ROADVIEW

Robust Automated Driving in Extreme Weather

Project No. 101069576

Deliverable 7.1

Test plan regarding the most appropriate test Method

WP7 – X-in-the-Loop Test Environment

Authors	Jonathan Robinson (WMG), Graham Lee (WMG), Hazim Zainudin (WMG), Valentina Donzella (WMG), Qasim Sadiq (WMG), Yuri Poldena (THI), Maikol Funk Drechsler (THI), Sébastien Liandrat (CE)
Lead participant	WMG
Delivery date	21 May 2024
Dissemination level	Public
Type	Report

Version 04

Co-funded by
the European Union

Innovate
UK

Co-funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Climate, Infrastructure and Environment Executive Agency (CINEA). Neither the European Union nor the granting authority can be held responsible for them. Project grant no. 101069576.

UK participants in this project are co-funded by Innovate UK under contract no.10045139.
Swiss participants in this project are co-funded by the Swiss State Secretariat for Education, Research and Innovation (SERI) under contract no. 22.00123.

Revision history

Author(s)	Description	Date
Jonathan Robinson (WMG), Graham Lee (WMG), Hazim Zainudin (WMG), Valentina Donzella (WMG)	Draft deliverable	17/11/2023
Jonathan Robinson (WMG), Valentina Donzella (WMG), Yuri Poldena (THI), Maikol Funk Drechsler (THI), Sébastien Liandrat (CE)	Revision 1	15/01/2024
Jonathan Robinson (WMG)	Revision 2	15/02/2024
Jonathan Robinson (WMG), Qasim Sadiq (WMG), Valentina Donzella (WMG)	Final version	29/02/2024
Andreia Cruz (accelCH)	Formatting and formal check (v01)	29/02/2024
Valentina Donzella (WMG)	Final version Sec.1 (page 8) – Added a clarification on failures; Sec. 1.1.1 (page 9) added clarification on metrics and pass/fail criteria; Sec. 2 (page 10) – clarification on scope and V2X testing. (v02)	08/05/2024
Mario Ceccarelli (accelCH)	Logo & acknowledgement update	21/05/2024

Contents

List of Figures

List of Tables

Partner short names

CE	Centre d'études et d'expertise sur les risques, l'environnement, la mobilité et l'aménagement
KO	Konrad GmbH
LUA	Lapin Ammattikorkeakoulu Oy
THI	Technische Hochschule Ingolstadt
VTI	Statens Vag- och Transportforskningsinstitut
WMG	The University of Warwick

Abbreviations

ADS	Automated Driving Systems
BSI	British Standards Institution
D	Deliverable
DDT	Dynamic Driving Test
EC	European Commission
EU	European Union
HEU	Horizon Europe
HiL	Hardware in the Loop
M	Month
MS	Milestone
ODD	Operational Design Domain
SAE	Society of Automotive Engineers
SiL	Software in the Loop
ViL	Vehicle in the Loop
VRU	Vulnerable Road Users
WP	Work Package
XiL	X-in-the-Loop

Executive summary

Objectives

The objective of this deliverable is to define and create a test plan for the testing to be completed in WP7 for the X-in-the-Loop (XiL) testing. This task includes defining what components will be tested in XiL, where they will be tested, the test cases to be implemented, as well as touching on the metrics that define success (these will be presented in detail in D7.2). Finally, the test cases must be made available in OpenX standard format to enable interoperability with a variety of simulators.

Through the creation of this test plan, the XiL testing rigs can be developed with greater knowledge of what will be tested. Simulation based methods will be able to be validated against those with greater levels of hardware such as tests conducted in real test tracks to demonstrate their appropriateness and the combination of test cases and metrics will allow the performance of ROADVIEW components to be evaluated.

Methodology and implementation

The methodology used includes compiling and unifying existing definitions for use case, Operational Design Domain, scenario and test scenario, and test case within a ROADVIEW context and then the process of going from a use case and Operational Design Domain, through test scenarios, to create test cases. This process is then demonstrated by implementing one ROADVIEW specific use case and test scenario.

Outcomes

The results from this deliverable are a process for creating a full test plan for WP7, including a detailed demonstration of the methodology on a single use case and test scenario. The structure and process for expanding this to further test scenarios and different use cases has been defined. This enables easy creation of test plans for the remainder of the project.

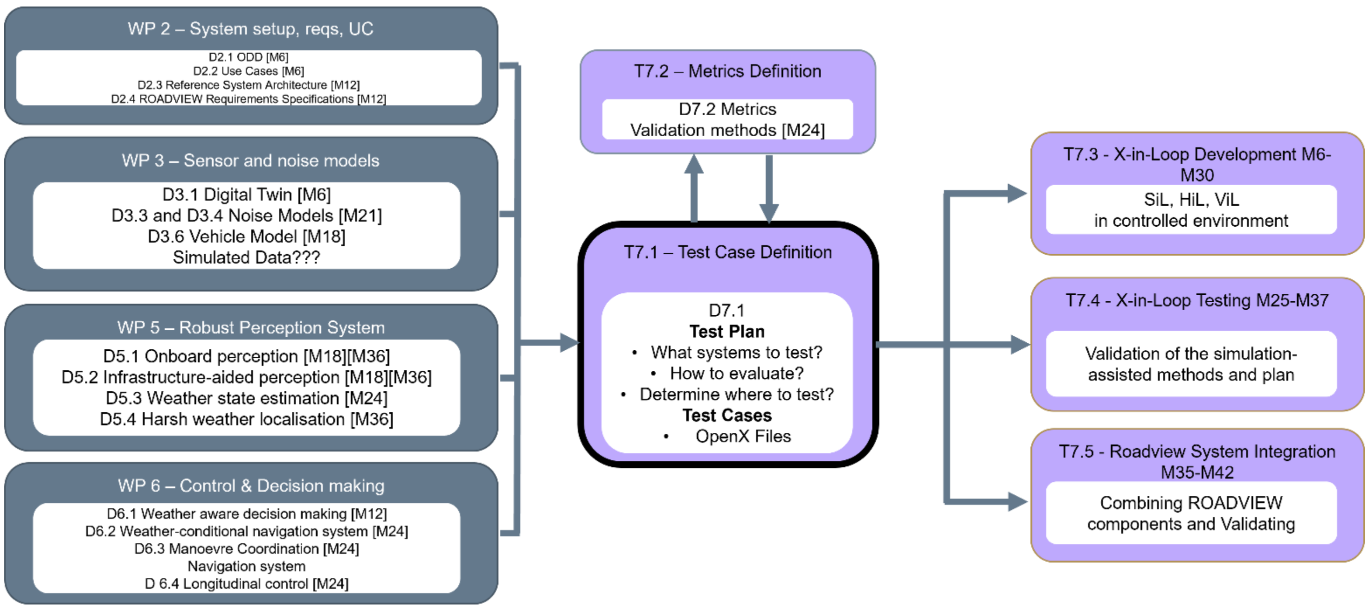
Next steps

The next steps following on from this deliverable will be to implement the test cases in the different environments for XiL testing.

1 Introduction

The purpose of this deliverable is to define a test plan for the X-in-the-loop (XiL) testing of the ROADVIEW system components. This document defines the methodology for going from a use case and Operational Design Domain (ODD), through test scenarios, to test cases, providing the means to create a test plan. This is necessary to ensure suitable coverage of test cases. However, in ROADVIEW, particular focus is placed on reducing the number of test cases to minimise the time and resource efforts needed to execute the test plan and to ensure that test cases can be implemented throughout the XiL testing environments and are applicable to the real world. This is due to time restrictions on the project that mean it is unfeasible to test every test case. Testing single sensors or system failures are out of scope, however they are partially covered by the work in WP5, and WP5 perception is incorporated in the tested autonomous pipeline (see T5.1 and the possibility of using LiDAR data in case of camera failures). Tast 7.1 involves defining the test plan for the XiL testing in Tasks 7.4 and 7.5, whereas Task 7.2 defines the metrics to be used to evaluate the generated test cases.

The relationship between the test plan and other parts of the ROADVIEW system is shown in Figure 1. The left side and in dark blue, are the main inputs into the test plan. These include definitions of use cases, ODD and system architecture as well as some of the tasks/deliverables that will be providing some of the components to test in XiL. These are potential inputs rather than confirmed inputs as many of these tasks are still in progress and hence their maturity will dictate whether they are suitable for testing in the XiL testing. The purple-coloured boxes show the tasks that have dependencies on the test plan. The metrics definition needs to be completed in parallel with this task as there is a need to ensure that there is a measure of success for each test. The three tasks positioned furthest right (T7.3, T7.4 and T7.5) are the development, execution, and validation of the XiL testing. These are dependent on the test plan as these are the steps required to implement the test plan.


WP 2 – System setup, reqs, UC
D2.1 ODD [M6]
D2.2 Use Cases [M6]
D2.3 Reference System Architecture [M12]
D2.4 ROADVIEW Requirements Specifications [M12]
WP 3 – Sensor and noise models
D3.1 Digital Twin [M6]
D3.3 and D3.4 Noise Models [M21]
D3.6 Vehicle Model [M18]
Simulated Data???
WP 5 – Robust Perception System
D5.1 Onboard perception [M18][M36]
D5.2 Infrastructure-aided perception [M18][M36]
D5.3 Weather state estimation [M24]
D5.4 Harsh weather localisation [M36]
WP 6 – Control & Decision making
D6.1 Weather aware decision making [M12]
D6.2 Weather-conditional navigation system [M24]
D6.3 Manoevre Coordination [M24]
Navigation system
D 6.4 Longitudinal control [M24]
T7.2 – Metrics Definition
D7.2 Metrics
Validation methods [M24]
T7.1 – Test Case Definition
D7.1
Test Plan
What systems to test?
How to evaluate?
Determine where to test?
Test Cases
OpenX Files
T7.3 - X-in-Loop Development M6-M30
SiL, HiL, ViL
in controlled environment
T7.4 - X-in-Loop Testing M25-M37
Validation of the simulation-assisted methods and plan
T7.5 - Roadview System Integration M35-M42
Combining ROADVIEW components and Validating
Figure 1 Related dependencies to the test plan

Throughout this document, the outputs and outcomes of T7.1 are presented. The methodology used to create a test plan is described step by step giving reasoning for decisions made. This methodology has been implemented for a single use case and test scenario and then a process put in place for extending this to further use cases and test scenarios.

1.1 High-level methodology

The methodology for creating a test plan has been presented in subsequent chapters. The process involves defining what is meant by terms such as use case, test scenario and test case in the context of ROADVIEW and then using these definitions to reduce abstraction step by step until test cases have been specifically defined. The **use case** provides a high-level and abstract description of the task without specifying the context for the task. **Operational Design Domain** (ODD) provides the bounds of expected operation of any system to be tested and links with the use case to provide conditions within which the automated systems should successfully operate. **Test scenario** provides contextual information. In ROADVIEW this is focussed on road layout (Urban, Rural, Highway), environmental conditions (Rain, Fog, Snow) and Vulnerable Road Users (VRUs). The test scenario describes what will happen in the testing phase without specifying values that parameters will take. Sometimes ranges will be detailed for parameter values at this stage, complementing the ODD. The final step is specifying all parameter values to create **Test cases**. This methodology has been implemented in later sections with focus on road layout, environmental conditions and VRUs to create ROADVIEW-specific test plan worked example.

1.1.1 Metrics and Analysis

A key part of a test plan is determining what success means for a test. What is meant by success for a test case depends on a number of factors such as what is being tested, the use case, the desired behaviour. The same test scenario can have many different success criteria. Therefore, the metrics for success and Key Performance Indicators must be defined for every test scenario, considering every part of the system. These metrics are to be presented in D7.2, so they will only be briefly introduced in this deliverable. Metrics given in this deliverable are examples. Metrics to be used for the project will be presented in D7.2. The criteria for passing the tests, based on the project (and manufacturers') requirements and current regulatory standards, will be further defined in D7.2, including the metrics used for evaluating the system.

In the general case, a metric is a mathematical function which, for a given scenario S and time t, returns a quantity. This quantity is a real number but can also take on an infinite value under certain boundary conditions. It generally corresponds to a physical quantity such as time, distance, or energy, or it can correspond to a rate of occurrence, success or failure. Some metrics are already defined on the scale of a complete scenario and are therefore no longer time-dependent, but very often it is necessary to apply an aggregation method to evaluate a complete scenario. For example, it may be necessary to look at the evolution of a metric over a given period or during a specific event.

There are a very large number of metrics described in the literature. Ideally, a metric should not require any prior choice on the part of the user, but most of them involve one or more parameters such as a threshold. Each metric is generally associated with, or at least particularly relevant to, a specific road context. As said before, the metric studied should therefore be chosen in direct relation to the test scenario in which it will be applied.

Existing metrics generally correspond to either object detection or motion planning. These are sometimes referred to as "low-level" or "high-level" respectively, since object detection corresponds to more direct processing of the data received by the sensors, while motion planning involves more complex decision-making. It should be noted that there are also metrics relating to driving comfort, which seek, for example, to avoid excessively high lateral or longitudinal acceleration (avoidance manoeuvres performed too late), these cases are not studied in ROADVIEW.

As part of ROADVIEW and WP7, the aim was to be able to evaluate each test case using metrics. It was also important that these metrics could be calculated both for tests carried out in XiL and for proving ground tests. Very often, metrics call upon quantities such as position, speed, acceleration or changes in acceleration of the various actors. Actors may be the ego vehicle, other vehicles, VRUs or other objects in a given context.

The aim is then to calculate the values of the metrics in these two situations and compare them to assess the relevance of the tests carried out in XiL.

2 Component and Systems to be Tested in WP7

Since this deliverable focuses only on the test plan to be implemented in the XiL testing, it is necessary to define what will be tested in XiL and what can be omitted and tested on test tracks and real-world. A description of which test cases or test scenarios are tested in Software-in-the-loop (SiL), Hardware-in-the-loop (HiL) and Vehicle-in-the-loop (ViL) is given in Section 6.1. The inputs to the XiL testing come from three work packages: WP3 - looking at creating sensors noise models for adverse weather conditions; WP5 - perception in adverse weather; WP6 - control and decision-making systems that utilise this perception. Explanation of which XiL environment these inputs are tested in (SiL, HiL or ViL) is given in Section 4. Due to partner organisations having different technology and equipment available to them, ROADVIEW has more than one system architecture for the development of automated systems in adverse weather. Since the XiL testing will be completed by THI at the CARISSMA test track facility, the architecture based on their equipment will be used. The focus of work package 7 is on improving XiL systems to address harsh weather conditions, and failure analysis will be conducted based on the limits of the detection algorithms. Furthermore, since V2X is less affected by weather conditions and requires additional hardware to integrate communication into XiL, the V2X features will be directly showcased in the demonstrations and not addressed in this WP.

The XiL testing will focus on the perception components as these are more relevant to SiL and HiL testing. These include initial data filtering, visibility detection, low-level sensor fusion, object detection, weather-type detection, free-space detection and slipperiness detection as shown in Figure 2. Selected combinations of these will be used in SiL, HiL and ViL, with the composition changing to suit the XiL method. For ViL, localisation, path planning and trajectory prediction will also be tested, giving early indications of any issues that may be present in the test track and real-world testing.

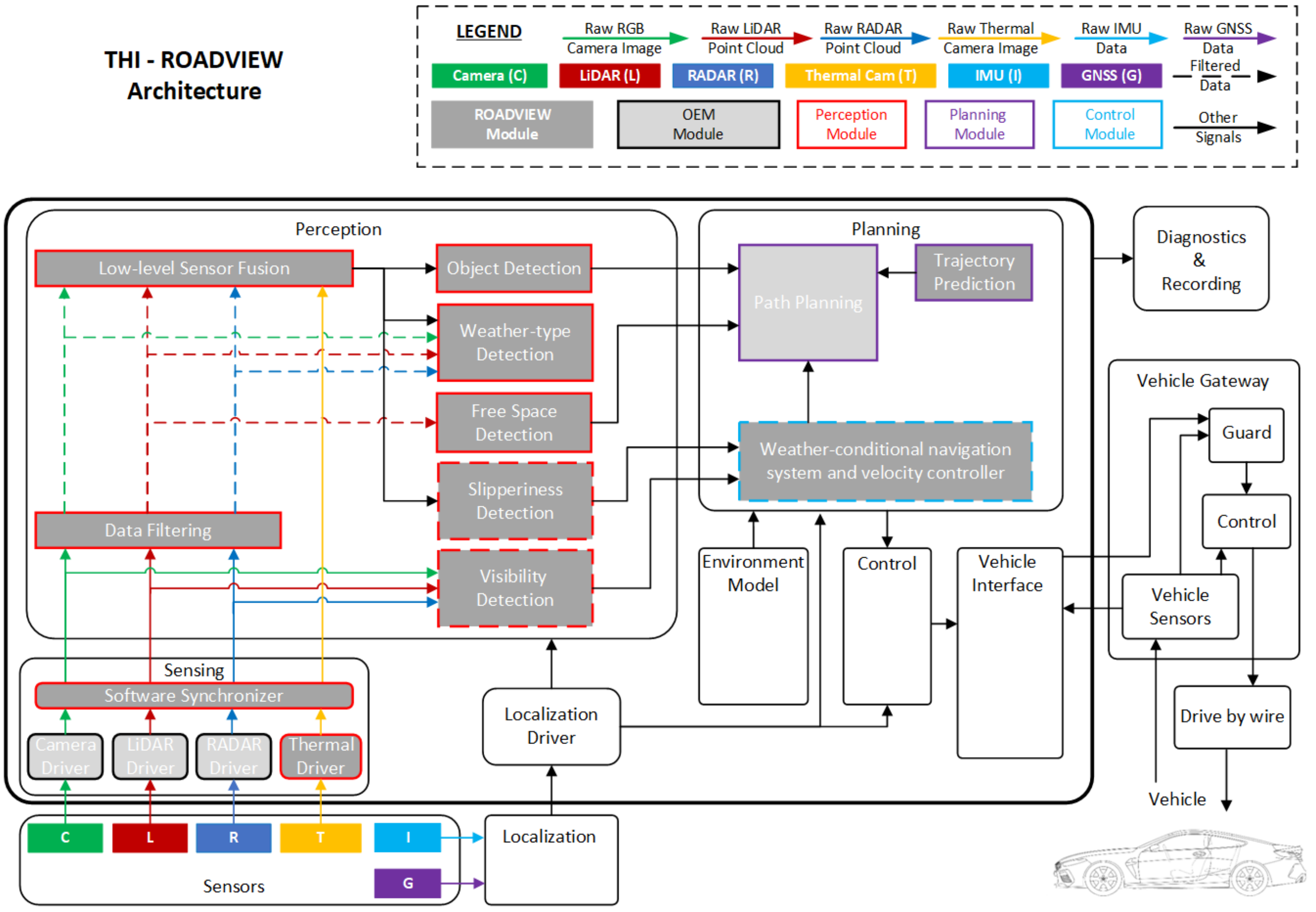


Figure 2 ROADVIEW Architecture for THI Test Rig, showing components to be tested and their interconnections.

3 Use Cases to Test Scenarios to Test Cases – measure KPIs

The use cases defined in D2.2 have been used as a base to define the test cases. The use cases for ROADVIEW are defined abstractly, which allows expansion upon the three main focuses of the ROADVIEW project, which will be explained in section 5.1 below. The process of going from use cases to test cases is defined and presented in this section, whereby the level of abstraction decreases until everything is specified in a test case.

The need to go from use cases to test cases with different levels of abstraction may not be immediately obvious, however, each distinction provides its own purpose. A use case is useful to specify what is being tested to a stakeholder who does not care about the intricacies of how it is tested. An example of this is shown in Figure 3 where the use case is ‘Interacting with a VRU’. This tells the stakeholder that the system’s reaction when a VRU is present will be tested but does not give them any extra information into the technical details of how it is tested. An ODD complements this by providing bounds or conditions under which the use case is tested as is explored further below.

The test scenario then provides contextual information to the use case. In the example in Figure 3, this includes the road type, the weather condition and the VRU that is present, three key areas of interest in the ROADVIEW project. Notice that specifics around the road type or weather condition etc. are not included here, such as the intensity of the rain. This is useful for stakeholders who wish to understand the types of situations in which the system will be tested within but does not include the entirety of the parameter values used for the testing. This enables test plans to be described without an enormous list of test cases with numerous permutations.

Finally, test cases provide the specific parameter values to each test scenario, such as initial speed and position or duration of a particular manoeuvres. This provides the level of detail necessary to effectively critique or recreate a test plan, particularly interesting for others working in the field.

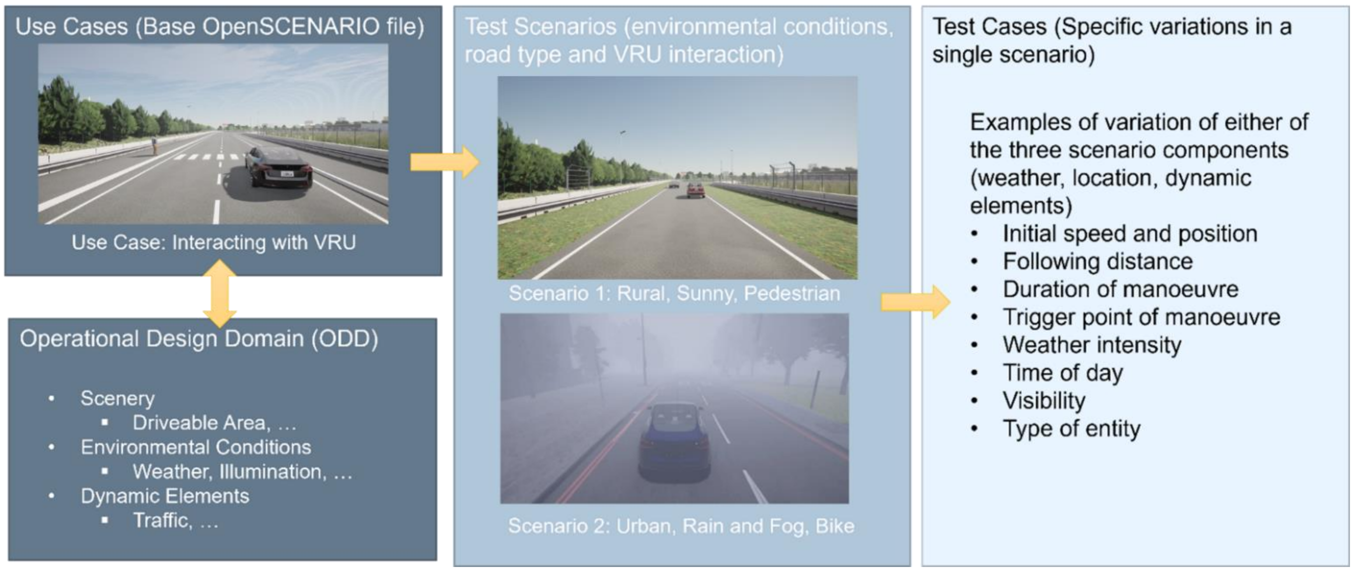

Use Cases (Base OpenSCENARIO file)
Use Case: Interacting with VRU
Operational Design Domain (ODD)
Scenery
Driveable Area, …
Environmental Conditions
Weather, Illumination, …
Dynamic Elements
Traffic, …
Test Scenarios (environmental conditions, road type and VRU interaction)
Scenario 1: Rural, Sunny, Pedestrian
Scenario 2: Urban, Rain and Fog, Bike
Test Cases (Specific variations in a single scenario)
Examples of variation of either of the three scenario components (weather, location, dynamic elements)
Initial speed and position
Following distance
Duration of manoeuvre
Trigger point of manoeuvre
Weather intensity
Time of day
Visibility
Type of entity

Figure 3 Use Cases to Test Cases Example Flow

3.1 Definition of Terms

To define the process of going from use cases to test cases, a clear definition of use cases, ODD, test scenarios, and test cases is needed. Some of these terms are used interchangeably or with differing meaning. The goal for the ROADVIEW project was to draw on definitions and standards to define these expressions in the context of ROADVIEW. This is intended to provide useful and clear definitions within the context of the project and this document, not to suggest a set of definitions to be used by those reading this deliverable.

An example of some differing terminology can be found when considering the definition of use case. Oxford languages defines a use case as “A specific situation in which a product or service could potentially be used” [1]. Alternatively, Meriam-Webster defines a use case as “a use to which something (such as a proposed product or service) can be put” [2]. The Oxford definition refers to the situation whereas the Meriam-Webster definition refers to the task. In ROADVIEW, these have been somewhat combined by referring to both the use, in this case different autonomous driving systems manoeuvres, and the situation which is the environment and road type as outlined in D2.1.

The ROADVIEW use cases are linked to an Operational Design Domain (ODD). British Standards Institution (BSI) define an ODD as “Operating conditions under which a given driving automation system or feature thereof is specifically designed to function” [3]. Society of Automotive Engineers (SAE) define an ODD as “Operating conditions under which a given driving automation system or feature thereof is specifically designed to function, including, but not limited to, environmental, geographical, and time-of-day restrictions, and/or the requisite presence or absence of certain traffic or roadway characteristics” [4]. Whilst these two definitions are aligned, the latter gives greater guidance on what might be included within an ODD. However, this demonstrates that what is defined within an ODD is somewhat open to interpretation. Hence, for the purposes of ROADVIEW, the ODD has been used as above but with particular focus on adverse and extreme weather conditions (see D2.1).

To test a system with a specific use case and ODD, test scenarios can be defined. ISO defines a scenario as a “sequence of scenes usually including the automated driving system(s) (ADS)/subject vehicle(s), and its/their interactions in the process of performing the dynamic driving task (DDT)” [5]. ASAM define a scenario as a “description of how the view of the world changes with time, usually from a specific perspective” [6]. ASAM OpenScenario allows road network, entities and storyboard – which also contains events and triggers – to be defined within its scenarios [7]. The Pegasus project has split the definition of a scenario by level of abstract where an initial functional scenario describes a high-level overview, logical scenarios add a range of values that each parameter can take, and then concrete scenarios provide specific values for each parameter [8]. In ROADVIEW, OpenScenario and ASAM definitions have heavily influenced the definition used for scenario, due to project outputs conforming to OpenX standards. Hence in ROADVIEW, a scenario is called a test scenario and describes a sequence of scenes, where a scene in turn consists of all entities (scenery and dynamic entities). A test scenario may also include events and triggers (see D2.1). This is in alignment with the Pegasus project except the terms used have changed. Functional scenario and logical scenario have been combined into the term test scenario and can still be broken down later. Concrete scenarios are instead termed test cases as is presented below.

Within a test scenario, several parameters can change in value without drastically changing the situation, task or environment. Rather than referring to these changes in value as a new test scenario, within ROADVIEW these are called test cases. For example, a rain intensity change from 5mm/h to 10mm/h would not constitute an entirely new test scenario. Therefore, in ROADVIEW, we denote a test case as a generic term for any type of test on any CAV implementation abstraction level. Common test case types include: software component tests, integration tests between software components, SiL tests, HiL tests, ViL tests, closed area vehicle tests, open road vehicle tests (see D2.1).

3.2 Methodology – Use Cases to Test Scenarios to Test Cases

Figure 4 shows the decomposition of a use case into a number of test cases and the connections with other elements such as the DDT. This diagram shows simplistically the relationship between different parts of the process. The rest of this section shows the methodology used to go from use cases to test cases with a worked example.

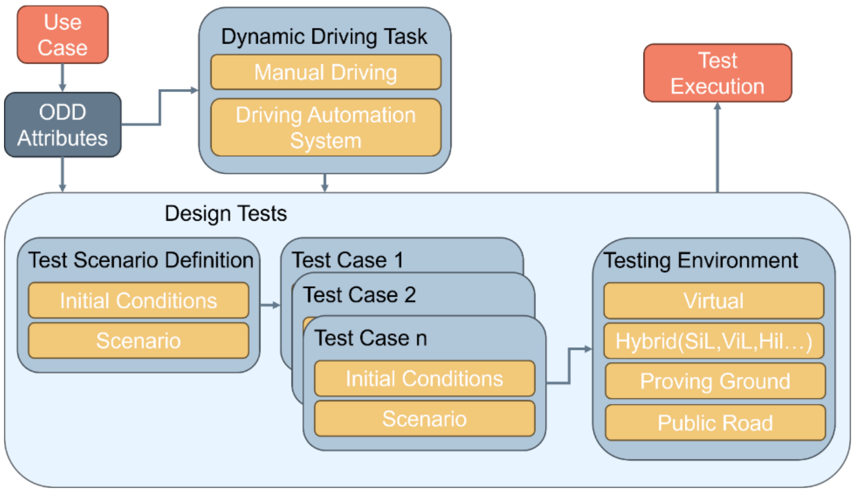


Figure 4 Flowchart of how use cases are expanded into many test cases.

3.2.1 Use Cases

The use cases are defined in D2.2, where there are 5 base use cases for the project to expand upon and use as a reference for testing the ROADVIEW system. These use cases are:

- **UC1:** Driving Straight
- **UC2:** Coming to a Full Stop
- **UC3:** Interacting with VRU
- **UC4:** Entering a Lane
- **UC5:** Exiting a Lane

These five use cases constitute abstract descriptions of the ego vehicle and the main manoeuvre the ego vehicle will be performing. These use case definitions will be defined within a specified ODD, before facilitating the definition of test scenarios by means of defining the attributes that are of interest to the ROADVIEW project.

As an example, we will take the driving straight use case description from D2.2 and provide potential test scenarios and test cases that could be created.

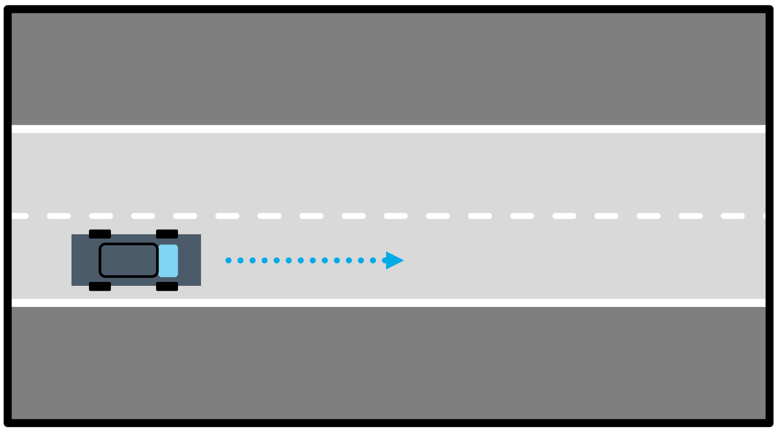


Figure 5 Driving straight use case from D2.2

Figure 5 shows the example driving straight use case extracted from D2.2. The general description of this use case is that the ego vehicle travels along its lane, where the expected behaviour of the ego vehicle is to travel along this lane at a target speed and react to other road users or VRUs (longitudinally, within the same lane).

3.2.2 Test Scenario and Initial Conditions

Test scenarios can be created once use cases are defined to enrich the description of the testing. Figure 6 shows the breakdown of various scenario elements to allow the formation of test scenarios. Those surrounded in red are of particular interest in ROADVIEW as they relate to road layout, environmental conditions and VRUs. Figure 7 relates the creation of test scenarios to the OpenScenario format.

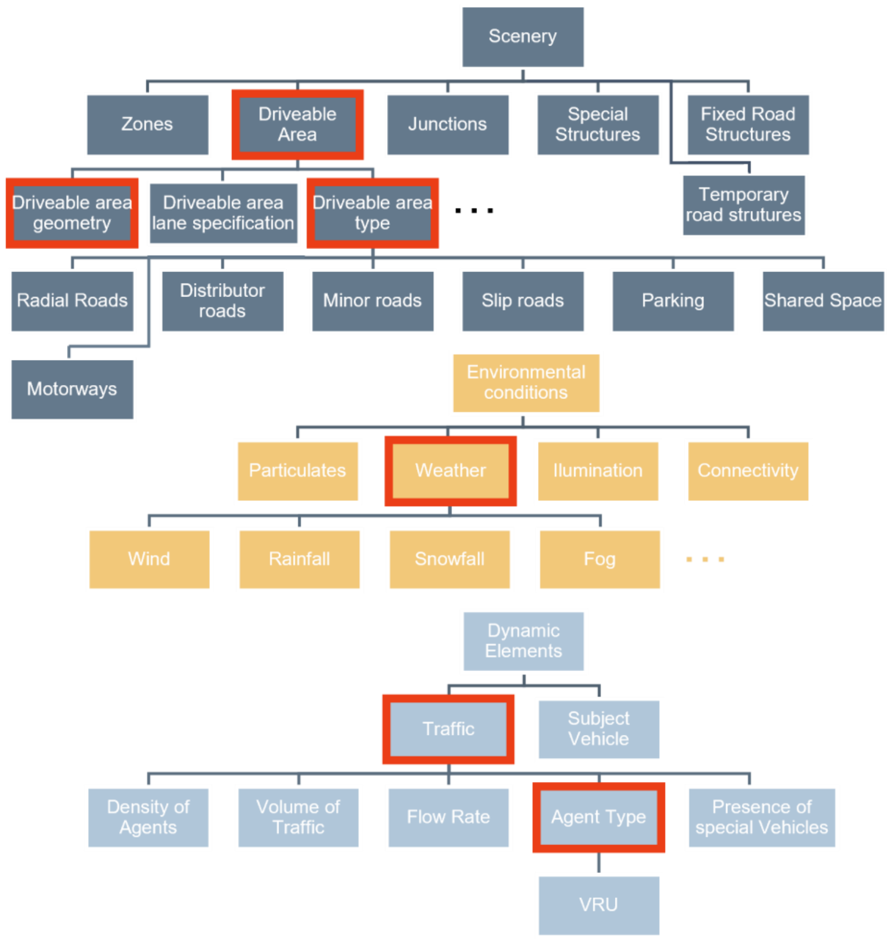


Figure 6 Breakdown of ODD attributes, highlighting areas of particular interest in ROADVIEW

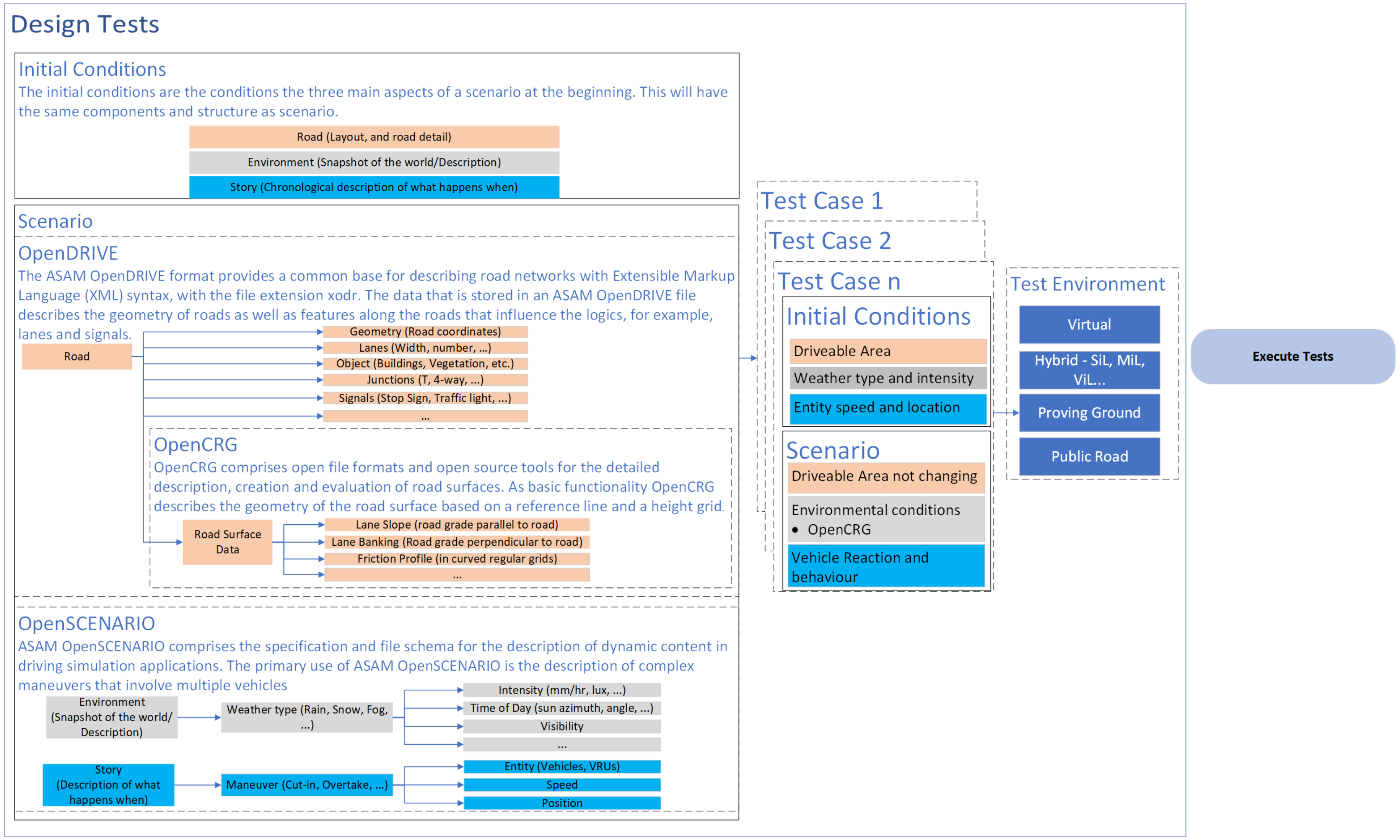

Design Tests
Initial Conditions
The initial conditions are the conditions the three main aspects of a scenario at the beginning. This will have the same components and structure as scenario.
Road (Layout, and road detail)
Environment (Snapshot of the world/Description)
Story (Chronological description of what happens when)
Scenario
OpenDRIVE
The ASAM OpenDRIVE format provides a common base for describing road networks with Extensible Markup Language (XML) syntax, with the file extension xodr. The data that is stored in an ASAM OpenDRIVE file describes the geometry of roads as well as features along the roads that influence the logics, for example, lanes and signals.
Road
Geometry (Road coordinates)
Lanes (Width, number, ...)
Object (Buildings, Vegetation, etc.)
Junctions (T, 4-way, ...)
Signals (Stop Sign, Traffic light, ...)
...
OpenCRG
OpenCRG comprises open file formats and open source tools for the detailed description, creation and evaluation of road surfaces. As basic functionality OpenCRG describes the geometry of the road surface based on a reference line and a height grid.
Road Surface Data
Lane Slope (road grade parallel to road)
Lane Banking (Road grade perpendicular to road)
Friction Profile (in curved regular grids)
...
OpenSCENARIO
ASAM OpenSCENARIO comprises the specification and file schema for the description of dynamic content in driving simulation applications. The primary use of ASAM OpenSCENARIO is the description of complex maneuvers that involve multiple vehicles
Environment (Snapshot of the world/ Description)
Weather type (Rain, Snow, Fog, ...)
Intensity (mm/hr, lux, ...)
Time of Day (sun azimuth, angle, ...)
Visibility
...
Story (Description of what happens when)
Maneuver (Cut-in, Overtake, ...)
Entity (Vehicles, VRUs)
Speed
Position
Test Case 1
Test Case 2
Test Case n
Initial Conditions
Driveable Area
Weather type and intensity
Entity speed and location
Scenario
Driveable Area not changing
Environmental conditions
• OpenCRG
Vehicle Reaction and behaviour
Test Environment
Virtual
Hybrid - SiL, MiL, ViL...
Proving Ground
Public Road
Execute Tests

Figure 7 Detailed Diagram demonstrating the link between OpenScenario and Test environments and how to create test scenarios and test cases

The ODD attributes of particular interest in ROADVIEW are the driveable area (area type and the geometry of the road), the weather (type) and the traffic agent type as shown in Figure 6. Throughout this deliverable, these are referred to as Road Layout, Environmental Conditions, and VRUs. Taking the driving straight example from section 3.2.1, this use case can be expanded into multiple test scenarios with different combinations of the three main attributes.

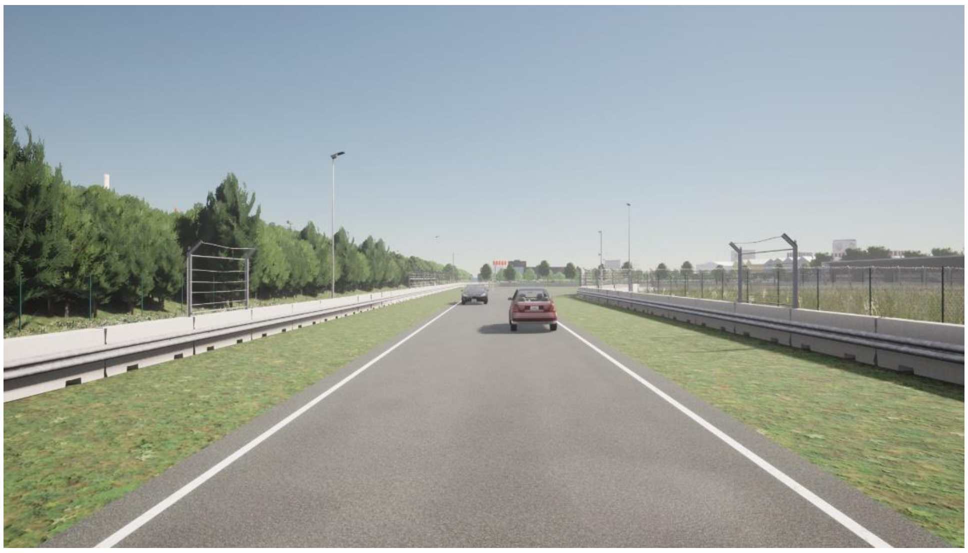


Figure 8 Driving straight, sunny and clear, rural road with oncoming vehicle.

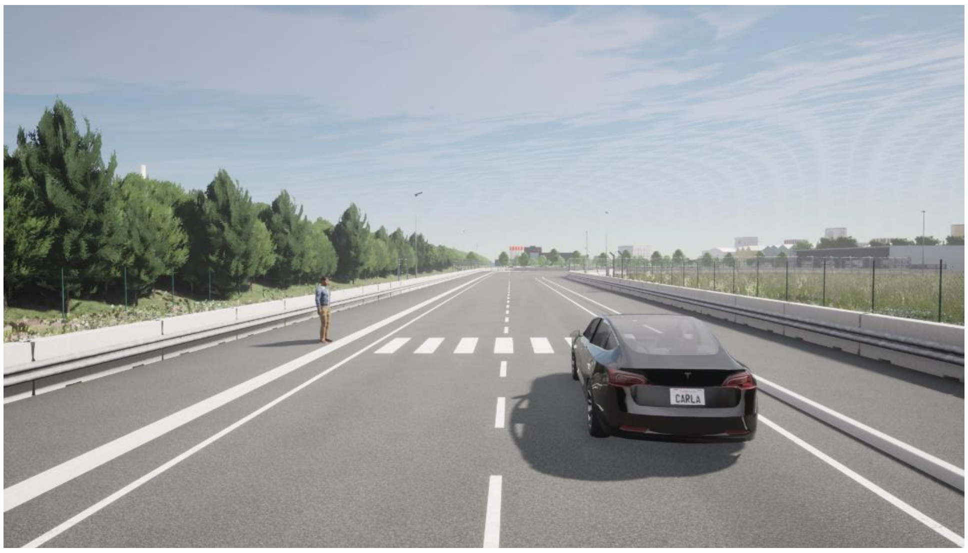


Figure 9 Driving straight, urban road, VRU interaction.

Figure 8 and Figure 9 show two different combinations of the three attributes mentioned to form two test scenarios. The first test scenario describes the ego vehicle driving straight following the lane through a rural road in sunny and clear weather, where the ego vehicle encounters an oncoming vehicle. The second test scenario describes the ego vehicle driving straight following the lane through an urban environment, where the ego vehicle encounters a pedestrian crossing the road. Using this method, depending on the number of attribute combinations available, of interest and possible to test, multiple test scenarios can be defined from one use case.

3.2.3 Test Cases

A test case, within the scope of ROADVIEW, is a more concrete description of the test scenarios mentioned above. This means that parameters such as the weather intensity, vehicle location and speed, and the road width are defined with specified values (e.g., 10mm/h rain intensity, 50kph vehicle speed and 7.5m road width). A change in these parameters would then be a different test case.

The list below shows a non-exhaustive list of the different parameters that could be changed to form different test cases depending on what component or system is to be tested.

Table 1 Table of example parameters for changing test cases

Category	Parameter
Dynamic Elements (Traffic)	Initial speed and position
	Following distance
	Duration of manoeuvre
	Trigger point of manoeuvre
	Type of entity
Environmental Conditions (Weather)	Weather intensity
	Time of day
	Visibility
Scenery (Driveable area)	Road width
	Road type

3.2.4 OpenX files – OpenSCENARIO files

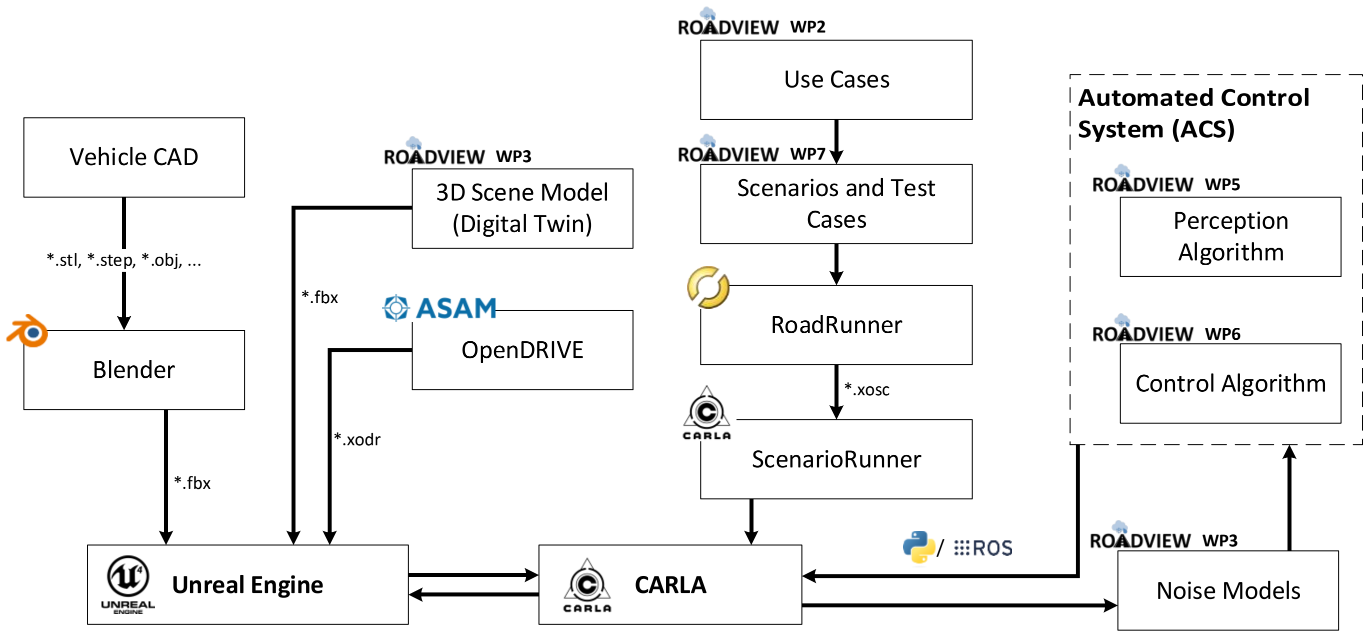


Figure 10 Structure of Architecture Blocks for Testing in Carla in WP7

The process of expanding from use cases to test scenarios and then to test cases has been detailed above. It is a ROADVIEW objective to have these test cases specified in OpenScenario v1.0 format for interoperability. With the defined test scenarios and test cases, the OpenScenario files can be created for each specific test case. This is completed by defining a base test scenario with temporary parameter values in Mathworks RoadRunner and then using a Python script to convert these base test scenario OpenScenario files into individual test case OpenScenario files defined by the test plan. The test plan itself is created using a software called minitab whereby a full factorial design of experiment is created from inputted parameters and their values. The software produces a table of labelled test cases where both standard order and run order are included, allowing both the possibility to randomise and repeat runs should this be seen as beneficial to the XiL testing.

The digital twin created in D3.1 along with an OpenDrive file are imported into RoadRunner to enable the creation of the base test scenario. Since RoadRunner does not support changing weather conditions, this is added manually to the base test scenario OpenScenario files before they are inputted into the Python script. These OpenScenario files can be added directly into the simulation software to implement the test cases. To do this in Carla, the execution is done by using the ScenarioRunner package.

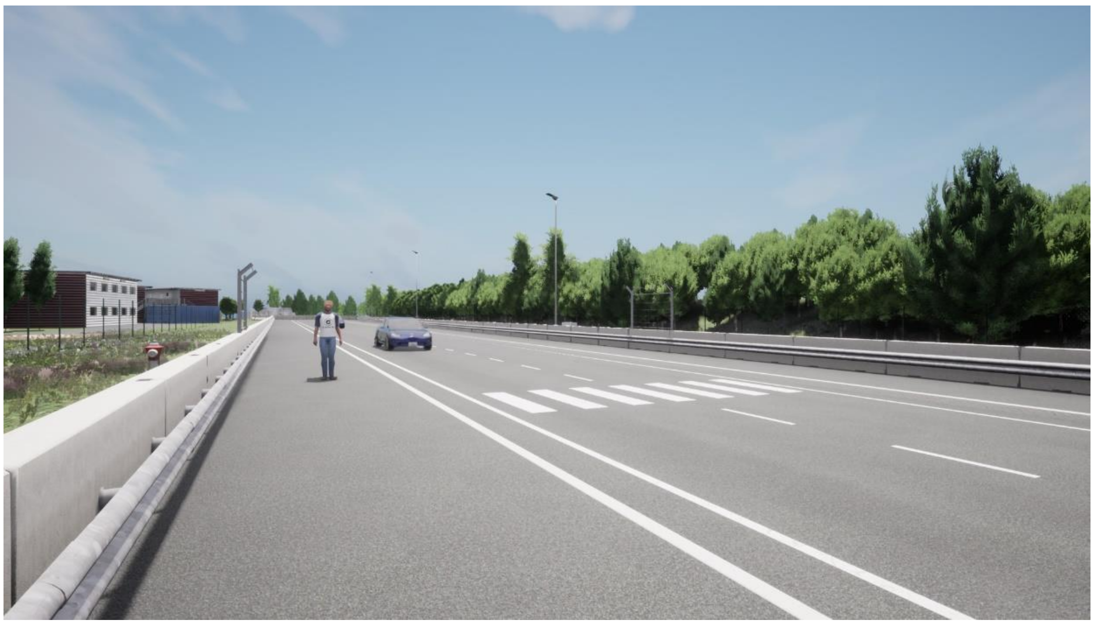


Figure 11 Sample of CARISSMA Digital Twin in CARLA showing vehicle and VRU, created in OpenSCENARIO format.

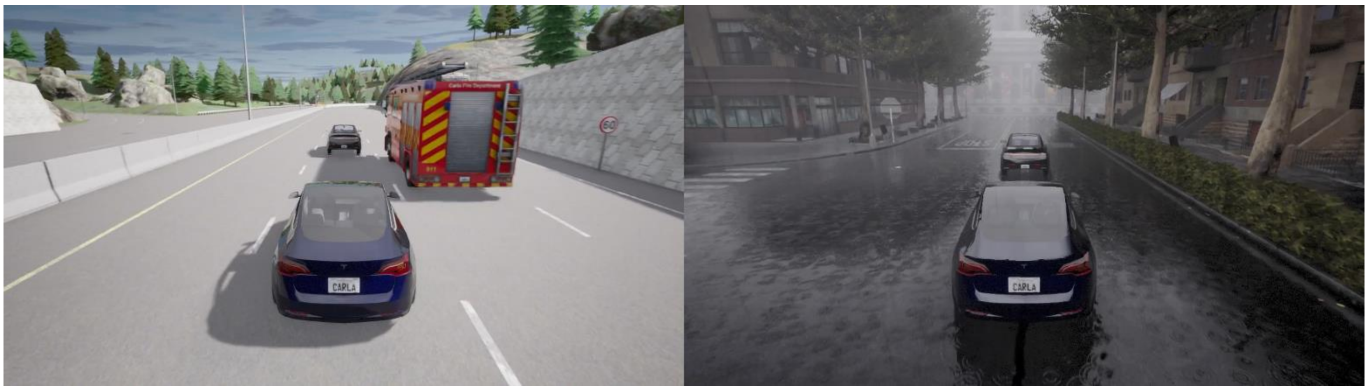


Figure 12 Snapshots of OpenSCENARIO files being used in CARLA with different weather, location and dynamic elements.

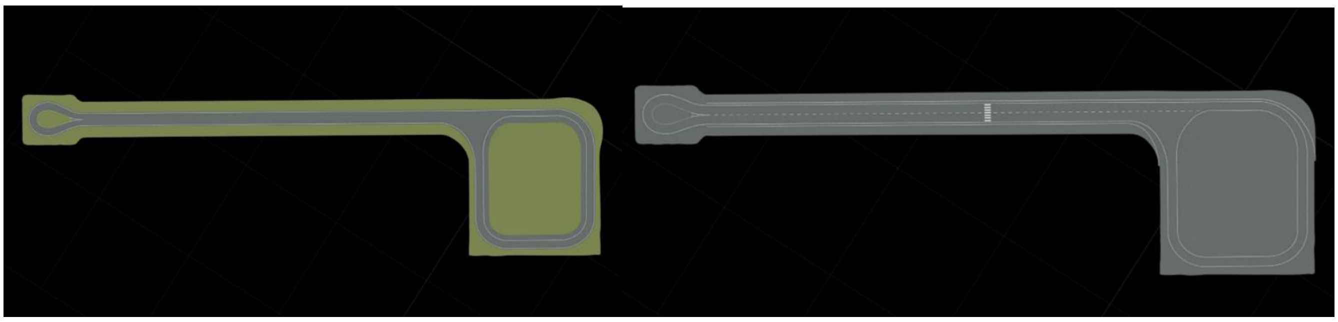


Figure 13 Snapshots of the track created in RoadRunner and imported into CARLA. Rural on the left and Urban on the right.

4 Test Facilities & Experimental Setups (Test Rigs)

4.1 Test Facility

As mentioned earlier, all the testing in WP7 will be based on the THI system architecture. This means that all ViL testing will be completed using the CARISSMA outdoor test facility at THI and any HiL testing will be completed using THI hardware. The CARISSMA test facility consists of an outdoor controlled environment with a dynamic area (60m x 70m), often used for sensor data collection, and an acceleration area (210m x 15m), often used for long range data collection. This facility enables testing with a stationary and moving vehicle and static and dynamic targets in an environment that is safe and controllable. Vehicle speeds up to 100km/h can be used in vehicle testing. Light levels and rain intensity can be measured but not controlled due to the nature of the open outdoor environment. Hence, likely rainfall rates are to be within 10mm/h and 20mm/h. The facility has lights akin to street lights on its perimeter enabling night testing if desired. **Error! Reference source not found.** shows the facility during the day in c loudy weather, at night, and during rainfall with intensity of 10mm/h. This facility also has a digital twin available from D3.1.

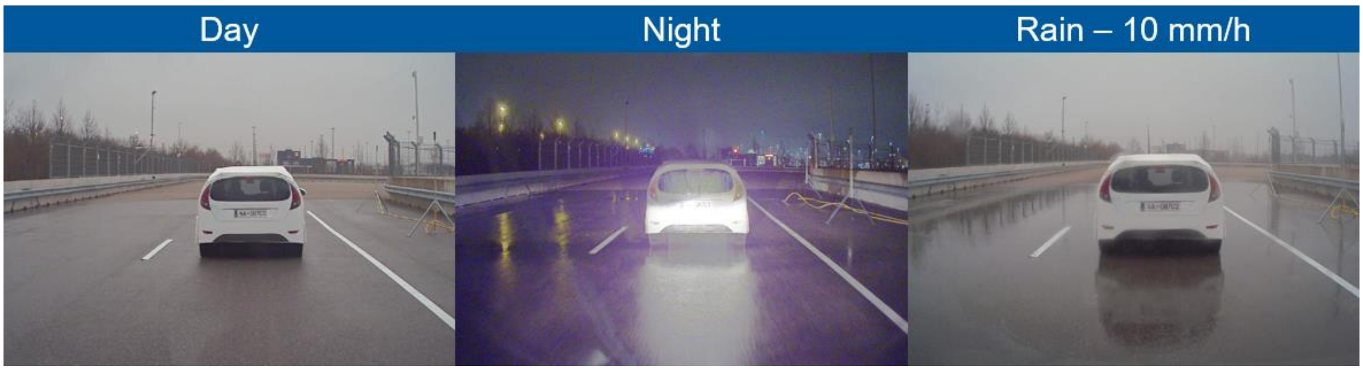


Figure 14 Images of CARISSMA outdoor test facility in different environmental conditions

4.2 Software-in-the-Loop

As mentioned throughout this deliverable, testing will be completed in SiL, HiL and ViL. For the SiL testing, selected software is Carla 9.14 and Carmaker 12. This deliverable focuses on Carla as this is open source, however the contents of this document will be applicable to both software packages. These will also be used for the HiL and ViL testing, for the parts of the system that are not present in hardware. The SiL testing will use the digital twin of the CARISSMA test facility produced in D3.1, providing the scene for the testing. Since the content of testing will be stored within the Carla project, this testing can be completed by any of the partners or a combination thereof. In SiL we can perform more testing due to the lower resource and time requirement. This means we can do more in depth testing of components. As we move to HiL and ViL the amount of testing we can perform will reduce.

4.3 Hardware-in-the-Loop

The HiL testing will use the digital twins within Carla, with some of the components replaced with hardware such as control units or sensors. This aims to lower the gap between the simulation and reality. For this, a very important part is the Realtime computer. A realtime computer is a specific type of computer that relays deterministically a new time. For example, in a vehicle simulation with 1000Hz, a new vehicle state will be calculated and ready for the simulation loop at exactly 1ms. In the CARISSMA HIL this work is done by the NI PXIe-1085. This is in an NI chassis which has other cards and capabilities, such as but not limited to, CAN, FlexRay, Broad R Reach communication, GPS simulation, ADC and DAC, Oscilloscope, Resistance simulator, current and voltage measurements, and FPGAs.

This realtime computer needs a host computer to configure its parameters, this work is done via the usage of a normal PC. This host computer can also “lend” its GPUs to the HiL system for image and point cloud generation, as this is a deeply hardware consuming effort, the host computer has two RTX4090 which are able to simulate the sensors in the simulation. Once the simulated data leaves the realtime/host computer the data is then sent to the sensor stimulation.

The Sensor Stimulators manipulate the data in the HiL. There are two main methods of stimulating sensors, over the air (OTA) and direct data injection (DDI). With OTA, there is an air gap between the injector and the sensor and DDI is where the sensors are stimulated directly inside the sensor stack, which maintains the timings. The idea in both cases is that from the perspective of the sensor it is observing real data, with minimal deviation from reality.

Table 2 Types of Stimulator available in CARISSMA facilities

Sensor Type	Stimulation	
	OTA	DDI
Camera	CameraBox (THI)	CAMULATOR(KO) and VIB(IPG Automotive)
Radar	VRTS (KO)	Not Currently Available
LiDAR	VLTS (KO)	Not Currently Available

Error! Reference source not found. shows the available sensor stimulation at the CARISSMA HiL Laboratory. T he CameraBox was developed internally at THI [9]. This consists of an OTA stimulation of camera using a 7 inch monitor placed in front of a camera, inside a box. CAMULATOR developed by KO, is a DDI stimulation method for GMSL2/3 cameras [10]. The VIB is developed by IPG Automotive and it stimulates only cameras GMSL1 [11],

For RADAR, available is the Vehicle Radar Target System (VRTS) from KO [12], it consists of an anechoic chamber with the RADAR sensor in one side and the stimulator in the other, as the RADAR sensor would need to see targets to its sides, it is rotated so that the Azimuth from the RADAR point is modified. It also counts with an elevator to change the point height. If the OTA performance is not sufficient, DDI or simulation techniques can be investigated.

The LiDAR can be stimulated using the Vehicle LiDAR Target System (VLTS) from KO [13]. It is a similar product to the VRTS but with key differences, where what essentially is a 3D image is shown to the stimulated sensor.

4.4 Vehicle-in-the-Loop

The ViL testing will be completed at the outdoor test facility at CARISSMA. This will involve the driving of a BMW M8 (shown in Figure 15) in a controlled environment communicating with Carla to provide contextual information to the test scenario. This means the vehicle can drive on an open test track whilst the system believes it is in the test scenario portrayed in Carla or CarMaker. The vehicle actuators will be integrated with the simulation system such that there is two-way communication between the state of the vehicle and the perception of the environment. The vehicle dynamics model will be validated here against the real response of the vehicle.

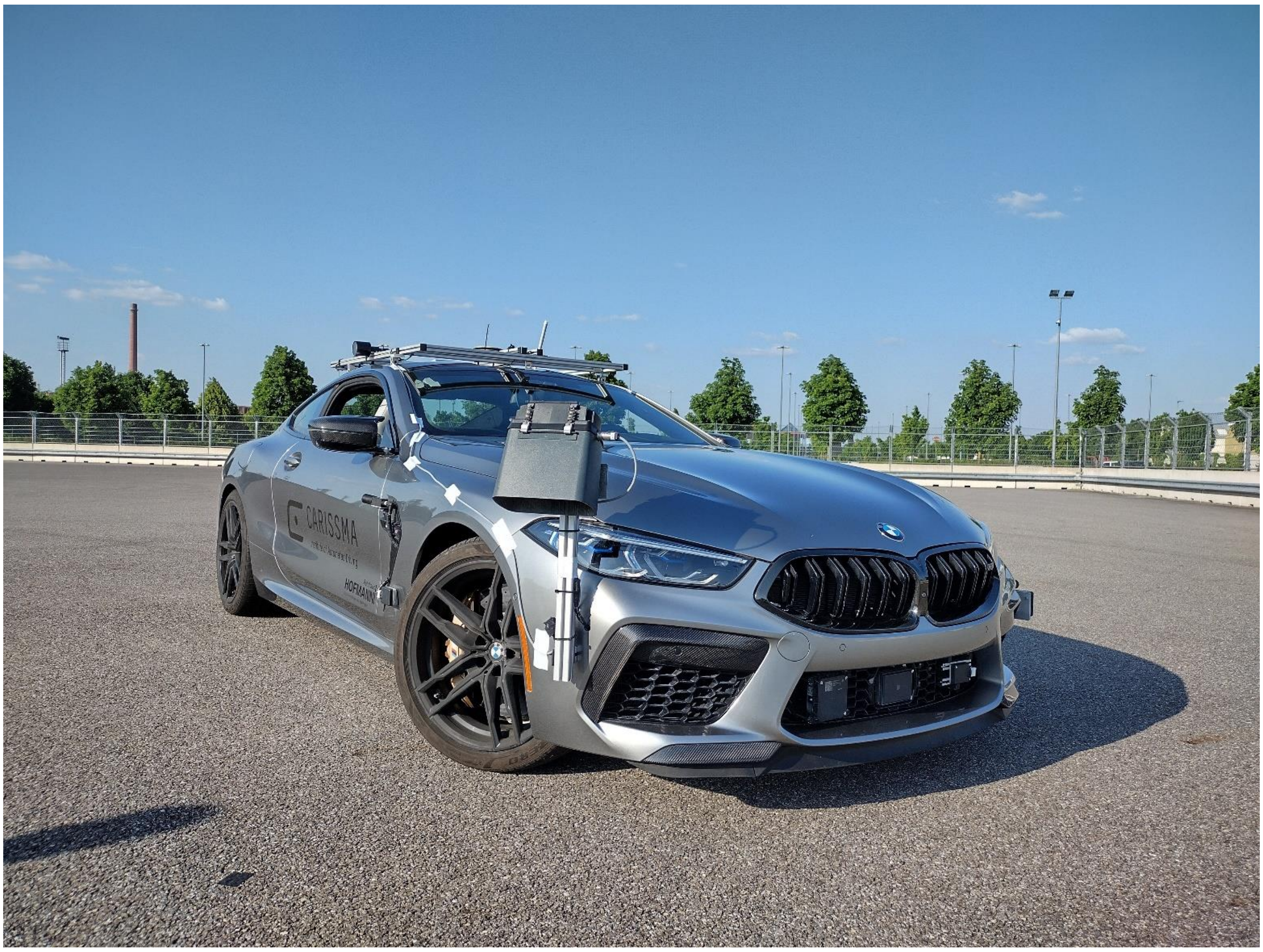


Figure 15 BMW M8 Test Vehicle

5 Methodology

After defining the terms and the methodology for creating test cases in section 3, we are equipped to implement this methodology to create a test plan for the XiL testing. In this deliverable we implement the methodology for a single test scenario, acting as a worked example, demonstrating how this can be used for other test scenarios later. Before presenting the test plan, we first outline the process taken to create the test plan, justifying any decisions made.

5.1 Creation of Test Scenario and Test cases

For this deliverable, the methodology has been defined to create test cases from use cases via test scenarios. In this subsection, this methodology will be put into practice for a single use case and test scenario to demonstrate its effectiveness. This can later be expanded to the entire test plan.

The chosen use case will be UC3: Interacting with a VRU. This use case is the most appropriate as it can include the three main areas of focus for ROADVIEW: Road layout, environmental conditions and VRUs. It is worth noting here that due to limitations in the test environment at THI, only UC1, UC2 and UC3 will be tested in XiL. The test scenario chosen includes a pedestrian wishing to cross at a pedestrian crossing in an urban environment with rain. A pedestrian at a crossing is an extremely common occurrence on public roads and hence is a logical situation to test, with this situation most likely to occur in urban environments so this is the most sensible combination. The weather condition chosen was rain, perhaps the most common of the weather conditions.

To create test cases based on this test scenario, there are some important factors to consider. Firstly, the parameters that will have fixed values and those which will be configurable must be defined. Secondly, the number of values that the configurable parameters can take must be defined. This must consider both coverage but also limiting the number of test cases. Of course, one would like to test every possible combination of every possible situation that could occur, but this is often not practical nor cost-effective. Due to project time and resource restrictions, it was decided that within ROADVIEW, there would be a maximum of 48 test cases for each test scenario. This particular value was chosen as it factorises nicely as $(2)^4(3) = 48$ allowing flexibility to choose number of parameters and the number of values they can take e.g. 4 parameters will 2 values and 1 parameter with 3 values.

This limited number of test cases for a vast number of possible parameters and parameter values, meant that clever grouping of parameters would be necessary to ensure reasonable coverage. The number of oncoming vehicles, the number of adjacent vehicles (if multiple lanes per direction), the spacing between them, the distance an ego vehicle is from a leading vehicle etc., can all be grouped under the parameter Traffic Density. One would expect that as traffic density increases, both the number of oncoming vehicles and adjacent vehicles present would be expected to increase and the spacing between them decrease. Applying the same logic, the distance to a leading vehicle would also decrease. This can be expanded further to variables such as vehicle speed, whereby the speed at which a vehicle travels at, is linked to the density of traffic. Within ROADVIEW, just parked vehicle number and spacing and the leading vehicle initial position will be included under the parameter 'Traffic Density'. This is to reduce complexity of the test scenarios, particularly for HiL and ViL environments.

The grouping of certain parameters assists with reduced test case number but there will be parameters that are linked to this grouping where it would not be desired to group them. For example, there are situations where the ego vehicle speed would be an extremely important parameter to change given different traffic densities, for example testing an Adaptive Cruise Control system on a highway. However, the important factors may change dependent upon the test scenario. Pedestrian parameters such as speed and position will be of high importance for UC3 in an Urban environment but are not going to be important for a highway road layout where no VRU is present.

Hence, this test plan will define a list of configurable parameters for UC3 which can later be extended to other use cases and test scenarios. The maximum number of values each parameter can take will also be defined to assist with ensuring the maximum number of test cases for a particular test scenario is not exceeded. As mentioned earlier, the total number of test cases for a test scenario will be limited to 48. These configurable parameters and the maximum number of values they can take are presented below for UC3.

- **UC3 Interacting with a VRU**
  - Weather Intensity (Rain intensity, Fog Visibility, Snow Intensity)

        - 4 values (clear, 3 intensity values)
    - Ego Vehicle Target Speed
        - Maximum 2 values
    - VRU Behaviour
        - Maximum 2 values
        - Included Parameters:
            - Speed
            - Initial Position
            - Waiting/Crossing
    - Traffic Density
        - Maximum 3 values (Low, Moderate, High)
        - Included Parameters:
            - Leading Vehicle Initial Position
            - Parked Vehicles Number (Urban Only)
            - Parked Vehicles Spacing (Urban Only)

The weather intensity will always take 4 values unless there is a strong reason to the contrary as adverse weather is one of the three main focuses of ROADVIEW, the others being road layout and VRU's. Beyond this, as many as three of the remaining parameters can be included with one taking 3 values and the remaining two taking 2 values. This would be a total of 48 test cases. These parameters are then converted into a test plan for each test scenario using a full factorial design of experiment.

5.2 Metrics

As described in the methodology section, metrics can be selected for each test case. The aim is then to calculate and compare the values of the metrics in proving ground and XiL. In line with what is to be evaluated in the ROADVIEW project, metrics relating to object detection, tracking and motion planning have been listed. At the time of writing, this list is not set in stone and continues to evolve. The metrics listed are purely examples, for project metrics please refer to D7.2. Nevertheless, the list below gives an overview of possible metrics that could be calculated for UC3.

UC3: Interacting with a VRU potential metrics:
- Intersection over Union + First Detection applied on pedestrian
- Intersection over Union + First Detection applied on road markings
- Roadview detection system + First detection
- Object detection - distance
- Intersection over Union + recall/sensitivity applied on pedestrian
- Intersection over Union + precision/confidence applied on pedestrian
- Intersection over Union + f1-score
- ROADVIEW detection system + recall, precision, and f1-score
- Time To Closest Encounter (TTCE) + minimum
- Deceleration Rate to Avoid Crash (DRAC)
- Pedestrian Risk Index (PRI)

As an example, the methodology is developed here through the Time To Closest Encounter (TTCE) metric, which is a variation of the very often used Time To Collision metric for a non-zero distance d. The aim of this metric is to evaluate how close the vehicle has come (temporally speaking) to being at a distance d or less from the VRU. In other words, the vehicle performed well if it mitigated the risk to the VRU by maintaining a distance of at least d.

$$DCE(A_1, A_2, t) = \min_{dt \geq 0}\left(p_1(t+dt), p_2(t+dt)\right)$$

$$TTCE(A_1, A_2, t) = arg \min_{dt \geq 0} \ distance(p_1(t+dt), p_2(t+dt))$$

This metric must be associated with a DMM (Dynamic Motion Model), i.e. a hypothesis concerning the dynamic behaviour of the two actors. If, for example, a constant speed is assumed for both vehicles, the metric is calculated as follows:

$$TTCE(A_1, A_2, t) = \frac{p_2(t) - p_1(t) + \frac{l}{2} + DCE}{v_2(t) - v_1(t)}$$

Where:
- $p_i(t)$ is the position of actor i
- $V_i(t)$ is the celerity of actor i
- l is the length of the car

For information purposes, and to understand the idea behind the metric, it would be possible to assume a braking capacity for each vehicle. Thus, for the same distance, the time to collision would be different.

The version described above is a simple version. Other things to note are:
- This metric should be calculated for a given test case both in the XiL and in the proving ground tests.
- The results would then be compared to assess the relevance of the tests carried out in the XiL compared with those carried out in the proving ground.
- The metric presented here, like many others, is given for each given instant t. The aim is generally to compare the results of a metric at the scale of a whole scenario, i.e. a test case in the case of ROADVIEW 7.1.
- An aggregation method needs to be chosen to obtain a single value.
- This can be an aggregation at the stage of calculating the metric for the complete test case. For example, in the case of TTCE, it could be interesting to take the minimum value corresponding to the situation closest to an accident encountered during the test case.
- The final stage involves comparisons between the minimum obtained in the XiL tests with the minimum obtained in the proving ground tests.

6 Implementation of Worked Example

6.1 Test Plan

The Test plan has been further split to specify the test cases that will be implemented in each XiL environment. Section 5.1 has described the configurable parameters that will be used in UC3. These have been used to define test cases for the test scenario below, where UC3 is tested within an urban environment in rainy weather. The SiL testing will include all the test cases as described above. The HiL and ViL will include a reduced number of test cases, created by fixing one or more of the configurable parameters.

6.1.1 SiL

Table 3 shows the test cases for the SiL testing based upon the UC3 worked example specified in Section 5.1 The configurable parameters are repeated below and the values they can take specified. These values are examples and do not necessarily reflect the values that will be used in the testing.

- Rain Intensity
    - 10mm/h
    - 25mm/h
    - 50mm/h
    - 100mm/h
- Ego Vehicle Target Speed
    - 30kph
    - 50kph
- VRU Behaviour
    - Crossing (Pedestrian crosses the road without waiting)
    - Waiting (Pedestrian waits until it is safe to cross the road)
- Traffic Density
    - Low
    - Moderate
    - High

Table 3 Test cases for worked example in SiL

StdOrder	RunOrder	Rain Intensity	Ego Vehicle Target Speed	VRU Behaviour	Traffic Density
1	40	0	30	Waiting	Low
2	20	0	30	Waiting	Moderate
3	19	0	30	Waiting	High
4	30	0	30	Crossing	Low
5	26	0	30	Crossing	Moderate
6	16	0	30	Crossing	High
7	23	0	50	Waiting	Low

StdOrder	RunOrder	Rain Intensity	Ego Vehicle Target Speed	VRU Behaviour	Traffic Density
8	14	0	50	Waiting	Moderate
9	25	0	50	Waiting	High
10	4	0	50	Crossing	Low
11	10	0	50	Crossing	Moderate
12	46	0	50	Crossing	High
13	13	25	30	Waiting	Low
14	18	25	30	Waiting	Moderate
15	33	25	30	Waiting	High
16	6	25	30	Crossing	Low
17	36	25	30	Crossing	Moderate
18	1	25	30	Crossing	High
19	47	25	50	Waiting	Low
20	24	25	50	Waiting	Moderate
21	29	25	50	Waiting	High
22	22	25	50	Crossing	Low
23	44	25	50	Crossing	Moderate
24	38	25	50	Crossing	High
25	21	50	30	Waiting	Low
26	9	50	30	Waiting	Moderate
27	17	50	30	Waiting	High
28	5	50	30	Crossing	Low
29	37	50	30	Crossing	Moderate
30	34	50	30	Crossing	High
31	35	50	50	Waiting	Low
32	42	50	50	Waiting	Moderate
33	8	50	50	Waiting	High
34	41	50	50	Crossing	Low
35	39	50	50	Crossing	Moderate
36	15	50	50	Crossing	High

StdOrder	RunOrder	Rain Intensity	Ego Vehicle Target Speed	VRU Behaviour	Traffic Density
37	3	100	30	Waiting	Low
38	32	100	30	Waiting	Moderate
39	12	100	30	Waiting	High
40	7	100	30	Crossing	Low
41	28	100	30	Crossing	Moderate
42	11	100	30	Crossing	High
43	45	100	50	Waiting	Low
44	27	100	50	Waiting	Moderate
45	2	100	50	Waiting	High
46	43	100	50	Crossing	Low
47	31	100	50	Crossing	Moderate
48	48	100	50	Crossing	High

6.1.2 HiL

Table 4 shows the test cases for the worked example for the testing within HiL. These are created by taking the test cases in SiL and fixing the ‘Traffic Density’ to ‘low’ thus reducing the number of test cases by a factor of 3. This reduction was chosen as HiL testing is more time consuming than SiL and hence a reduction in test cases was desired. Since the traffic density mostly effects perception due to obstructions, and perception is tested in SiL, the need more different traffic densities is reduced when testing in HiL.

Table 4 Test cases for worked example in HiL

StdOrder	RunOrder	Rain Intensity	Ego Vehicle Target Speed	VRU Behaviour
1	14	0	30	Waiting
2	15	0	30	Crossing
3	9	0	50	Waiting
4	1	0	50	Crossing
5	5	25	30	Waiting
6	8	25	30	Crossing

StdOrder	RunOrder	Rain Intensity	Ego Vehicle Target Speed	VRU Behaviour
7	7	25	50	Waiting
8	11	25	50	Crossing
9	4	50	30	Waiting
10	6	50	30	Crossing
11	2	50	50	Waiting
12	10	50	50	Crossing
13	16	100	30	Waiting
14	12	100	30	Crossing
15	3	100	50	Waiting
16	13	100	50	Crossing

6.1.3 ViL

Table 5 shows the test cases for the worked example in ViL. These have been reduced to just 8 test cases by fixing ‘Ego Vehicle Target Speed’. This is needed because ViL testing is very time consuming. The fixing of ‘Traffic Density’ to ‘Low’ means fewer entities need to be placed in the scene and the number of test cases reduced by a factor of 3. Fixing of speed reduces test case number by a further factor of 2. Rain Intensity is kept at 4 values as harsh weather is a key theme of ROADVIEW and VRU behaviour remains at 2 values as VRUs are the focus of this use case. Hence, ‘Ego Vehicle Target Speed’ was the obvious choice to fix for the purpose of reducing test cases.

Table 5 Test cases for the worked example in ViL

StdOrder	RunOrder	Rain Intensity	VRU Behaviour
1	2	0	Waiting
2	4	0	Crossing
3	3	25	Waiting
4	6	25	Crossing
5	8	50	Waiting
6	1	50	Crossing
7	7	100	Waiting
8	5	100	Crossing

6.2 Dissemination of Resource

6.2.1 Test Case to OpenScenario Script

The script used to create each test case as an OpenScenario file has been made available to the public. This script takes the test cases (as above) and a base OpenScenario file as an input and outputs all the OpenScenario test case files as per the number of test cases. This script is written in Python 3.9. The script comes with a readMe document and can be found in the ROADVIEW GitHub. This script is designed to work with any table of test cases and OpenScenario file combination provided they have been implemented according to the readMe file provided.

6.2.2 OpenScenario Files

The OpenScenario files created during the ROADVIEW project are being made available. This will allow the test cases to be implemented in other simulation software. These of course are ROADVIEW specific but enable repeating of test cases should that be desired. They also provide worked examples of the code above which may aid understanding. The files can also be found in the ROADVIEW GitHub.

7 Conclusions

Through the completion of this task, the methodology for creating a test plan has been defined in the context of ROADVIEW. This takes into account components to be tested, test rigs, XiL environments, ROADVIEW focus points and resource requirements to describe how effective XiL testing can be created. This has been demonstrated for a worked example enabling a full ROADVIEW test plan to be created by following the aforementioned methodology. The decision was taken not to define this plan at present as specifics of the test plan will be affected by the outcomes and restrictions from other ROADVIEW work packages. Hence, specifying precisely what tests will be executed at this point in the project was deemed counterproductive and potentially restrictive.

However, through the creation of a detailed methodology, it is now clear what is needed to create the full XiL test plan within ROADVIEW and the time required to do so is significantly reduced. This methodology is specific to ROADVIEW and XiL testing within, however it is more widely applicable. Apply the same reasoning with a different focus and different restrictions and this methodology can be applied to create a test plan in other contexts and applications.

8 References

[1] Oxford Learner's Dictionaries, «Definition of use case noun from the Oxford Advanced Learner's Dictionary,» [Online]. Available: https://www.oxfordlearnersdictionaries.com/definition/english/use-case. [Zugriff am 23rd October 2023].

[2] Meriam-Webster Dictionary, «Defition of a use case noun,» [Online]. Available: https://www.merriam-webster.com/dictionary/use+case. [Zugriff am 23 October 2023].

[3] BSI, «PAS1883 Operational Design Domain (ODD) taxonomy for an automated driving system (ADS) - Specification,» 2020. [Online]. Available: https://www.bsigroup.com/globalassets/localfiles/en-gb/cav/pas1883.pdf. [Zugriff am 11 December 2023].

[4] SAE International, «J3016_202104: Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles,» 30 April 2021. [Online]. Available: https://www.sae.org/standards/content/j3016_202104/. [Zugriff am 23 October 2023].

[5] ISO, «ISO 34501:2022(en) Road Vehicles - Test scenarios for automated driving systems - Vocabulary,» 2022. [Online]. Available: https://www.iso.org/obp/ui/#iso:std:iso:34501:ed-1:v1:en. [Zugriff am 23 October 2023].

[6] ASAM, «ASAM OpenScenario: User Guide v1.0.0,» [Online]. Available: https://www.asam.net/index.php?eID=dumpFile&t=f&f=3496&token=df4fdaf41a8463e585495001cc3db3298b57d426#_foreword. [Zugriff am 23 October 2023].

[7] ASAM, «OpenScenario: ScenarioDefinition,» [Online]. Available: https://releases.asam.net/OpenSCENARIO/1.0.0/Model-Documentation/content/ScenarioDefinition.html. [Zugriff am 23 October 2023].

[8] Pegasus Projekt, «Requirements & Conditions - Stand 4: Scenario Description,» [Online]. Available: https://www.pegasusprojekt.de/files/tmpl/PDF-Symposium/04_Scenario-Description.pdf. [Zugriff am 11 December 2023].

[9] F. Reway, A. Hoffmann, D. Wachtel, W. Huber, A. Knoll and E. Ribeiro, "Test Method for Measuring the Simulation-to-Reality Gap of Camera-based Object Detection Algorithms for Autonomous Driving," 2020 IEEE Intelligent Vehicles Symposium (IV), p. 1249–1256, 2020.

[10] KONRAD TECHNOLOGIES Gmbh, «Camulator Camera Direct Injection,» 2023. [Online]. Available: https://www.konrad-technologies.com/adas-products/camulator-camera-direct-injection. [Zugriff am 23 11 2023].

[11] IPG Automotive, «Video Interface Box X | IPG Automotive,» 2023. [Online]. Available: https://ipg-automotive.com/en/products-solutions/hardware/video-interface-box-x/. [Zugriff am 23 11 2023].

[12] KONRAD TECHNOLOGIES GmbH, «Radar Test,» 2023. [Online]. Available: https://www.konrad-technologies.com/adas/radar-test. [Zugriff am 23 11 2023].

[13] KONRAD TECHNOLOGIES GmbH, «Lidar Test,» 2023. [Online]. Available: https://www.konrad-technologies.com/adas/lidar-test. [Zugriff am 23 11 2023].