Autonomous Vehicles to Evolve to a New Urban Experience

DELIVERABLE

# D6.3 First Iteration Controlled environment vehicle safety evaluation report

Co-funded by the Horizon 2020 programme of the European Union

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 769033

# Disclaimer

This document reflects only the author's view and the European Commission is not responsible for any use that may be made of the information it contains.

# Document Information

| **Grant Agreement Number** | **769033** |
|---|---|
| Full Title | Autonomous Vehicles to Evolve to a New Urban Experience |
| Acronym | AVENUE |
| Deliverable | D6.3 Fist Iteration Controlled environment vehicle safety evaluation report |
| Due Date | 31.10.2020 |
| Work Package | WP6 |
| Lead Partner | CEESAR |
| Leading Author | Jérôme Ehrhardt, AVL LMM |
| Dissemination Level | Public, restricted |

# Document History

| **Version** | **Date** | **Author** | **Description of change** |
|---|---|---|---|
| 0.1 | 31.03.2020 | AVL LMM | First draft |
| 0.2 | 22.04.2020 | Jérôme Ehrhardt | Document update |
| 0.3 | 24.07.2020 | Jérôme Ehrhardt | Document update |
| 0.4 | 20.10.2020 | Clément Val | Restructuring/harmonisation of document, update of VIF contribution, English improvement |
| 1.0 | 23.10.2020 | Jérôme Ehrhardt | Update with conclusion |
| 1.1 | 26.10.2020 | Dimitri Konstantas | Review |
| 1.2 | 28.10.2020 | Johannes Jany-Luig | Review |
| 1.3 | 30.10.2020 | Clément Val | Final corrections |
| 1.4 | 26.11.2021 | AVL Team | Revision according to PO feedback |

# Table of Contents

Disclaimer ..... 2
Document Information ..... 2
Document History ..... 2
Table of Contents ..... 3
Acronyms ..... 5
Executive Summary ..... 7
1 Introduction ..... 1
1.1 On-demand Mobility ..... 1
1.2 Autonomous Vehicles ..... 1
1.2.1 Autonomous vehicle operation overview ..... 2
1.2.2 Autonomous vehicle capabilities in AVENUE ..... 3
1.3 Preamble ..... 3
2 Vehicle Safety Evaluation Methodology ..... 5
2.1 Overview ..... 5
2.1.1 Scope of Task 6.1 ..... 9
3 Simulation environment architecture ..... 11
3.1 Simulation Inputs ..... 11
3.2 Internals ..... 12
3.3 Simulation Outputs and Risk Assessment ..... 13
4 Toolchain implementation ..... 14
4.1 CARLA implementation ..... 14
4.2 VTD based toolchain ..... 17
4.2.1 Importing scenarios developed with CARLA ..... 19
4.2.2 Infrastructure environment ..... 20
4.2.3 Vehicle model ..... 20
4.2.4 Toolchain testing ..... 28
5 Safety assessment ..... 31
5.1 Data reading and use case isolation ..... 31
5.2 Evaluation Criteria ..... 32
5.2.1 Deceleration Safety Criterion ..... 32
5.2.2 Lateral Safety Criterion ..... 33
5.2.3 Ego Decision Criterion ..... 34
5.2.4 Minimum Distance to Lane ..... 35
5.2.5 Response Delay Criterion ..... 36

5.2.6 Reach Target Speed ..... 37
5.3 Evaluation and reporting ..... 38
6 Conclusions ..... 40
7 Appendix ..... 42
7.1 AVL-DRIVE AD ..... 42
7.2 Objective Assessment of Subjective Criteria ..... 42
7.3 Automated and Real-Time Assessment ..... 42
7.4 AVL-DRIVE AD process ..... 43

# Acronyms

| | |
|---|---|
| ADAS | Advanced driver assistance systems |
| ADS | Automated Driving Systems |
| AI | Artificial Intelligence |
| AM | Autonomous Mobility |
| API | Application Protocol Interface |
| AV | Autonomous Vehicle |
| BM | Bestmile |
| BMM | Business Modelling Manager |
| CARLA | Open-source simulator for autonomous driving research |
| CAV | Connected and Autonomous Vehicles |
| CB | Consortium Body |
| CERN | European Organization for Nuclear Research |
| D7.1 | Deliverable 7.1 |
| DC | Demonstration Coordinator |
| DI | The department of infrastructure (Swiss Canton of Geneva) |
| DMP | Data Management Plan |
| DSES | Department of Security and Economy - Traffic Police (Swiss Canton of Geneva) |
| DTU | Technical University of Denmark |
| test track | test track |
| EAB | External Advisory Board |
| EC | European Commission |
| ECSEL | Electronic Components and Systems for European Leadership |
| EM | Exploitation Manager |
| EU | European Union |
| EUCAD | European Conference on Connected and Automated Driving |
| F2F | Face to face meeting |
| FEDRO | (Swiss) Federal Roads Office |
| FOT | (Swiss) Federal Office of Transport |
| GDPR | General Data Protection Regulation |
| GIMS | Geneva International Motor Show |
| GNSS | Global Navigation Satellite System |
| HARA | Hazard Analysis and Risk Assessment |
| IPR | Intellectual Property Rights |
| IT | Information Technology |
| ITU | International Telecommunications Union |
| KPI | Key Performance Indicator |
| LA | Leading Author |
| LIDAR | Light Detection And Ranging |
| MEM | Monitoring and Evaluation Manager |
| MiL | Model In the Loop |
| MT | MobileThinking |
| OCT | General Transport Directorate of the Canton of Geneva |
| ODD | Operational Domain Design |
| OEDR | Object And Event Detection And Response |
| OFCOM | (Swiss) Federal Office of Communications |
| PC | Project Coordinator |
| PEB | Project Executive Board |
| PGA | Project General Assembly |
| PRM | Persons with Reduced Mobility |
| PSA | Group PSA (PSA Peugeot Citroën) |
| PTO | Public Transportation Operator |
| PTS | Public Transportation Services |
| ROD | Road Network Editor |
| QRM | Quality and Risk Manager |
| QRMB | Quality and Risk Management Board |
| RN | Risk Number |
| SA | Scientific Advisor |
| SAE Level | Society of Automotive Engineers Level (Vehicle Autonomy Level) |
| SAN | (Swiss) Cantonal Vehicle Service |
| SDK | Software Development Kit |
| SiL | Software-in-the-Loop |
| SLA | Sales Lentz Autocars |
| SLAM | Simultaneous Localization and Mapping |
| SMB | Site Management Board |
| SoA | State of the Art |
| SOTIF | Safety Of The Intended Functionality |
| SWOT | Strengths, Weaknesses, Opportunities, and Threats. |

| | |
|---|---|
| T7.1 | Task 7.1 |
| TM | Technical Manager |
| TOF | Target Object Front |
| TPG | Transport Publics Genevois |
| UITP | Union Internationale des Transports Publics (International Transport Union) |
| V2I | Vehicle to Infrastructure communication |
| ViL | Vehicle-intheLoop |
| VSM | value stream map |
| VTD | Virtual Test Drive |
| WP | Work Package |
| WPL | Work Package Leader |

# Executive Summary

To gain relevance and therefore acceptance, autonomous shuttles will require higher operating-speeds and ability to operate without on board safety drivers, on more flexible routes. This causes significant safety challenges, which are addressed in AVENUE in the context of tasks 6.1 and 6.2.
Safety assessment in those tasks focuses on issues which are particular to autonomous shuttles. This corresponds to the recently introduced concept of *Safety of the Intended Functionality* (SOTIF), i.e. the identification and mitigation of threats resulting from inadequacy between one vehicle's capacities (e.g. situational awareness resulting from its sensors and perception algorithms, decision model, reaction time) and the conditions in which it is used (e.g. speed, weather, surroundings, other users' behaviour).
Within this context, Task 6.1's aim is to carry out controlled environment trials to assess that performance targets are met, before evolutions are deployed on the field. The controlled environment allows to test and validate all autonomous driving functions integrated in the shuttle regarding safety of its passengers and its surrounding environment.  It can be based on real tests or simulations. For simulation, the environment conditions (roads, traffic lights, players…) could be defined with high accuracy and all use cases could be studied whatever their risk of accident which is an advantage to consider such an environment.
A methodology and a toolchain are designed and implemented to allow this evaluation to largely be done using numerical simulation, although ideally, these will be consolidated by physical tests. To that effect, experimental procedures will be defined to closely match conditions of worst-case scenarios and reproduce them in a safe way (i.e. dummy obstacles and empty vehicle). Safety-critical scenarios will be analysed and evaluated in controlled (simulated) environment to mitigate the involved risks. This task aims at identifying and designing appropriate quality measures for safety and comfort and to evaluate to which degree safety of the automated vehicle function has been achieved. This includes the definition of the metrics to evaluate the automated vehicle functions and the subjective safety feeling. Those metrics will be implemented in the AVL-DRIVE AD software, which will then become capable of assessing the drivability of automated vehicles in different driving conditions.

In this report, a methodology for safety evaluation is described firstly following by a description of the simulation environment architecture based on AVL tool chain. Secondly, the safety assessment based on AVL DRIVE AD tool is introduced to explain how different safety criterions are evaluated based on real tests or simulated data.

# 1 Introduction

AVENUE aims to design and carry out full-scale demonstrations of urban transport automation by deploying, for the first time worldwide, fleets of autonomous minibuses in low to medium demand areas of 4 European demonstrator cities (Geneva, Lyon, Copenhagen and Luxembourg) and 2 to 3 replicator cities. The AVENUE vision for future public transport in urban and suburban areas, is that autonomous vehicles will ensure safe, rapid, economic, sustainable and personalised transport of passengers. AVENUE introduces disruptive public transportation paradigms on the basis of on-demand, door-to-door services, aiming to set up a new model of public transportation, by revisiting the offered public transportation services, and aiming to suppress prescheduled fixed bus itineraries.

Vehicle services that substantially enhance the passenger experience as well as the overall quality and value of the service will be introduced, also targeting elderly people, people with disabilities and vulnerable users. Road behaviour, security of the autonomous vehicles and passengers' safety are central points of the AVENUE project.

At the end of the AVENUE project's four-year period, the mission is to have demonstrated that autonomous vehicles will become the future solution for public transport. The AVENUE project will demonstrate the economic, environmental and social potential of autonomous vehicles for both companies and public commuters while assessing the vehicle road behaviour safety.

## 1.1 On-demand Mobility

Public transportation is a key element of a region's economic development and the quality of life of its citizens.

Governments around the world are defining strategies for the development of efficient public transport based on different criteria of importance to their regions, such as topography, citizens' needs, social and economic barriers, environmental concerns and historical development. However, new technologies, modes of transport and services are appearing, which seem very promising to the support of regional strategies for the development of public transport.

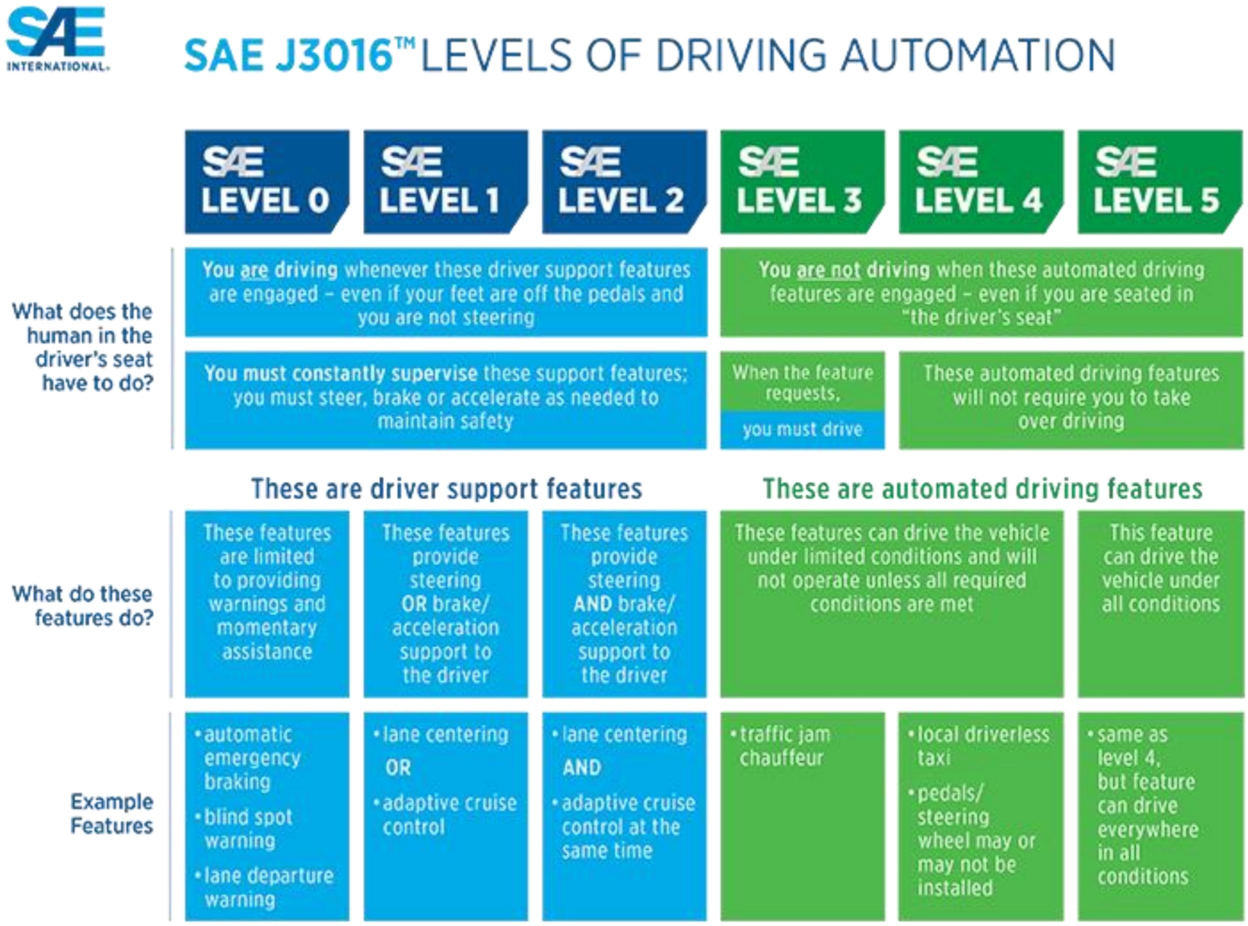
On-demand transport is a public transport service that only works when a reservation has been recorded and will be a relevant solution where the demand for transport is diffuse and regular transport inefficient. On-demand transport differs from other public transport services in that vehicles do not follow a fixed route and do not use a predefined timetable. Unlike taxis, on-demand public transport is usually also not individual. An operator or an automated system takes care of the booking, planning and organization.

It is recognized that the use and integration of on-demand autonomous vehicles has the potential to significantly improve services and provide solutions to many of the problems encountered today in the development of sustainable and efficient public transport.

## 1.2 Autonomous Vehicles

A self-driving car, in the AVENUE project referred to as **an Autonomous Vehicle** (**AV**), is a vehicle that is capable of sensing its environment and moving safely with no human input. The choice of "autonomous" vs "automated" was made in AVENUE since, in the current literature, most of the vehicle concepts have a person in the driver's seat, utilize a communication connection to the Cloud or other vehicles, and do not independently select either destinations or routes for reaching them, thus being "automated". The automated vehicles are considered to provide assistance (at various levels) to the driver. In AVENUE, there will be no driver (so no assistance will be needed), while the route and destinations will be defined autonomously (by the fleet management system). The target is to reach a system comprising of vehicles and services that independently select and optimize their destination and routes, based on the passenger demands.

In relation to the SAE levels, the AVENUE project will operate SAE Level 4 vehicles.






## 1.2.1 Autonomous vehicle operation overview

In AVENUE, two levels of control of the AV are distinguished: micro-navigation and macro-navigation. Micro-navigation is fully integrated in the vehicle and implements the road behaviour of the vehicle, while macro-navigation is controlled by the operator running the vehicle and defines the destination and path of the vehicle, as defined the higher view of the overall fleet management.

For micro-navigation, Autonomous Vehicles combine a variety of sensors to perceive their surroundings, such as 3D video, lidar, sonar, GNSS, odometry and other types of sensors. Control software and systems, integrated in the vehicle, fusion and interpret sensors' information to identify the current position of the vehicle, detecting obstacles in the surrounding environment, and choosing the most appropriate reaction of the vehicle, ranging from stopping to bypassing the obstacle, reducing its speed, making a turn etc.

For the macro-navigation, the Autonomous Vehicle receives the information from either the in-vehicle operator (in the current configuration with a fixed path route), or from the remote control service via a dedicated 4G/5G communication channel, for a fleet-managed operation. The fleet management system takes into account all available vehicles in the services area, the passenger request, the operator policies, the street conditions (closed streets) and send route and stop information to the vehicle (route to follow and destination to reach).

### 1.2.2 Autonomous vehicle capabilities in AVENUE

The autonomous vehicles employed in AVENUE fully and autonomously manage the above defined, micro-navigation and road behaviour, in an open street environment. The vehicles are autonomously capable to recognise obstacles (and identify some of them), identify moving and stationary objects, and autonomously decide to bypass them or wait behind them, based on the defined policies. For example, with small changes in its route, the AVENUE shuttle is able to bypass a parked car, whereas it will slow down and follow behind a slowly moving car. The AVENUE vehicles are able to handle different complex road situations like entering and exiting a round-about in the presence of other fast running cars, stopping in zebra crossings, or communicating with infrastructure via V2I interfaces (e.g. red light control).
The shuttles used in the AVENUE project technically can achieve speeds of more than 60km/h. However, this speed cannot be used in the project demonstrators for regulatory and safety reasons. Under current regulations, the maximum authorised speed is 25 or 30km/h, depending on the site. In the current demonstrators, the speed does not exceed 23km/h, with an operational speed of 14 to 18km/h. Another, even more important reason for limiting the vehicle speed is safety of passengers and pedestrians. Due to the fact that current LIDAR systems have a range of 100m and the obstacle identification is done for objects not further than 40 meters, and considering that the vehicle must safely stop in case of an obstacle on the road (which will be "seen" at less than 40 meters distance), we cannot guarantee a safe braking at speeds above 25km/h. Technically, the vehicle could perform a harsh break and stand still within 40 meters at higher speeds (40-50km/h), but then the break process would be very harsh, such that passenger safety could not be guaranteed. The project is working in finding an optimal point between passenger and pedestrian safety.

## 1.3 Preamble

Making autonomous shuttles relevant in the public transportation landscape requires improving quality of service (higher operating speeds, on-demand service) and reducing dependency on human operators (i.e. transition from on-board safety operators to remote monitoring). **This poses serious safety and security challenges, which are the focus of WP6.**

Passengers' and other road users' safety is addressed in tasks 6.1 and 6.2. Task 6.1 aims at assessing safety in a controlled environment (test tracks and simulation), whereas 6.2 concentrates on actual field operations and related hazards. Both tasks are intimately interleaved in a common methodology which is explicated in deliverable *D6.1 First Iteration Methodology for Safety Evaluation*.

Security is addressed in task 6.3 which focuses on making the services provided within AVENUE robust to hacking attempts (i.e. cybersecurity), but also supports development of automatic detection of threats to passengers security through, for instance, automated video processing.

It is worth noting that safety and security are vast domains which can only partially be addressed within the scope of such project. **The activities in WP6 therefore concentrate on threats which are specific to**

**autonomous shuttles**. WP6 aims at supporting operations during the AVENUE project to ensure that current best-practices are applied, but also at improving the state of the art and to **provide advice which can be used in future deployments**. Therefore, WP6 doesn't focus on issues which are very specific to AVENUE (e.g. specific vehicle model used in operations), but rather aims at providing universal findings and recommendations for autonomous shuttles operating in an urban environment.

This deliverable D6.3 describes the methodology for a *controlled* environment safety evaluation and the current state of implementation for task 6.1, notably the implementation of a **scenarios simulation toolchain**.

### Relation to other tasks and deliverables

As previously written, tasks 6.1 and 6.2 are interleaved in a common methodology. This methodology has been thoroughly described in *D6.1 - First Iteration Methodology for Safety Evaluation.* In addition to the overall methodology, this deliverable also contains first results of the *injury risk study*, which provides insights for criteria to be used when evaluating simulation outputs to simultaneously assess risk for passengers and surrounding pedestrians.

# 2 Vehicle Safety Evaluation Methodology

The following section is a reminder of the overall methodology, presented in *D6.1 First Iteration Methodology for Safety Evaluation*, with a focus on the contribution of task 6.1 to this global approach.

## 2.1 Overview

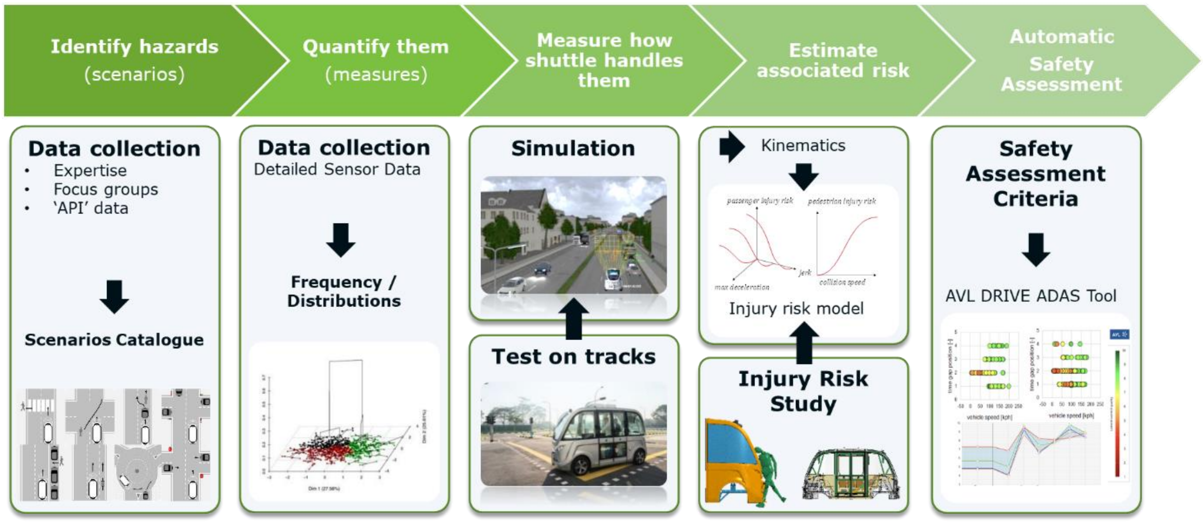
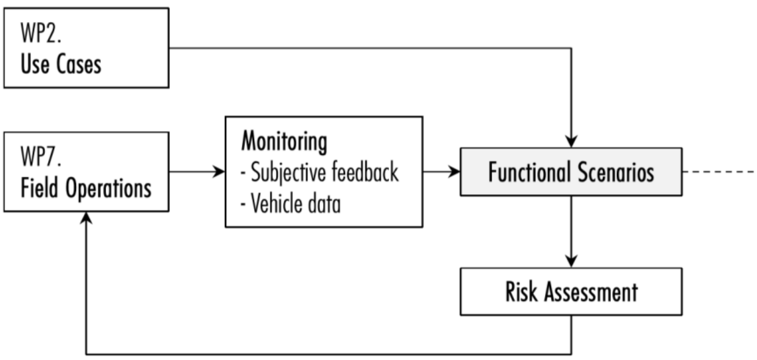



**Figure 1. T6.1 & 6.2 Safety Evaluation**

The methodology presented here was developed during the first months of the project. It relies on multiple skills (objective and subjective data collection and analysis, safety critical scenarios definition and categorization, injury risk assessment, computer simulations…), which are brought by WP6's partners. More specifically:

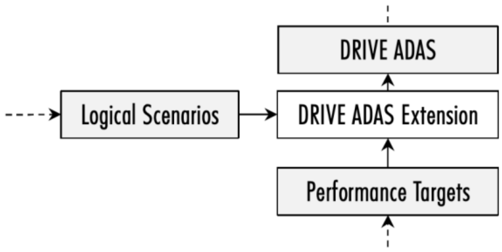
- Both subjective and objective data are collected from the test sites (WP7) and combined with use cases (i.e. future plans, WP2) to identify safety relevant scenarios. Based on those scenarios, a preliminary safety assessment will be carried out.



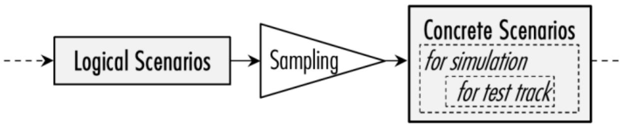

- Safety-relevant scenarios which are specific to autonomous shuttles will be selected and further described, in a quantitative way (i.e. by measurable parameters and their possible range and/or distribution).


- An injury risk study, taking into account the geometry of autonomous shuttles has been carried out. It delivered risk functions based on the most important parameters (e.g. passengers injury risk during a braking, based on their position and deceleration profile).

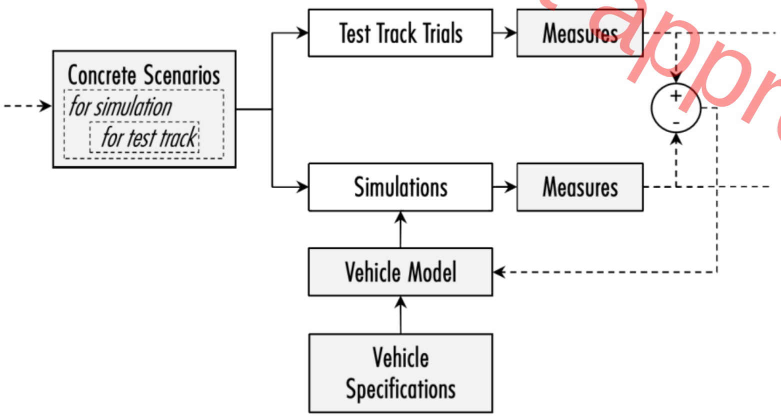

- Relevant scenarios need to be detected and the associated Key Performance Indicators need to be compared to Performance Targets. To this end, AVL's software AVL-DRIVE AD will be extended.



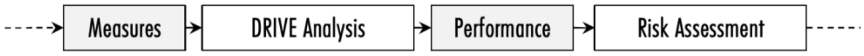

- Some instances of the relevant scenarios will be sampled (i.e. parameters values will be fixed), either to sweep the entire parameter space and build a representative set for a comprehensive risk estimation, or to explore boundary conditions (i.e. conditions where the desired outcome is known, e.g. "avoid hurting any pedestrian that would run in front of the vehicle within a 10m headway or more").




- Those scenarios will be simulated, and some of them reproduced, if possible, on a test track, to improve the vehicle model used in the simulation.




- Results from those controlled environment tests and simulations will be run through AVL-DRIVE AD to provide a refined safety assessment.




The complete process is summarised in Figure 2 below, which also delineate tasks 6.1 and 6.2 perimeters.

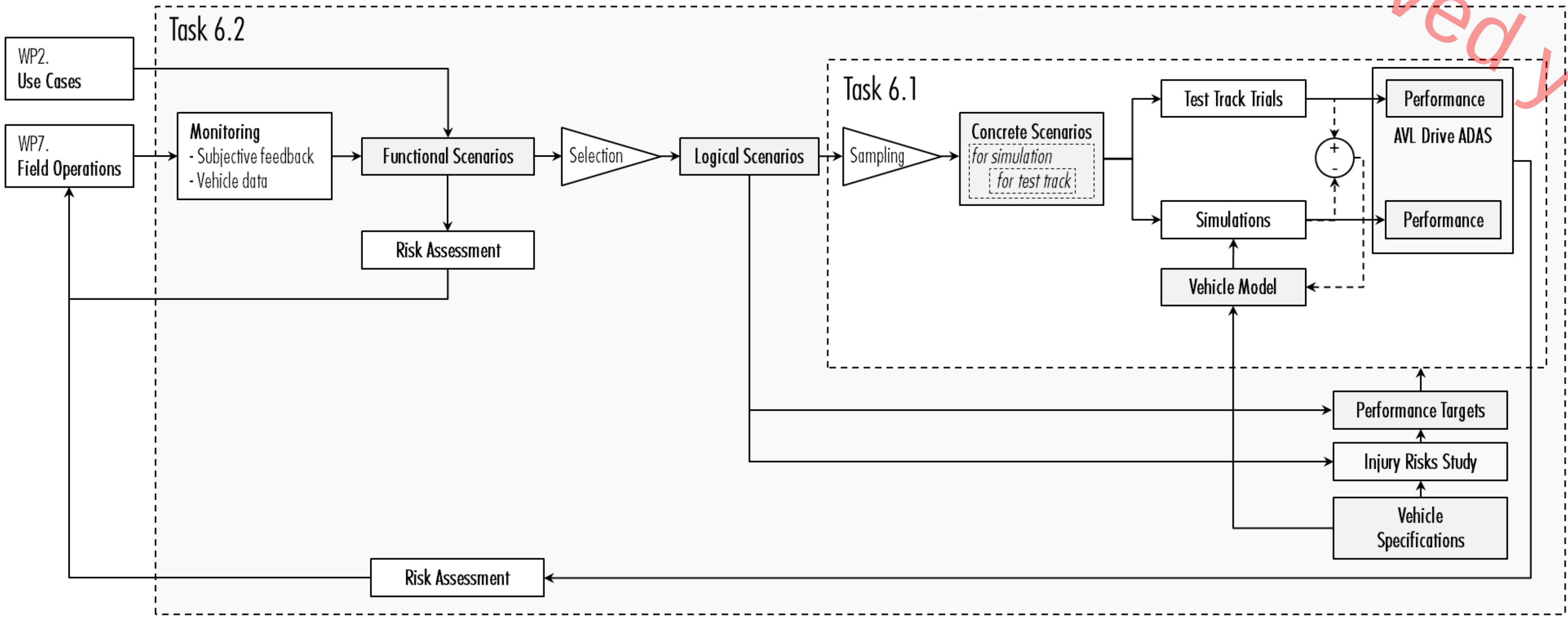



**Figure 2. Tasks 6.1 and 6.2 methodology and interactions**

### 2.1.1 Scope of Task 6.1

As previously described, Task 6.1 is responsible of the Vehicle Safety Evaluation based on the logical scenarios provided by Task 6.2:

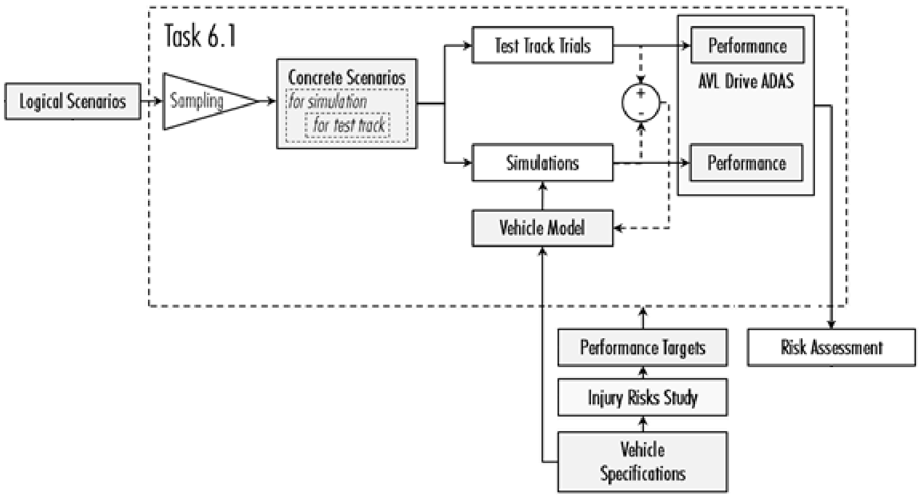



**Figure 3. Task 6.1 scope**

In order to perform this risk assessment, first a conversion of logical scenarios into concrete scenarios will be done with a definition of all variation and parameters we are considering as critical.
Thus, a controlled environment is defined, either in simulation or at the proving ground, in order to implement a safety evaluation in a way which can be applicable before the deployment of shuttle operation.

#### 2.1.1.1 Simulation

Simulation activities were carried out to perform quantitative safety evaluation in a number of relevant traffic scenarios. Those simulations rely on a virtual environment including infrastructure geometry, traffic lights, other road actors, and various weather conditions, among others.

Several approaches to developing a simulation environment were carried out, working towards the following goals:

- Being able to simulate complex traffic scenarios, initially provided in the form of logical scenarios[1] ;
- Being able to vary parameters defining each of those logical scenarios over a wide range of possible values ;
- Creating a simplified/generic vehicle model, which not only can be tuned to match current vehicle characteristics to an acceptable level, but can also be used to explore potential evolutions such as, e.g., sensor locations.

[1] D6.1 thoroughly defines *functional*, *logical* and *concrete* scenarios

- Implementing an environment where a much more detailed and specific vehicle model *could* also be used, i.e. provide a virtual and realistic proving ground.

Initial implementation efforts, carried out by VIF, relied on CARLA, an open-source simulation tool under development. Despite considerable improvements made within the course of AVENUE so far, CARLA proved not to be mature enough to reach the aforementioned goals.

As a consequence, AVL made some additional efforts to provide the WP6 partners with an appropriate simulation toolchain based on proprietary software which is still compatible with the original scenarios implemented by VIF to arrive at a sustainable solution. Details are given in chapter 4.

Simulations produce outputs such as speed, acceleration, relative position to obstacles, in a wide variety of scenarios. Those outputs will then be given as inputs to AVL-DRIVE AD, which will calculate KPIs and compare them with performance targets, derived from Task 6.2's injury risk study, allowing:

- Public transport operators to assess whether the current capabilities of the vehicle are compatible with their plans;
- Vehicle manufacturers to experiment with potential evolutions of their product, to allow new use cases to be addressed.

#### 2.1.1.2 Real tests

To claim any credibility, simulations need to be confronted to (and then tuned to match) real situations. Test tracks allow observing the reactions of the actual vehicle to events which can be produced without any safety concern using dedicated infrastructure and dummies and/or soft targets to represent vulnerable road users and surrounding vehicles.

As a limited amount of proving ground tests are planned in WP6.1, those will focus on the most critical scenarios in order to verify the compliance of simulations with actual vehicle behaviour in similar situations.

This report thus focuses on **simulations**, as proving ground tests could not be carried out yet. It first introduces the general architecture of a traffic scenario simulation solution, and then discusses the various implementations which have been made.

The approach based on the AVL tool chain is flexible and generic. To carry out simulations of the two use cases considered, data needed are as follow:

- Sensor's data to adapt the field of view, range, latency
- Kinematic and dynamic data to calibrate vehicle behavior: lateral and longitudinal controllers
- Decision making module
- Environment

To tune the controllers, proving ground trials describing both the deceleration and braking behavior and lateral behavior of the shuttle are requested.

# 3 Simulation environment architecture

Within the context of Task 6.1, simulations aim at **confronting the autonomous shuttle – and its characteristics – to specific situations**, and measure the resulting kinematics (position, speed, acceleration). Those kinematics properties will then be used to assess whether the simulated situations are handled by the vehicle in a safe way.

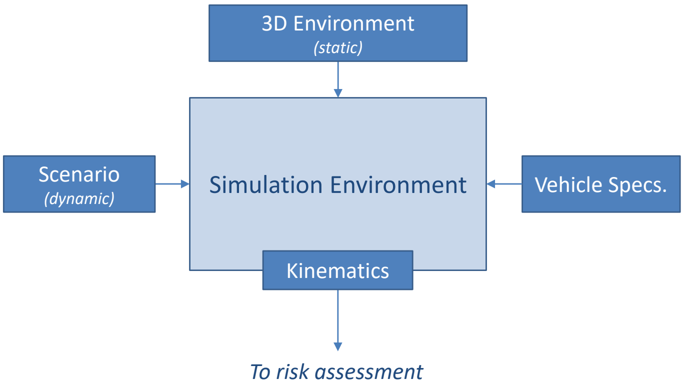



**Figure 4. Scenarios simulation scope**

## 3.1 Simulation Inputs

The inputs of a simulation therefore are:

- **The vehicle specifications**, which include sensors location and performance, perception and control software behaviour, but also physical characteristics such as weight and braking performance;
- **A scenario**, which describes the intended manoeuver of the autonomous vehicle, but also the behaviour of surrounding actors (e.g. other vehicles, pedestrians) or more generally, the dynamic content of the simulation which is not under the autonomous vehicle's direct control. This may also include, for instance, traffic lights states and weather. Scenarios are provided to task 6.1 as *logical scenarios.* Those consist in high level descriptions of vehicle and other surrounding actors manoeuvers, identification of parameters contributing to the scenario outcome (i.e. initial positions, manoeuvers characteristics, static obstacles or visual occlusions), and possible values or distributions of those parameters. As one simulation corresponds to one specific set of parameters values, i.e. to a *concrete scenario*, sampling from logical scenarios and translation to a concrete scenario description file format must be performed prior to simulation.
- **A virtual environment**, which describes the static properties (e.g. geometry) of surroundings.

All those inputs have important impact on the simulation outcome and may be independently changed and combined to study the contribution of various factors to the resulting performance.

# 3.2 Internals

The simulation environment itself consists of various components:

- A **3D environment simulation tool**, which places and moves the different actors of the scenario in the chosen virtual environment. It includes **virtual sensors** which generate streams of data based on the relative positions of objects and sensors' own properties. Those virtual sensors may simulate complex physical phenomena to generate realistic raw data, relying on fine simulation of external conditions such as e.g. weather. In such case, perception algorithms themselves can be included in the simulation and tested. They may also and otherwise generate higher level data, bypassing the fine simulation of perception to directly provide detected objects based on simpler properties (field of view, range…).
- A **controller**, which generates steering, acceleration and braking commands based on the sensors' streams. This controller may be the actual software embedded in a specific vehicle under study, in which case we will describe the process as a "*Software-in-the-Loop" (SiL)* simulation. It may also be a simplified version with a simpler parameter set to ease wide range parametric studies.
- A **dynamic model of the vehicle**, which models the way that the vehicle reacts to the commands provided by the controller, depending on its mechanical characteristics (e.g. weight). This model therefore takes the controller commands as inputs and returns the motion characteristics (i.e. kinematics) of the vehicle under study. These kinematics are the desired output of the whole simulation, but are also fed back to the 3D environment simulation tool to properly move the vehicle in the environment. This continuously changes its perspective of the simulated world, and the resulting sensors streams, as the simulation goes on.
- Finally, as this architecture requires various software modules to communicate together as a closed-loop system, some kind of **communication framework** is usually required.

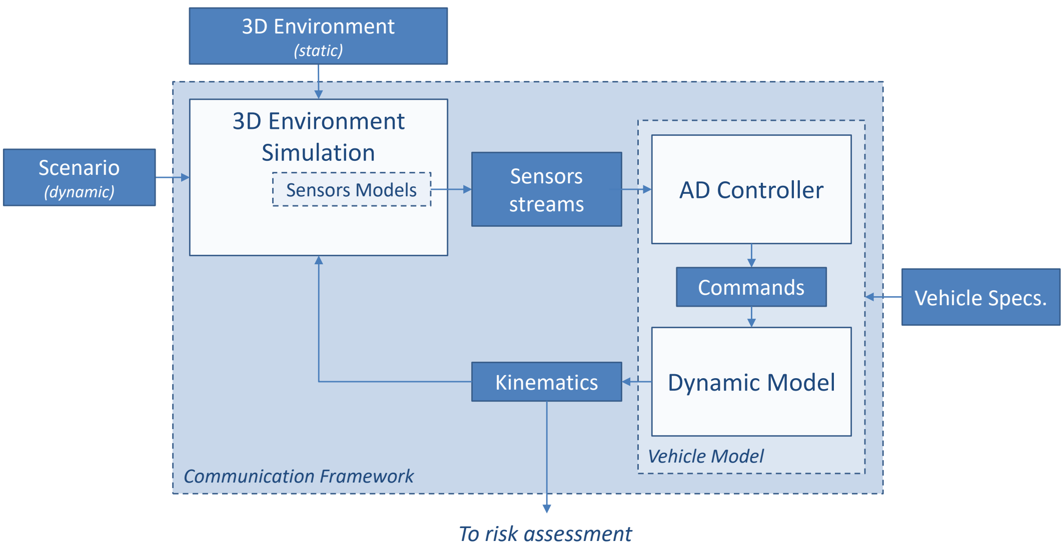



**Figure 5. Simulation environment components**

As the aim in AVENUE isn't to provide fine-tuned simulations reproducing very specific vehicle behaviour under very specific conditions, but rather to identify the sensitivity of safety to the main characteristics

(e.g. sensors amount and location, detection latency) of a potentially not-yet-created vehicle, “ideal sensors” providing high level detected objects and simplified controller logic were favoured.
Two successive approaches to implement a toolchain corresponding to the aforementioned architecture were experimented. The first one was built around the CARLA simulation tool, an open source tool under continuous development, and the other one around VTD, an established commercial package.
We will present both approaches, including their strength and limitations.

## 3.3 Simulation Outputs and Risk Assessment

The outputs of the simulations are the **kinematics**, i.e. position, speed and acceleration profiles of the shuttle and other interacting actors. Those will be used to assess the risk associated with each simulation, based on criteria defined from Task 6.2 findings.

AVENUE’s WP6 investigations concentrate on:

- Discomfort and injury risk for passengers in case of harsh braking, which are characterised by their deceleration profiles;
- Injury risk for vulnerable road users (starting with pedestrians, expanding to two-wheelers if resources allow) in case of collision, identified by relative position and speed of the shuttle and the vulnerable road users.

Key Performance Indicators (KPIs) such as discomfort or head injury risks, estimated from Task 6.2’s risk curves, will be implemented in the AVL-DRIVE AD tool to be calculated from those kinematics.

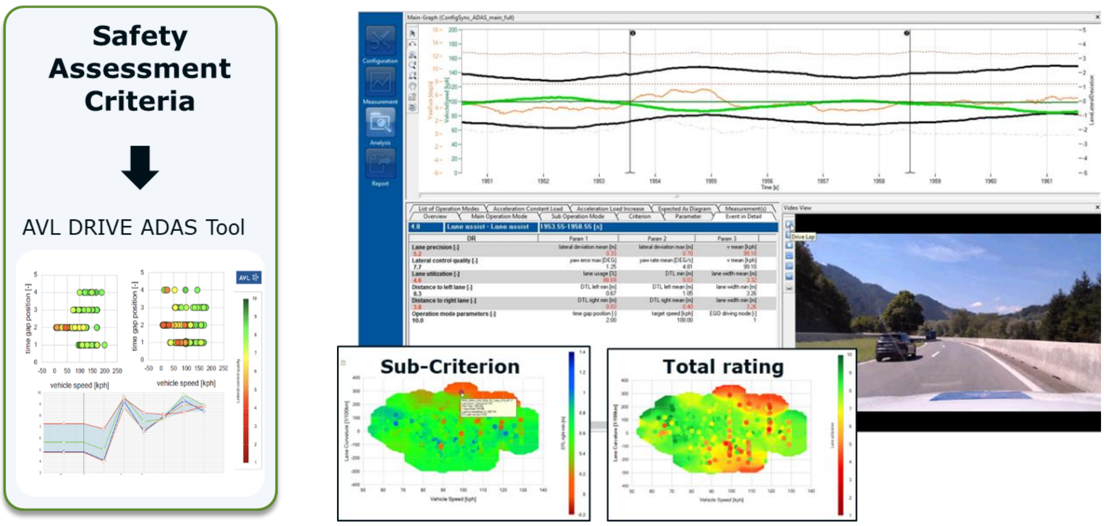



**Figure 6.** AVL-DRIVE AD **Safety Evaluation**

Determining whether the result of a simulation is acceptable or not, requires defining *performance targets*. Those correspond to thresholds on the aforementioned KPIs.

As defining performance targets might require accepting a certain level of risk to allow the implementation of the service, their acceptance level within the project will be under the responsibility of public transport operators.

# 4 Toolchain implementation

The previous chapter introduced general notions about traffic scenarios simulation, describing what was expected from them, and a typical toolchain architecture which would address such needs. We will here describe here what has actually been implemented, to address the specific needs and constrains of the AVENUE project.

## 4.1 CARLA implementation

The first implementation attempt was based around the CARLA simulation tool[2], which is a free and open source simulation platform for development, training, and validation of autonomous driving system. It is based on the Epic Unreal videogame engine, which provides the technical underpinnings to create realistic and lively virtual 3D worlds. It implements virtual sensors representing various technologies (LIDAR, computer vision, more recently radars) and provides an open API, with which multiple clients can, in real time, read information from the virtual world and impose actions to its various actors. CARLA also provides assets which can readily be used such as multiple virtual environments, including urban layouts, and a multitude of vehicle models, buildings, pedestrians, street signs, etc. to populate them. Besides that, CARLA offers a wide range of environmental conditions, including weather and time of day. CARLA continuously evolves thanks to a very active ecosystem, which not only improves the simulation core, but also develops multiple tools which can interface with it to answer specific needs.

**Figure 7. (a) Map of** CARLA **town, (b)** CARLA**'s 3D environment**

[2] https://carla.org/

An initial implementation for AVENUE was made using readily available assets. This implementation had the following characteristics:

- **Scenarios** were specified in the **OpenScenario** format, which is a scenario description standard currently under development[3]. This standard provides the data model and a file format specification for the description of dynamic content in driving simulation. The data model is structured around the concept of a *storyboard*, which is subdivided in *stories*, *acts* and *sequences*. A story can describe the driving manoeuvres of one single vehicle or several actors, potentially interacting with each other. Stories consist of acts, which are triggered when specific conditions are met. The detailed driving behaviour of an actor is described via *events* (i.e. when does it happen?) and *actions* (i.e. what happens?). Actions may be related to one vehicle and can include speed changes, lane changes or instruction to drive to a specified position. Actions may also be related to the environment and can include the change of a traffic light or the occurrence of a traffic jam. The file format itself follows the XML standard, with a specific schema, corresponding to the possibilities of the OpenScenario data model. The file extension is *.xosc*.
- The **virtual environments** were picked from the ‘towns’ (i.e. urban) environments provided with the CARLA distribution.
- The **3D Simulation Tool** of course was CARLA, which had to be complemented with dedicated software to improve OpenScenario support. OpenScenario itself still is under development and therefore is a moving target. Implementation in CARLA when its use in AVENUE started was still in its infancy. To improve support, the CARLA scenario loader was created in the context of a master’s thesis at TU Graz[4].
- The **controller** was the ‘autopilot’ also readily available in CARLA. This controller follows waypoints and generates commands based on sensor streams in CARLA. Its behaviour cannot be controlled for it to match the behaviour of a specific autonomous vehicle, and it can only run inside the CARLA environment.
- The **dynamic model of the vehicle** was a VW T2 from the standard CARLA assets, which we hoped would have vehicle dynamics sufficiently similar to those of an autonomous shuttle.

**Figure 8. Scenario in CARLA with VW T2 and autopilot mode on**

[3] https://www.asam.net/standards/detail/openscenario/

[4] C. Pilz (2019). Master’s thesis at Graz university of technology. Development of a scenario simulation platform to support autonomous driving verification. [Online]. Available: https://github.com/MrMushroom/CarlaScenarioLoader/blob/master/oneside_final.pdf.

This approach allowed running some of the desired scenarios in a short timeframe. It presented however many limitations, which proved hard to overcome:

- CARLA's OpenScenario support, although improving, still is limited for some aspects (e.g. driving backward, moving laterally), which prevents some relevant scenarios to be implemented.
- Virtual towns provided in CARLA can't easily be modified so that their characteristics would match, to an acceptable level, the kind of infrastructure that AVENUE's autonomous shuttles evolve into. Creating new virtual environments dedicated to AVENUE assessment would also be too challenging.
- Replacing CARLA's autopilot by a dedicated controller is possible, but again, although promising solutions seem to emerge in the form of a bridge with ROS[5], allowing using, for instance, a customised version of Autoware[6], the effort would be too important for the project.
- Finally, creating a physical model really corresponding to the shuttle's main characteristics would require lots of efforts, for lack of efficient tooling.

Ultimately, due to all the limitations, it was decided **not** to run final simulations in CARLA, but to use it as a test bed to develop scenarios in OpenScenario format, for them to be used in a new toolchain based around Vires VTD.

This platform, as described below, allows to deal with critical use cases using an accurate dynamic vehicle model and a flexible interface to integrate any AD functions or to run in direct coupling with any other tools.

Therefore, the effort of VIF is directed towards scenario development in the OpenScenario standard, which although not entirely mature, seems to be future-proof. Around 40 out of 60 selected scenarios have been implemented. The rest of them will be implemented once the functionalities required for their testing in CARLA are available, which is expected in the upcoming version 1.0 of the software.

[5] ROS: Robot Operating System. https://www.ros.org/
[6] https://www.autoware.org/

## 4.2 VTD based toolchain

Due to the difficulties met in implementing a toolchain based around CARLA which would meet the requirements of the task, a new approach was favoured. It resulted in the implementation of a modular toolchain, built around the proprietary Vires VTD (Virtual Test Drive) simulation tool.

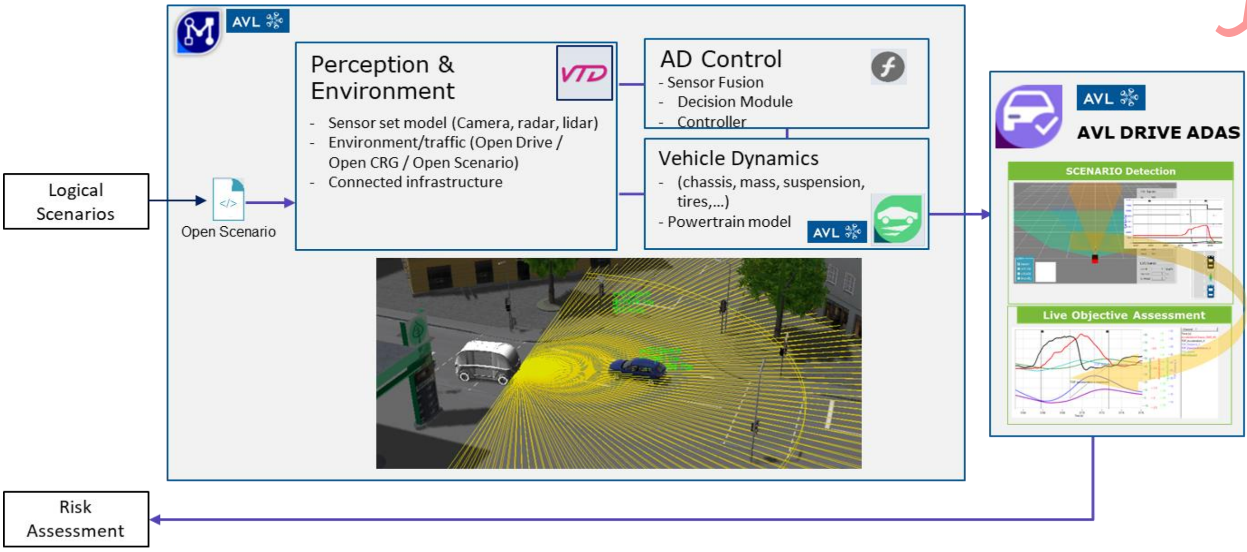



**Figure 9. Schematic representation of the toolchain**

The toolchain is composed by many modules, each performing a specific role in the simulation context, and communicating based on an open co-simulation platform, **AVL Model.CONNECT**:

- The **virtual environments** used in the simulation can be created using **Road Designer ROD**, which provides an interactive road network editor with extensive libraries of 3d objects, textures etc. that can be used to create a realistic 3D environment.
- The **3D Simulation Tool** is **Vires VTD**. It models the environment (roads, weather, etc.) and the other actors of the simulation (vehicles, pedestrians, etc.), as well as the vehicle's sensors. It also generates the 3D image visualization of the simulation.
- The **Controller**, i.e. the algorithm responsible for computing the vehicle's actions, reading sensor information, running decision-making and control algorithms, and outputting the required signals for the actuators, is an AVL made software consisting of a longitudinal controller and a lateral controller, and is described further below.
  NOTA: Due to lack of access to Navya vehicle's internal implementation, the controller designed to perform T6.1 activities will follow a vehicle trajectory based on virtual lanes, which is not directly the same concept than used by Navya shuttle, focusing more on the map than on lanes.
- The **dynamic model of the vehicle** is implemented in **AVL VSM**. It models the vehicle and its dynamics, calculates the response to the actuators (pedals, steering wheel) and outputs the vehicle state (speed, position, etc.).

Model.CONNECT couples all the above-mentioned tools into one single environment and manages the interaction between them. It provides an interface for configuring the simulation settings and parameters, and for visualizing the results.

As already discussed, AVL-DRIVE AD monitors the various simulation variables generated by the other programs in order to evaluate the driving performance (in terms of safety, comfort, etc.) and attribute a score to the events during simulation (lane keeping, cut-ins, etc.).

The aim of this toolchain is first to allow virtual testing of a wide range of situations, but it can also be used with real data measurement. As a matter of fact, the safety evaluation tool can also be connected to the real vehicle to allow performing "Model-in-the-Loop (MiL)", as shown on Figure 9, or Vehicle-in-the-Loop (ViL) testing, as seen below.

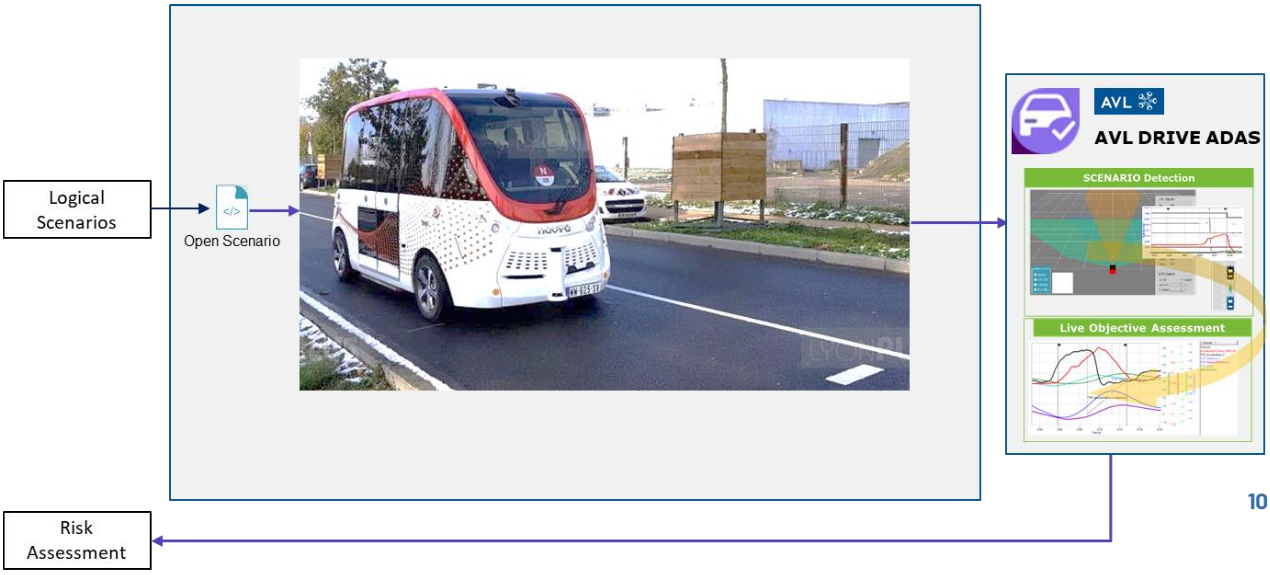



**Figure 10. Vehicle-in-the-Loop assessment**

The following sections describe how those tools were configured and used to implement realistic simulations of an autonomous shuttle's behaviour.

### 4.2.1 Importing scenarios developed with CARLA

Scenario files generated in OpenScenario format (.*xosc*) by CARLA Simulator unfortunately cannot be directly read by VTD to be used in the toolchain, because both CARLA's and VTD's support for this format is still limited. They both support their own subset of the standard which don't entirely overlap. Some compromises can be made, tough, so that those scenarios can be simulated in the toolchain.

The first main incompatibility is the lack of an OpenSceneGraphics file (.*osgb*) in CARLA scenarios. This file contains the 3D database for the image generator. CARLA only provides the OpenDrive file (.*xodr*) with the road description, while VTD needs both these files. By using Road Designer (ROD) by VTD, one can generate an empty .*osgb* for a given .*xodr*. It will contain only the road and no landscape, but it can nevertheless be used in VTD. If a landscape is needed, the user must manually add it to every road segment, for example by applying macros.

**Figure 11. Adding landscape in ROD**

The second main incompatibility is the actions sequence. Since actions in CARLA scenarios are not interpreted by VTD, it is necessary that the user exporting the CARLA scenario describe all the players' actions and their sequencing in another file (e.g. a text file or a spreadsheet) so that the user importing it in VTD can read it and replicate those actions in order to obtain the same sequencing. Some conditions are not supported by VTD (e.g. the beginning of an action being triggered by the end of another), which means that some adaptations must be made to mimic the original behaviour (e.g. calculating the triggering action's duration and using it as the start delay for the triggered one).

Once these steps are accomplished, the exported scenario can then be used in the toolchain (possibly with compromises) and the simulations can be performed.

CARLA Simulator provides four example scenarios in OpenDrive format, two of which have been already exported to VTD through this process.

## 4.2.2 Infrastructure environment

The environment in VTD's scenarios (e.g. players, infrastructure) can be defined according to the user's needs to meet the requirements of each use case. Players such as vehicles and pedestrians can be positioned anywhere in the scenario and their movement can be controlled through actions, such as speed variation, lane change, etc. Traffic signs can also be added and can be detected by the available sensors. It is also possible to add traffic lights to the intersections, with which the vehicle under simulation can interact. Traffic lights can be configured so as to be synchronized with each other, and the timing of each semaphore phase can be individually set.

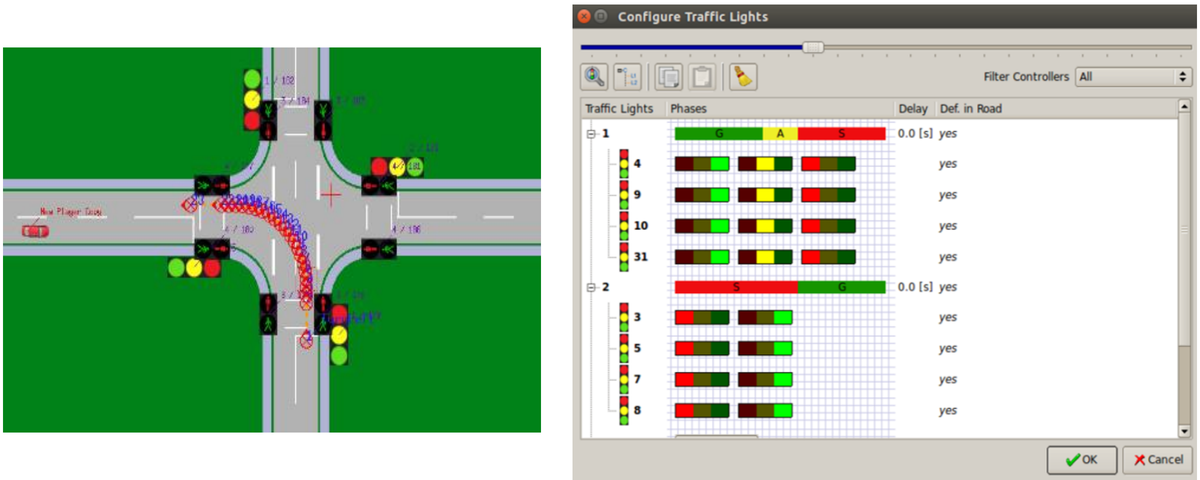



**Figure 12. Configuration of traffic lights in** VTD

## 4.2.3 Vehicle model

The representation of the vehicle in the toolchain is decomposed in three main parts: mechanical description, sensors, and controllers. The mechanical description is made in a VSM project which is then imported to Model.CONNECT. The sensors are defined in VTD's configuration files. Controllers can be made in many ways and can be defined either directly in Model.CONNECT or in external software imported into the model.

The vehicle model was as close as possible to NAVYA vehicle based on static data provided by NAVYA (mass, dimensions, tire model, …).

After D6.3, further calibration of the controllers based on dynamic measurements has been performed. No information was available to tun the decision making.

The chosen simulation scenario was artificially created, but considering the real-world situation best possible (e.g., street dimensions, object positions, …). The first use case represents an actual situation observes by KEOLIS at Lyon and the second use case represents the most challenging actual situation ranked by the operators.

### 4.2.3.1 Mechanical description (chassis)

All vehicle dynamics parameters, such as aerodynamics, powertrain description, tires' constants, etc. are defined externally in a VSM project. Multiple profiles can be defined for each parameter set in the same project.

As an example, the dynamic modelling of the Navya shuttle was done by inserting the information from the vehicle's specification into VSM's configuration windows, as shown below:

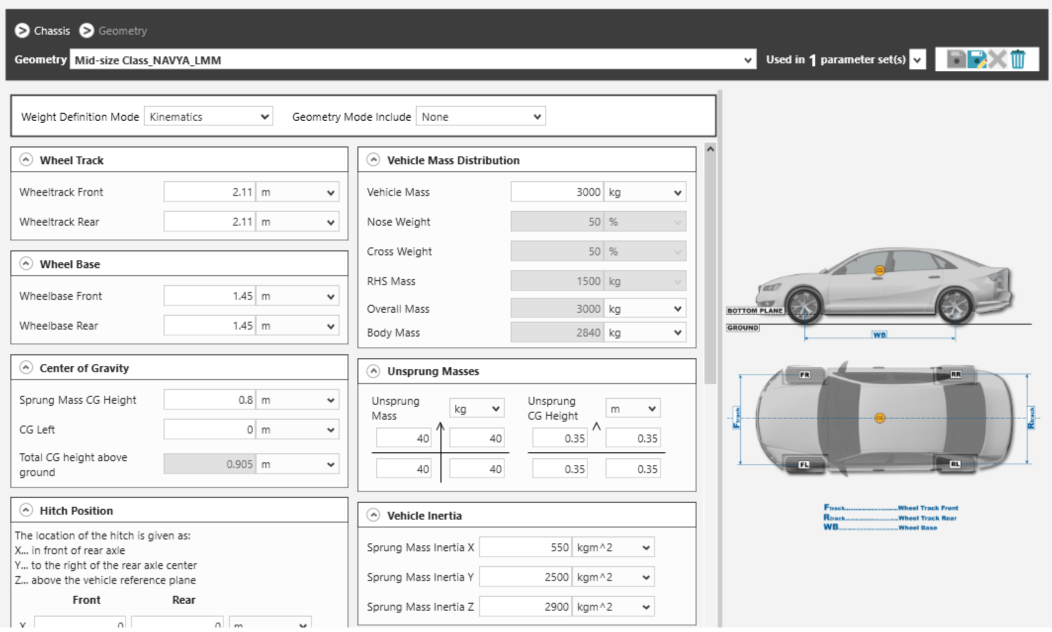



**Figure 13. Configuration of the shuttle's geometrical parameters in VSM**

Some of the shuttle's modelling parameters were directly available in the specification, while others needed further calculations in order to be obtained. Some of the information was arbitrarily defined, such as the steering wheel's maximum angle, since the shuttle doesn't have a steering wheel. Other data such as the wheels' maximum steer angle were derived from the minimum turning radius provided in the specification and the shuttle's geometric properties, by means of a bicycle model.

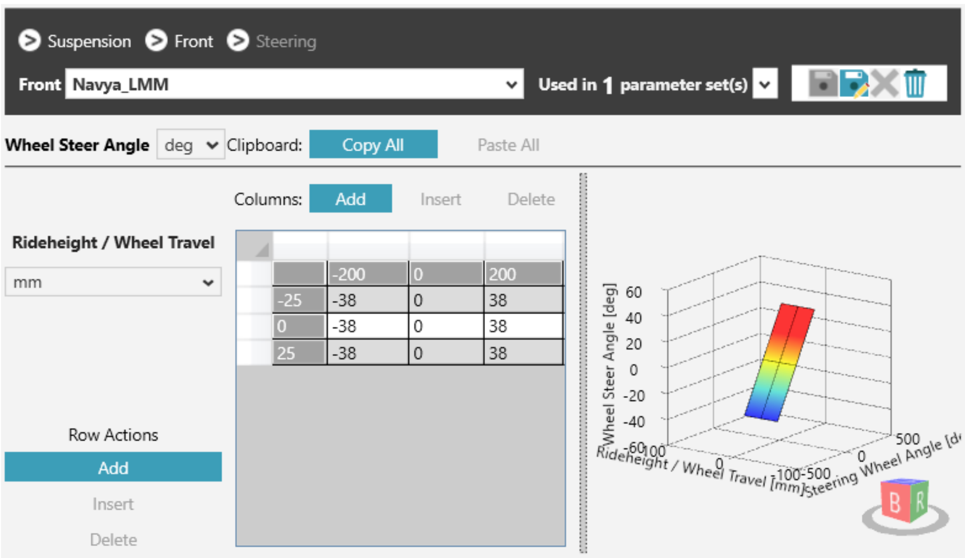



**Figure 14. Configuration of steering parameters in VSM**

Once the project file is exported into Model.CONNECT, these configurations become available for the user to select and all its parameters are available. They can also be modified by the user if and when needed.

### 4.2.3.2 Sensors

VTD provides multiple sensor plugins, which retrieve the environment information and send what they perceive to Model.CONNECT through a TCP port. Among the available sensors is the *PerfectSensor*, which provides accurate information on objects such as vehicles, pedestrians, cyclists, road marks, traffic signs, traffic lights and obstacles in general. All of the sensors' parameters (e.g. position, range, and field of view) can be configured in VTD's configuration files. Other sensors can also be configured, such as radars and LIDARs, which provide more faithful physical modelling and therefore better representation of a specific technology's limitations. Additionally, custom sensors can also be developed by the user if needed.

For the Navya shuttle, two sets of four sensors each are used, one on each side of the vehicle. Each set consists of two 180-degree LiDAR sensors on the bottom, one 360-degree LiDAR on top of the vehicle and one camera. Since the intent is to base the evaluation on a high-level description of the system's performance, and not to validate very specific perception algorithms, those were all modelled as *PerfectSensors* in VTD, and configured according to the information (range, field of view, etc.) provided in the specifications.

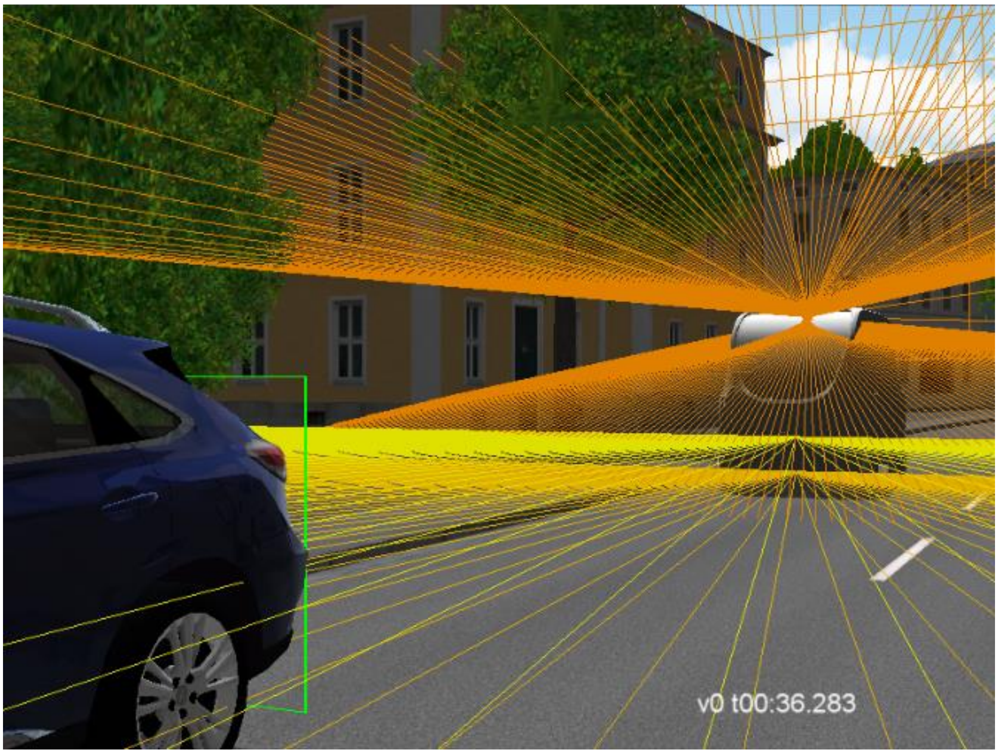



**Figure 15. Navya's LIDAR sensors in** VTD

### 4.2.3.3 Controller

The precise behaviour of Navya's controller is unknown. It is our understanding that it relies primarily on real-time kinematics GNSS for location, which under good conditions (i.e. proper reception of real-time corrections from the base station) provides centimetric accuracy. It is unknown to us whether this is complemented by a SLAM algorithm relying on sensors and high definition map data.

The path to follow and target speed in each section are defined when deploying the shuttle in a new site, i.e. the shuttle follows 'virtual rails' at a predefined speed. Objects detected by the sensors are fused together to obtain a 360° view with a range of about 40m. The sensors' modelling approach we chose, taking into account their location and specs, allows reproducing blind spots. The shuttle's speed and in some cases its trajectory are corrected in real-time to account for surrounding objects. The system is helped for some use cases with V2X communication, to, for instance, synchronise traffic lights around an intersection with the shuttle.

Our objectives when implementing a controller were:

- To validate the complete simulation toolchain;
- To implement a behaviour similar to what we just described, in a way which would allow tuning its main parameters;
- To allow replacing this first controller at a later stage with the actual vehicle controller, should it be desired.

Since we use *PerfectSensors*, perception is already dealt with and considered to be perfect. Only the *decision* layer therefore is implemented in the controller. It consists of a *longitudinal* controller and a *lateral* controller, which jointly, generate the appropriate commands on the vehicle's controls. Those are described below.

All functions of decision making, and controllers are developed during the project as no input was available. As AVL tool chain is modular, the interface is open to integrate any more realistic vehicle intelligence and then replace the AVL controllers and decision-making module used.

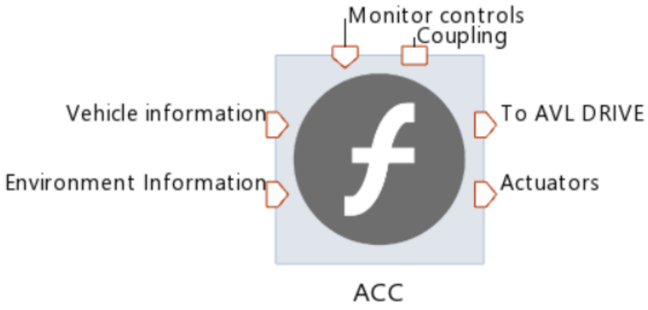
In this report, a PID controller is used for the longitudinal behavior and a predefined path crossing algorithm for the lateral controller. The target was not to evaluate a specific AD function, but to validate the overall methodology.

#### 4.2.3.3.1 Longitudinal controller

The longitudinal controller developed for the toolchain controls the longitudinal speed, and is capable of keeping a desired distance in time (time gap) between the ego vehicle (the autonomous vehicle considered) and the TOF (Target Object Front, e.g. a preceding vehicle) while following a predefined path. This path, used as railway, is kept by the lateral controller.

The controller has four sets of inputs:

- Environment information: Data coming from VTD regarding the ego vehicle's front sensor, such as TOF speed and distance.
- Vehicle information: Data from VSM, such as ego speed.
- Monitor controls: time gap and speed setpoints and controller gains
- Coupling: information from the lateral controller, such as maximum speed.




The controller was implemented using Model.CONNECT's function block, which allows for a C-like programming of its behaviour. It has two modes of operation: one when a TOF vehicle is within range and another for free lane.

- When no target vehicle is ahead, the controller behaves by regulating the ego vehicle's speed to the desired setpoint. It directly calculates the values for the acceleration and brake pedals, based on the error between current and target speed.
- When a vehicle is present within the sensor's range, the controller calculates the desired distance between the two vehicles (based on the time gap setpoint and current ego speed) and compares it to the measured distance, calculating their difference. The relative speed between both vehicles (whose desired value is always zero) is also computed. Then, the distance error, its integral and the relative speed are multiplied by constants (gains) and those three components are added, resulting in the desired relative speed between the vehicles. This relative speed is then added to the current speed, resulting in a target speed value which is then sent to VSM's longitudinal controller. A simplified diagram is found below.

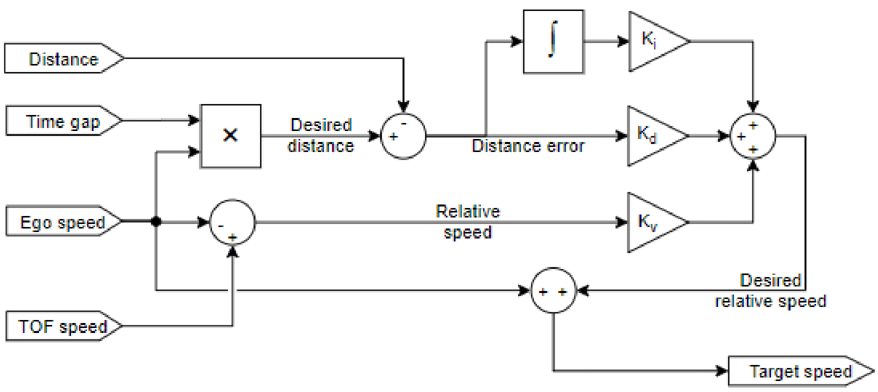



**Figure 16. Longitudinal controller diagram**

Other features are also included in the controller, such as a minimum distance threshold and a capping of the error to limit acceleration and deceleration values.

To validate the controller on the largest dynamic range achievable by the shuttle, a use case has been created in which the ego vehicle could run until 60 km/h when it encounters a standing vehicle ahead, which causes the ego to perform a full brake. Afterwards, the TOF vehicle starts accelerating at 0.5 m/s² for about 500 m, after which it starts to brake with a deceleration of -4 m/s² until a full stop is reached. The performance of the controller's response can be observed in the figure below, which represents the acceleration and brake phase. It can be noticed that the time gap (in green) remains constant during the whole period.

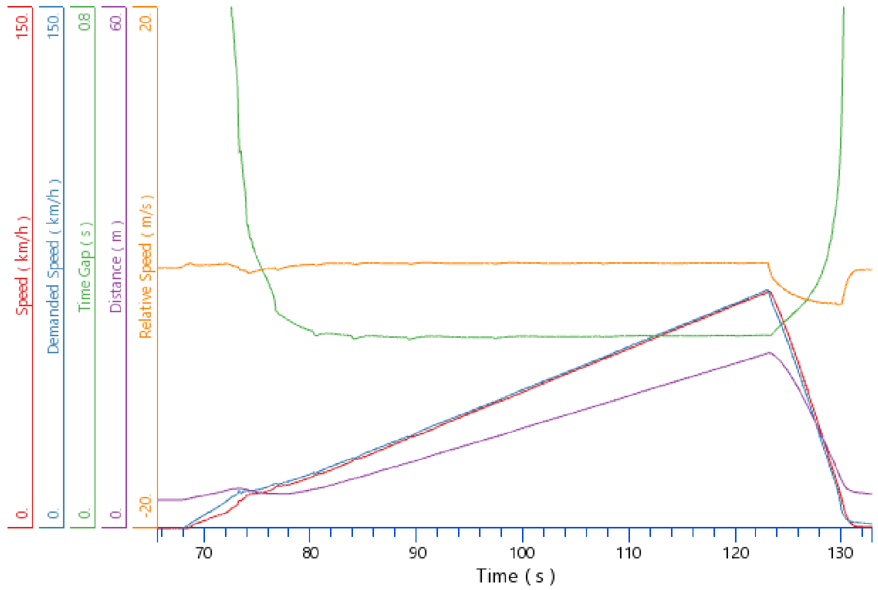


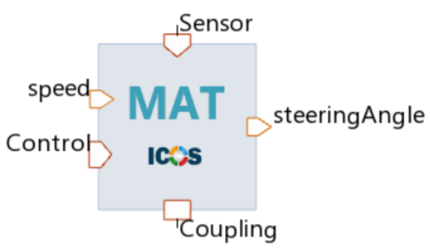

**Figure 17. Longitudinal controller's response**

#### 4.2.3.3.2 Lateral controller

The lateral controller is responsible for keeping the vehicle aligned with the road and in the center of the lane. It is also capable of performing lane changes. It reads the road mark information through VTD's sensors or from map and RTK, and calculates the appropriate angle for the steering wheel so as to keep the vehicle centered.

Like the longitudinal controller, this lateral controller can be replaced by any other if there is a correspondence between their interface signals, those being:

- Sensor (input): road mark information from VTD
- Control (input): target lane and controller gains
- Coupling (in/out): interface with longitudinal controller, such as maximum speed
- Speed (input): ego speed from VSM
- Steering angle (output): actuator to VSM




The implementation of this controller was made with Simulink, using Model.CONNECT's ICOS interface.

The sensor information consists of unordered array of mark information such as lateral distance relative to the vehicle, relative yaw angle, curvature and curvature derivative. The algorithm (block *MarkBuilder*), then groups the information in all those arrays into different mark objects, each with its own individual information.

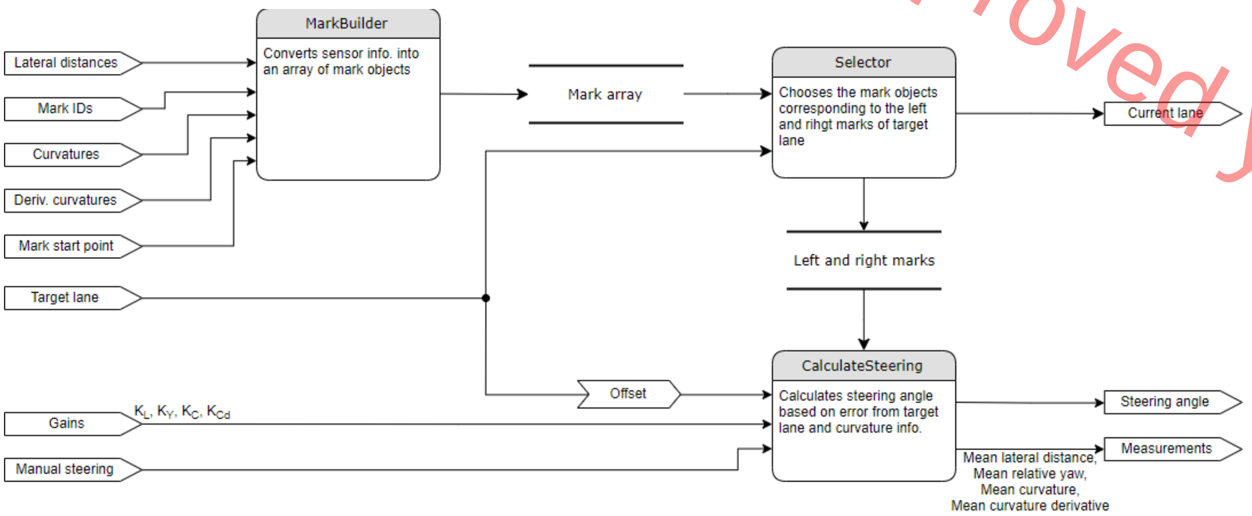



**Figure 18. Diagram of lateral controller**

If the map and RTK could be provided, the predefined path can be used to create *virtual* lines and use them as inputs for the lateral controller.

These mark objects are then filtered (by block *Selector*): only those corresponding to mark segments adjacent to the vehicle are kept, while those relative to segments further away in the road are discarded. They are also sorted from right to left and a continuity algorithm computes the current lane number based on current and past mark information. Then, the two marks (left and right) adjacent to the target lane are selected and sent to the *CalculateSteering* block, which calculates the errors such as lateral error relative to the desired offset from the center of the lane (usually zero) and angular error relative to the desired angle between vehicle and road (always zero, that is, a complete alignment).

Both errors are multiplied by their respective gains, $K_d$ and $K_y$. The curvature and its derivative are also multiplied by their gains $K_c$ and $K_{cd}$. These four components are then added to give the desired angular speed, which is in turn converted to a steering wheel angle based on the current linear speed and a regression model. This angle is then delivered as the output to VSM.

In case one of the two marks is not detected, the controller is still able to operate well by using the past information of the lane's width to reconstruct the missing mark. If both marks are lost, the *ManualSteering* input allows the user to directly control the steering angle in order to reposition the vehicle.

Below is an example of the controller's response to an abrupt change in curvature from a straight line to a left turn.

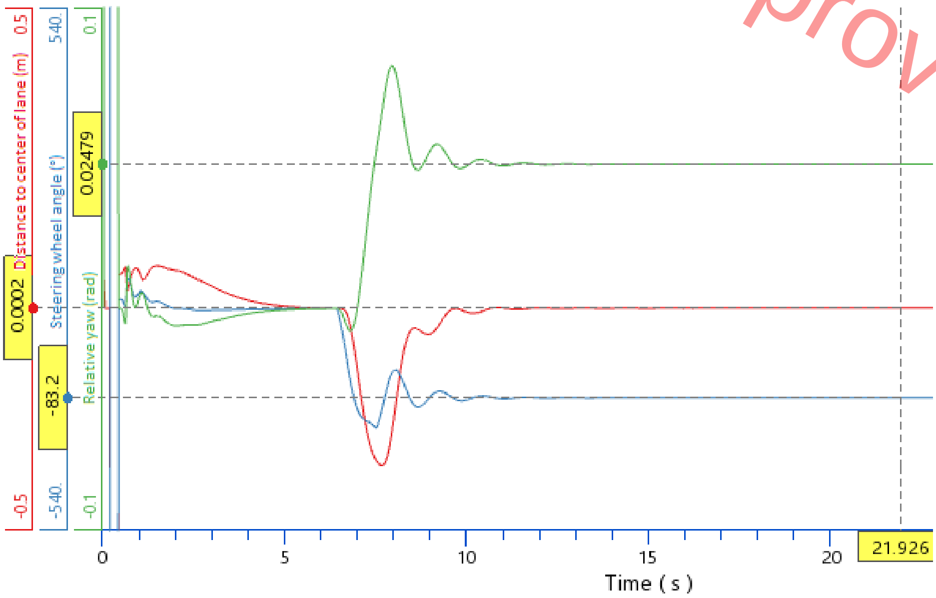



**Figure 19. Lateral controller response**

We can observe that the distance error is brought to zero, and its overshoot is quick and does not exceed 30 cm. We can also observe below the behaviour during lane changes (in blue, the vehicle's trajectory and, in black, the detected marks):

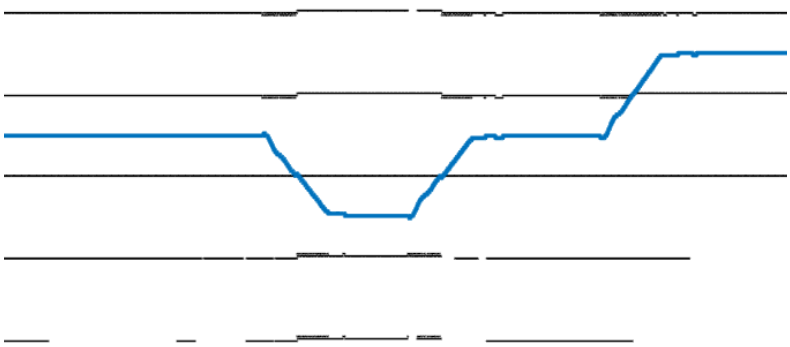


**Figure 20. Lane change behaviour**

Finally, both lateral and longitudinal controllers are coupled in terms of maximum allowed lateral acceleration. *maxAccel* input in the *Control* port (whose default value is 2 m/s²) is used to calculate the maximum allowed longitudinal speed based on the current curvature radius.

### 4.2.3.4 Vehicle model validation

Data collection measurement would be useful to refine and verify the simulation environment, for it to better match the actual behaviour of the vehicles used in AVENUE. The following types of data are considered:

- 'Events' dataset:
  - To identify signals required for the toolchain, in order to adapt the simulation environment with data types comparable to those of the real vehicle;

  → This dataset allows establishing signals with a realistic behaviour and define a safety evaluation method usable in virtual and real controlled environment.

- A test track dataset (i.e. rich, continuous data, collected in controlled environment such as a test track, in predefined scenarios) will be constituted on defined critical scenarios:
  - For direct safety assessment in select scenarios
  - To tune the behaviour of the vehicle model used in simulation

## 4.2.4 Toolchain testing

To test the complete toolchain, some scenarios, extracted from AVENUE's scenarios catalogue, have been implemented. They represent an input from T6.2, feedback from CEESAR expertise and test carried out by operators at Lyon, Copenhagen and Geneva.

To simulate them within AVENUE's context, one real example of Lyon test site has been reproduced:

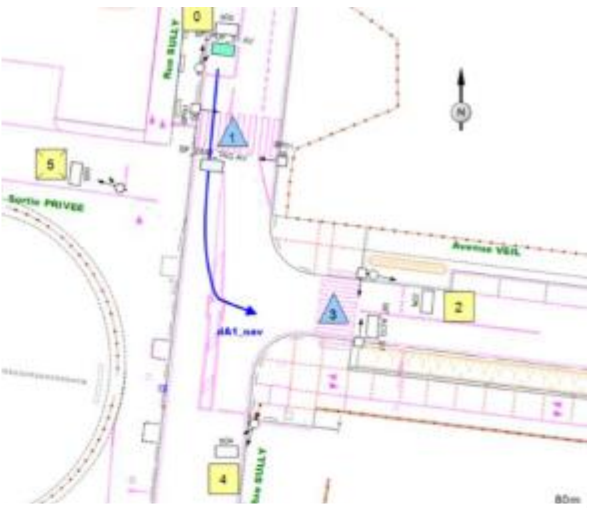



Figure 21. Lyon test site example

The turn left situation from Lyon test site has been reproduced in simulation environment regarding all specific road markings, distances of the road, pedestrian crossings and connected traffic lights.

All moving objects (car, bicycle, pedestrian will also be implemented with regards to the tested scenario.

The connected infrastructure also allows to validate traffic lights sequences' calibration, to enhance traffic conditions (reduce shuttle braking or standstill sequences).

Based on this intersection, several scenarios out of AVENUE scenarios catalogue can be covered for safety evaluation. In a first step, the following scenarios have been considered:

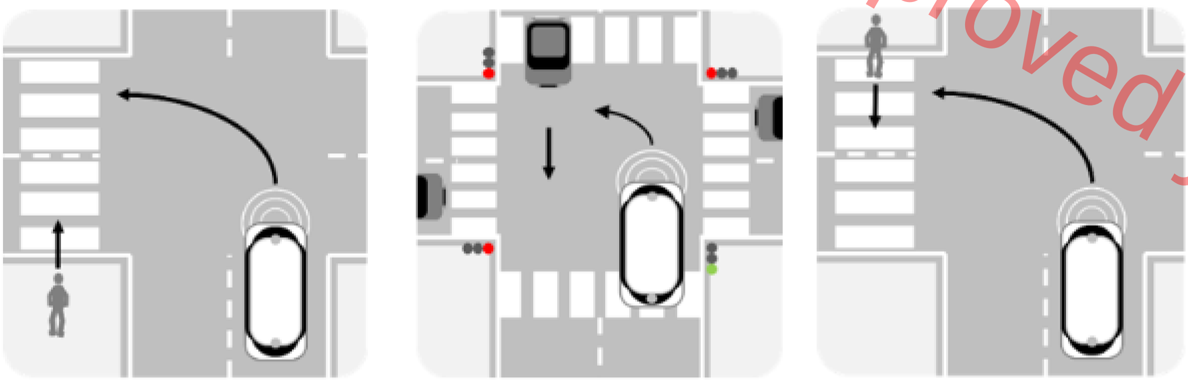


**Figure 22. Extract AVENUE scenario catalog**

Bellow some illustration of the implementation, with an ego shuttle based on T6.1 assumptions:


**Figure 23. Use case with environment configured for sensor evaluation**

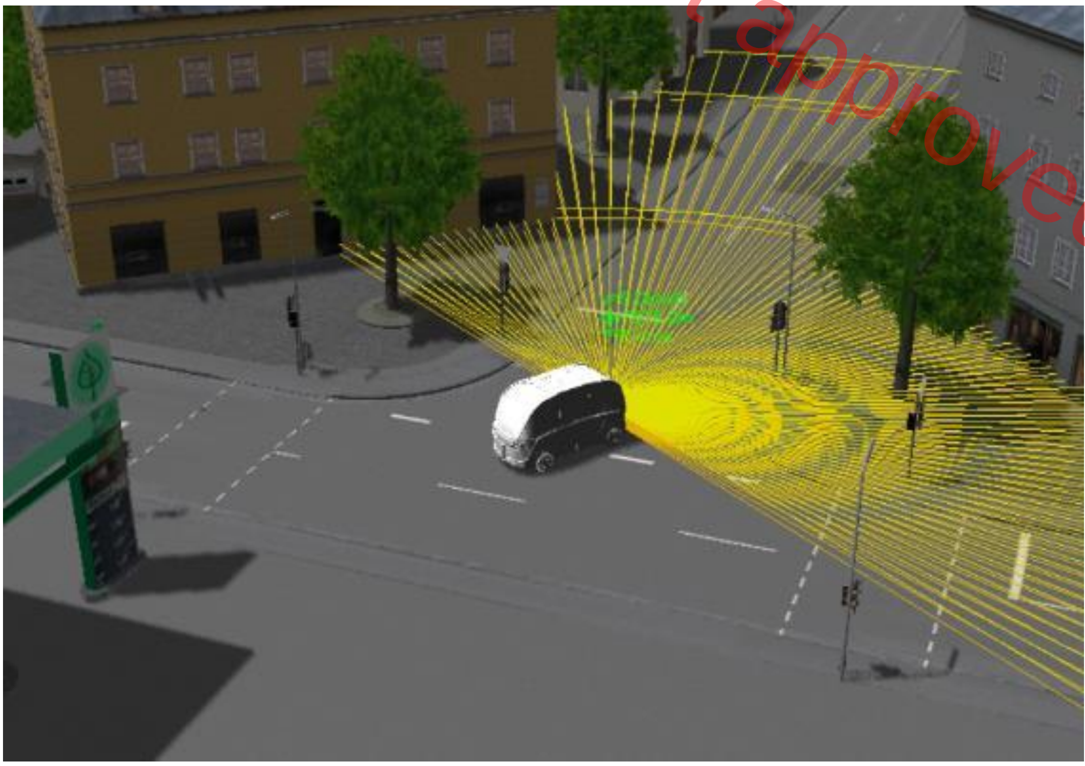


**Figure 24. Virtual environment with 2D Lidars**

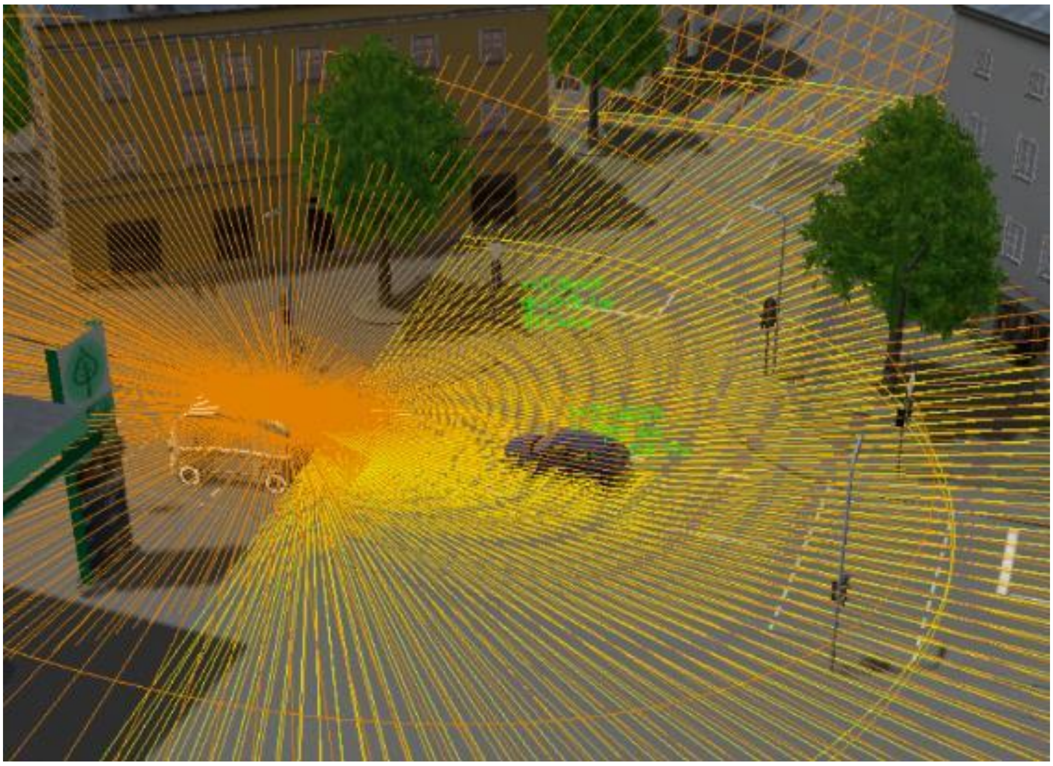


**Figure 25. Virtual environment with all sensors (2D and 3D Lidars)**

The developed methods are not specific to any kind of vehicle (except for the KPIs, which are of course related to the AVENUE use cases). Due to the lack of availability of the NAVYA software to WP6, the current vehicle project software could not be reproduced 1:1.

# 5 Safety assessment

Simulations and eventual test-track trials provide kinematic measures, which can then be used to assess safety. This chapter describes how safety can be assessed in both cases by using AVL-DRIVE AD, by describing how such evaluation is done for “Turn Left” use case (see 4.2.4 page 28).

## 5.1 Data reading and use case isolation

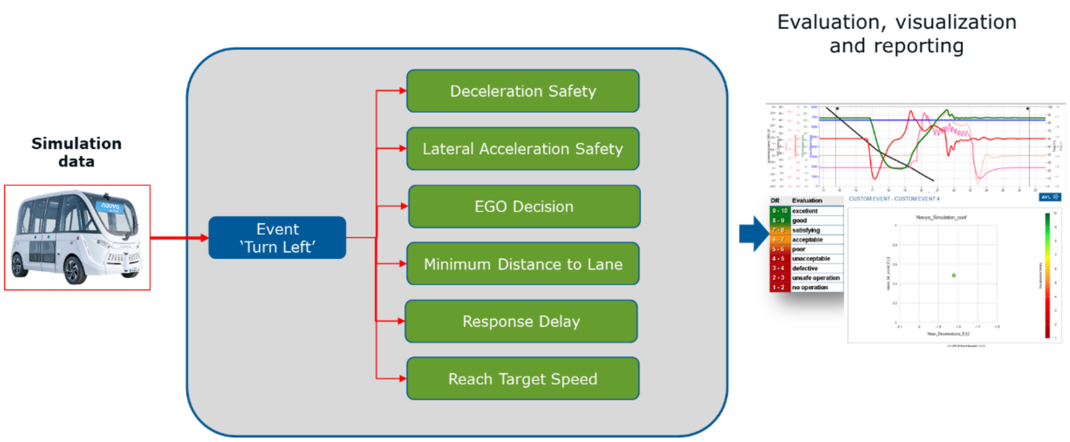
The vehicle safety evaluation method relies on AVL-DRIVE AD. This tool can directly read data from simulations running in the toolchain previously described. It requires some mandatory signals to evaluate the shuttle’s performance. These signals are sampled with a 10 Hz frequency.

**Table 1. List of signals**

| Signal | Unit | Type | Description |
|---|---|---|---|
| AccelerationChassis | m/s² | Acceleration | Longitudinal acceleration at the vehicle's center of gravity |
| AccelerationLateral | m/s² | Acceleration | Lateral acceleration at the vehicle's center of gravity |
| LaneDistance_L | m | Length | Left lane lateral distance to Ego center |
| LaneDistance_R | m | Length | Right lane lateral distance to Ego center |
| SteeringWheelAngle | deg | Angle | Ego steering wheel ange |
| TOF_Distance_X | m | Length | longitudinal relative distance of Target front |
| TOF_Distance_Y | m | Length | lateral relative distance of Target front |
| TOF_ID | m/s² | Identification | Target of front Identification |
| TOF_Speed_X | km/h | Velocity | longitudinal relative speed of Target front |
| TargetSpeed | km/h | Velocity | Target longitudinal speed |
| VehicleSpeed | km/h | Velocity | Ego speed |

AVL-DRIVE AD has a trigger event mode which is used to isolate a part of the measurement to be analysed. This tool allows the system to trigger several events at the same time. Using this method, events can be categorized into operation modes to focus the analysis on specific parts (pedestrian events, turning events…).

To evaluate “Turn Left” use case, we triggered an event to isolate data that needs to be analysed. We calculated six criteria over the event’s duration, to evaluate shuttle’s safety on different aspect of its behaviour. Each criterion gives a rate to evaluate the performance of the vehicle. A global rate is calculated from these criteria to evaluate the event.




AVL-DRIVE AD – *Safety criteria architecture*

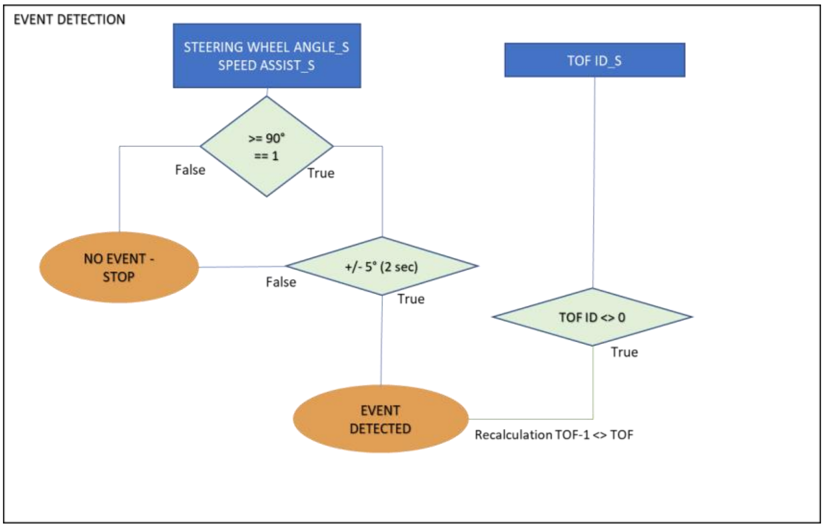
The event detection works with input signals from the simulation. To detect “Turn Left” use case, it focuses on the turn left action and TOF presence. To trigger the event, start and end conditions need to be verified:

- Start condition:
    - SteeringWheelAngle is greater than 90°,
    - SpeedAssist is equal to 1 (active).

  Event starts 5 seconds before the turn if the previous conditions are fulfilled.

- End condition:
    - SteeringWheelAngle must be between -5° and 5° for at least 2 seconds.

- Start recalculation:
    - If TOF ID is different from 0 on the event, start point is recalculated from the beginning of TOF presence.




*Use Case 45_1 - Event detection*

NOTA: in terms of vehicle parameters, the assessment of longitudinal jerk is one of the principal relevant criteria for passenger Safety. This parameter could not be considered until know, as virtual data are not representative for jerk evaluation. This parameter will be considered only based on real data that will be provided during the real test of critical driving situations.

# 5.2 Evaluation Criteria

To evaluate the shuttle’s safety in scenario “Turn Left”, six criteria are selected. Each one addresses one part of the vehicle’s performance.

## 5.2.1 Deceleration Safety Criterion

The **Deceleration Safety Criterion** evaluates the shuttle’s significant longitudinal decelerations during the event. If the vehicle must slow down in consequence of its environment, we want to make sure that the brake process is performed in a safe way for the passengers. This criterion calculates the number of significant decelerations to verify the smoothness of the vehicle's behaviour. The more the shuttle anticipates, the less it will have to brake and the better the safety will be.

Deceleration Safety uses the following signals as input:

- Acceleration_Chassis_SMO40 (m/s²)

In this criterion, some physical parameters are calculated to improve the analysis:

- Deceleration_x_max: maximum deceleration during the event.
- Number_Decelerations: number of *significant* decelerations area on the event. A deceleration is considered significant when it is lower than -1.5 m/s². The area is defined around this value with a threshold of -0.5 m/s².
- Mean_Decelerations: it is the mean of maximum value of deceleration area.
- Standard_deviation_Decelerations: it is the standard deviation of maximum value of deceleration area.

Calculation method:

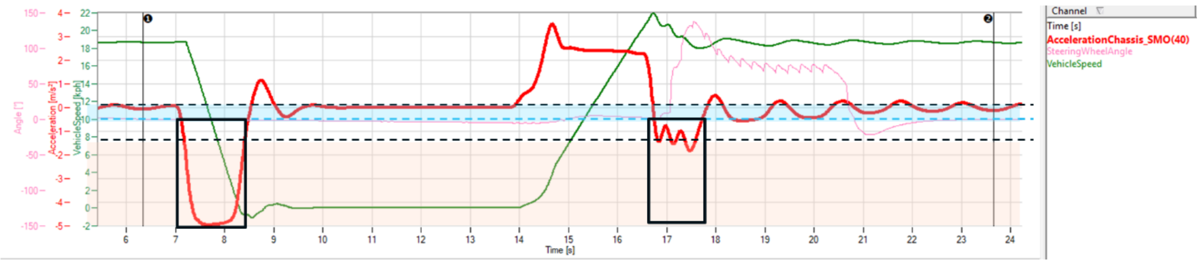



**Figure 26. Deceleration Safety Criterion**

The criterion finds areas (black squares) where the signal Acceleration Chassis exceeds -1.5 m/s² (orange surface) and returns to values greater than -0.5 m/s² (blue surface). This area is considered as a deceleration.

The higher the maximum deceleration is, the lower the rating will be.

## 5.2.2 Lateral Safety Criterion

**Lateral Safety Criterion** evaluates the shuttle's significant lateral acceleration during the event. As the Deceleration Safety criterion, this one calculates the level of significant lateral acceleration to verify the lateral smoothness of the vehicle behaviour.

Lateral Safety uses the following signals as input:

- Acceleration_Lateral_SMO40 (m/s²)

In this criterion, some physical parameters are calculated to improve the analysis:

- Lateral_Acceleration_max: it is the maximum value for absolute lateral acceleration during the event.

- Number_Lateral_Accelerations: it is the level of significant lateral acceleration area on the event. A lateral acceleration is counted when it is greater or less than +/- 1.5 m/s². The area is defined around this value with a threshold of +/- 0.5 m/s².
- Mean_Lateral_Accelerations: it is the mean of maximum absolute value of lateral acceleration area.
- Standard_lateral_accelerations: it is the standard deviation of maximum absolute value of lateral acceleration area.

Calculation method:

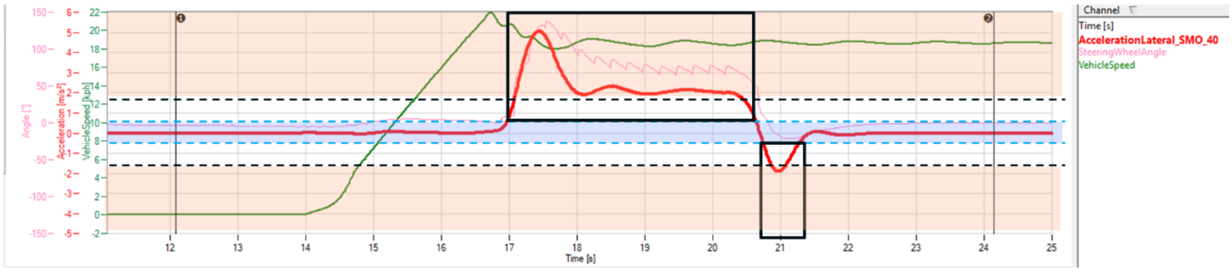



**Figure 27. Lateral Safety Criterion**

The criterion finds areas (black squares) where the signal Acceleration Lateral exceeds +/- 1.5 m/s² (orange surface) and returns to values lower than +/- 0.5 m/s² (blue surface). This area is considered as a lateral acceleration.

The higher the maximum acceleration is, the lower the rating will be.

### 5.2.3 Ego Decision Criterion

**Ego Decision Criterion** evaluates the decision-making of shuttle considering its environment. It assigns a binary score depending on the action taken. We check if the ego's reaction is consistent or not with its environment.

Ego Decision uses the following signals as inputs:

- TOF_Distance_X (m)
- Acceleration_Chassis_SMO40 (m/s²)
- Vehicle_Speed (kph)
- TOF_Speed_X (kph)

In this criterion, some physical parameters are calculated to rate the event:

- Action_to_do: it is a binary variable that gives the expected reaction of the shuttle. It is considering the environment signals, as TOF related ones, to estimate the time to collision. With this time we evaluate if the shuttle can go through the crossroad before TOF or if it must slowdown.
- Action_done: it is also a binary variable that gives the shuttle's reaction. Considering Acceleration_Chassis and Vehicle Speed signals, it evaluates if ego slowed down or not.

Calculation method:

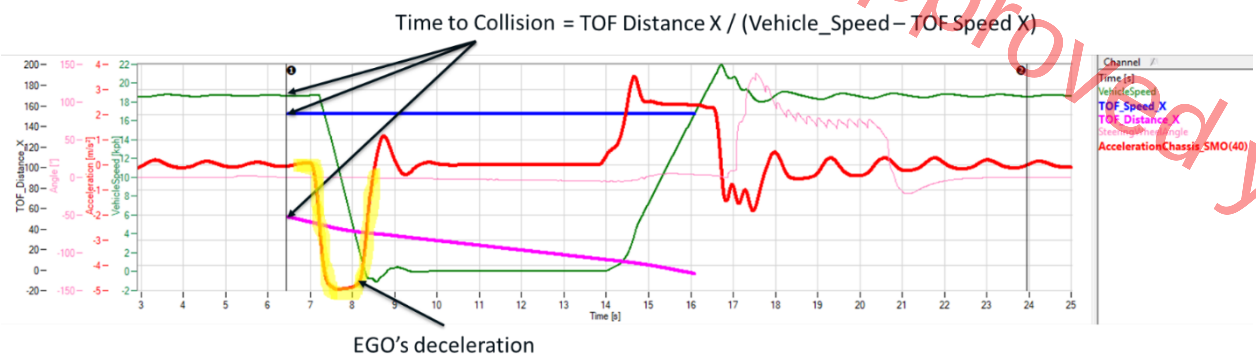



**Figure 28. ego Decision Criterion**

The parameters are calculated on the start of the event, before the turn. If there is a TOF, it calculates time to collision with Vehicle Speed, TOF Distance X and TOF Speed X. If this time is greater than 10 seconds, we consider that the ego can go through the crossroad and turn before TOF. Otherwise, should slowdown or brake. This calculation is used in the action_to_do parameter.

On the other side, we look at Acceleration Chassis and Vehicle Speed to evaluate what the ego's reaction is. If Vehicle Speed is less than 8 kph or Acceleration Chassis is less than -1m/s² before the turn, we consider that the ego is braking. Otherwise, we consider that the ego keeps its speed to cross the road. This calculation is used in the action_done parameter.

To rate the criterion, value of both parameters is compared. If the actions are consistent, a good rating is given.

## 5.2.4 Minimum Distance to Lane

**Minimum Distance to Lane** evaluates the minimum distance to the left lane when TOF Distance X is at its minimum, in case where TOF has priority over the EGO. It ensures that ego stays in his lane when TOF is on its side. The distance to lane is calculated in percentage to normalise for lane width.

Minimum Distance to Lane uses the following signals in input:

- TOF_Distance_X (m)
- Lane_Distance_Left (m)
- Lane_Width (m)
- Vehicle_Width (m)

In this criterion, some physical parameters are calculated to improve the analysis:

- Dist_min_lane_perc: it is the value in percentage of the minimum distance to left lane.
- Dist_min_lane: it is the real value of the minimum distance to left lane.
- Lane_width: it is the value of the lane width, i.e. the distance between left and right lane.

Calculation method:

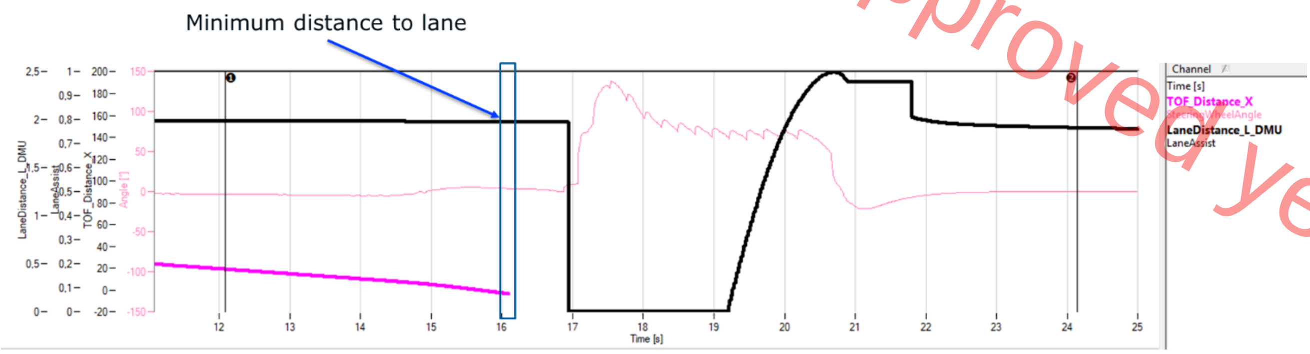



**Figure 29. Minimum Distance to Lane Criterion**

The criterion is calculated only if TOF Distance X signal has non-null values. We are looking for the TOF Distance X minimum positive value and the distance to the left lane at this moment. If the distance is equal to 0 (loss of sensor information), we look for the last non-null value on LaneDistance_left signal.
This value calculates the minimum distance to the left line as a percentage relative to the size of the lane and the vehicle width. The distance calculated is between the left wheel and the lane.

The higher the percentage is, the better the rating.

## 5.2.5 Response Delay Criterion

**Response Delay Criterion** evaluates the time between the loss of TOF in control vision and EGO's first reaction to start turning. This is a comfort-oriented criterion for shuttle's passengers.

Response delay uses the following signals in input:

- Acceleration_Chassis_SMO40 (m/s²)
- TOF_Distance_X (m)

In this criterion, some physical parameters are calculated to rate the event:

- Response_delay: it is the time between the minimum positive distance with TOF, before it goes out of control horizon, and the ego first reaction when Acceleration Chassis is greater than 0.4 m/s² before the turn.

Calculation method:

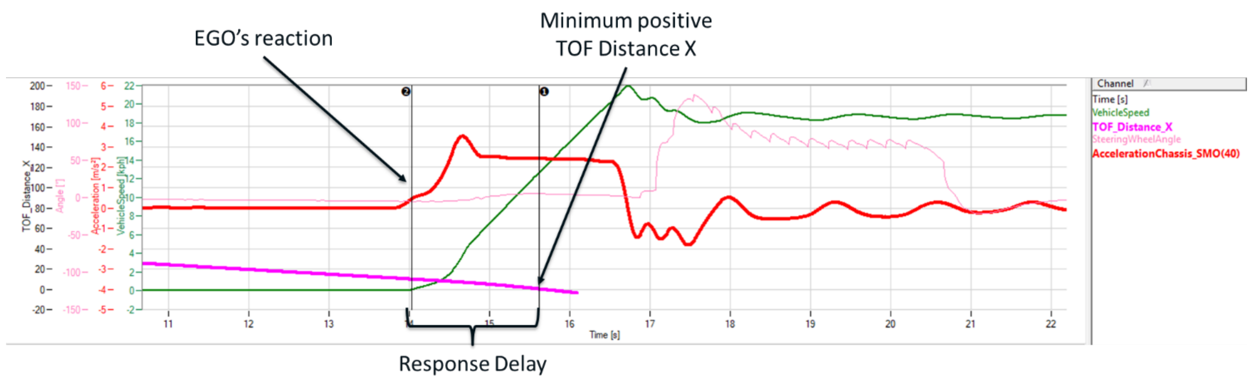



**Figure 30. Response Delay Criterion**

We look for the time of the ego reaction before the turn where Acceleration_Chassis is greater or equal to 0.4 m/s² and the time when TOF Distance X is minimal and positive. This response delay calculated can be negative. In this case, ego anticipates the TOF behaviour which is good for comfort.

The lower the response delay is, the higher the rating.

## 5.2.6 Reach Target Speed

**Reach Target Speed** criterion evaluates the time to reach the target speed and measure an eventual overshoot. This is a comfort criterion to evaluate the EGO's reactivity.

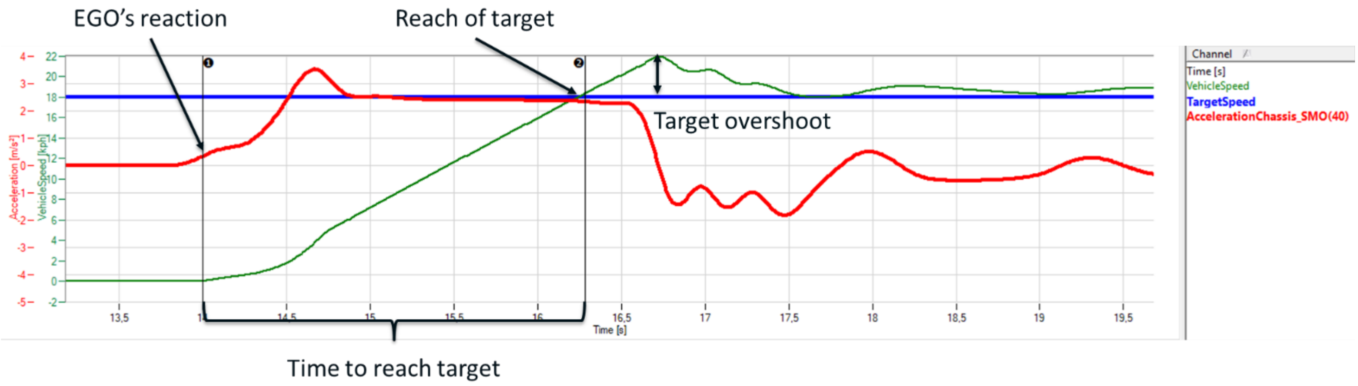
Reach Target Speed uses the following signals in input:

- Acceleration_Chassis_SMO40 (m/s²)
- Vehicle Speed (kph)
- Target Speed (kph)

In this criterion, some physical parameters are calculated to rate the event:

- Time_to_reach_target: it is the time between EGO's first longitudinal reaction before turning and the reach of target speed.
- Overshoot_speed: it is the difference between maximum speed reached and target speed. It is calculated in percentage compared to the target speed.

Calculation method:




*Reach Target Speed Criterion*

For the overshoot speed, it calculates the maximum value of Vehicle Speed after the EGO's reaction. It compares it to the target speed value and translate it in percentage to rate it in the matrix.
For the time to reach speed, it compares the first time when Acceleration Chassis is greater than 0.4 m/s² and the second time when Vehicle Speed is equal to Target Speed signal. If the target speed is never reach on the event, the time to reach speed is forced to 999 seconds.

The shorter the time to reach speed, and the smaller the overshoot are, the better the rating will be.

# 5.3 Evaluation and reporting

After data processing, AVL-DRIVE AD provides the tools to complete the use case analysis. A graphic area allows analysing the chosen signals. Every signal recorded in the data can be analysed in this graphic. A video viewer can read a media file corresponding to the data.

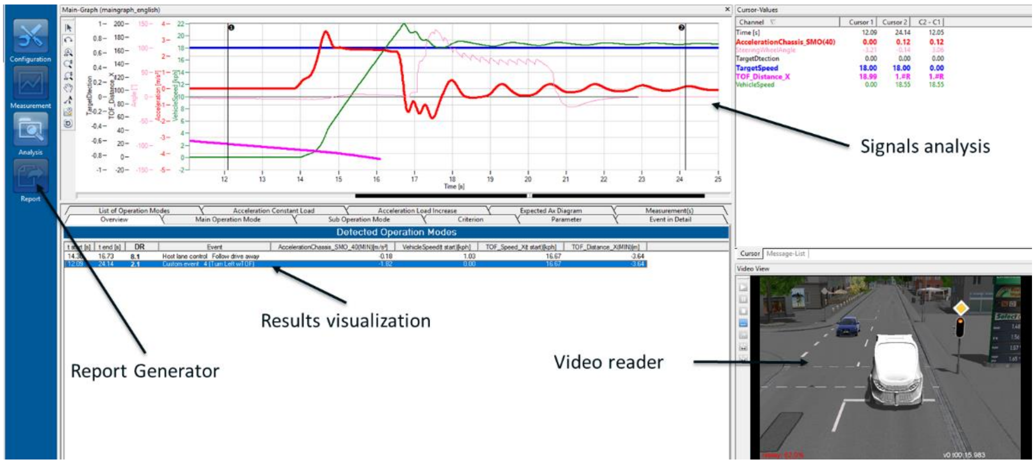



**Figure 31. Overview of** AVL-DRIVE AD **environment**

Event detection results and criteria calculation are available in the visualization window. All the triggered events and corresponding rates are available to be quickly analysed.
For example, on shuttle simulation data, all the criteria for the 45_1 use case are calculated and showed in this window. For each criterion, the values of corresponding parameters are also calculated in this interface. It is possible to set a target rate to reach before processing data. If the rate is under this target, it appears in red in the window to draw attention on it.

| 10.0 | Custom event - 4 (Turn Left wTOF) | 14.78-26.60 [s] | | |
|---|---|---|---|---|
| **DR** | **Param 1** | **Param 2** | **Param 3** | **Param 4** |
| ---- | 0 | 100 | 0 | 21 |
| **Deceleration Safety [-]** 3.0 | Deceleration_Maximal_D [m/s²] -5.09 | Number_Decelerations_D [-] 2.00 | Mean_Decelerations_D [-] -3.62 | Standard_deviation_Decelerations_D [-] 2.08 |
| **Lateral Acceleration Safety [-]** 4.0 | lateral_acceleration_max_D [m/s²] 3.52 | number_lat_accel_D [-] 2.00 | mean_lat_accel_D [-] 2.92 | standard_dev_lat_accel_D [-] 0.85 |
| **Reach target speed [-]** 8.0 | overshoot_speed_D [%] 16.00 | time_to_reach_speed_D [s] 2.09 | | |
| **EGO decision [-]** 9.9 | action done_D [-] 1.00 | action to do_D [-] 1.00 | | |
| **Minimum Dist_Lane_L [-]** 9.9 | dist_min_lane_perc_D [%] 99.53 | dist_to_lane_D [-] 0.82 | lane_width_D [-] 3.75 | |
| **Response Delay [-]** 9.8 | response_delay_D [s] -1.84 | | | |

**Figure 32. Criteria calculation results**

AVL Report Generator can be used to easily report the work with trace of results, and a complete set of graphics, which permits a better view of tests realized on the shuttle. With this tool, we can generate detailed reports of simulation data.

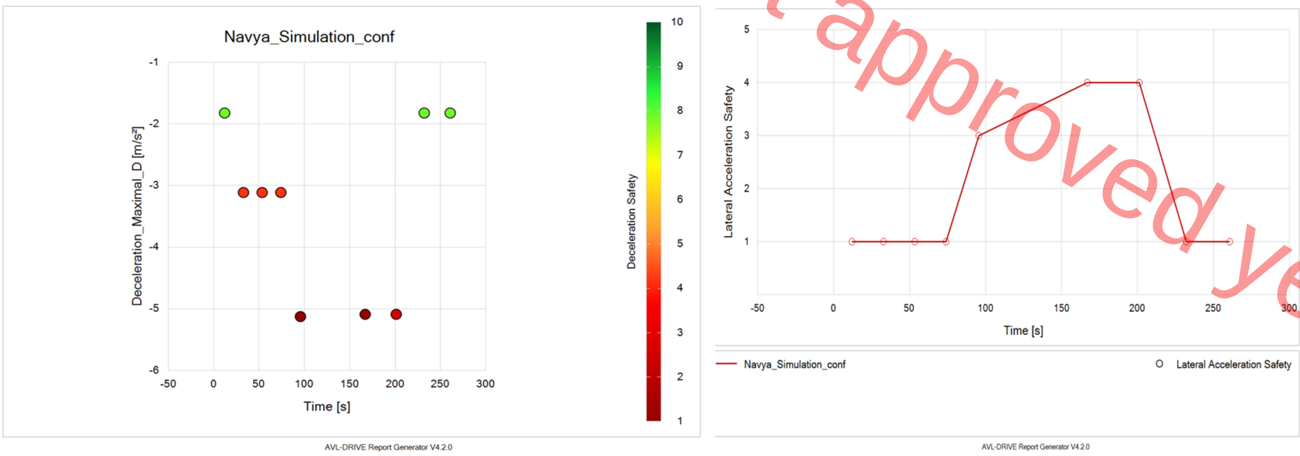



**Figure 33. Graphics available in Report Generator**

Complementary data can improve the analysis with generated graphics. This tool gives the possibility to focus on specific events, criteria or physical parameters calculated in AVL-DRIVE AD, and display them in graphics (2D, 3D, Bar chart, spider chart, target chart, dispersion or cumulative chart). These graphics are useful when a massive simulation is done. It allows comparing several simulations with different parameters in one chart.

# 6 Conclusions

The aim of *WP6 - Safety Evaluation* is to define a method to assess and measure the vehicle behavior with regards to safety aspects, both for passengers and surrounding road users. The safety evaluation in T6.1 builds on several metrics and KPIs provided by the injury risk evaluation performed in T6.2. It includes safety metrics as well as relevant thresholds for safety acceptance.

This requires, however, the availability of real-world measurement data to correlate, further develop, and to fine-tune the safety metrics. Unfortunately, these real data have not been available for WP6 to date, so it has been necessary to start with many assumptions and boundaries to guarantee an acceptable quality referring to the safety evaluation method. A correlation with real test drives for critical scenarios will still be necessary to finalize the safety assessment methodology. Field data were not available at that point in time. However, vehicle responses during deceleration and braking teste have been used in D6.4 to tune the controllers afterwards.

Based on this assumption, T6.1 created a virtual environment with the limited available data. The aim was to develop a validation toolchain with an open approach, in order to allow using the same toolchain for both virtual and real validation.

As an additional level of flexibility, an open-source simulation environment (CARLA Simulator) was first selected. This open-source environment had some restriction to perform all use cases, and it was decided to replace it by another virtual environment, Vires VTD. The validation toolchain based on Vires VTD is fully operational and used as a base for AVENUE safety evaluation. The objective within T6.1 is to eventually provide this validation toolchain also with the CARLA virtual environment.

The safety evaluation method developed in T6.1 is focusing on safety metrics, considering injury risks, vehicle controller comfort, perception performance limitations, and perceived safety for passengers and pedestrians or vehicles around. The safety criteria are generic in order to be applied to several each use cases using the same physical parameters and acceptance thresholds. An automatic detection of a critical use case is performed live and the associated safety assessment too. This allows having an automated processing of T6.1 safety evaluation both in virtual and real environment. The final evaluation report will also consider some recommendations in order to improve the overall safety of the global system (vehicle + infrastructure)

To conclude, a first iteration of WP6 safety evaluation methodology is today finalized and validated for one critical scenario (with several variations) in a virtual environment. The upcoming work will focus on other critical scenarios, to extend both the amount and the maturity of the safety metrics.

A final validation of WP6 activities will require the performance of the same critical scenarios in real-world in order to release AVENUE safety evaluation methodology.

In the next report D6.4, the following section will be included:

- Calibration of controllers based on the real tests
- Massive simulations of two use cases
- Edition of KPI surfaces

- Recommendation to reduce unsafe zones

# 7 Appendix

## 7.1 AVL-DRIVE AD

Autonomous Driving (AD) and Advanced Driver Assistant Systems (ADAS) are increasingly finding their way into series production vehicles. This includes features such as highway pilot, parking pilot, adaptive cruise control, lane keep assist, braking assist, and other systems. And it raises development challenges in the area of perceived safety and comfort.

For drivers the feeling of safety is generally higher when they are able to fully control the vehicle themselves. As AD features take over more driving tasks in the future, OEMs must find ways to objectively assess and reproduce driving maneuvers and scenarios. This is to ensure driver confidence in the vehicles, and must take place in all development phases and in different development environments.

## 7.2 Objective Assessment of Subjective Criteria

With AVL-DRIVE™ AD, we perform an objective human-centric assessment of assisted and automated driving characteristics. This enables OEMs to specify and develop driving features that result in a pleasing experience for the driver. The technology has already been used in the application of KPI target setting in the early phases of development, to support a 'right-first-time' approach. In the calibration and validation phase its fully automated evaluation algorithm and test report generation totally replaces the data analysis. Testing efficiency and calibration time can be significantly reduced.

## 7.3 Automated and Real-Time Assessment

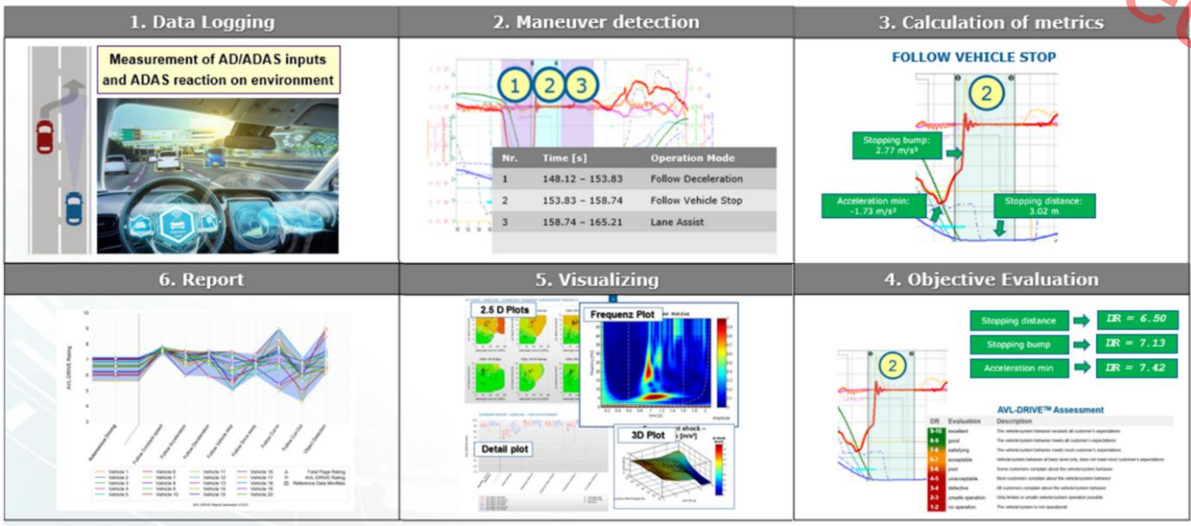
Our tools and methods help you reduce complexity and development effort and produce market-leading products that meet your goals. And this includes AVL-DRIVE AD.

Our assessment technology enables performance KPI target setting, competitor features benchmarking, and features verification and validation both on real road and in virtual environments. This supports frontloading and high-performance testing. The outcome is efficient development, quality improvements and a reduction in costs.

Automated and real-time assessment of autonomous driving features will become more vital as the industry moves into AD levels 3 and 4. Consumer confidence is, therefore, a mandatory goal for commercial viability. As the transformation of the industry gathers speed, it is tools and methods such as AVL-DRIVE AD that will help OEMs lead the market and meet the requirements of the end user.

# 7.4 AVL-DRIVE AD process

This is the AVL-DRIVE AD process:




1) We record the data with our sensors radar, camera, GPS or vehicle information.
2) Based on those signals, each ADAS maneuver is detected automatically in real time. As example, “Follow constant speed”, “Follow acceleration”, Target Object Front approach”
3) For each maneuver, some parameters are calculated to analyze the reaction of our vehicle.
4) Some criteria are defined with those parameters. Based on those parameters, ratings are calculated.
5) A deepest analysis is performed with some several plots
6) A report is generated automatically with the plots