51	3	SB - Stand By (L3)	0	8000	good, sunny						
52	3	SB - Stand By (L3)	0	216		onramp to motorway					
53	3	SB - Stand By (L3)	216	4290		2 lane motorway					
54	3	SB - Stand By (L3)	4290	5850		1 lane motorway (construction zone)					
55	3	SB - Stand By (L3)	5850	7830		2 lane motorway					
56	3	SB - Stand By (L3)	7830	8000		motorway exit					
57	3	SB - Stand By (L3)	0	4100			80				
58	3	SB - Stand By (L3)	4100	5900			60				

Figure 22 Example part of the table containing relevant pre-programmed events along the route in the driving simulator.

3.5. Decision Logic

As the main purpose of the TUC driving simulator study in WP3 was to provide a highly standardized (and optimally identical) experience of the Mediator system to the user, the DL was only implemented in a rather simplistic and static way. Main aim was to assess differences in user reactions to the same Mediator decisions and actions. Thus, in principle the DL in the TUC prototype had the same role as in the vehicle prototype, such as recommend switches between the automation states according to availability and TTAF/TTAU. However, all these decisions were pre-programmed dependent on the actual meter on the road, the trip number and/or user actions such as intervening by pedals or steering wheel. Therefore it was not necessary to implement sophisticated algorithms for DL in the TUC prototype – most decision were based on simple IF-THEN statements.

3.6. HMI and Use Cases

The main focus of the driving simulator study at TUC in WP3 was on the drivers' perception of the Mediator system and its functionalities. Thus, the HMI played an important role. For this matter, a sophisticated HMI was implemented into the simulator. The HMI concept was designed by the colleagues from TU Delft and adapted to the use cases, driving scenarios and spatial conditions of

the driving simulator. The HMI concept included the following components (see Figure 23, also Figure 18, Figure 25, Figure 28):

- A screen behind the steering wheel displaying driving relevant data (e.g., speedometer, driving mode, TTAF/TTAU, available driving mode for the upcoming road section), instructions to the drivers (e.g., take over requests) and information about the context (e.g., detection and distance of a traffic jam)
- A total of three LED strips on the dashboard, the steering wheel as well as ambient lighting indicating the current driving mode by dedicated colours, the intended or necessary changes in the driving mode as well as remaining time by different illumination width and pulsation frequency
- A sound system for sound alerts and spoken messages, integrated in the screen behind the steering wheel
- A laptop mounted in front of the centre console to run a secondary task

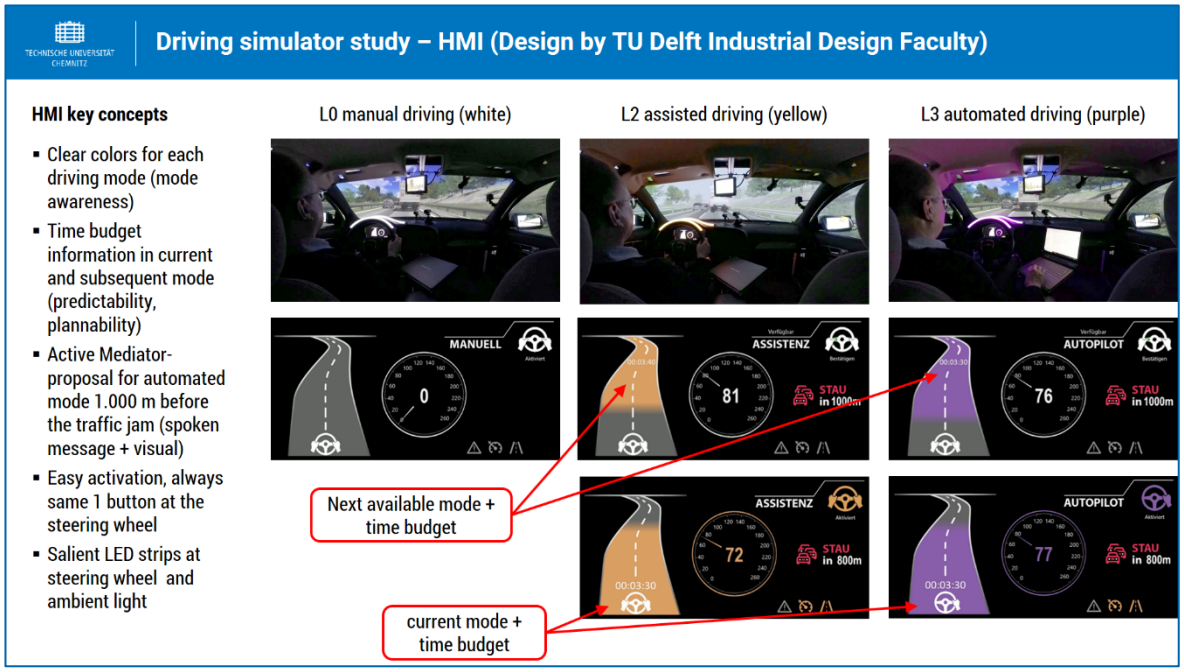

Figure 23 Key HMI elements and examples of the three automation modes in the TUC driving simulator prototype.

The driving simulator study focused mainly on comfort transitions of control from manual to automated driving, simulated automation degradation and related transitions of control by the human driver as well as driver characteristics. Hence, take over requests from the driver or the Mediator system were examined, planned or unplanned transitions from automation to manual driving due to degraded automation fitness (such as dense fog), uncomfortable driving manoeuvres (i.e., close approach scenarios) and the influence of driver characteristics (e.g., age, gender, driving experience). Five use cases were covered by the driving simulator study with the respective technical implementation of the HMI:

- **Use Case 2 - Driver takes back control:** The driver uses the HMI to indicate a desire to take back vehicle control. The Mediator system reacts by confirming that the driver is fit enough to drive and guiding the takeover.

- **Use Case 3b - Comfort takeover (human to automation):** Comfort takeover from human to automation initiated by the Mediator system: The Mediator system detects an event, such as receiving a text message or an upcoming traffic jam, from which it concludes that the driver comfort could be improved. The system reacts by suggesting a takeover to automation.
- **Use Case 5a - Mediator initiated takeover (automation to human):** Planned takeover from automation to human initiated by the Mediator system: The automation indicates that the current route leads to automation unfitness as it will leave its operational design domain. The Mediator system reacts by preparing the driver for and guiding the driver through a non-urgent takeover.
- **Use Case 6b - Comfort CM switch on:** The Mediator system suggests switching on driving in "Continuous Mediation" mode due to comfort-related reasons: The Mediator system detects sufficient fitness for driving in CM from which it concludes that the driver comfort could be improved and reacts by suggesting to switch on CM.
- **Use Case 9 - CM shuts off suddenly:** While driving in CM, the automation fitness degrades due to dense fog and automation can no longer perform its driving task. The Mediator system reacts by communicating to the driver that "Continuous Mediation" mode is switching off and manual takeover needs to be performed within 10 s (Figure 18).

3.7. Hardware set-up

The hardware for the Mediator driving simulator prototype consisted of a powerful Windows laptop as core component (“Mediator laptop”, Figure 14, green front light), connected via USB to an Arduino Due for controlling the three LED strips and a secondary 14-inch screen with sound output as instrument cluster behind the steering wheel. The Mediator laptop was connected via standard LAN connection to the driving simulator network. The second laptop (Figure 14, blue front light) was used for real-time displaying driving data and was not part of the Mediator prototype. An Intel NUC was used to control the TUC driver state monitoring and recording (Figure 14, monitor on right side). In addition, two computers provided by Cygnify that accessed the four driver state cameras (FLIR, Blackfly S USB3) and recorded HD images while transferring the camera data in VGA resolution to the TUC computer were integrated. The Mediator laptop contained the complete Mediator software programmed in NI Labview (see next chapter) including the modules HMI, decision logic, automation state and driving context. The automation itself was part of the driving simulator software itself (automation module of SILAB) and could activated and deactivated by pressing a button at the left side of the steering wheel.

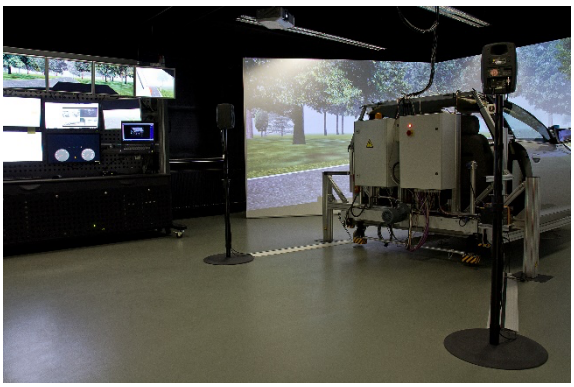

Figure 24 TUC driving simulator before adding the Mediator lab prototype (top left), full scenery during a Mediator trip with the three Mediator computers in the back (bottom) and closer look (top right) to the Mediator laptop showing the instrument cluster (green front light) and data display laptop (blue front light).

Figure 25 Hardware for controlling the three WS2812b LED strips based on Arduino Due with the PC power supply (top left), after integration under the engine hood of the driving simulator (right) and active in SB automation mode with footwell lighting and laptop for secondary tasks (bottom left)

All three LED strips displaying the automation state were WS2812b LED strips powered by 5V. The ambient light LED strip consisted of a 1 m strip with 100 LED and was mounted as footwell lighting for both the driver and passenger area (Figure 25). The led strip at the steering wheel consisted of 51 LED and the strip behind the steering wheel of 67 LED, both with a density of 144 LED/m (Figure 25). As the power consumption of all led strips could have reached about 50 watts (full bright white on all LED), the strips were powered by a PC power supply (Figure 25). All strips were connected to an Arduino Due and controlled using the FastLed library (see next chapter).

3.8. Software set-up

The software for the TUC driving simulator prototype consisted of several parts: 1) The core software component on the Mediator Windows laptop, programmed in NI Labview (Figure 26) and 2) the software for controlling the led strips on the Arduino Due, programmed in C++. The Mediator laptop received all driving data from the SILAB driving simulator software via UDP messages over an Ethernet connection. All components of the decision logic, driving context and automation state were integrated in the Labview code (Figure 26), adapted to the four different trips and the events occurring at certain route meters. These events triggered the visual HMI elements at the instrument cluster screen (like time budgets, colour fading, Mediator messages about availability of automation modes, announcement of the traffic jam etc.) as well as pre-recorded audio messages. In addition, the Labview software component triggered pre-programmed led patterns at the Arduino Due, such as colour changes according to the current driving mode, the decreasing width of the led lighting in accordance with the remaining time budget and escalation patterns such as blinking during the last 10 s of the time budget. The Labview code also contained a data logging component where all Mediator HMI actions were recorded at 60Hz and saved to a csv file, synchronized to the driving data from the simulator software.

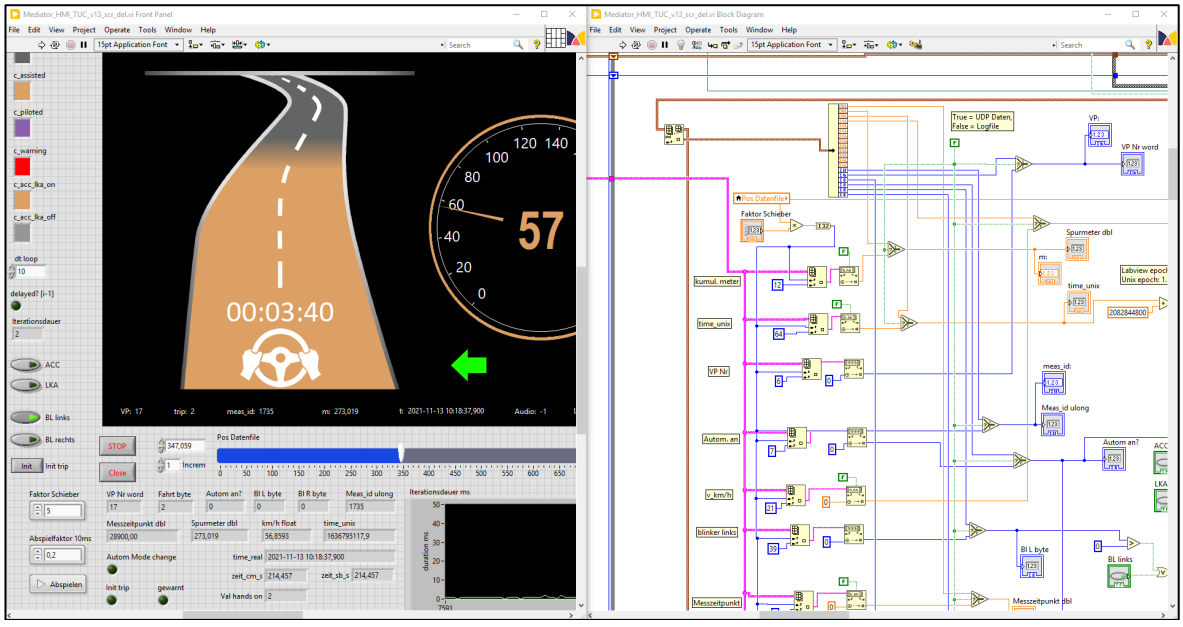

Figure 26 Core software component on the Mediator Windows laptop in the NI Labview software development environment with parts of the front panel (left) and block diagram (right).

3.9. Incremental integration and testing procedure

Similar to the lab prototype described in chapter 2 and the vehicle prototypes, the driving simulator prototype was also developed incrementally and stepwise outside the actual driving simulator in a lab. This development strategy was also important due to organisational reasons: the driving simulator is a shared resource at the university with dedicated time slots for each study, which does not allow for long component development and testing. In addition, the driving simulator was closed for longer periods of time due to COVID, which made it even more important to develop most hardware and software components outside the simulator. These were the development steps:

Step 1: Similar to the vehicle prototype, one of the most important first steps was to set up a replay system of driving simulator data in the Labview environment (Figure 27). This means that a csv-logfile from a previous driving simulator study with the exact data structure was used as template for developing and testing all Mediator functions. This csv-file could be loaded into the Labview Mediator program code, replayed at various positions and selectable speed. In addition, the csv-file could be edited manually to simulate potential user actions such as deactivating automation by pressing the brake pedal or turning the steering wheel. This procedure allowed for developing more than 90% of the final prototype in the driving simulator. The last 10% mainly consisted in setting up the network connections and technical communication in the driving simulator lab, fine-tuning of message timings, positioning of display elements and adjusting brightness of led-lights as well as extensive stability tests.

Figure 27 Mediator software prototype with replay system of driving simulator data in the Labview environment. The graph at bottom right shows the current execution timings in ms.

Step 2: After having the replay system in place, the second step was to implement all visual HMI components designed by TU Delft as real-time controls in Labview. For this, standard Labview components such as gauges, picture rings, number and text fields as well as colour ramps were

adapted to the Mediator design (Figure 27). Continuous tests with the replay data made sure that all elements worked properly. In addition, standardized audio messages were recorded as mp3-files using the AI voice generator play.ht.

Step 3: The third step aimed at programming and testing the Arduino code component in C++ for controlling the LED strips. All led pattern according to the HMI design by TU Delft were implemented into software code using the FastLED-Library (https://github.com/FastLED) and tested with first just one strip. After successful implementation of all patterns, the two additional strips were added. Initially, an Arduino Uno was used, however, longer lab tests with all the three LED strips showed timing problems due to the high amount of LED (https://github.com/FastLED/FastLED/wiki/Interrupt-problems). The problem could be solved by switching to a more powerful Arduino Due.

Continuous testing in the lab environment: During the whole development and integration of all hardware and software components, extensive tests were performed in the lab (Figure 28). Next to the correct functioning of all components, the software execution timings and potential delays played an important role in the development. The minimum target frequency of the whole system was 60 Hz to be aligned with the incoming driving simulator data, which means that all HMI update actions including logging needed to be completed within 16 ms. To ensure and monitor execution timings, timed loops of the Labview software environment were used, showing the exact milliseconds for each processing cycle (Figure 27 bottom right). Using this strategy, bottlenecks could be identified (such as display elements consuming too much milliseconds) and replaced by alternative solutions. Long term lab test over several hours ensured that no errors occurred due to overheating problems or cumulative delays e.g. in the data logging part. Actually, the Mediator driving simulator prototype did not fail even once during the user study with all 74 participants.

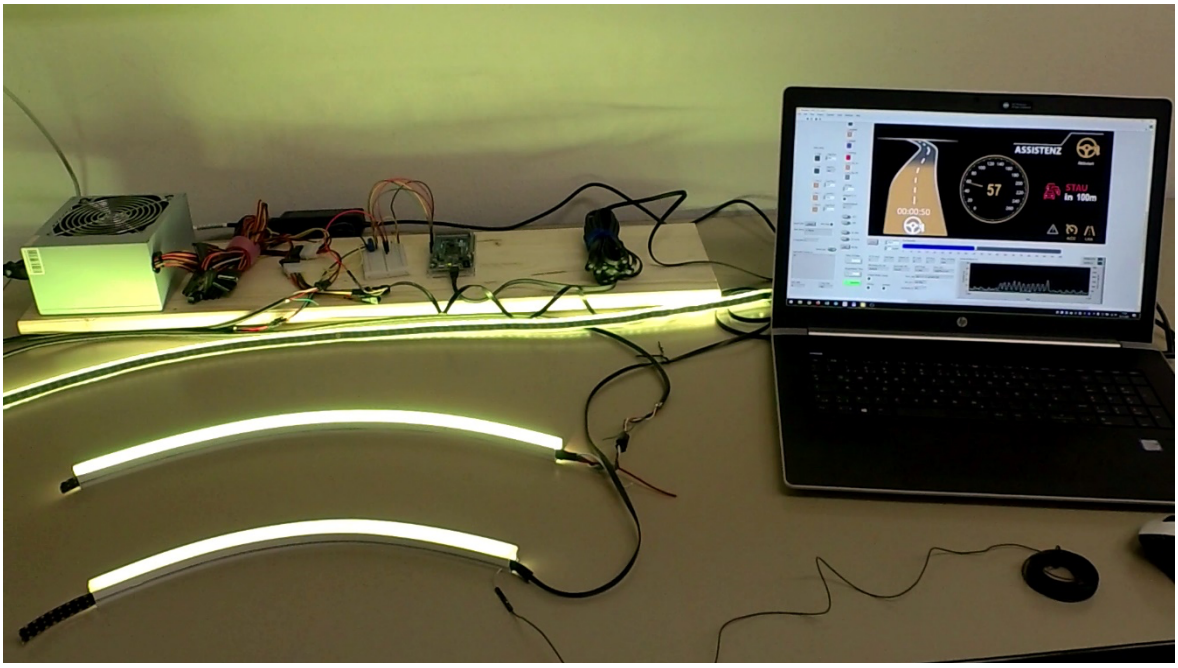

Figure 28 Lab test setup of the driving simulator prototype including the three LED strips.

Testing in the driving simulator environment: After successful completion of all lab tests and about one month before starting the user tests, the whole system was integrated and extensively tested

in the driving simulator (Figure 25). A standard procedure before starting the data collection with the real participants was to record a complete dataset with an experimenter as participant (Figure 30). This dataset served multiple purposes: 1) test the full experimental setup with all Mediator components, actions, HMI messages etc. from the perspective of the participant, 2) check the data logging and data analysis chain, 3) serve as reference dataset shared with all Mediator partners and 4) serve as data source for showing pictures of e.g. the face video at public events like conferences without having privacy problems.

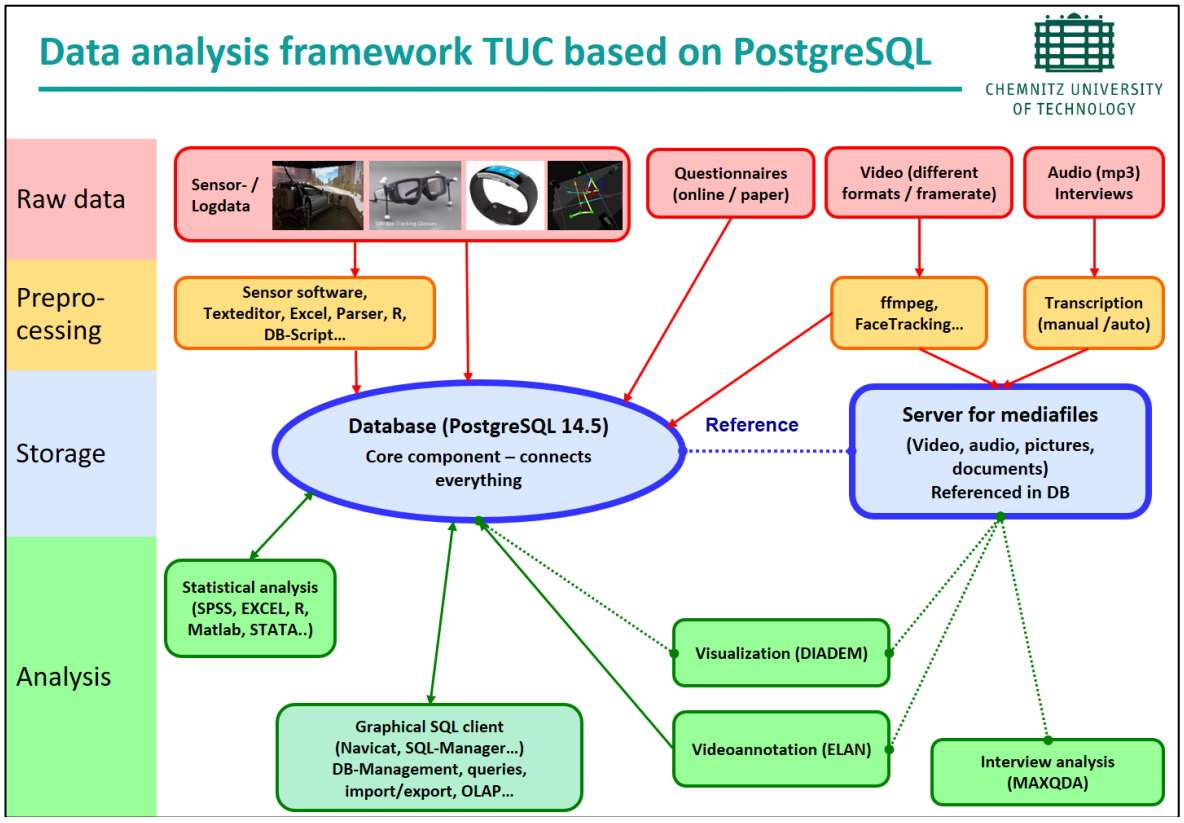

Figure 29 Data analysis framework of TUC based on the relational open source database system PostgreSQL 14.5.

The reference dataset was processed in the data analysis framework of TUC (Figure 29), based on the relational database system PostgreSQL. Within the database framework, several standardized test scripts were executed to check e.g. for delays in the simulation as well as completeness and plausibility of all data. Visual checks of the video data synchronized with all other data using NI Diadem (Figure 30) made sure that all data were correctly synchronized over the full period of recording. All subsequent datasets from the 74 study participants were processed the same way. The database system allows for efficient management of the data as well as for exporting the data in various formats for other Mediator partners.

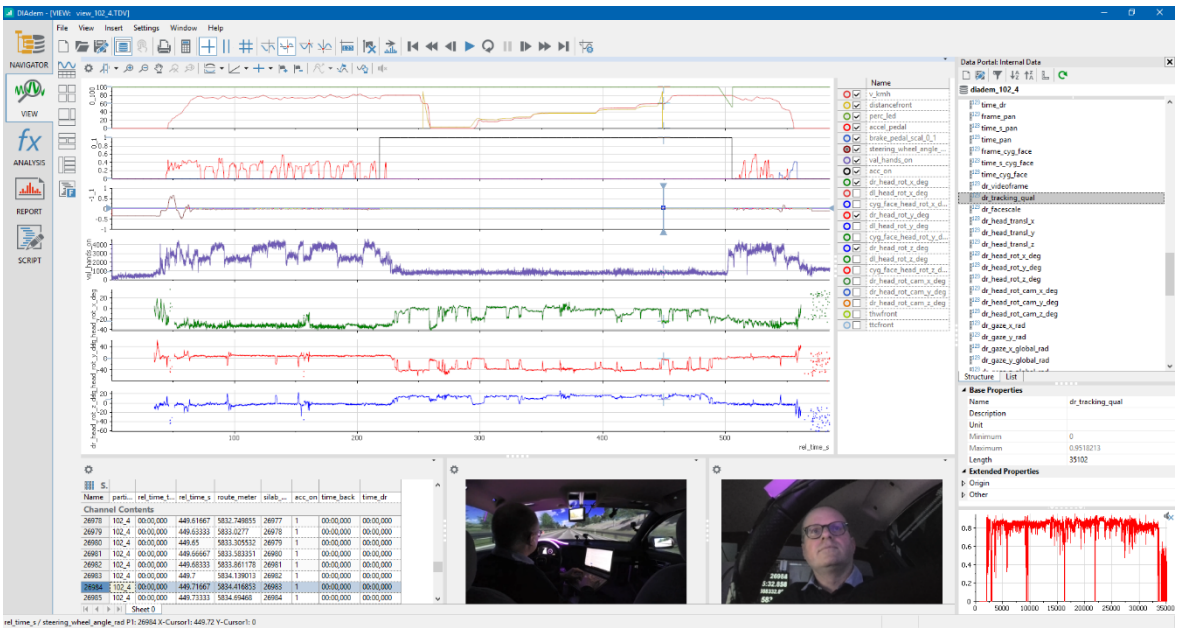

Figure 30 Exemplary visualisation of synchronized data from the experimenter reference dataset in the data visualisation environment NI DIADEM.

4. Conclusions

This report has described the conceptual approach behind the main lab prototypes, the specific versions of the main Mediator components used prototype, the set-up of the lab prototypes in terms of hardware and software, and the main development steps and investigations that were performed.

One important approach that was used, which helped efficient development and testing a great deal, was the approach of mixing and matching real, real-time operating sensor and software components with simulated replacements, where those simulated replacements are realised in particular by replaying previously logged data. In this way, realistic data streams from for example Automation State could be realised in the lab setting.

Another important and useful approach that was used was incremental integration and testing. This means that developing and testing of integrated component functioning was done by adding individual main components one by one, having the important advantage that not all complexity is addressed all at once. The tests were very diverse in nature, in terms of type of test and the test-passed criteria, and typically focused on software system integration and simple acceptance tests, in preparation for the real-world prototypes (in particular the vehicles); leaving extensive numerical (accuracy, acceptance, etc.) evaluation to MEDIATOR's Evaluation Work Package, WP3.

This also points to limitations of these lab prototypes. Extensive and exhaustive evaluation of overall functioning and accuracy of included components was left to MEDIATOR's later evaluation stages. For the driving simulator lab prototype, another limitation is that the implementation only works for the specifically designed route in the driving simulator; i.e. events happening on certain specific route meters, fog appearing at certain meters, close approach to a traffic jam at certain meters, etc. At the same time, for that prototype and its intended use (see also deliverable D3.3), this high experimental standardisation corresponds to the very nature and advantage of driving simulator experiments.

Given that with the main lab prototype(s) correct joint functioning of the main components in an integrated Mediator system was demonstrated and tested (to the extent possible in these lab settings), the next important step was final fine-tuning and implementation in the vehicle prototypes, the driving simulator prototypes, and the purely computer simulation prototype focusing on Decision Logic. These are all described in the public MEDIATOR deliverables associated with Work Package 3.