Scenario Description Language

(SDL)

Version 8.1

WMG
THE UNIVERSITY OF WARWICK

Contents

1. Introduction

The complexities associated with Automated Driving Systems (ADSs) and their interaction with the environment pose challenges for their safety evaluation. Traditionally, 'number of miles driven' has been suggested as a measure to demonstrate ADSs maturity. However, ADSs would need to be driven for 11 billion miles to demonstrate they are 20% better than human drivers [1], which leads to the need for scenario based testing [2]. EuroNCAP (The European New Car Assessment Programme) have highlighted the role of scenario based testing for the safety evaluation of ADASs and ADSs in their 2025 roadmap [3]. Furthermore, Khastgir et. al. suggested that for ADASs and ADSs, the focus needs to be on *"how a system fails"* as compared to *"how a system works"*. This leads to the focus on the scenarios which expose failures, i.e. Hazard Based Testing [4].

The main motivation of developing the Scenario Description Language (SDL) is to facilitate the scenario-based testing workflow and to support the associated regulatory requirements. However, being an emerging and evolving concept, the industry is still yet to form a unified definition of the term 'scenario' itself. Go and Carroll [5] define scenario within the context of system design as a description that contains actors, their environment and goals, and sequences of actions and events. Geyer et al. [6] then concluded that, under the context of ADSs, a scenario includes at least one situation within a scene and the ongoing activities of one or both actors. Later, Ulbrich et al. [7] introduced the definition of scenario for ADSs as follows:

> *'A scenario describes the temporal development between several scenes in a sequence of scenes. Every scenario starts with an initial scene. Action & events as well as goals & values may be specified to characterise this temporal development in a scenario. Other than a scene, a scenario spans a certain amount of time.'*

A recent study from Gelder et al. [8] further defines a scenario as a description of the characteristics of the ego vehicle, its activities and/or goals, its environment, and all the events that are relevant to the ego vehicle.

It is important to first differentiate and better understand the terms used in this document, i.e., use case (functional scenario), test scenario (logical scenario) and test case (concrete scenario). A use case describes the system behaviour as a sequence of actions linking the result to a particular actor. A test scenario is a specific path through a use case, i.e., a specific sequence of actions. A test case is a set of test case preconditions, and inputs, developed to drive the execution of a test item to meet test objectives [9]. A use case can correlate to multiple test scenarios, and a test scenario can result into multiple test cases. A similar concept is also proposed in a later study in which three levels of scenarios were proposed: functional scenario, logical scenario, and concrete scenario. The 'functional scenario' sits at the most abstract level and can result into multiple logical scenarios, and one logical scenario can result into multiple concrete scenarios. The 'logical scenario' describes parameter using ranges, and the 'concrete scenario' uses concrete values [2][10]. As an addition, Neurohr et al. [10] have recently tended that by adding a fourth layer between the functional scenario and logical scenario called an 'abstract scenario'.

2. User Stories, Goals and Requirements

An important aspect of developing and storing scenarios for scenario-based testing is the need for appreciation about the diversity of its end users (e.g., autonomous vehicle (AV) technology developer, simulation test engineer, real-world test engineer, regulators, the public etc.). Each of these end users have varied requirements at different levels of abstraction.

- AV technology developers would favour a common structure for scenarios in order to share and re-use scenarios across systems and organisations.
- Test engineers would want a high level of specificity to have an objective understanding and be able to execute the scenario on a test platform (e.g., simulation platform, test track).
- Regulators would want a common structure at a higher abstraction level to enable non-specialists to understand the test scenarios. At the same time, they would want to ensure OEMs share similar common structures.
- The public or end users would also want to understand the high-level test scenarios to achieve 'informed safety'.

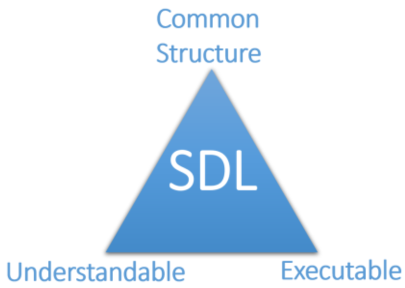


Figure 1 - Targets for Developing the SDL

These user stories have shaped the goal of the Scenario Description Language (SDL) to have a **common structure**, be **understandable** and **executable,** as shown in Figure 1. While there are synergies in the requirements from the various end-users, some are also competing (e.g., executability vs high level of abstraction). Hence there is a need for different levels of abstraction for the Scenario Description Language. In the SDL concept, a two-level abstraction approach has been developed, as shown in Figure 2. SDL Level 1 is mapped to the functional scenarios level presented by 'Use Cases', where characteristics will be described in a qualitative manner (e.g., speed: high). SDL level 2 is mapped to the logical scenarios level presented by 'Scenarios', where characteristics are described using ranges (e.g., speed: 5 to 10 m/s). The test cases or concrete scenarios can be covered using SDL level 2 format, however instead of parameter ranges it will be a concrete value (e.g., speed: 7.5 m/s).

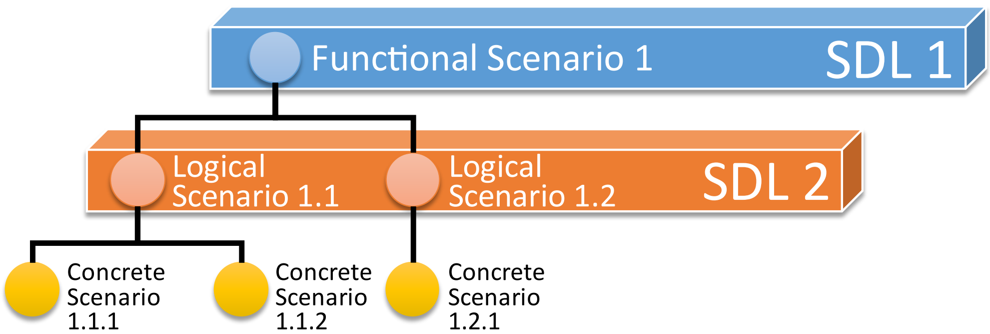


Figure 2 - Two-Level Abstraction of SDL Mapped to Levels of Scenarios

In addition to the high-level goals, these user stories also inspired the following language requirements – composability, extendibility, portability/re-use, and readability.

2.1. Composability

Composability deals with the inter-relationships of components, a highly composable systems provides components that can be selected and assembled in various combinations to satisfy user requirements. The system components in here can be further divided into scenario elements and logical functions, as below:

- Scenario elements include the objects and activities which belong to the SDL domain model. Scenario elements composability indicates the ability of constructing complicated scenarios based on individual elements. Taking the road layout as an example, user needs to be able to compose a complete city road network which consists of junctions and roads from the basic SDL scenery elements.
- Function composition represents the ability of combining simple functions to build complicated operating logic. Examples of function composition in python programming language are if-elif-else, while, etc. The result of each function is passed as the argument of the next until the exit condition is reached. Taking the scripted dynamic elements of SDL as an example, function composition can be used to form the execution logic of a manoeuvre sequence, or synchronised parallel manoeuvres.

2.2. Extensibility

Extensibility is a measure of the ability to extend a system and the level of effort required to implement the extension. Extensions can be achieved through the addition of new functionality or through modification of existing functionality. Extensibility in the context of SDL can be illustrated in two ways: 1) the extension of the underlying domain model of SDL, 2) the extension of the language structure. The first one can be achieved by designing a flexible domain model and allows for modifications and additions via the SDL. The second one can be achieved by modularisation of the language elements, which, as mentioned in the composability section, means that new modules can be added and integrated into the language concept.

2.3. Packaging and Re-useability

Packaging and re-useability refer to the usability of the same elements in different environments. Within the context of SDL, packaging and re-useability are realised in two different use cases: 1) portability across different usage environments, and 2) portability across different scenarios. The first use case can be illustrated by positioning the SDL as a platform independent domain specific language, which means its usage environment should range from simulation to real world, covering MIL, SIL, HIL and test tracks. Such capability of SDL has been demonstrated in the UK wide consortium OmniCAV [11], where SDL Level 1 description was used by the test drivers for executing in a real world environment, and SDL Level 2 was used for simulation execution. The second use case can be achieved by designing a library concept of the SDL language, enabling the user to package certain content from one scenario and re-use it as part of another scenario definition.

2.4. Readability

Readability applies to both natural language and structured Domain Specific Languages (DSLs), though in different forms. In DSLs, features such as comments, choice of structure, and choice of names can determine the readability. For SDL language definition, a mixed approach was followed which combines the natural language terms in a structured DSL format. This can be further demonstrated using the two-level abstraction approach, where SDL level 1 emphasises on the human readability and SDL level 2 focuses more on the machine readability.

3. Scenario Creation

Although this document focuses on the scenario description format, it is important to know where scenarios originate. The creation of scenarios can be data-driven [12][13], knowledge-driven [14], or a combination of the two. A data-driven approach utilises the available data to identify occurring scenarios, and a knowledge-driven approach utilises expert knowledge to identify hazardous events systematically and create scenarios.

Eight different scenario generation approaches have been investigated, as shown in Figure 3. Three of them are currently implemented in the OmniCAV project (options 1, 2, and 3 illustrated in Figure 3), and the other five methods have further been developed as part of a safety assurance framework for ADS. This document will cover the individual elements at a high level, further details can be found in the references provided.

For option 1, the publicly available STAT19 [15] UK accident dataset was analysed to identify accident hotspots and scenario parameters which contribute to causation of accidents carrying high levels of severity [11]. For option 2, anonymised insurance claim records were also analysed to identify the trends in near-miss events that lead to insurance claims [16]. For option 3, an extension to the Systems Theoretic Process Analysis (STPA) method was used to analyse the characteristics of the ADS architecture and identify system failures and hazardous situations [17]. The analysis was then converted into a set of logical scenarios together with their corresponding pass/fail criteria.

In addition to the options 1, 2 and 3, options 4 to 8 have been explored. Option 4 uses the formal analysis approach with the highway code rules for scenario generation. Each of the highway code rules describes a hypothetical driving scenario with the corresponding behaviour and ODD (Operational Design Domain) elements. The ODD is a specification set out by the manufacture of an ADS and it defines the operating conditions within which the ADS can operate safely [18]. Formal models are generated via a model template to create the mathematical representations of those scenarios, collecting the combinations of ODD and behaviour parameters. The analysis reports the manoeuvre parameters that are near the boundary of violation and produces scenarios that represent this set of violations. Option 5 uses similar formal representation to option 4, but applied to the ODD of the ADS. Inspired by the works in [14][19], option 6 uses an ontology-based scenario generation approach. An ontology defines all the classes within the domain. By using a well-developed ontology (with the associated rules and properties) and a highly abstract scenario description of interest at the function level/set of conditions, the abstract information can be detailed and used to generate a large number of logical scenarios that can satisfy the initial conditions.

In addition to the above scenario generation methods, the existing scenarios already defined in the standards, regulations or guidelines (option 7) can also be utilized for the testing of ADSs, for example the scenarios set out in ISO22737 [20] and EuroNCAP [3]. ISO22737 has been developed for Low-Speed Automated Driving systems (LSAD) and the EuroNCAP provides a set of testing scenarios for the safety assurance of vehicles. An example of a EuroNCAP scenario converted into a logical scenario was previously illustrated in this paper section IV-B [21]. Option 8 includes the scenarios that occur during real world trials and deployments. Such scenarios might have not been considered pre-deployment but are key learnings.

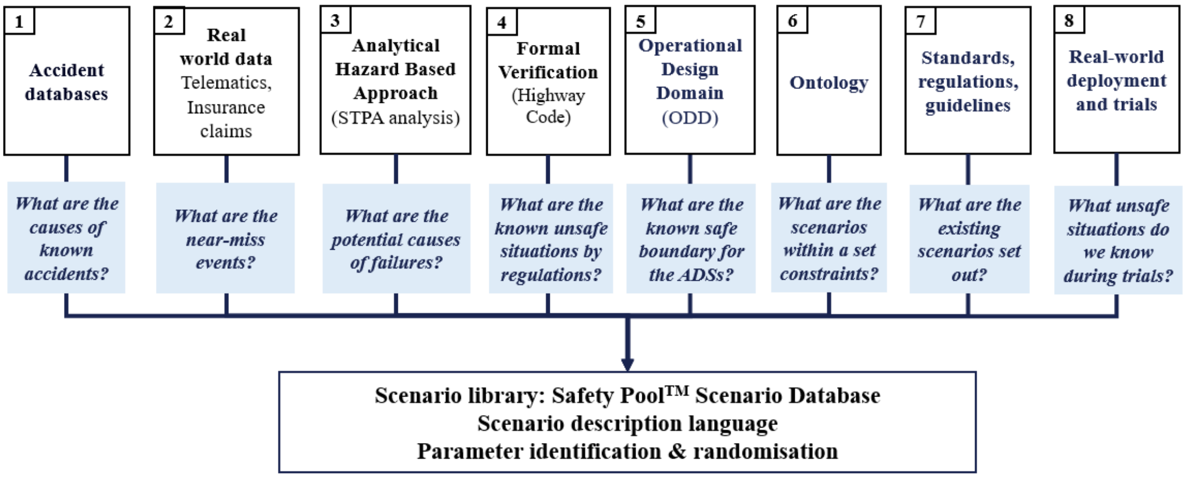


Figure 3 - Scenario Generation Methods in OmniCAV [11]

4. Upper-Level SDL Concepts

The upper-level concepts used to construct a scenario in SDL involve answering the following questions to uniquely describe scenarios:

1. Actors - **Who** is involved into this scenario?
2. Activities - **What** is happening in this scenario?
3. Locations - **Where** is this scenario performed?
4. Events – **When** can this activity take place?
5. Ambient Conditions – **What** is the ambient settings or environmental conditions?

Using these upper-level concepts, one could construct a natural language based format to describe a situation as follows:

Actor is doing activities in location when event at ambient condition.

The two examples in Figure 4 show how SDL statements are mapped to this sentence structure:

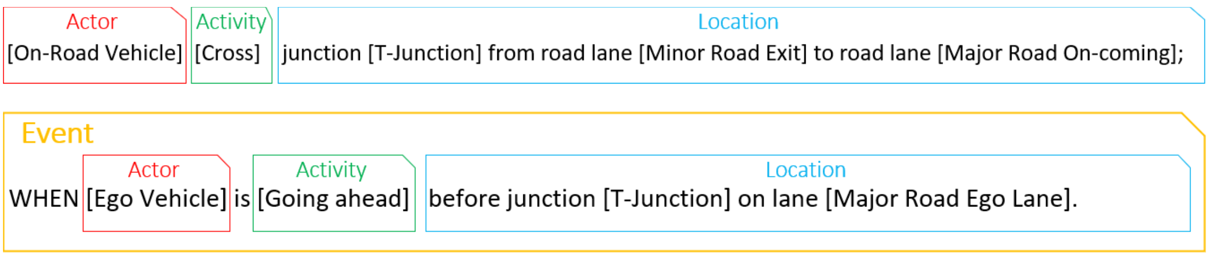


Figure 4 – SDL Statements Mapped to Sentence Structure

The following sections will describe how all these upper-level concepts can be linked to domain-level concepts for defining individual scenarios. The dynamic element of SDL is used to form the ‘actors’ and describe how they can ‘perform’ their activities. To define activities, manoeuvre phases which span across the time dimension will be used for each actor. Subsequently, scenery elements (‘location’) and environment elements (‘ambient condition’) are specified around them to create a unique scenery and environmental setting. The scenery elements (e.g., junction/road/lane descriptions) will be referenced within the dynamic elements, as seen in the 'Location' section of Figure 4.

5. Domain-Level SDL Concepts

Within the domain-level, several main SDL concepts have been identified and developed for describing a scenario. At a high level, they can be categorised into the following:

- **Scripted Dynamic Elements (individual actor)**
  This consists of the descriptions of the behaviours as well as the actor types for those actors whose behaviour characteristics need to be explicitly controlled.
- **Non-scripted Dynamic Elements (macroscopic traffic)**
  This contains the description of the macroscopic traffic behaviour that surrounds the vehicle under test, the behaviour of the individual actors within the traffic will not be described.
- **Scenery Elements**
  These include all the traffic infrastructures, road topology, junction details and road structures, this also incorporates temporary changes.
- **Environment Elements**
  These describe the global ambient conditions, including weather conditions, lighting conditions, air particulates as well as data & communication conditions.
- **Ego Vehicle**
  This refers to the vehicle under test (VUT) or subject vehicle, SDL will not provide detailed behaviour description for the VUT as this will be determined by the corresponding vehicle control software.

5.1. Dynamic Elements

Dynamic elements provide the information related to the behaviours of the actors as well as the actor categories. The actor categories include road users (i.e., those who can legally travel on the road such as vehicles, animal riders), pedestrians and animals. For the behaviour aspect, they are defined using a combined information of **relative manoeuvres** and **absolute manoeuvres**:

- **Relative manoeuvres** refer to behaviours that indicate the relation between one actor and another actor or a scenery object, such as *cut-in, towards.* Such manoeuvres cannot be completed by one single actor alone.
- **Absolute manoeuvres** refer to the manoeuvres that can be performed by a single actor without any relation to others, such as *drive, turn left.* Such manoeuvres can be directly translated to vehicle control input parameters and applied to the intended actor.

The concept of dynamic elements can be split into two main categories: **scripted traffic** and **non-scripted traffic**. The following section will explain both categories in details.

5.1.1. Scripted Traffic

For the SDL scripted traffic, the full dynamic behaviour description is included - it is usually required for the non-ego actors who directly influence the behaviour of the ego vehicle. The description will define the behaviours relative to the ego vehicle, other non-ego actors and scenery objects within a scenario.

Figure 5 illustrates the categorisations of the actor types. This categorisation should serve as the foundation or base model, it is **extensible** based on users' requirements. For example, animals could be sub-categorised into horses, cows or dogs etc. Same applies for cars, where it could be further categorised into estate or coupe etc. The majority of the provided vehicle classes fall under the on-road vehicle category as they are intended for manoeuvring over roads, junctions, and other types of purposely designed drivable areas. One thing to note is that under the Road Vehicles (and Uses) Regulation 1986, amended 2003, bicycles as well as motorcycles are also categorised as vehicles.

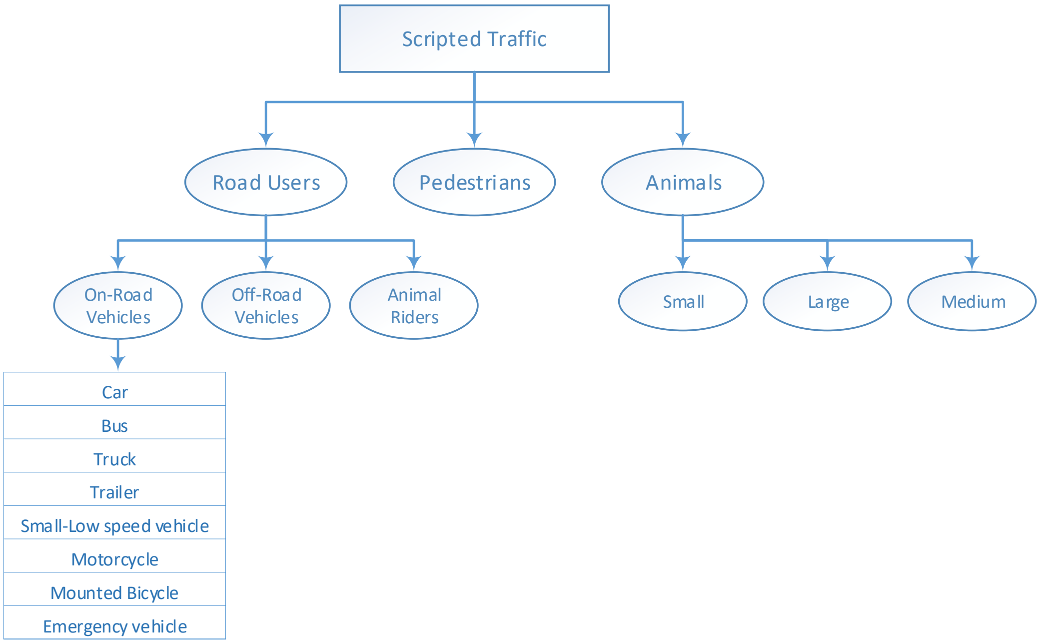


Figure 5 - SDL Scripted Traffic Categorisation

5.1.1.1. Road User Manoeuvres

By investigating into the expected behaviour of on-road vehicles such as cars, motorcyclists, trucks, buses etc, a set of absolute manoeuvres has been identified, as shown in Table 1. These absolute manoeuvres can be performed by the individual vehicle without considering their relations to other actors or scenery objects, and they can be translated directly into vehicle control input parameters (such as throttle, steering angle and brake) for execution across different environments.

ID	Absolute manoeuvres
1	Drive
2	Stop
3	Lane change right
4	Lane change left
5	Turn right
6	Turn left
7	Reverse

Table 1 - Road User Absolute Manoeuvres

However, absolute manoeuvres alone are not sufficient to uniquely describe the actor behaviour if no initial conditions or previous phases exist. For example, in Figure 6a-d, all the manoeuvre cases can be referred to as V1 turning right since V1's relation to V2 or the crossroad junction are not described. To remove this ambiguity, two other parameters are introduced: **relative manoeuvre** and **relative heading angle**.

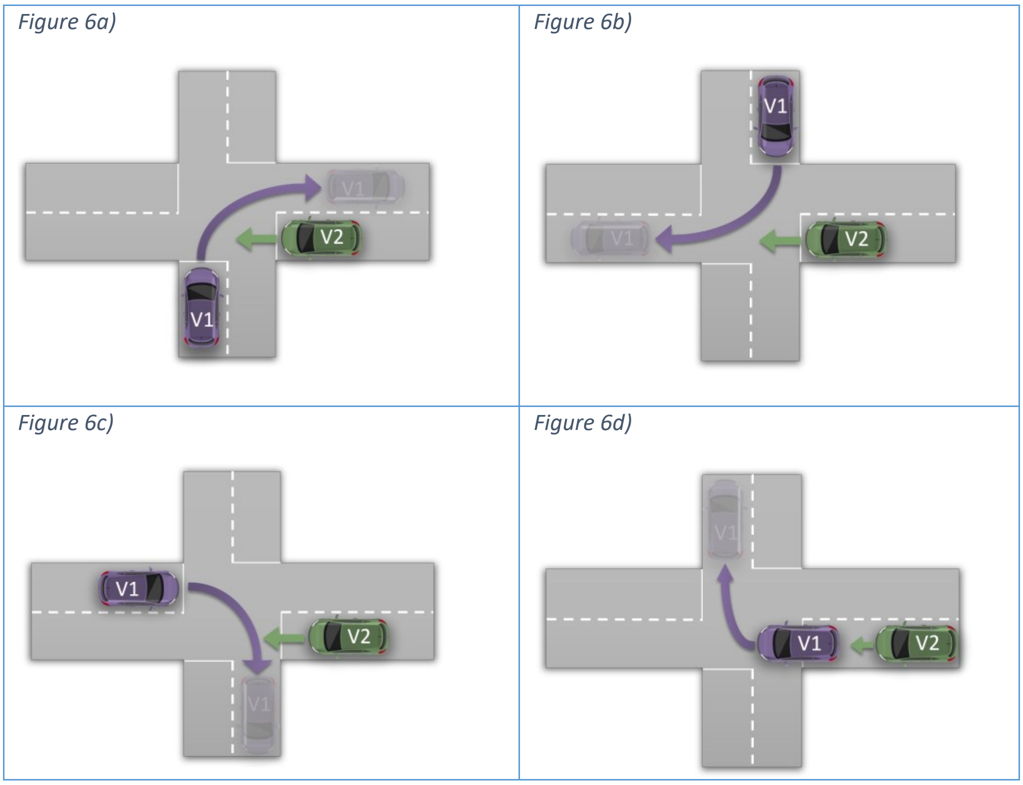


Figure 6 - Absolute Manoeuvre Example

Figure 7 (a-c) illustrate how adding relative manoeuvre and relative heading angle can result in the scenario shown in Figure 6a. As can be seen, Figure 7a displays a scenario where an absolute manoeuvre is used to describe V1; it is not related to V2 at this stage. Figure 7b further adds the relative heading angle between V1 and V2 and, given the heading direction of V2, the heading direction of V1 can be obtained. However, even with absolute manoeuvre and relative heading angle specified, V1 can still be either driving towards V2 or away from V2 (as seen in Figure 7b). Figure 7c further includes the relative manoeuvre of V1 by stating that V1 is driving towards V2 before the turning, which enables a singular scenario to be described.

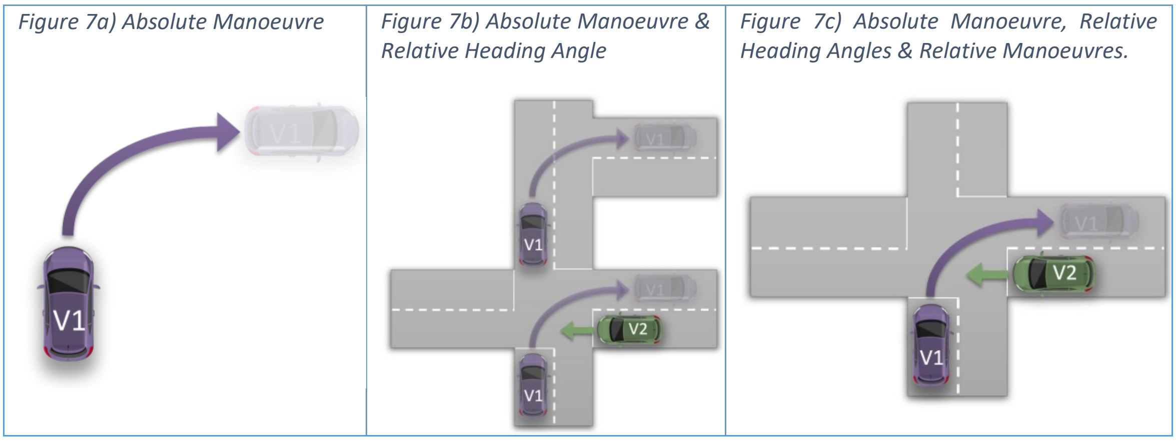


Figure 7 - Illustration of Scenario Concretisation Levels

After adding the relative manoeuvres and relative heading angles, the unique descriptions of the four examples shown in Figure 6 can be described as below:

- Figure 6a: V1 turning right, cut-in and then cut-out of V2's lane, V1 has an initial relative heading angle of 270 degree to V2, and a final relative heading angle of 180 to V2.

- Figure 6b: V1 turning right and cut-in of V2, V1 initial relative heading angle is 90 degree and final heading angle is 0 degree to V2.
- Figure 6c: V1 turning right, cut-in and then cut-out of V2, V1 initial relative heading angle is 180 and final heading angle is 90 degree to V2.
- Figure 6d: V1 turning right and cut-out of V2, V1 initial relative heading angle is 0 degree and final heading angle is 270 degree to V2.

Table 2 summarises the heading angle and relative manoeuvre types for the road users that are used for both SDL level 1 and level 2. For the relative heading angle, the qualitative descriptions (i.e., same direction, opposite direction, etc) are used for SDL level 1, and the actual value ranges are used for SDL level 2. For the relative manoeuvre type, both levels share the same descriptions.

Relative heading angle
Same direction – 0 degree
Opposite direction – 180 degree
Perpendicular direction – 270 or 90 degree
Arbitrary input angle

Relative manoeuvre type
Cut-in
Cut-out
Towards
Away
Cross

Table 2 - Road Users Relative Heading Angle and Relative Manoeuvre Types

To further establish a common relative heading angle definition for the SDL and for users to easily identify relevant values, the heading angle compass is introduced, as shown in Figure 8. In Figure 8, the black vehicle marked by ‘Vx’ is the reference vehicle, and the purple vehicle marked by ‘V1’ has a relative heading angle to Vx. Hence if taking the V1 car that’s at the 90° rotation for example, one could say that “V1 has a relative heading angle of 90 degrees to Vx”.

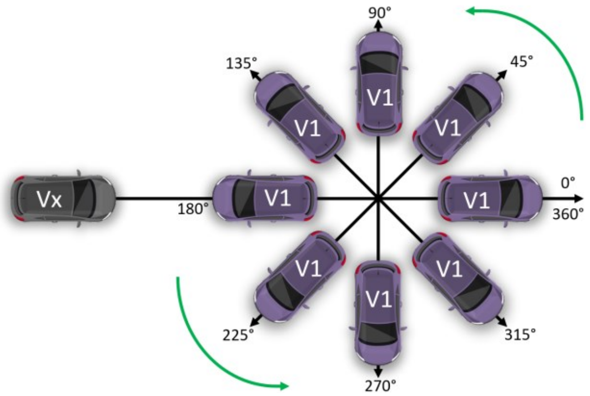


Figure 8 - Road User Relative Heading Angle Compass

Another key element for describing the road user manoeuvre is the relative position. The following example will demonstrate that without the relative position information it will not be able to instantiate unique cases for certain scenarios. Figure 9 illustrates two different scenarios of lane change left cut-in with the same relative heading angle between V1 and V2. In order to differentiate between the two, the relative position compass is introduced as shown in Figure 10.

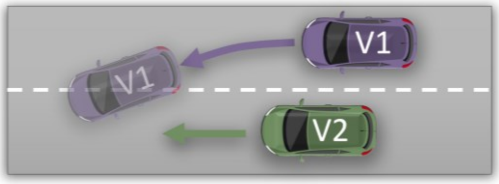
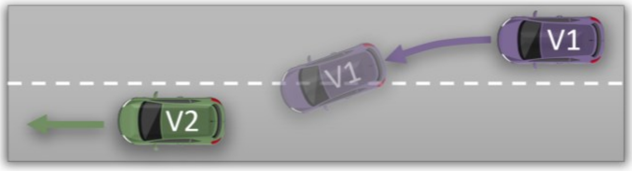
Figure 9a)	Figure 9b)
	

Figure 9 - Two Potential Variations of the Cut-In Manoeuvre

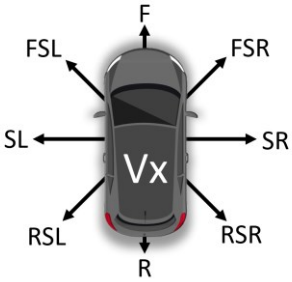


Figure 10 - Vehicle Relative Location Compass

The relative position compass is used to specify the position of one actor to another. Within the relative position compass, *F* corresponds to the front position, *FSR* is front side right, *SR* is side right, *RSR* is rear side right, *R* is rear, *RSL* is rear side left, *SL* is side left and *FSL* is front side left. After applying the relative position, Figure 9a becomes V1 is changing lane left cut-in at FSR position to V2, and Figure 9b becomes V1 is changing lane left cut-in at RSR position to V2.

So far, the concepts of absolute manoeuvre, relative manoeuvre, relative position, and relative heading angles have been introduced. Figure 11 summaries their combined effect for defining scenarios using three examples.

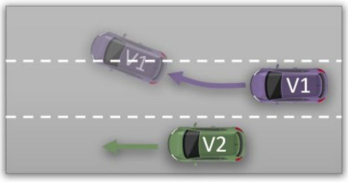
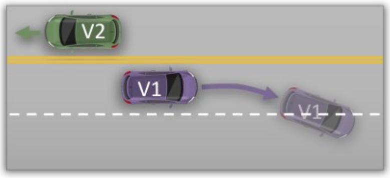
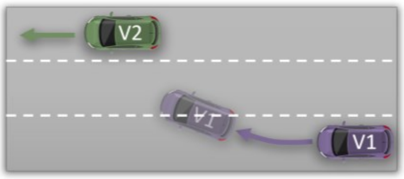
Figure 11a)	Figure 11b)	Figure 11c)
		
Lane change right, Cut-out, RSR (**315°** between V1 and V2 at mid-point of manoeuvre)	Lane change right, Cut-out, RSL (**135°** between V1 and V2 at mid-point of manoeuvre)	Lane change right, Cut-in, RSL (**315°** between V1 and V2 at mid-point of manoeuvre)

Figure 11 - Three Examples Illustrating The Concepts Of Absolute Manoeuvre, Relative Manoeuvre, Relative Position And Relative Heading Angle

After combining an absolute manoeuvre with a relative manoeuvre, a combine SDL manoeuvre type is created. In SDL level 2, combined manoeuvres are used to describe actor behaviours. Table 3 illustrates the complete SDL manoeuvre matrix. There are 33 manoeuvre matrix IDs, the first 30 rows are the results of combining the absolute and relative manoeuvres, the last 3 rows indicate collide and stop manoeuvre as well as the stopped state, which are manoeuvre matrix IDs individually.

No.	Manoeuvre matrix ID	Absolute manoeuvre	Relative manoeuvre
1	Drive_CutIn	Drive	Cut-in
2	Drive_CutOut	Drive	Cut-out
3	Drive_Towards	Drive	Towards
4	Drive_Away	Drive	Away
5	Drive_Cross	Drive	Cross
6	LaneChangeRight_CutIn	Lane change right	Cut-in
7	LaneChangeRight_CutOut	Lane change right	Cut-out
8	LaneChangeRight_Towards	Lane change right	Towards
9	LaneChangeRight_Away	Lane change right	Away
10	LaneChangeRight_Cross	Lane change right	Cross
11	LaneChangeLeft_CutIn	Lane change left	Cut-in
12	LaneChangeLeft_CutOut	Lane change left	Cut-out
13	LaneChangeLeft_Towards	Lane change left	Towards
14	LaneChangeLeft_Away	Lane change left	Away
15	LaneChangeLeft_Cross	Lane change left	Cross
16	TurnRight_CutIn	Turn right	Cut-in
17	TurnRight_CutOut	Turn right	Cut-out
18	TurnRight_Towards	Turn right	Towards
19	TurnRight_Away	Turn right	Away
20	TurnRight_Cross	Turn right	Cross
21	TurnLeft_CutIn	Turn left	Cut-in
22	TurnLeft_CutOut	Turn left	Cut-out
23	TurnLeft_Towards	Turn left	Towards
24	TurnLeft_Away	Turn left	Away
25	TurnLeft_Cross	Turn left	Cross
26	Reverse_CutIn	Reverse	Cut-in
27	Reverse_CutOut	Reverse	Cut-out
28	Reverse_Towards	Reverse	Towards
29	Reverse_Away	Reverse	Away
30	Reverse_Cross	Reverse	Cross
31	Collide	Collide	
32	Stop	Stop	
33	Stopped	Stopped	

Table 3 - Road User Manoeuvre Matrix

In SDL level 1, users can further attach qualitative speed descriptions to each of the manoeuvre IDs; the available qualitative descriptions can be used for either absolute or relative speed. For relative speed they are [*accelerating, speeding up, at constant speed, decelerating, slowing down*]. At the SDL level 2 abstraction level, the speed will be described with a value range instead. The example below shows how users could combine the manoeuvre matrix ID with the speed description to reach the corresponding SDL level 1 description:

Vehicle1 **drives, accelerating towards** Vehicle2 at its front left.

So far, all the elements related to the road user manoeuvres have been introduced, which are: **absolute manoeuvres, relative manoeuvres, relative positions, relative heading angles, absolute speed, relative speed.**

In SDL level 2 there is an expectation that manoeuvres will be immediately executable. In order to execute these manoeuvres in a consistent and repeatable way they need to be defined clearly from a semantic perspective. Table 4 describes the semantics for absolute manoeuvres.

Absolute Manoeuvre	Active Component	Terminating condition (when WHILE condition isn't stated)	Holding Component
Drive	Accelerating to target speed.	Reaching target speed.	Continue ahead at target speed.
Lane Change Right	Vehicle trajectory changes towards the neighbouring lane on the right. Accelerating to target speed (if necessary).	Vehicle is positioned one lane to the right, in the lane centre.	Continue ahead at target speed in new lane.
Lane Change Left	Vehicle trajectory changes towards the neighbouring lane on the left. Accelerating to target speed (if necessary).	Vehicle is positioned one lane to the left, in the lane centre.	Continue ahead at target speed in new lane.
Turn Left	Vehicle trajectory will arc into the equivalent direction lane of a road positioned to its left.	Vehicle is positioned in the target road in the centre of the target lane.	Continue ahead at target speed in new road.
Turn Right	Vehicle trajectory will arc into the equivalent direction lane of a road positioned to its right.	Vehicle is positioned in the target road in the centre of the target lane.	Continue ahead at target speed in new road.
Reverse	Accelerating to target speed.	Reaching target speed.	Continue at target speed.
Stop	Vehicle slows to 0 speed (if necessary) or is stationary.	Vehicle is stationary.	Vehicle is stationary.
Stopped	Vehicle is stationary.	Vehicle is stationary.	Vehicle is stationary.

Table 4 - Semantic Meaning of Absolute Manoeuvres.

5.1.1.2. Pedestrian/Animal Manoeuvres

Pedestrians and animals are the main types of vulnerable road users (VRUs) and hence being able to describe their behaviours within scenarios is of key importance. Following the same manoeuvre concept for the road users dynamic description, the pedestrians/animals manoeuvres can also be divided into absolute and relative. Three basic types of absolute manoeuvres for pedestrian have been identified, which are **walk**, **run** and **slide**; and three types of relative manoeuvre have been identified, which are **cross** (cross agent's lane), **away** (moving away from agent) and **towards** (moving towards agent). In addition, 'stop' is also included as a standalone manoeuvre type. Their combinations are shown in Table 5.

No.	Manoeuvre matrix ID	Absolute manoeuvre	Relative manoeuvre

1	Walk_Towards	Walk	Towards
2	Walk_Cross	Walk	Cross
3	Walk_Away	Walk	Away
4	Run_Towards	Run	Towards
5	Run_Cross	Run	Cross
6	Run_Away	Run	Away
7	Slide_Towards	Slide	Towards
8	Slide_Cross	Slide	Cross
9	Slide_Away	Slide	Away
10	Stop	Stop	

Table 5 - Pedestrian/Animal Manoeuvre Matrix

Similar to the road user manoeuvres, the relative position and the relative heading angle of the pedestrians are also necessary in order to describe their behaviour. However, due to the low-speed nature of most pedestrian/animal manoeuvres, their relative speed becomes an optional parameter and only the absolute speed is compulsory. To summarise, all the elements needed for the pedestrian/animal manoeuvre descriptions are **absolute manoeuvres**, **relative manoeuvres**, **relative positions**, **relative heading angles** and **absolute speed**.

Figure 12 is used to illustrate these elements, assuming that the pedestrian is stationary. In this example, the absolute manoeuvre and relative manoeuvre will be 'Stop', the relative position to V1 will be 'F', the relative heading angle to V1 will be 45 degrees and the absolute speed will be 0.

Figure 12 - Example of a Snapshot Involving a Pedestrian and a Vehicle

5.1.2. Non-Scripted Traffic

The increasing demand for more realistic simulations [22] to enhance the testing capabilities has raised the requirements of more intelligent actors to be represented in scenario executions and thus also in scenario definitions. To cater to this requirement, in a simulation setting, the non-scripted (goal based) traffic description is used to define intelligently controlled traffic, which can be based on AI decisions or surveillance camera recordings.

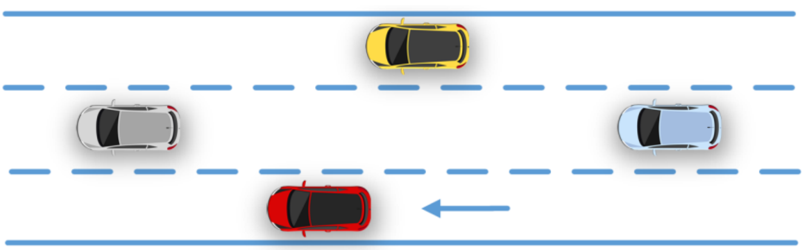


Figure 13 - Example of Non-Scripted Traffic

Figure 13 illustrates an example of a simple non-scripted traffic scenario and within this figure all the four vehicles are non-scripted vehicles. A direct observation of the characteristics of this traffic could include the distance/range (200m), the number of vehicles involved (4 vehicles) and vehicle average speed (70 mph). These properties will then be further converted into the defining characteristics that make up 'traffic' – density, volume and speed. **Density** defines how many vehicles on average are expected within the observed road section, which is measured as vehicles per distance. **Volume** defines the rate of vehicle appearance/disappearance at the observed road section, which is measured as vehicles per duration of time. And **speed** defines how fast the vehicle will travel, which is measured as distance per time. One can derive the third parameter by knowing the other two parameters, since *volume = density × speed*. However, if only one parameter is known then the traffic will not be uniquely instantiated. Table 6 illustrates cases where different metrics of traffic share the same value when only one parameter is specified.

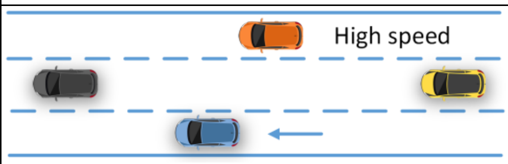
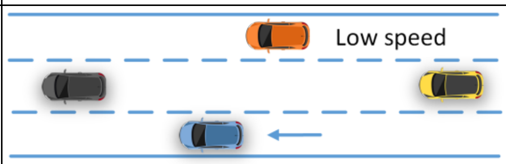
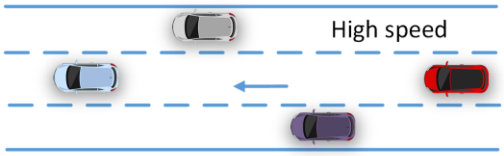
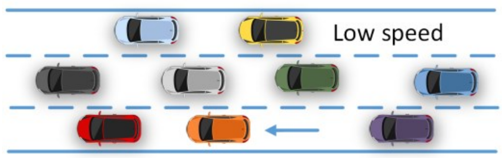
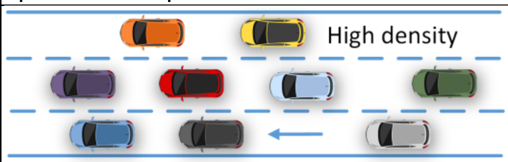
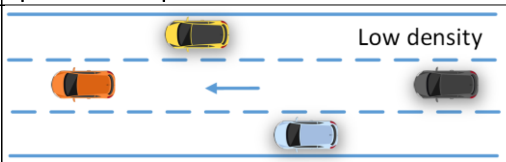
	Traffic 1	Traffic 2
Density @ 4 per section	 High speed Speed = 70 mph	 Low speed Speed = 10 mph
Volume @ 1 per minute	 High speed Speed = 70 mph	 Low speed Speed = 10mph
Speed @ 70 mph	 High density Density = 9 per section	 Low density Density = 4 per section

Table 6 - Illustration of Traffic Situations When Only one Parameter is Specified

To construct the description for non-scripted traffic behaviour, the speed and volume are used in the SDL concept. In addition, the start and end points of the traffic are also required, these allow the placement of the traffic within the map environment. Actor types, or the composition of the macroscopic traffic, and the presence of special vehicles can also be specified to increase the fidelity of the described traffic.

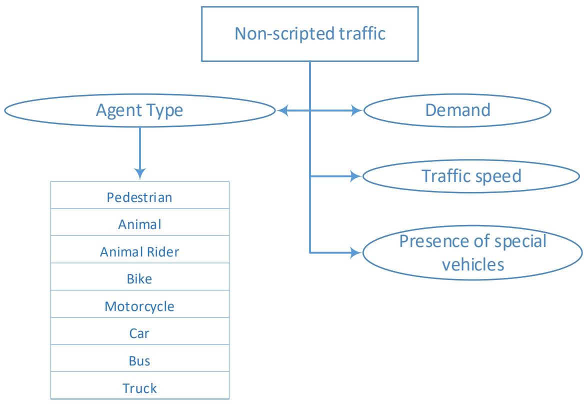


Figure 14 - Non-Scripted Traffic Parameters

5.2. Scenery Elements

Figure 15a) *Figure 15b)* *Figure 15c)*

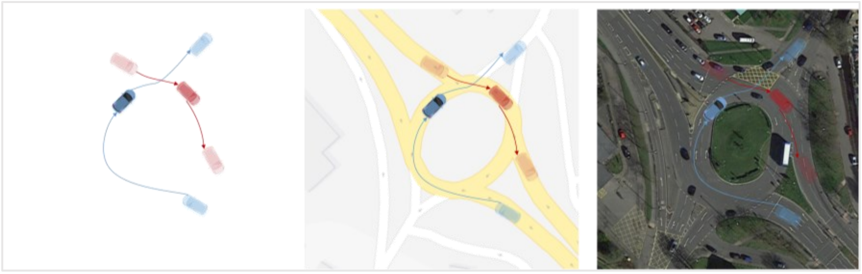


Figure 15 - Scenery Information Layers Example

The scenery elements provide descriptions of the static and quasi-static elements within a scenario using junctions and roads as the building blocks. Static elements refer to the scenario elements that cannot change state or position (such as buildings, dustbin), whereas quasi-static elements refer to the scenarios elements that can change state but cannot change position (such as automated toll bar, gate). Figure 15 illustrates the graphical representations for the two different levels of the scenery elements. Figure 15a displays two vehicles manoeuvre within an empty space without any defined scenery elements. Figure 15b provides only the road geometries and junction connections; this corresponds to the concept of SDL level 1 abstraction aimed at function scenario (or use cases) level. Figure 15c adds more details to the scenery elements such as lane specifications, road structures, traffic control, roundabout details etc, this corresponds to the detail level of SDL level 2 aimed at logical and concrete scenarios (or test case) level. By providing the scenery descriptions, the question of 'where does the scenario take place?' can be addressed.

In theory, SDL level 2 scenery elements could contain a large number of attributes in order to represent real world scenarios. However, for a scenery to be relevant, it needs to be part of the Operational Design Domain (ODD) of the ADS. ODD refers to the operating conditions underwhich an ADS can perform safely [23]. In order to effectively describe the scenery while maintaining a compact format, the attribues from the BSI PAS 1883 ODD taxonomy have been used to form part of the domain model for the SDL, the high-level elements are shown in Figure 16.

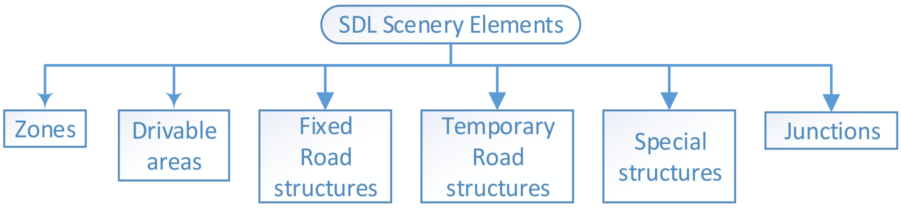


Figure 16 – Scenery Elements Main Categorisation

5.2.1. Zones

The **zones** element specifies any special areas that might differ from the typical road conditions. This could be due to local regulatory requirements (such as low-emission zone), or ambient conditions applied to a local area (GPS interference zone). The **zone** element will include but is not limited to: Geo-fenced areas; Traffic management zones; School zones; Regions or States and Interference zones (e.g., dense foliage or loss of GPS due to tall buildings).

5.2.2. Drivable Area

The **drivable area** defines the layout and the surface characteristics of roads and other designated drivable spaces. For any individual section of a road, it provides a description of the elements that are directly related to the road users' manoeuvrability. At a high level, the **drivable area** includes: the road type; geometry; lane specification; roadway edge features and road surface types/conditions, as shown in Figure 17.

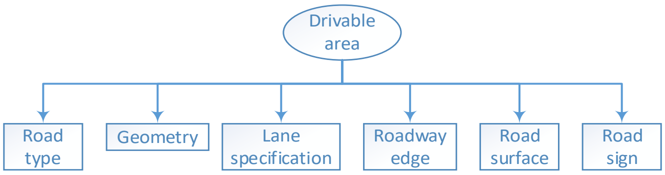


Figure 17 – Scenery Drivable Area Categorisation

5.2.2.1. Road Type

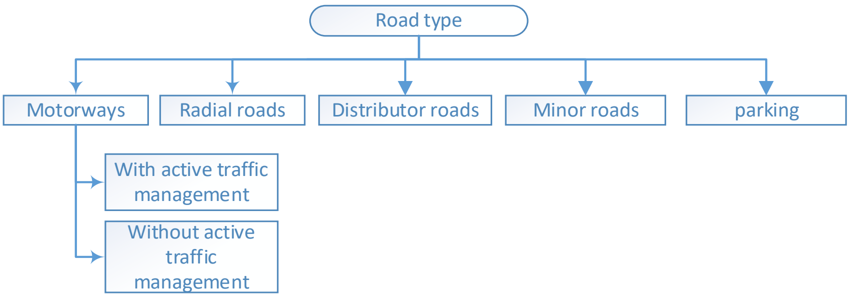


Figure 18 – Road Type Categorisation

Within the **drivable area** category, the **road type** element can be further divided into five main types: Motorways; Radial roads; Distributor roads; Minor roads and Parking.

Motorways are high traffic roads where non-motorised vehicle and pedestrian are prohibited. Radial roads (A-roads) are high density traffic roads that connect motorways to distributor roads. Distributor roads (B-roads) connect A-roads with minor or local roads, generally having low to moderate capacity. Minor roads or local roads provide access to residential areas and other local developments. Furthermore, motorways are classified as 'with activate traffic management' or without.

5.2.2.2. Geometry

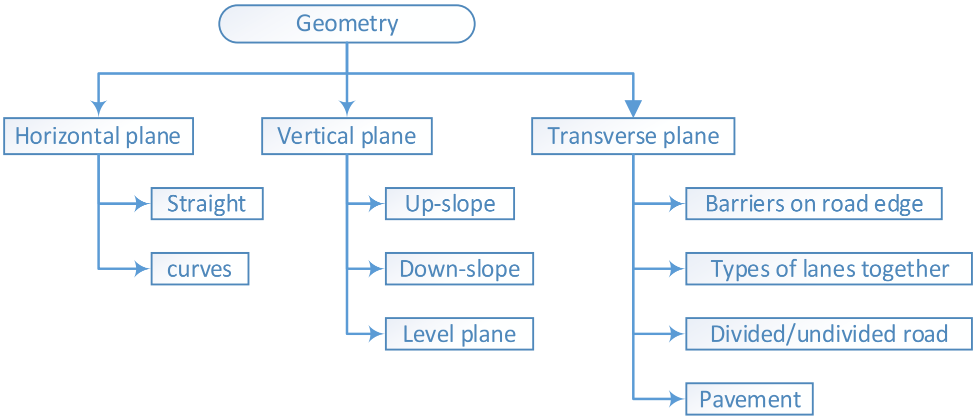


Figure 19 – Geometry Categorisation

The **geometry** defines the shape of the road layout in three-dimensional space, which consists of the horizontal plane, the vertical plane and the transverse plane. The horizontal alignment can be visualised as the road being projected onto the horizontal plane. In this plane the road layout can be classified as a straight or curvy road. If curved, the curvature will need to be specified. The vertical alignment can be visualised as the elevation property of the road central line, in this alignment it can be classified into up-slope, down-slope and level plane with their corresponding slope values being specified in the SDL description. The transverse plane can be visualised as the cross section of the road layout, consisting of the road edge, lanes, lane markings, etc. In addition, the banking angle can also be specified within the transverse plane description.

5.2.2.3. Lane Specification

Within the SDL, individual roads can be further broken down into lane components. To define the lane specification, the following characteristics need to be specified:

1. Lane dimensions, such as lane width.
2. Lane markings (broken line, solid line etc).
3. Lane type, which can be further categorised into bus lane, traffic lane, cycle lane, tram lane, emergency lane or another special lane.
4. Number of lanes in the road.
5. Traffic direction (left hand, or right hand).

5.2.2.4. Road Sign and Roadway Edge

All the road signs will be classified into three main groups based on their functionality: information signs, regulatory signs, and warning signs. Furthermore, they can be variable (such as electronic signs) or uniform, full-time or temporary. The road signs individual sign name along with the categories and sign properties need to be specified.

The roadway edge illustrates the details at the outer boundary of the road element. Possible roadway edge parameters include line markers, road shoulder (paved/gravel/grass), roadway edge barriers (grating/rail/curb/cones etc) and temporary line markers.

5.2.2.5. Road surface

The road surface indicates the condition of the road, the surface features and the surface type. The road surface conditions can be weather induced conditions such as icy roads, flooded roadways, mirages, snow, standing water or wet road. The road surface features may include damages caused by the traffic such as cracks, potholes, ruts and swells. The road surface type may include loose surface, segmented or uniform surface.

5.2.3. Junctions

Within an SDL scenery description, all the information is composed in the form of a 'roads and junctions' network. As the elements for road descriptions are already covered in the **drivable** area, this section will cover the elements for junction description. At a high level, **junctions** can be divided into **intersections** and **roundabouts.** As shown in Figure 20, an intersection can be further classified into T-junction, staggered, Y-junction, grade separated, straight and crossroads. Roundabouts can be further classified into normal roundabouts, large roundabouts, compact roundabouts, double roundabouts, and mini roundabouts. Furthermore, both 'intersections' and 'roundabouts' need to specify whether they are signalised or non-signalised.

Within a roads and junctions network each junction links two or more roads together, hence it is important to specify the roads and lanes connection as well as the connecting angles between them when defining junction characteristics. With the individual road characteristics defined, as well as their connection details and junction characteristics defined, the complete scenery settings can be described.

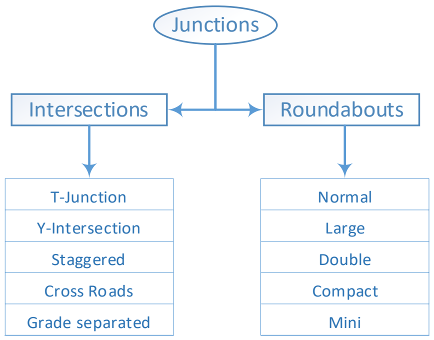


Figure 20 – Scenery Map Junctions Categorisation

5.2.4. Special Structures, Fixed Road Structures, Temporary Road Structures

Road structures are the features that are not required to construct the logical and drivable roads and junctions network. However, in a real-world scenario scenery setting, they are important ODD features which might impact the driving behaviours.

Three categories of structures are considered in SDL and also in the PAS 1883 taxonomy: special structures, fixed road structures and temporary road structures (as shown in Figure 21). Special structures are those permanent structures that can be placed directly on the drivable area, such as bridges, rail crossings and a toll plaza. Fixed road structures are the structures that are not drivable by road users, but can contribute towards the scenery settings, such as streetlights, buildings. Temporary structures could be placed both on the drivable area or non-drivable area and they contain objects that have a temporary nature. Please note that this temporary nature might not necessarily need to be reflected within one scenario, but rather it represents the common understanding of such a group of objects in the real world, with examples containing road works or a refuse collection.

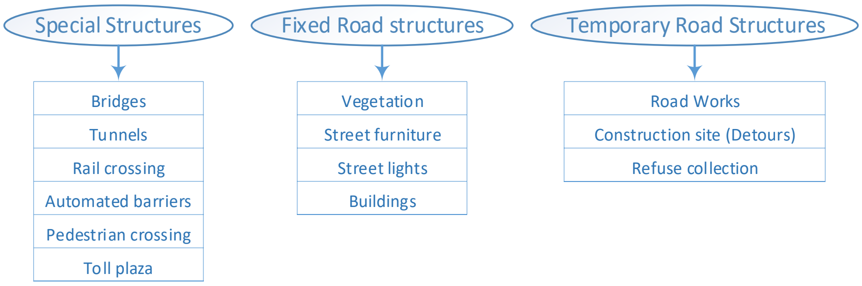


Figure 21 – Road Structures Categorisation

5.3. Environment Elements

Environment elements are considered as the last layer of information to be defined by SDL. The environment elements are classified into weather, particulates, illumination, connectivity, and time, as illustrated in Figure 22:

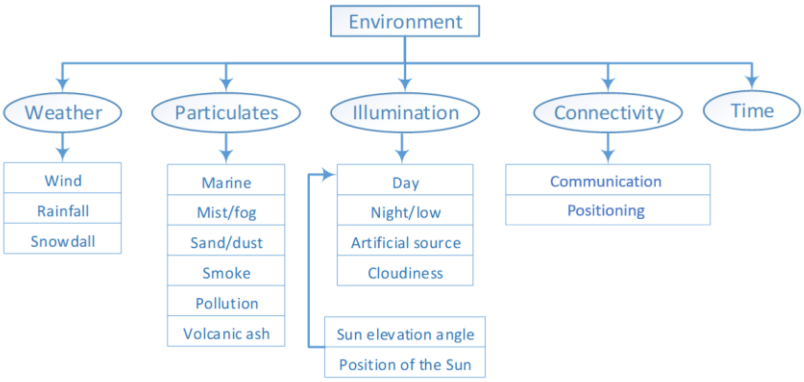


Figure 22 – Environment Elements Categorisation

Within the weather category, it includes wind, rainfall and snowfall. This can be further detailed into different levels using their measuring metrics, such as heavy rain or light rain. The illumination includes daylight or night, and for daylight cases where the Sun is the light source, the Sun's elevation angle and position also need to be specified. Cloudiness will also need to be described using the 'okta' measuring unit which defines the amount of cloud covers at any given location.

5.4. Ego Vehicle

The SDL (scripted and non-scripted) dynamic elements described so far are intended to define the behaviours of scripted actors. Scripted actors represents the vehicles, pedestrians and animals whose behaviours are completely defined within the scenarios; the scripted actors are used to set the test situations for the vehicle/system under test. On the other hand, the vehicle/system under test is referred to as the **Ego Vehicle** in the SDL concept. The SDL is not intended to control the ego vehicle behaviour but can only influence its behaviour and might act dynamically based on its reactions.

As can be seen in Figure 23, the ego vehicle's observed behaviour can be described in a similar way to agent vehicle absolute manoeuvres. It consists of **going ahead, stopped, reversing, lane**

changing (right/left), turning (left/right). In addition, the location and speed of the ego vehicle can also be specified, in SDL level 1 they will be described qualitatively whereas in SDL level 2 they will be described quantitatively.

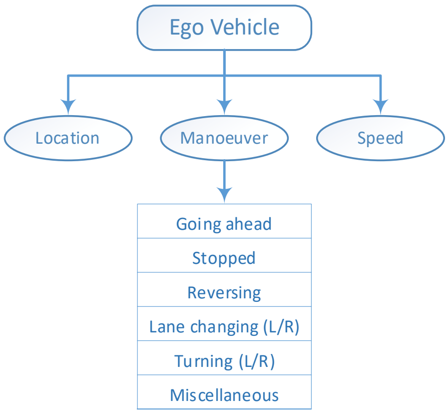


Figure 23 – Ego Vehicle Observed Information

The Ego vehicle's observed behaviour description can be formed using the same SDL statement format as with the actors (agents), after replacing the actor (e.g., V1, V2, PD1) with 'Ego' vehicle as shown below:

When Ego is doing action in world…

Example:

When Ego is stopped ahead of T-Junction1 and the traffic light ahead is green.

Such complete (heavily scripted) Ego observed behaviour descriptions are normally used as the triggers for the scripted agents' behaviours. Furthermore, within each activity phase of an agent's behaviour, the relative speed and relative position can be set relative to the Ego vehicle, and they can be specified as the expected observed behaviour of the Ego. The next section will illustrate how all the domain elements can be used to for SDL description.

6. SDL Format

So far, the SDL domain concepts covering dynamic, scenery and environment aspects have been introduced, which formed the domain model behind the language concept. In this section, how these domain concepts can be used to form individual descriptions will be illustrated. This includes the SDL format structure and logic for all three main elements: dynamic, scenery and environment.

6.1. Dynamic Description Format (Scripted Traffic)

6.1.1. Road User Behaviour Description Format

For the behaviour description, the overall structure contains two parts: initialisation phase and activity phases.

The initialisation phase sets out the initial road and lane for each actor. The relative heading angles and relative positions between actors can also be specified.

For the activity phases, a behaviour tree style description format is utilised. An activity phase consists of a manoeuvre phase, as well as a trigger condition. If the manoeuvre is omitted in one phase, it is assumed that the previous manoeuvre is still being performed under a holding condition. Each actor contains one or more activity phases in a sequential relation to represent the temporal progression; when two actors are performing activities at the same time the activity phases between the two actors are in a parallel relation.

Figure 24 illustrates the logic of an SDL behaviour description consisting of three actors, where the first two actors each have two activity phases and these two actors are performing actions at the same time. In the 1st Actor's description for phase 1 (Activity 1-1), there is a manoeuvre happening, caused by the trigger 1-1. In the 2nd Actor's description for phase 1 (Activity 2-1), there is a parallel manoeuvre performed by this actor. Then in the 2nd Actor's description for phase 2 (Activity 2-2), there is just another manoeuvre. The third actor starts to perform activities after the first two actors have stopped for the remaining of the scenario. The 4th Actor has asynchronous activity blocks to actors 1, 2 and 3, thus showing how asynchronous phases are possible. Such behaviour style dynamic description can also be utilised during implementation to monitor and check whether the intended scenario has been executed at runtime.

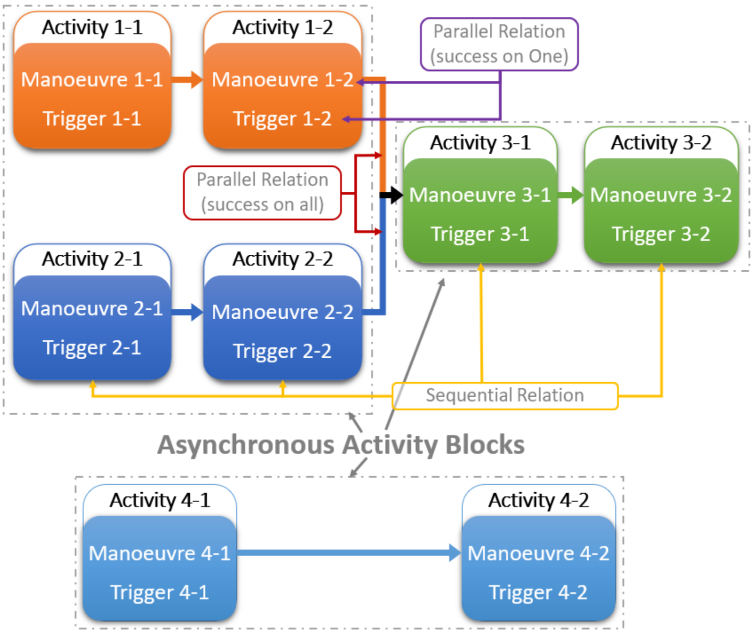


Figure 24 - Dynamic Behaviour SDL Definition Architecture

The term **event** in the SDL concept represents the instantaneous snapshots where trigger conditions are met. The list below shows common types of triggers:

- **Time based** [Example: after certain time duration]
- **Actor's dynamic behaviour** [Example: when Ego vehicle reaches 30mph]
- **Actor's location on map** [Example: when Ego vehicle is 2-3 meters from Junction J1]
- **Scenery elements' event** [Example: when Traffic light T1 is Green]
- **Environment element's event** [Example: when precipitation is rain]
- Or combination of one or more types joined by logical operators of AND, OR and NOT.

From a semantic perspective, the dynamic events of a scenario can be characterised as follows. Each actor's 'activity' (as in Figure 24 above) is part of a Synchronised Serial Manoeuvre Sequence (SSMS), made up of a sequence of phased manoeuvres. In Figure 24, an SSMS is represented by the grey dotted box that groups the various actors' manoeuvre sequences together. The progression of the scenario through the SSMS, and in turn its phases, must be clearly defined for the language to be executable. It must be clear when describing a scenario how an actor might progress from one phase to the next, and in turn how an SMSS is initiated or terminated. The definition of this execution semantic is presented below.

Synchronised Serial Manoeuvre Sequence (SSMS):

- An SSMS may contain multiple manoeuvre sequences, each pertaining to an actor.
- Each sequence of an SSMS organises an actors behaviour into a series of phased manoeuvres.
- An SSMS has a WHEN trigger, which triggers all actors' contained within it, to begin with their respective Phase 1s.
- Phases across different sequences, but within the same SSMS, having identical index values, operate synchronously.
- Any two SSMSs taken together may operate asynchronously from each other.

Semantics of Phases:

- Within the same SSMS, the different actors begin identically numbered phases simultaneously.
- For an actor, a phase consists of a manoeuvre, manoeuvre parameters, and a WHILE condition that must hold true while that phase is in operation.
- Manoeuvres can be of different types. For simplicity, we call manoeuvres not having lateral motion (drive, reverse, stop) drive manoeuvres.
- All manoeuvres can be thought of as having two components, representing the manoeuvre action (active component) and the default action (holding/default component).
- The drive manoeuvres hence, can be viewed as having a 'speed change' action (if at all), and a 'speed maintain' action. If at the beginning of the driving manoeuvre, the desired speed is already achieved, then only the latter is relevant.
- The non-drive lateral manoeuvres can therefore now be thought of as having the drive manoeuvre 'speed maintain' component, as the default action component of their manoeuvre.

End of a Phase & Moving to the Next Phase:

- A phase is considered 'complete' once the active component of all actors' manoeuvres, in the same phase, have completed, OR when any WHILE condition linked to that phase is invalidated. Hence, so long as all WHILE conditions of the phase hold true, any actor having completed its active manoeuvre component continues with a default drive action until all actors in the same phase complete their respective active manoeuvre components.
- A drive-only phase is unique, and hence, in addition to the conditions above,
    - the completion of a phase containing a drive manoeuvre necessarily requires either a WHILE condition associated with the drive manoeuvre to become invalid OR
    - only in the case where there is no associated WHILE condition for the drive manoeuvre, earlier conditions apply (once the active 'speed change' component of the drive action is complete, the drive phase is considered complete, and no hold component is applied).

For SDL level 2, the syntax for the dynamic components of a scenario are defined below; cascading from a high level breakdown of the scripted agent behaviour format, to the individual syntax of each section.

SDL level 2 scripted agent behaviour format:

Format	**Initialisation (see syntax below)** **Synchronous Serial Manoeuvre Sequence (SMSS)+ (see syntax below)** **END (see syntax below)** **+ : as used above, indicates that at least one SMSS is necessary in the scripted agent behaviour, however there can be many SMSSs in a single scenario description. This section is repeatable.**

Table 7 - Format of Scripted Agent Behaviour

SDL level 2 Initialisation format:

Syntax	**INITIAL :** **A_TYPE [A_ID_1] in [R_ID_1.L_ID_1] (at [AP])? (at heading angle [AH])?** **AND A_TYPE [A_ID_2] in [R_ID_2.L_ID_2]** **(AND with a [D_1] offset of [O_1] to [A_ID_a] (AND with a [D_2] offset of [O_2] to [A_ID_b])?)?** **AND at relative position [RP] with relative heading angle [RA] to [A_ID_c]** **(AND Global timer [Tx] = [DOUBLE])?** **(AND Local timer [tx] = [DOUBLE])?**
Parameters	**A_ID_x**: Agent ID x. E.g., V1, V2, Ego, CTL2, CS1, CU1. **A_TYPE**: Actor type. E.g., Vehicle, Pedestrian, Cyclist **AP**: Absolute position described in two dimensions: **X , Y**. **AH**: Absolute heading angle described as a range H_1 **to** H_2. **R_ID_x.L_ID_x**: Road ID x and lane ID x. E.g., R1.L1, R2.L2. **RP**: Relative position described in the relative position compass. E.g., F, SR, FSR. **RA**: Relative heading angle described in the heading angle compass. E.g., 180, 270. **D_x**: Directional can either be longitudinal or lateral. **O_x**: Offset value, can be positive or negative to indicate direction. **Tx**: Global timer, which runs throughout the scenario. **tx**: Local timer, which runs within each manoeuvre phase. **?:** Indicate optional statement.

Table 8 - Road Users Initialisation Description Format SDL Level 2

SDL level 2 Synchronised Serial Manoeuvre Sequence (SSMS) format:

Syntax	**WHEN: [A_ID_x] is [M_ID] Context** **DO: SMS (see syntax below)** **Context (Position): in [R_ID_x.L_ID_x | J_ID]** **Context (Speed): at a speed of [DOUBLE RANGE]**
Parameters	**A_ID_x**: Agent ID x. E.g., V1, V2, Ego, CTL2, CS1, CU1. **M_ID**: Manoeuvre matrix ID. **R_ID_x.L_ID_x**: Road ID x and lane ID x. E.g., R1.L1, R2.L2. **J_ID**: Junction ID where manoeuvre is happening.

Table 9 - Syntax for describing the Synchronised Serial Manoeuvre Sequence.

SDL level 2 Serial Manoeuvre Sequence (SMS) format:

<table>
<tr><td>Syntax</td><td>PHASE X: [M_ID] [J_ID, AS, A] [A_ID_1: RS, RP]

(WHILE: Condition)?

Condition (Position): WHILE: [A_ID_2] [Directional] offset to [A_ID_3] is Op [O]
Condition (Timer) : WHILE: [tx] [Op] [T]</td></tr>
<tr><td>Parameters</td><td>M_ID: Manoeuvre matrix ID.
J_ID: Junction ID where manoeuvre is happening, ‘-‘ if not within a junction.
AS: Absolute speed of the scripted agent.
A: Acceleration of the scripted agent.
A_ID_x: Relative agent ID. The same scripted agent can be A_ID_1, A_ID_2 and A_ID_3.
RS: Relative speed of the agent relative to A_ID_1.
RP: Relative position to the agent relative to A_ID_1.
Directional: Lateral or longitudinal.
Condition: Position or Timer condition.
Op: Operators such as >, <, ==, !=, <=, >=.
O: Offset value, can be either positive or negative to indicate directions.
tx: Local timer, which runs within each manoeuvre phase.
T: Time value
?: Indicate optional statement.</td></tr>
</table>

Table 10 – Syntax for describing a Serial manoeuvre Sequence (contained within an SSMS).

SDL level 2 END format:

<table>
<tr><td>Syntax</td><td>END (: Dynamic condition)?

Dynamic Condition (Timer): [A_ID_x] Op [DOUBLE]
Dynamic Condition (Position): [A_ID_x] [Lateral | Longitudinal] offset to ‘[’ ID ‘]’ Op [DOUBLE]
Dynamic Condition (Collision): Collision
Op: < | <= | > | >= | == | !=</td></tr>
<tr><td>Parameters</td><td>Op: Operators such as >, <, ==, !=, <=, >=.
A_ID_x: Relative agent ID. The same scripted agent can be A_ID_1, A_ID_2 and A_ID_3.</td></tr>
</table>

Table 11 - Syntax for describing the 'End' or termination of a Scripted Agent Behaviour.

As illustrated in Table 10, the manoeuvre phase syntax contains three conditions: target position, target speed and target acceleration. The execution logic is designed that if target position and the target speed are not reached, then the target acceleration will be maintained. However, if target speed has been reached then the acceleration will be set to zero and the target speed will be maintained. If the target position is reached at any given time, the manoeuvre phase will be terminated, as displayed below:

<table>
<tr><td>Execution logic</td><td>IF relative position is not reached:
    IF target speed is not reached:
        A_ID M_ID with acceleration of a
    Else: A_ID M_ID with acceleration of 0
Else: End of M_ID</td></tr>
</table>

Table 12 - Road Users Manoeuvre Execution Logic

For SDL level 1, the initialisation and manoeuvre phases syntax follows a similar composition as the SDL level 2, however they contain qualitative descriptions and are at a more abstract level with less information. The structure is formatted to replicate natural language. A standard sentence structure is provided to describe initialisation and a separate standard sentence structure is provided to describe the manoeuvre. The sentence structure at this higher abstraction level has been made flexible, with multiple options for describing phases.

SDL level 1 scripted agent initialisation format:

Syntax	**There are/is INT ActorType(s), ID_1 ((, ID_2 …) and ID_x).** **IDx ((, IDx …) and IDx) are/is connectivity capable.** **ID_1 is in RoadID ((, ID_2 is in RoadID, …) and ID_x is in RoadID).**
Parameters	**IDx**: Agent ID x. E.g., Vehicle1, Vehicle2, Ego, ConnectedTrafficLight1. The ID may not have any spaces. **RoadID**: Road ID x. E.g., Road1, Road2… ActorType: i.e., vehicle, pedestrian, cyclist, connected structure. **(content)**: Brackets around *content* imply optional repetition of *content* depending on number of actors.

Table 13 - SDL Level 1: Road Actor Initialisation Description Format

SDL Level 1 scripted agent synchronised serial manoeuvre sequence

Syntax	**When** WhenCondition **[(,**WhenCondition**) and** WhenCondition**] (** ManoeuvrePhases **) .** **([Also, when / Next, when / Also, as / Next, as / Also, / Next,]** FirstManoeuvrePhase **(** OtherManoeuvrePhases **))** **OR** **ID operates** autonomously**.**
Parameters	$WhenCondition_i$, $ManoeuvrePhases_i$: Sequence of phases in the i^{th} manoeuvre for a vehicle. Different vehicles can synchronise their manoeuvre phases by placing them within the same synchronised manoeuvre sequence. **(content)**: Brackets around *content* implies optional repetition of *content* depending on number of manoeuvre phases and actors. **[content]**: Square brackets around *content* implies optional use, depending on the level of detail desired by the end user. In some scenarios this detail may not be necessary.

Table 14 - Scripted Agent Manoeuvre Sequence Format

Format: SDL Level 1 Scripted Agent Manoeuvre Phases

When Condition Syntax 1	**ID_1** Manoeuvre **[,** SpeedCondition **[,]]** ActionLocation **[(** PositionalPreposition **ID_2** **(,** PositionalPreposition **ID_3) and** PositionalPreposition **ID_4] [at its** RelativePosition**]**
When Condition Syntax 2	**the traffic light** ActionLocation **ID_1 is** TrafficLightColour
First Manoeuvre Phase	**, ID_1** Manoeuvre **[,** SpeedCondition **[,]]** ActionLocation **[(** PositionalPreposition **ID_2** **(,** PositionalPreposition **ID_3) and** PositionalPreposition **ID_4] [at its** RelativePosition**]**
Manoeuvre Phase Syntax 1	**[, / .] ID_1 [then]** Manoeuvre **[,** SpeedCondition **[,]]** ActionLocation **[(** PositionalPreposition **ID_2** **(,** PositionalPreposition **ID_3) and** PositionalPreposition **ID_4] [at its** RelativePosition**]**

When Condition Syntax 1	**ID_1** Manoeuvre **[,** SpeedCondition **[,]]** ActionLocation **[(** PositionalPreposition **ID_2** **(,** PositionalPreposition **ID_3)** **and** PositionalPreposition **ID_4] [at its** RelativePosition**]**
When Condition Syntax 2	**the traffic light** ActionLocation **ID_1 is** TrafficLightColour
First Manoeuvre Phase	**, ID_1** Manoeuvre **[,** SpeedCondition **[,]]** ActionLocation **[(** PositionalPreposition **ID_2** **(,** PositionalPreposition **ID_3)** **and** PositionalPreposition **ID_4] [at its** RelativePosition**]**
Manoeuvre Phase Syntax 2	**[, and / . And] [then] [ID_1]** Manoeuvre **[,** SpeedCondition **[,]]** ActionLocation **[(** PositionalPreposition **ID_2 (,** PositionalPreposition **ID_3)** **and** PositionalPreposition **ID_4] [at its** RelativePosition**]**
Parameters	**IDx**: Agent ID x. E.g., V1, V2, Ego. The ID may not have any spaces. Manoeuvre: Manoeuvre ID. E.g. is driving, is stopped. going ahead, drives towards, turning right across etc. SpeedCondition: Relative speed description E.g. accelerating, speeding up, at constant speed, decelerating, slowing down. ActionLocation: Action direction. E.g. towards, ahead of, across, in front of, behind, from, with, ahead, away. TrafficLightColour: Traffic Light Active Colour. E.g. green, amber, red. PositionalPreposition: Preposition associated with position. E.g. near, on, onto, into etc. RelativePosition: Relative position of one actor with another. E.g. front, rear, left, right, front left, rear right, etc. **(content)**: Brackets around *content* implies optional repetition of *content* depending on number of manoeuvre phases and actors. **[content]**: Square brackets around *content* implies optional use, depending on the level of detail desired by the end user. In some scenarios this detail may not be necessary.

Table 15 - Scripted Manoeuvre Phase Format

6.1.2. Example 1: SDL Scripted Agent Manoeuvres

To demonstrate the description of individual manoeuvre phases and the use of event trigger conditions, the scenario shown in Figure 25 is created. The SDL level 1 and level 2 descriptions are provided for the three manoeuvre phases, with the vehicles' relative positions being used as the triggers for the phase changes. Please note that this is not a complete scripted dynamic description as example 2 will focus on how sequence and parallel activity phases can be constructed. The T-Junction is labelled as TJ1. The road numbers (R1, R2 and R3) are marked next to each road alongside an arrow indicating the road direction. 'L1' and 'L-1' indicate the lane numbers for each road.

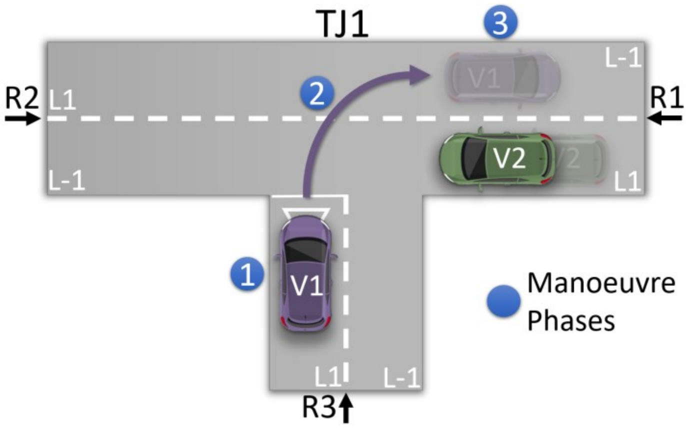


Figure 25 - T-Junction Turn Right Cut-In Example

SDL Level 1:

When Vehicle2 is driving, Vehicle1 drives towards Vehicle2 at its front left, and turns right across Vehicle2. Then Vehicle1 drives away from Vehicle2 at its right.

SDL Level 2:

```
WHEN: [V2] is [Going_Ahead]
 DO: [V1]
   PHASE 1: [Drive_Towards] [-, 15 to 20, 0 to 2] [V2: 0 to 1, FSL]
   PHASE 2: [TurnRight_Cross] [TJ1, 20 to 22, 0 to 2] [V2: 5 to 6, F]
   PHASE 3: [Drive_Away] [-, 20 to 22, 0 to 2] [V2: 5 to 6, SR]
```

6.1.3. Example 2: Complex Scripted Agent Dynamic Description

Example 2 is a more complicated scenario than example 1, as can be seen in Figure 26. The Ego vehicle will be driving towards a signalised pedestrian crossing and turning right at the T-junction (TJ1). Along the route, it will encounter two sub-scenarios:

1. V2 is driving in the opposite direction on Ego's path whilst overtaking V1.
2. The pedestrian (PD1) is walking across the road at the crossing beside the T-junction.

This example intends to illustrate how parallel (synchronised) and sequential manoeuvres phases can be constructed, and to demonstrate the composition of a complete scripted agent behaviour description. The SDL level 2 description for this example is displayed below Figure 26.

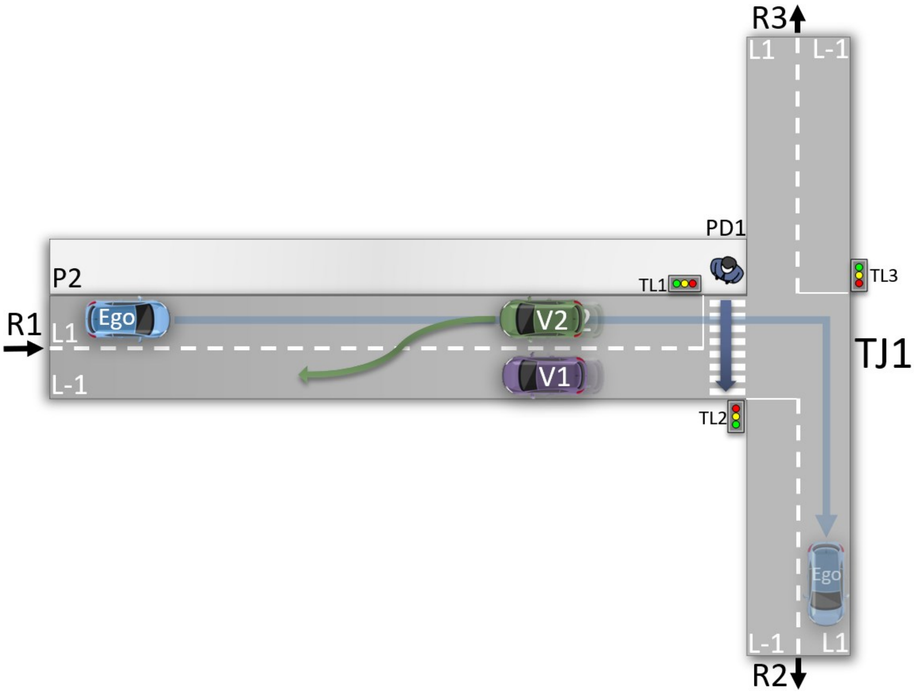


Figure 26 – Multi-situation Scenario

SDL Level 2:

```
INITIAL: Vehicle [EGO] in [R1.L1]
   AND Vehicle [V1] in [R1.L-1] AND at relative position [FSR] with relative heading angle [175 to 185] to [EGO]
   AND Vehicle [V2] in [R1.L1] AND at relative position [SR] with relative heading angle [-5 to 5] to [V1]
   AND Pedestrian [PD1] in [R1.P2] AND at relative position [FSL] with relative heading angle [265 to 275] to
[EGO]
WHEN: [EGO] is [Going_Ahead]
 DO: [V1]
   PHASE 1: [Drive_Towards] [-, 15 to 25, -1 to 1] [EGO: -15 to -5, FSR]
   PHASE 2: [Drive_Towards] [-, 15 to 25, -1 to 1] [EGO: -15 to -5, FSR]
   PHASE 3: [Drive_Towards] [-, 15 to 25, -1 to 1] [EGO: -15 to -5, FSR]
 AND: [V2]
   PHASE 1: [Drive_Away] [-, 20 to 30, -1 to 1] [V1: 0 to 5, FSR]
    WHILE: [V2] [Longitudinal] offset to [V1] < [7]
   PHASE 2: [LaneChangeLeft_CutIn] [-, 20 to 30, 1 to 2] [V1: 0 to 5, FSR]
   PHASE 3: [Drive_Away] [-, 30 to 40, -1 to 1] [V1: 10 to 20, F]
WHEN: [EGO] is [Going_Ahead] AND [TL1] is [Amber] AND [PD1] is [Going_Ahead] in [R1.P2]
 DO: [PD1]
   PHASE 1: [Walk_Towards] [Crossing1, 1 to 5, 0 to 1] [EGO: -25 to -30, FSL]
   PHASE 2: [Walk_Cross] [Crossing1, 1 to 5, -1 to 1] [EGO: -25 to -30, F]
   PHASE 3: [Walk_Away] [Crossing1, 1 to 5, -1 to 0] [EGO: -25 to -30, FSR]
END: [EGO] in [R2.L1]
```

In the initialisation section of the SDL level 2 description above, the spawn positions of V1, V2 and PD1 are specified according to the syntax. In the manoeuvre phase, it consists of two ‘WHEN’ conditions, the first ‘WHEN’ corresponding to when the ego vehicle encounters V1 and V2, the second ‘WHEN’ corresponding to when the ego encounters the pedestrian. In addition, this scenario also sets out an end condition which when satisfied the scenario will terminate.

Figure 27 displays an abstract structure from the previous SDL level 2 description. It can be seen that within the first ‘WHEN’ condition (highlighted in red box), there are two synchronised manoeuvre phases in a parallel relation. Within each activity phases that form the manoeuvre phases, it can be further divided into activities and triggers. Activities represent the manoeuvre characteristics of scripted actors, and triggers are in charge of switches between phases.

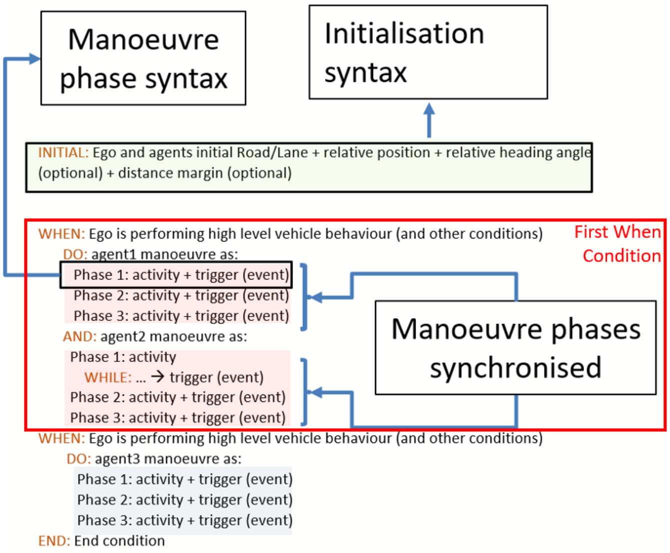


Figure 27 - SDL Level 2 Dynamic Element Composition

6.1.4. Example 3: SDL Pedestrian Manoeuvre

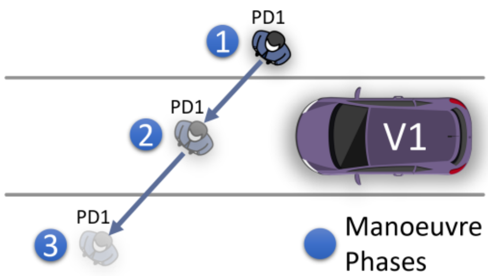


Figure 28 - Pedestrian Manoeuvre Example

Figure 28 displays a scenario where a pedestrian crosses the road in front of V1 (vehicle under test) in a diagonal direction. The pedestrian’s behaviour over time can be divided into three temporal parts:

1. When the pedestrian walks towards V1’s lane.
2. When the pedestrian crosses in front of V1.
3. When the pedestrian walks away from V1.

To illustrate how the pedestrian manoeuvre sequence can be described in both SDL level 1 and level 2, the descriptions below are provided.

SDL Level 1:

There is 1 vehicle, Vehicle1. There is 1 pedestrian, Pedestrian1. Vehicle1 and Pedestrian1 are in Road1. When Vehicle1 is going ahead, Pedestrian1 walks towards Vehicle1 at its front right, walks across in front of Vehicle1 and then walks away.

SDL Level 2:

```
INITIAL: Vehicle [V1] in [R1.L1]
    AND Pedestrian [PD1] in [R1.P2] AND at relative position [FSR] with relative heading angle [45 to 50] to
[V1]
WHEN: [V1] is [Going_Ahead]
 DO: [PD1]
   Phase 1: [Walk_Towards] [-, 1 to 4, 0 to 1] [V1: -25 to -30, FSR]
   Phase 2: [Walk_Cross] [-, 1 to 5, 0 to 1] [V1: -25 to -30, F]
   Phase 3: [Walk_Away] [-, 1 to 5, 0 to 1] [V1: -25 to -30, FSL]
END: [PD1] in [R1.P2] AND at relative position [FSL] to [V1]
```

Note: For any dynamic element, if the actor's state become stopped or stationary, the subsequent phases will be skipped until it starts manoeuvring again.

6.2. Dynamic Description Format (Non-Scripted Traffic)

A similar approach to the scripted traffic can also be utilised to form the non-scripted traffic description. As described in the domain concept section, the macroscopic traffic can be defined using the start/end position and any two of volume, speed and density. The start position and the end position can be used to form the initialisation section of the traffic description; the volume, speed and density can be used to create individual traffic. Centroids, in the SDL concept, are used to represent both starting position and end position, they are points within the roads and junctions network before any traffics are specified and the direction of traffic can be determined by using any two centroids.

SDL level 2 non-scripted traffic initialisation description format:

Syntax	**CENTROID [C_ID] in [R_ID] at [M] to [OBJ] AND … AND Centroid [C_ID] in [R_ID] at [M] to [OBJ]**
Parameters	**C_ID:** Centroid ID **R_ID:** Road ID **M:** Distance margin to reference object **OBJ:** Reference scenery and dynamic object

Table 16 - Non-Scripted Traffic Initialisation Description Format SDL Level 2

SDL level 2 non-scripted traffic characteristics description format:

Syntax	**TRAFFIC [Traffic_ID]: [C_SRC_ID: R_ID.L_ID, C_DEST_ID: R_ID.L_ID] at [Density, Speed]**
Parameters	**Traffic_ID:** ID for the traffic **C_SRC_ID, C_DEST_ID:** Centroid Source, Destination ID **R_ID:** Road ID **L_ID:** Lane ID **Density:** Traffic density measuring vehicles/distance **Speed:** Traffic speed measuring distance/time

Table 17 - Non-Scripted Traffic Characteristic Description Format SDL Level 2

Unlike in SDL level 2, the traffic condition will not be discussed or specified in great detail in SDL level 1 as there is no necessity to accurately model it. Rather there is an optional addition to the road description, within the wider 'scenery' description, which can describe the traffic level in the scenario.

SDL level 1 non-scripted traffic manoeuvre format (contained within the scenery description format):

Syntax	**(RoadID ((, RoadID …) and RoadID) is/are** GeometryType **,** RoadType **[with** TrafficType **traffic] .)**
Parameters	TrafficType**:** E.g. no, light, some, moderate, heavy, a lot of. **[content]**: Square brackets around *content* implies optional use, depending on the level of detail desired by the end user. In some scenarios this detail may not be necessary. **(content)**: Brackets around *content* imply optional repetition of *content* depending on the choice of the scenario author.

Table 18 - Non-Scripted Traffic Characteristic Description Format SDL Level 1

6.2.1. Example 1: SDL Non-Scripted Road User Manoeuvre

In this non-scripted traffic setting example, the traffic is intended to be defined along the main roads (R2, R1) within a T-junction (TJ1) and the traffic needs to travel in both directions (L1 and L-1) on this road. Please note that non-scripted traffic only defines the target traffic behaviour, whether such definition can be achieved during execution is determined by the individual scenarios. However, such target behaviours can be used as the criteria to evaluate whether the intended scenario has taken place or not.

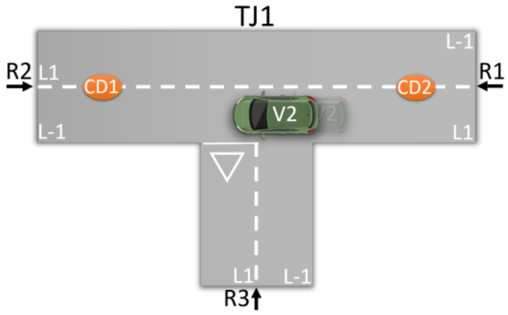


Figure 29 - Non-Scripted Traffic Example Scenario

SDL Level 1:

Road1 is a straight, distributor road with moderate traffic.

SDL Level 2:

INITIAL: CENTROID [CD1] in [R2] at [50 to 70] to [TJ1] AND centroid [CD2] in [R1] at [50 to 70] to [TJ1]
DO:
TRAFFIC [Traffic1]: [CD1: R2.L1, CD2: R1.L-1] at [55 to 60, 40 to 50]
TRAFFIC [Traffic2]: [CD2: R1.L1, CD1: R2.L-1] at [55 to 60, 40 to 50]

6.3. Scenery Elements Definition

There are two approaches for defining the scenery elements of a scenario. The first approach is to define the scenery layout according to the navigation route of the ego vehicle, within this approach only the relevant parts of the map will be described. If the ego vehicle's route contains repetitive sections, the description might duplicate previous sections. Furthermore, such approach does not account for the possibilities that ego vehicle takes an unintended route, as such routes are most likely not provided in the description. The second approach defines the scenery elements independently from any agents, it lists all the roads and junctions contained within a map and define their connections and orientations. Such description method allows the user to construct any maps in a scalable and organised manner. In both SDL level 1 and level 2, the second approach will be used.

Following this second approach, SDL considers any scenery settings as a roads and junctions network, as shown in Figure 30.

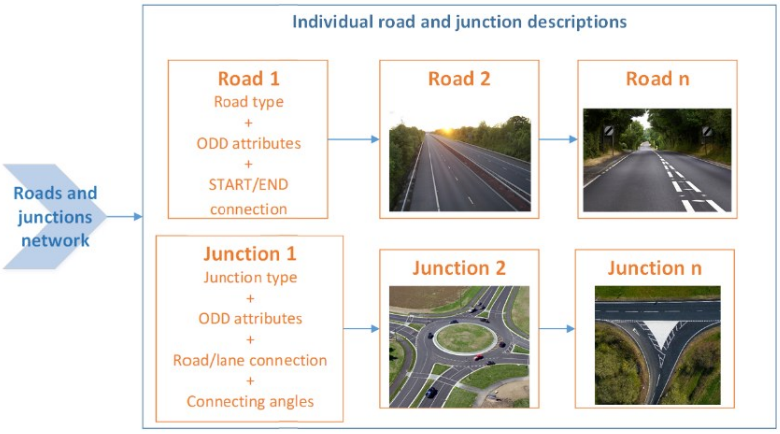


Figure 30 - SDL Scenery Element Structure

The main content body is therefore consisting of road descriptions and junction descriptions. Each road and junction will be described individually using their types and their associated ODD attributes. In addition, for each junction, their connecting roads and connecting lanes as well as connecting angles are also required. This will be referenced to the individual road description and allow the composition of the entire scenery.

To illustrate the components within a scenery description, the T-junction example shown in Figure 31 is used.

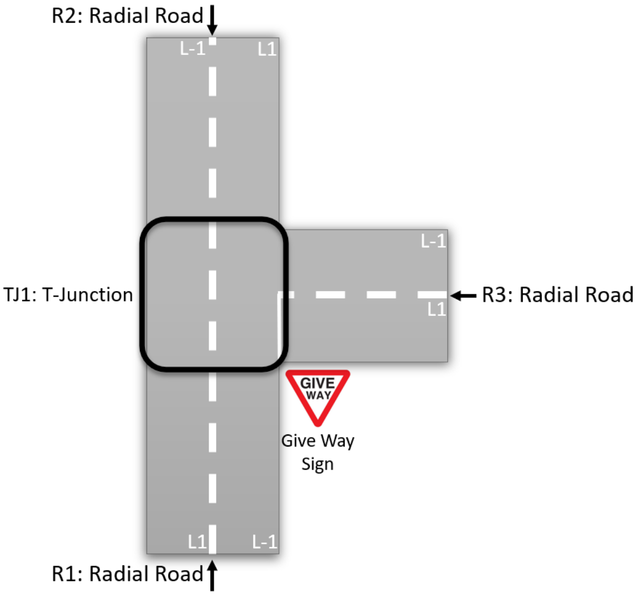


Figure 31 - Roads and Junctions Network – T-Junction Example

It can be seen in Figure 31 that the T-junction connects three radial roads that all have a straight horizontal geometry. The three roads each contain two lanes with opposite travel direction. At the joint between road 3 lane 1 and the T-junction, there is a give way road sign. These three roads and one T-junction form a simple roads and junctions network in this case, and the corresponding SDL level 2 and level 1 descriptions are provided to demonstrate how such scenery is composed. SDL level 1 shares the same key information as SDL level 2 but uses qualitative natural language description format, therefore it is pertinent to show the format separately.

SDL level 2 Scenery format:

Syntax	**SCENERY ELEMENTS :** **DO : Map - roads and junctions network [S_ID] as :** JUNCTIONS ROADS
Parameters	S_ID: Name for the scenery. JUNCTIONS: A list of junctions – adhering to the format for junctions. ROADS: A list of roads – adhering to the format for roads.

Table 19 - Road Users Initialisation Description Format SDL Level 2

SDL level 2 Junction format:

Junctions Syntax	**Junctions : None** **OR** **Junctions :** JUNCTION_DESCRTIPTIONS
Junction Syntax	**J_ID : Junction type [J_TYPE] as [J_ID]** **which has [CC_TYPE] connection control** **and [CN] connection(s) with [ROAD_LIST]** **Angles between roads [ROAD_ANGLES_MAP] [ROAD_ANGLES]** **Road lane connections [ROAD_LANE_CONNECTIONS]** **Dimensions [ROAD_DIMENSIONS]** **(Road signs ROAD_SIGNS)?** **(Traffic light TRAFFIC_LIGHT as [ID -> R_ID_x.L_ID_x:])**
Parameters	J_ID: Junction ID. E.g., TJ1, RA2. J_TYPE: Junction type. E.g., T-Junction, Y-Junction, Staggered, Crossroads, Straight, Grade Separated. Normal roundabout, Large roundabout, Compact roundabout, Double roundabout, Mini roundabout. CC_TYPE: Connection Control Type. E.g., Traffic light, Give way, No. CN: Number of roads connected at the junction. ROAD_LIST: Comma separated list of road IDs. E.g., R1, R2, R3. R_ID_x.L_ID_x: Road ID x and lane ID x. E.g., R1.L1, R2.L2. ROAD_ANGLES_MAP: Map of connected roads. E.g., R1 -> R2, R1 -> R3. ROAD_ANGLES: Comma separated angle ranges mapping in-order to the road angle maps. E.g., 30 to 35, 55 to 65. ROAD_LANE_CONNECTIONS: Map of connected lanes. E.g., R1.L2 -> R2.L1 + R3.L1. ROAD_DIMENSIONS: Of the form "**Diameter : Range**" OR "**Width : Range , Depth : Range**". ROAD_SIGNS: Type of road sign. E.g., Traffic Light Warning, Give way, Stop. TRAFFIC_LIGHT: Description of traffic light at the junction. **?:** Indicate optional statement.

Table 20 - Road Users Initialisation Description Format SDL Level 2

SDL level 2 Roads format:

Syntax	**Roads : None** **OR** **Roads:** **ROAD_DESCRTIPTIONS**
Road Syntax	**R_ID : START (: [J_ID_S])?** **Road type [RT] as [R_ID] with zone as [ZONE] AND speed limit of [SL] in (a/an) [ENV] environment with Number of lanes [NL] as [RLL]** **Road traffic direction [TDIR] Lane type [LT] Lane markings [LM]** **(Road surface type [ST])? (with surface condition [SC])? (AND surface feature [SF])?** **Horizontal road geometry [HG] (with curvature radius of [CR])?** **Vertical road geometry [VG]** **Transverse road geometry [TG] with [RF] roadside feature** **Road banking angle [BA]** **(Roadway edge features [REF])?** **(Fixed road structures [FRST] at [R_ID])** **(Temporary road structures [TRST] at RLN)** **('Special' 'road' 'structure' '[SRST]' 'as' '['SRS_ID ']' 'at' '['DOUBLE']' 'distance' 'from' ('START')/('ID'))*** **Road sign [RS] as [RS_ID] at ['DOUBLE'] 'distance' from 'START'** **Length [RL] AND Lane width [RW/SL]** **END (: [J_ID_E])?**
Parameters	**R_ID**: Road ID. E.g., R1.L1, R3.L-1. **SRS_ID**: Special Road Structure ID. E.g., Pedestrian Crossing. **RS_ID:** Road Sign ID **J_ID_S**: ID of the Junction where the road starts (if at all it starts at a junction). E.g., TJ1, RA2. **J_ID_E**: ID of the Junction where the road ends (if at all it ends at a junction). E.g., TJ1, RA2. RT: Road type. E.g., Motorway, Radial road, Distributor road, Minor road, Parking. ZONE: Type of zone. E.g., N/A, custom zone description as a string in quotes. SL: Speed limit as a double range. ENV: Type of environment. E.g., Rural, Urban. **NL**: Number of lanes RLL: Comma separated list of road-lane IDs of the form Rx.Ly. E.g., R1.L2, R1.L1, R1.L-1, R1.L-2. TDIR: Direction of traffic. E.g., Left-handed, Right-handed. LT: Type of lane. E.g., Traffic lane, custom description as a string in quotes. LM: Type of lane marking. E.g., Broken line, Solid line. ST: Type of Road Surface. E.g., Loose, Segmented, Uniform. SC: Condition of Road Surface. E.g., Dry, Wet, Snow, Contaminated, Icy, Mirage, Flooded. SF: Feature on the Road Surface. E.g., Cracks, Potholes, Ruts, Swells, N/A. HG: Description of the horizonal geometry. CR: Curvature radius mapping for curved segments of the road. VG: Type of vertical geometry of the road. E.g., Up-slope, Down-slope, Level plane. TG: Transverse geometry of the road. E.g., Divided, Undivided, Pavement. RF: Roadside feature. E.g., No, custom description as a string in quotes. BA: Banking angle of the road. E.g., Level Plane, Left Bank – Shallow, Left Bank – Deep, Right Bank – Shallow, Right Bank – Deep. REF: Road edge feature. E.g., Shoulder (grass), Solid barriers, Temporary line markers, Pavement, Line markers, Shoulder (paved or gravel). FRST: List of Fixed Road Structures with position on the road. SRST: List of Special Road Structures. E.g., Pedestrian Crossing, Bridge, Communications Unit, Speed bump. Custom description as a string in quotes. RS: Road sign. E.g., Speed Limit, Give way. TRST: List of Temporary Road Structures. E.g., Accident site, custom description as a string in quotes. RLN: Road Lane Position of temporary road structures. RL: Road Length as a range. RW: Road Width as a range. SW: List of Segment Widths as ranges. **?:** Indicate optional statement.

Table 21 - Road Users Initialisation Description Format SDL Level 2

From the SDL level 2 description and the description composition, it can be seen that all the junctions and roads are described under separate sections. With the junction section, the junction type, associated ODD attributes, the road/lanes connection and connecting angles are specified. Within the road section, the road type, associated ODD attributes, and the START/END connection of each road are specified. All the ODD attributes can be referenced to the underlying domain model of the language introduced in the previous section as they support the attributes contained with BSI PAS 1883. Furthermore, users can specify additional attributes using the SDL extension mechanism. The road/lane connections indicate all the intended legal links contained within the scenario. When specifying the connections, it links a single road and lane ID to all possible connections in the form of $R_x.L_x \rightarrow R_{x1}.L_{x1} +..+ R_i.L_i$. In addition, the connecting angle between roads also needs to be specified in order to form the intended geometry of the roads and junctions network. This can be done by using a clockwise rotation from the first road's direction to the second road's direction. See the 3 examples in Figure 32 for clarification.

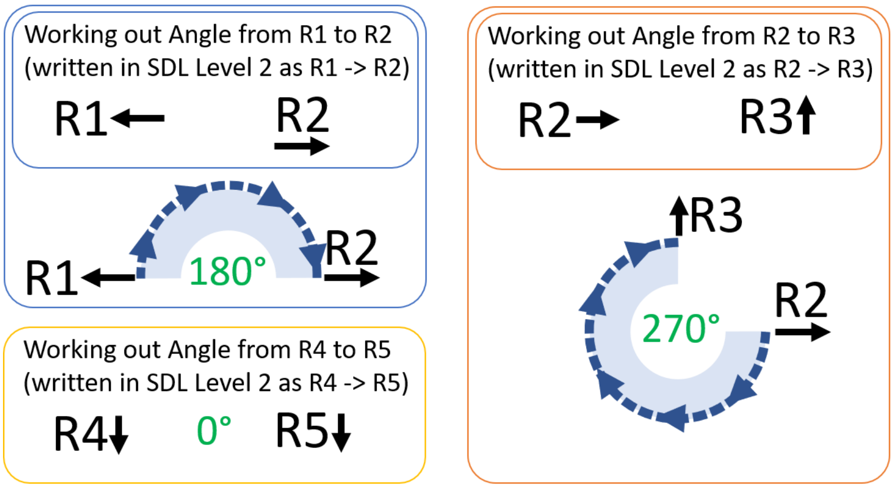


Figure 32 – Road Angle Calculation Examples

For each road, once it's direction has been defined, the lane numbering can be worked out using a common structure. On the left side of the direction of the road (the arrow direction), the lane numbering is positive numbering, and on the right side it is negative numbering simply by convention, as seen in the examples in Figure 33. For road 5 (R5), there are 2 pedestrian lanes, and these are treated in a similar manner to normal lanes except for the 'L' is replaced by a 'P'.

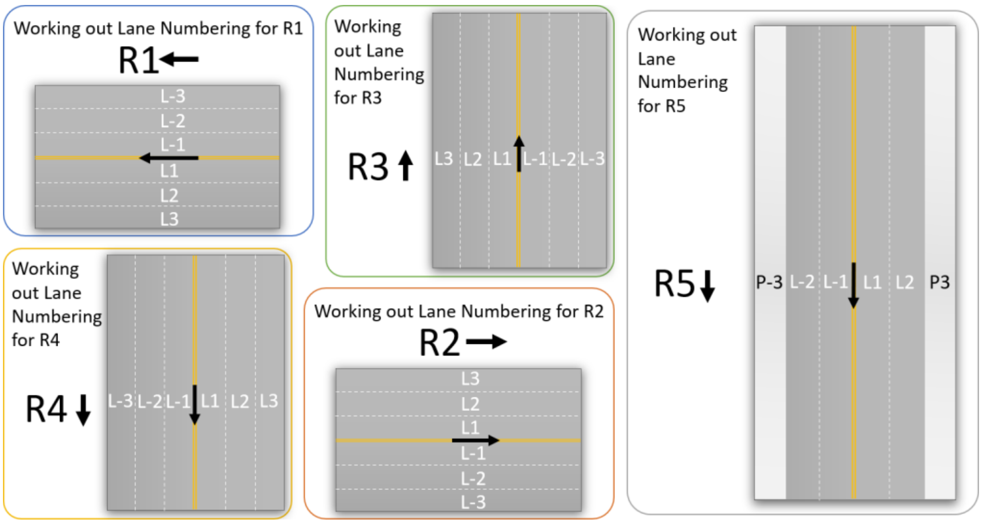


Figure 33 - Lane Numbering Convention Examples

Now that road angles and lane numbering have been defined, the level 2 scenery description for the (Figure 31) T-Junction example can be seen in Table 22.

Example SDL level 2 description	Description compositions
DO: Map - roads and junctions network [Network1] as: **Junctions:** TJ1: Junction type [T-Junction] as [TJ1] which has [Give way] connection control and [3] connections with [R1, R2, R3] Angles between roads [R1 -> R2, R1 -> R3, R2 -> R3] [175 to 185, 265 to 275, 85 to 95] Road lane connections [R1.L1 -> R2.L-1 + R3.L-1, R2.L1 -> R1.L-1 + R3.L-1, R3.L1 -> R1.L-1 + R2.L-1] Dimensions [Width: 5 to 7, Depth: 7 to 10] Road signs [Give way] at [R3.L-1]	DO: Map - roads and junctions network [Network1] as: **Junctions:** Junction type + Associated ODD attributes + Roads/lanes connections + Connecting angles
Roads: R1: START Road type [Radial road] as [R1] with zone as [N/A] AND speed limit of [40] in an [Urban] environment with Number of lanes [2] as [R1.L1, R1.L-1] Road traffic direction [Left-handed] Lane type [Traffic lane] Lane markings [Broken line] Horizontal road geometry [Straight] ... Length [50 to 100] AND Lane width [4 to 4.2] END [TJ1] R2: … R3: …	**Roads:** Road type + Associated ODD attributes + START/END connection

Table 22 - SDL Level 2 Scenery Description for the T-Junction Example

Table 23 displays the SDL level 1 description for the same T-junction example. As illustrated, the SDL level 1 description only contains very high-level qualitative descriptions of the junctions and roads. For the junctions, the required information is junction type, road connection and the connection direction between roads. For the road description, only the road type and geometry are required. However, the user can include other optional ODD attributes into the SDL level 1 description if needed.

The format of the description is as follows:

SDL Level 1 scenery format:

Scenery Syntax	DescriptionOf-Junctions DescriptionOf-Roads
Junctions Syntax	**There is no junction present.** ---------------- OR ----------------- **(There is a** JunctionType **, JunctionID, which has connections with RoadID ((, RoadID …) and RoadID). (RoadID to RoadID is** ConnectionType**.))**
Roads Syntax	**(RoadID ((, RoadID …) and RoadID) is/are** GeometryType **,** RoadType **[with** TrafficType **traffic] .)**

Scenery Syntax	DescriptionOf-Junctions DescriptionOf-Roads
	(Also RoadID ((, RoadID …) and RoadID) has/have RoadFeatureTypeList **.)**
Parameters	RoadID: E.g. Road1, Road2. The ID may not have any spaces. GeometryType: E.g. straight, curved RoadType: E.g. Motorways, Radial roads JunctionID: E.g. TJunction1, Crossroads2. The ID may not have any spaces. JunctionType: E.g. Motorways, Radial roads ConnectionType: E.g. straight ahead, turning right/left TrafficType**:** E.g. no, light, some, moderate, heavy, a lot of. RoadFeatureTypeList: Comma separated list of features. E.g. tunnel, traffic lights, lane marking etc. The second to last and last entry, an "**and**" is used. **[content]**: Square brackets around *content* implies optional use, depending on the level of detail desired by the end user. In some scenarios this detail may not be necessary. **(content)**: Brackets around *content* imply optional repetition of *content* depending on number of roads, connections and features.

Table 23 - SDL Level 1: Scenery Format

Example SDL Level 1 description	Description compositions
There is a T-Junction, Junction1, which has connections with Road1, Road2 and Road3. Road1 to Road2 is straight ahead. Road1 to Road3 is turning right. Road3 to Road2 is turning right.	**Junctions:** Junction type and road connections + Road connection directions
Road1, Road2 and Road3 are straight, radial roads.	**Roads:** Road type and geometry
Also, Road1, Road2 and Road3 have broken lane markings.	**Optional road features:** Additional features E.g. lane marking

Table 24 – SDL Level 1 Scenery Description for the T-Junction Example

6.3.1. Example 1: SDL Scenery Description Using Real-world Route

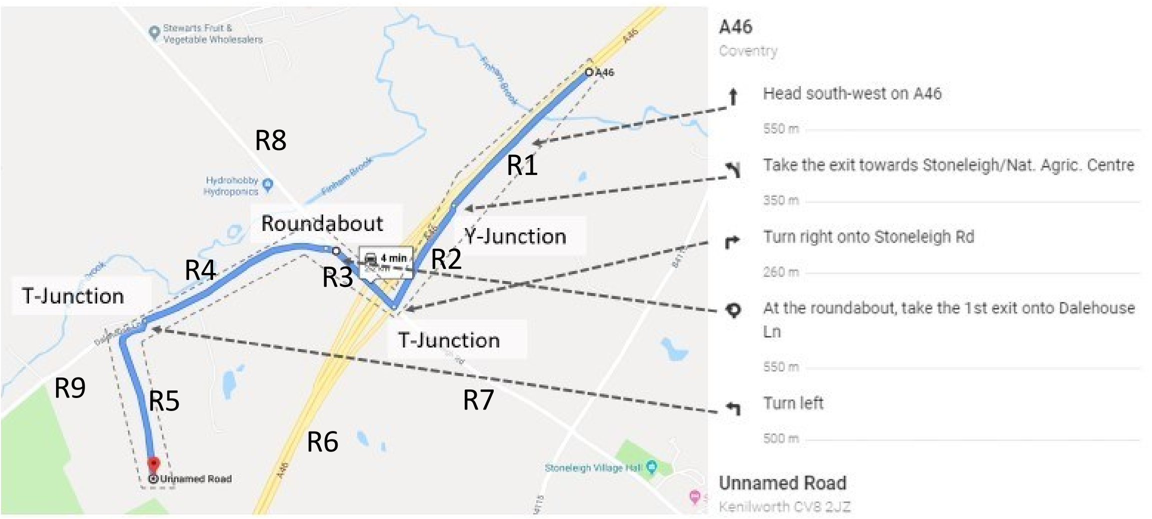


Figure 34 – SDL Scenery Demo Example Using UK Based Real-World Map Section

In order to illustrate a complete SDL scenery description, this section will use a real-world route in the UK to demonstrate. As shown in Figure 34, the scope of this description is only contained within the dashed boundary (which is only the roads and junctions that directly form the blue coloured route). The scenery elements contain a radial road (A46), a Y-junction, four distributor roads (A46 exit road, Stoneleigh Rd, Dalehouse Ln, and Unnamed road), two T-junctions, and one roundabout. The corresponding lengths of each road can be referenced on the right side of Figure 34.

Below are the SDL level 1 and level 2 descriptions of this scenery setting, please note due to repetitive reasons, parts of the SDL level 2 description are hidden.

SDL Level 1:

There is a Y-Junction, Junction1, which has connections with Road1, Road2 and Road6. Road1 to Road2 is to the left. Road1 to Road6 is straight ahead. There is a T-Junction, Junction2, which has connections with Road2, Road3 and Road7. Road2 to Road3 is to the right. Road2 to Road7 is to the left. Road3 to Road7 is straight ahead. There is a normal roundabout, Junction3, which has connections with Road3, Road4 and Road8. Road3 to Road4 is to the left. Road3 to Road8 is straight ahead. Road4 to Road8 is to the left. There is a T-Junction , Junction4, which has connections with Road4, Road5 and Road9. Road4 to Road5 is to the left. Road4 to Road9 is straight ahead. Road5 to Road9 is to the left. Road1 and Road6 are straight, radial roads. Road2, Road3, Road6, Road7, Road8 and Road9 are straight, distributor roads. Road4 is a curved, distributor road. Road5 is a straight, minor road.

SDL Level 2:

DO: Map - roads and junctions network [Network1] as:
Junctions:
YJ1: Junction type [Y-Junction] as [YJ1] which has [No] connection control and [3] connections with [R1, R2, R6]
  Angles between roads [R1 -> R2, R1 -> R6] [25 to 35, 0 to 5]
  Road lanes connections [R1.L1 -> R2.L1 + R3.L1, R3.L-1 -> R1.L-1]
  Dimensions [Width: 5 to 7, Depth: 7 to 10]
  Road signs [Give way] at [R2.L1]
 # Do the same for all the other junctions
 # TJ1:...
Roads:
R1: START
Road type [Radial road] as [R1] with zone as [N/A] AND speed limit of [60] in an [Urban] environment with
  Number of lanes [3] as [R1.L1, R1.L2, R1.L3]
  Road traffic direction [Left-handed]
  Lane type [Traffic lane]
  Lane markings [Broken line]
  Horizontal road geometry [Straight]
  Vertical road geometry [Level plane]
  Transverse road geometry [Divided] with [No] roadside feature
  Length [600 to 700] AND Lane width [4 to 4.2]
END [YJ1]

 # Do the same for all the other roads
 # R2:

6.3.2. Example 2: Temporary Road Structures

As part of the high-level ODD attributes, this example illustrates how a temporary road structure can be specified within the SDL level 1 and level 2 descriptions - the descriptions presented will be omitting the rest of the attributes. It can be seen in Figure 35 that the road layout consists of a T-junction connecting two main roads and a perpendicular joining road. At the entry of the joining road to the main road, there is a temporary accident site needs to be described in this scenario. The descriptions below illustrate how such a feature can be described. From the descriptions, it is shown that in SDL level 1, further ODD attributes apart from the road type and geometry are optional, whereas in SDL level 2 the descriptions are mandatory.

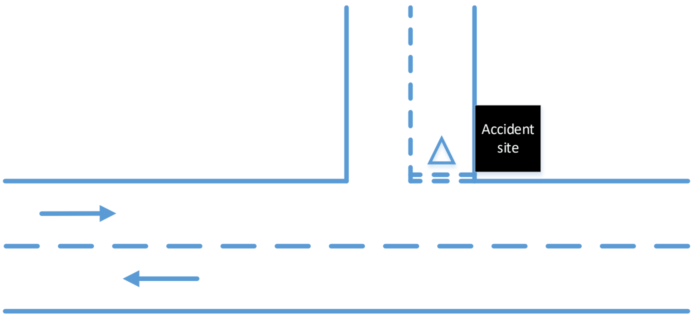


Figure 35 – Temporary Road Structure Example

SDL Level 1:

Road1 and Road2 are straight distributor road. Also, Road1 has "an accident site".

SDL Level 2:

DO: Map - roads and junctions network [Network 1] as:
Junctions:
TJ1: …
Roads:
R1: START
Road type [Distributor road] as [R1] with zone as [N/A] AND speed limit of [60] in an [Urban] environment with
  Number of lanes [2] as [R1.L1, R1.L-1]
  Road traffic direction [Left-handed]
  Lane type [Traffic lane]
  Lane markings [Broken line]
  Horizontal road geometry [Straight]
  Vertical road geometry [Level plane]
  Transverse road geometry [Undivided] with [No] roadside feature
  Temporary road structure [Accident site] at [R1.L-1] *# MANDATORY and default to be described in SDL level 2*
  Length [600 to 700] AND Lane width [4 to 4.2]
END [TJ1] …

6.3.3. Example 3: Signs and Traffic Lights

This example shows how traffic signs and traffic lights can be addressed in SDL. Since traffic lights normally have a target signalling direction or road section, it is important that this information is captured within the description. As displayed in Figure 36, a signalised T-junction links two main roads and a joining road. There are three traffic lights joints between each road and the T-junction. In addition, on one of the roads, there is a traffic light warning sign that can also be described.

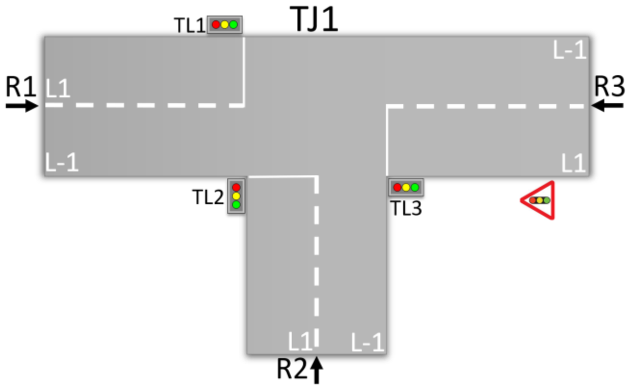


Figure 36 - Signal and Traffic Light Example

SDL Level 1:

There is a T-Junction, Junction1, which has connections with Road1, Road2 and Road3. Road1 to Road2 is to the right. Road2 to Road3 is to the right. Road3 to Road1 is straight ahead. Also, Junction1 has traffic lights.

SDL Level 2:

DO: Map - roads and junctions network [Network1] as:
Junctions:
TJ1: Junction type [T-Junction] as [TJ1] which has [Traffic light] connection control and [3] connections with [R1, R2, R3]
Angles between roads [R1 -> R2, R1 -> R3, R2 -> R3] [265 to 275, 175 to 185, 265 to 275]
Road lane connections [R1.L1 -> R2.L-1 + R3.L-1, R2.L1 -> R1.L-1 + R3.L-1, R3.L1 -> R1.L-1 + R2.L-1]
Dimensions [Width: 5 to 7, Depth: 7 to 10]
Road signs [Give way] at [R3.L1]
Traffic light [Without pedestrian crossing] as [TL1 -> R1.L1, TL2 -> R2.L1, TL3 -> R3.L1]

6.4. Environment Elements

As introduced in the domain concept section, the SDL environment attributes are referenced in the BSI PAS 1883. All the default attribute data types, measuring metrics and units are also specified within PAS 1883. From a scenario perspective, environmental conditions are treated as a global ambient condition in SDL, it is applied across the defined scenery area. The description of environmental conditions can be constructed by simply listing all the related attributes together with their values. The list can also be given a standard name space for re-usability and packaging purposes.

SDL level 2 Environment format:

Syntax	**ENVIROMENT ELEMENTS : DO : [E_ID]** **(Wind [WS])?** **Cloudiness [C]** **(Particulates [PT])?** **Rainfall [RT : RR]** **(Snowfall [ST (visibility : SR)])?** **Time of the day [TR/Any]** **Illumination [LT] with [LS] as light source** **(at [DE] degree elevation AND [LP] position)?** **(Connectivity [CT])?**
Parameters	E_ID: Environment ID. E.g., ENV1 WS: Wind speed as a range. C: Cloudiness in oktas. PT: Particulate type. E.g., Marine, Mist and Fog, Sand and Dust, Smoke and Pollution, Volcanic Ash, None. RT: Rainfall type. E.g., Light Rain, Moderate Rain, Heavy Rain. RR: Quantity of Rainfall as a range. ST: Snowfall type. E.g., Light Snow, Moderate Snow, Heavy Snow. SR: Visibility as a range.. TR: Time of day as a range. E.g., 5:30 to 9:30, 13:46 to 18:22. LT: Light type. E.g., Day, Night Lit, Night Dark, Artificial. LS: Light Source type. E.g., Sun, Headlamp, Street Lighting. DE: Range in Degrees of elevation of the sun – if used as light source. LP: Light Source relative position. E.g., F, SL, SR, etc. CT: Type of Connectivity (V2V – Cellular, Satellite, WiFi/V2I-GLOSA, Cellular, Satellite, WiFi) and positioning service (Galileo, GLONASS,GPS). **?:** Indicate optional statement.

Table 25 – SDL Level 2: Environment format

In SDL level 1, the descriptions of environmental attributes are qualitative and built into a structured sentence, while in SDL level 2 they are specified using values. The example shown in Figure 37 illustrates a situation where a list of environmental conditions are applied to the same T-junction scenery settings.

SDL Level 1 Environment Format:

Syntax	**During the** TimeDescription**,** **there is a** IlluminationDescription **(,** CloudDescription **) sky** **[,** AdditionalEnvironmentalFeatures **]** **[,** AdditionalEnvironmentalFeatures **]** **and** AdditionalEnvironmentalFeatures **).**
Parameters	TimeDescription: E.g. day, night.

Syntax	**During the** TimeDescription**,** **there is a** IlluminationDescription **(,** CloudDescription **) sky** **[,** AdditionalEnvironmentalFeatures **]** **[,** AdditionalEnvironmentalFeatures **]** **and** AdditionalEnvironmentalFeatures **).**
	IlluminationDescription: E.g. dark, light, bright. CloudDescription: E.g. lightly cloud covered, cloud covered, clear etc. AdditionalEnvironmentalFeatures **are:** RainDescription **,** WindDescription **,** SnowDescription **,** ParticulateDescription **.** RainDescription: E.g. light rain, moderate rain, heavy rain. WindDescription: E.g. a light breeze, a moderate breeze, a strong breeze etc. SnowDescription: E.g. light snow, heavy snow, moderate snow. ParticulateDescription: E.g. smoke, fog etc. **(content)**: Brackets around *content* imply optional repetition of *content* depending on number of environmental features.

Table 26 - SDL Level 1: Environment format

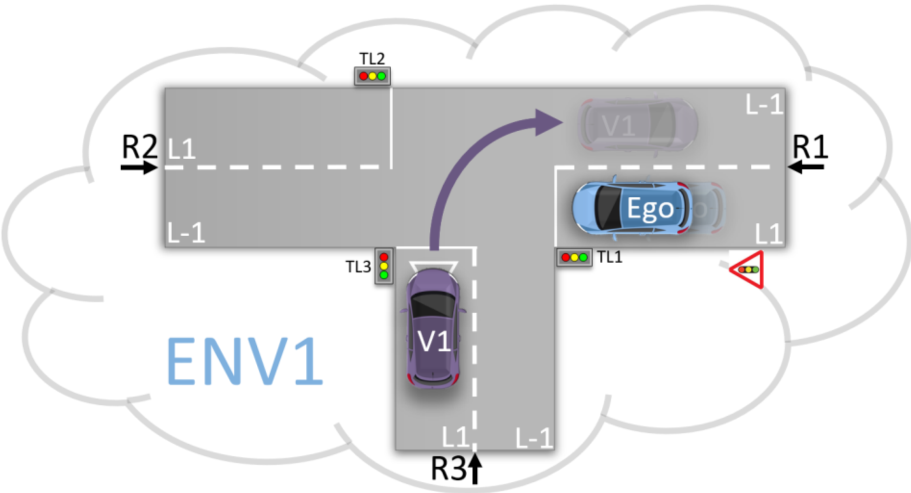


Figure 37 - Environmental Conditions Example

SDL Level 1:

During the day, there is a light, cloud covered sky, with mist and fog, rain and a moderate breeze.

SDL Level 2:

DO: [ENV1]
Wind [10 to 15]
Cloudiness [5 to 6] *# Cloudiness is described using oktas*
Particulates [Mist And Fog]
Rainfall [Moderate Rain: 3 to 5]
Time of the day [10:00 to 13:00]
Illumination [Day] with [Sun] as light source at [30 to 35] degree elevation AND [FSR] position

Please note that when the illumination source is the Sun, its elevation angle and position (in relation to ego) also need to be defined.

6.5. Packaging and Re-useability

For the portability and re-useability requirement, the SDL is designed such that any combination of SDL parameters can be grouped together and given a user defined namespace for later reference. Such a mechanism is available for all the main SDL elements: dynamic elements, scenery elements and environment elements. Examples below illustrate how custom namespaces can be specified.

Example 1 – scripted traffic dynamic description:

```
…
WHEN: [V1] is [Going ahead]
  DO: [PD1] manoeuvre as [my_manoeuvre_sequence]:
    Phase 1: [Walk_Towards] [-, 1 to 4, 0 to 1] [V1: -25 to -30, FSR]
    Phase 2: [Walk_Cross] [-, 1 to 5, 0 to 1] [V1: -25 to -30, F]
    Phase 3: [Walk_Away] [-, 1 to 5, 0 to 1] [V1: -25 to -30, FSL]
…
```

Example 2 – non-scripted dynamic description:

```
…
DO:
Traffic [Traffic 1] as [my_traffic]:
- Direction [CD1] to [CD2]
- Density [Moderate]
…
```

Example 3 – scenery description:

```
DO: Map - roads and junctions network [Network 1] as [my_map]:
Junctions:
TJ1: …
Roads:…
```

Example 4 – environment description:

```
DO: Environment [ENV1] as [my_env]:
 Wind [10 to 15]
 Clouds [5 to 6]
…
```

Such user defined namespaces and list of attributes can then be used in another scenario by simple stating '*DO: SDL elements as [User_define_name]*', by specifying such namespace to use, the attributes contained within the name will not need to be defined in the new scenario.

6.6. Extension Mechanism, Versioning, Author Information

As described in the domain concept section, the domain model used in the SDL is intended to be extensible. A similar extensible nature is also highlighted in the PAS 1883, and the domain model described so far only serves as a foundation or base. In practice, different users might want to further add custom attributes on top of the base domain model. For example, one might want to add a private test track as a road type and use it to construct scenarios. To satisfy this requirement, the SDL concept has included an extension mechanism. The idea is that the user would take a common base domain model and use the extension mechanism to modify the domain model within the scenario description. Such modifications will then be traceable to other users for the same file. The custom attributes can be assigned as metrics, types, values and units where necessary, and they can be seamlessly integrated with the base domain model. Within the scenario file, an extension is placed before composing the scenery, dynamic and environment statements.

The format below illustrates the syntax of the extension mechanism together with two examples. It can be seen that it includes *new attribute, parent attribute, metric, data type* and *attribute* *properties.* Please note that depending on the data type of the new attribute, it might not need the corresponding metrics, values and units, therefore the second line of the syntax is optional. Example 1 demonstrates how users could add a new attribute called '*my_rain*' under *rainfall* parent class. Furthermore, *my_rain* is defined using a rainfall rate with the units mm/hr and it is constrained using a double range between 0.1 and 2.5. Example 2 illustrates how one could add a new road type called '*my_test_track'* under the *road type* parent class. Since *my_test_track* does not require defining metrics, values or data type, the rest of the information is therefore not needed for this extension. One could consider that *road type* is an attribute with an enumerated data type, and example 2 shows how the enumerations can be extended.

Extension mechanism syntax:

```
Extension:
 Add new_attribute to parent_attribute
 (Add [metric] with data_type [attribute_properties] to new_attribute)? # '?' indicates optional statement
```

Example 1:

```
Extension:
 Add my_rain to rainfall
 Add [rainfall_rate] with double_range [0.1 to 2.5 mm/hr] to my_rain
```

Example 2:

```
Extension:
 Add my_test_track to road_type
```

In addition to the extension mechanism, the SDL version and the author information are also required to be present at the top of an SDL file, following the example listed below.

```
VERSION: 8.0
AUTHOR: John Doe
```

Using all the previous sections together, we arrive at a complete understanding of the SDL level 1 and 2 concept. In the following section are some full examples of scenarios that SDL level 1 and level 2 can describe.

7. SDL examples

7.1. Example 1

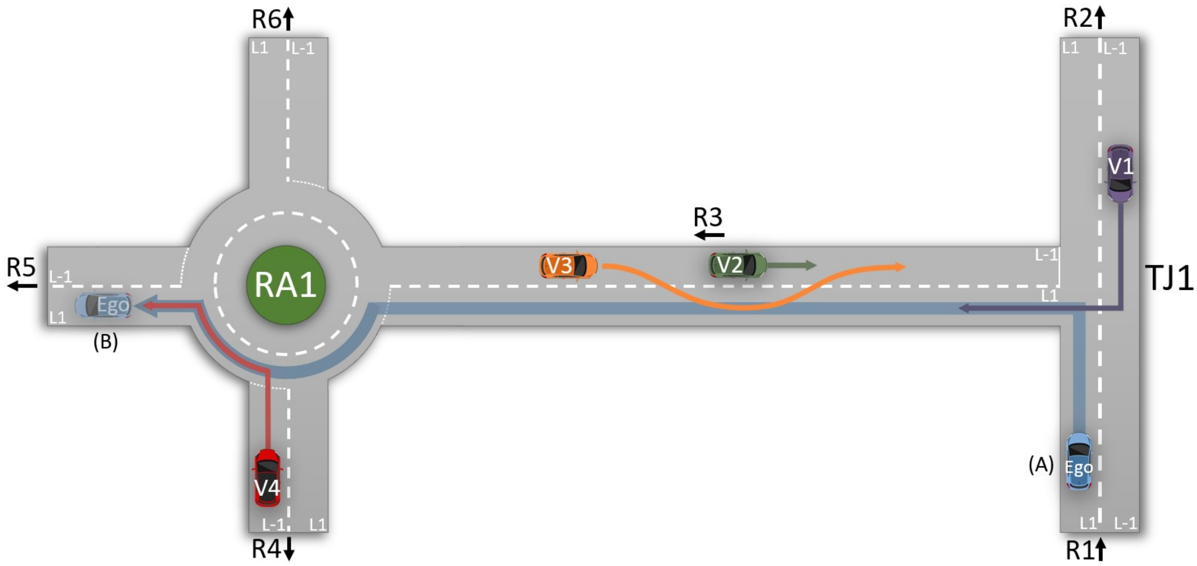


Figure 38 – Multi-Junction Example Scenario

Scenario summary: Ego vehicle is going from point (A) to point (B), with V1-V4 dynamic actors. Ego vehicle will encounter three situations along its route. The first situation involves an agent vehicle (V1) driving towards the Ego vehicle from the opposite lane and performing a turn right cut-in ahead of the Ego at a T-Junction (TJ1). The second situation involves two agents (V2 and V3); V3 changes lane to the Ego's lane and overtakes V2. In the third situation, the Ego is going forward at a roundabout (RA1) and an agent vehicle (V4) is cutting into the Ego's lane from the roundabout's next entry point, which is Road 4 (R4).

SDL Level 1:

VERSION: 8.2
AUTHOR: "Jason Zhang"

During the day, there is a light, cloud covered sky, with fog, heavy rain, and a moderate breeze.

There is a T-Junction, Junction1, which has connections with Road1, Road2 and Road3. Road1 to Road2 is straight ahead. There is a compact roundabout, Roundabout1, which has connections with Road3, Road4, Road5 and Road6. Road1 to Road2 is straight ahead. Road1 to Road3 is turning left. Road1, Road2, Road3, Road4, Road5 and Road6 are straight, distributor roads. Also, Road1, Road2, Road3, Road4, Road5 and Road6 have broken lane markings.

There are 5 vehicles, Ego, Vehicle1, Vehicle2, Vehicle3 and Vehicle4. Ego is in Road1, Vehicle1 is in Road2, Vehicle2 is in Road3, Vehicle3 is in Road3 and Vehicle4 is in Road4.
When Ego is driving ahead, Vehicle1 drives towards Ego, turns right onto Road3, cutting-in ahead of Ego at its front right, and then drives away. Next, as Vehicle2 drives towards Ego, Vehicle3 drives towards Vehicle2 at its rear. Vehicle3 then changes lane right, drives away from Vehicle2, and then changes lane left in front of Vehicle2. Next, as Ego drives towards Roundabout1, Vehicle4 turns left into Roundabout1, cutting-in ahead of Ego at its front left, drives away, and then turns left onto Road5.

SDL Level 2:

VERSION: 8.2
EXTENSION: None
AUTHOR: 'Jason Zhang'

SCENERY ELEMENTS:

DO: Map - roads and junctions network [Network1] as:
Junctions:
TJ1: Junction type [T-Junction] as [TJ1] which has [No] connection control and [3] connections with [R1, R2, R3]
Angles between roads [R1 -> R2, R1 -> R3, R2 -> R3] [0 to 5, 265 to 275, 265 to 275]
Road lane connections [R1.L1 -> R2.L1 + R3.L1,
R2.L-1 -> R1.L-1 + R3.L1,
R3.L-1 -> R1.L-1 + R2.L1]
Dimensions [Width: 5 to 7, Depth: 7 to 10]
RA1: Junction type [Compact roundabout] as [RA1] which has [No] connection control and [4] connections with [R3, R4, R5, R6]
Angles between roads [R3 -> R4, R4 -> R5, R5 -> R6, R6 -> R3] [265 to 275, 265 to 275, 265 to 275, 265 to 275]
Road lane connections [R3.L1 -> R3.L-1 + R4.L1 + R5.L1 + R6.L1,
R4.L-1 -> R3.L-1 + R4.L1 + R5.L1 + R6.L1,
R5.L-1 -> R3.L-1 + R4.L1 + R5.L1 + R6.L1,
R6.L-1 -> R3.L-1 + R4.L1 + R5.L1 + R6.L1]
Dimensions [Diameter: 4 to 4.5]

Roads:
R1: START
Road type [Distributor road] as [R1] with zone as [N/A] AND speed limit of [30] in an [Urban] environment with
Number of lanes [2] as [R1.L1, R1.L-1]
Road traffic direction [Left-handed] Lane type [Traffic lane]
Lane markings [Broken line]
Road surface type [Uniform] with surface condition [Dry]
Horizontal road geometry [Straight]
Vertical road geometry [Level plane]
Transverse road geometry [Undivided] with [No] roadside feature
Length [60 to 100] AND Lane width [4 to 4.2]
END: [TJ1]

R2: START [TJ1]
Road type [Distributor road] as [R2] with zone as [N/A] AND speed limit of [30] in an [Urban] environment with
Number of lanes [2] as [R2.L1, R2.L-1]
Road traffic direction [Left-handed] Lane type [Traffic lane]
Lane markings [Broken line]
Road surface type [Uniform] with surface condition [Dry]
Horizontal road geometry [Straight]
Vertical road geometry [Level plane]
Transverse road geometry [Undivided] with [No] roadside feature
Length [60 to 100] AND Lane width [4 to 4.2]
END

R3: START [TJ1]
Road type [Distributor road] as [R3] with zone as [N/A] AND speed limit of [30] in an [Urban] environment with
Number of lanes [2] as [R3.L1, R3.L-1]
Road traffic direction [Left-handed] Lane type [Traffic lane]
Lane markings [Broken line]
Road surface type [Uniform] with surface condition [Dry]
Horizontal road geometry [Straight]
Vertical road geometry [Level plane]
Transverse road geometry [Undivided] with [No] roadside feature
Length [200 to 300] AND Lane width [4 to 4.2]
END: [RA1]

R4: START [RA1]
Road type [Distributor road] as [R4] with zone as [N/A] AND speed limit of [30] in an [Urban] environment with
Number of lanes [2] as [R4.L1, R4.L-1]
Road traffic direction [Left-handed] Lane type [Traffic lane]
Lane markings [Broken line]
Road surface type [Uniform] with surface condition [Dry]
Horizontal road geometry [Straight]
Vertical road geometry [Level plane]

Transverse road geometry [Undivided] with [No] roadside feature
Length [60 to 100] AND Lane width [4 to 4.2]
END

R5: START [RA1]
Road type [Distributor road] as [R5] with zone as [N/A] AND speed limit of [30] in an [Urban] environment with
Number of lanes [2] as [R5.L1, R5.L-1]
Road traffic direction [Left-handed] Lane type [Traffic lane]
Lane markings [Broken line]
Road surface type [Uniform] with surface condition [Dry]
Horizontal road geometry [Straight]
Vertical road geometry [Level plane]
Transverse road geometry [Undivided] with [No] roadside feature
Length [60 to 100] AND Lane width [4 to 4.2]
END

R6: START [RA1]
Road type [Distributor road] as [R6] with zone as [N/A] AND speed limit of [30] in an [Urban] environment with
Number of lanes [2] as [R6.L1, R6.L-1]
Road traffic direction [Left-handed] Lane type [Traffic lane]
Lane markings [Broken line]
Road surface type [Uniform] with surface condition [Dry]
Horizontal road geometry [Straight]
Vertical road geometry [Level plane]
Transverse road geometry [Undivided] with [No] roadside feature
Length [60 to 100] AND Lane width [4 to 4.2]
END

DYNAMIC ELEMENTS:
INITIAL: Vehicle [Ego] in [R1.L1]
AND Vehicle [V1] in [R2.L-1] AND at relative position [FSR] with relative heading angle [175 to 185] to [Ego]
AND Vehicle [V2] in [R3.L-1] AND at relative position [FSL] with relative heading angle [265 to 275] to [Ego]
AND Vehicle [V3] in [R3.L-1] AND at relative position [R] with relative heading angle [0 to 5] to [V2]
AND Vehicle [V4] in [R4.L-1] AND at relative position [RSR] with relative heading angle [85 to 95] to [V3]
WHEN: [Ego] is [Going_Ahead] in [R1.L1]
DO: [V1]
PHASE 1: [Drive_Towards] [-, 25 to 35, 2 to 3] [Ego: -5 to 5, FSR]
PHASE 2: [TurnRight_CutIn] [TJ1, 30 to 35, 2 to 3] [Ego: 0 to 5, F]
PHASE 3: [TurnRight_CutOut] [TJ1, 30 to 35, -1 to 1] [Ego: 5 to 10, FSL]

WHEN: [Ego] is [Going_Ahead] in [R3.L1]
DO: [V2]
PHASE 1: [Drive_Towards] [-, 15 to 25, -1 to 1] [Ego: -15 to -5, FSR]
PHASE 2: [Drive_Towards] [-, 15 to 25, -1 to 1] [Ego: -15 to -5, FSR]
PHASE 3: [Drive_Towards] [-, 15 to 25, -1 to 1] [Ego: -15 to -5, FSR]
PHASE 4: [Drive_Towards] [-, 15 to 25, -1 to 1] [Ego: -15 to -5, FSR]
PHASE 5: [Drive_Towards] [-, 15 to 25, -1 to 1] [Ego: -15 to -5, FSR]
AND: [V3]
PHASE 1: [Drive_Towards] [-, 20 to 30, 0 to 1] [V2: 5 to 15, R]
WHILE: [V3] [Longitudinal] offset to [V2] > [7]
PHASE 2: [LaneChangeRight_CutOut] [-, 25 to 35, 4 to 5] [V2: 10 to 20, RSR]
WHILE: [V3] [Lateral] offset to [V2] < [4.2]
PHASE 3: [Drive_Towards] [-, 25 to 35, -1 to 0] [V2: 10 to 20, RSR]
PHASE 4: [Drive_Away] [-, 25 to 35, -1 to 0] [V2: 10 to 20, FSR]
WHILE: [V3] [Longitudinal] offset to [V2] < [7]
PHASE 5: [LaneChangeRight_CutIn] [-, 25 to 35, 4 to 5] [V2: 10 to 20, FSR]

WHEN: [Ego] is [Going_Ahead] in [RA1]
DO: [V4]
PHASE 1: [TurnLeft_CutIn] [-, 20 to 30, 4 to 5] [Ego: -10 to 0, FSL]
PHASE 2: [Drive_Away] [RA1, 20 to 30, 4 to 5] [Ego: -10 to 0, F]

WHILE: [V4] [Longitudinal] offset to [R5.L1] > [10]
PHASE 3: [TurnLeft_CutOut] [RA1, 20 to 30, 4 to 5] [Ego: -10 to 0, FSL]
END: [Ego] in [R5.L1]

ENVIRONMENT ELEMENTS:
DO: [ENV1]
Wind [10 to 15]
Cloudiness [5 to 6]
Particulates [None]
Rainfall [None:N/A]
Time of the day [10:00 to 13:00]
Illumination [Day] with [Sun] as light source at [30 to 35] degree elevation AND [FSR] position

7.2. Example 2

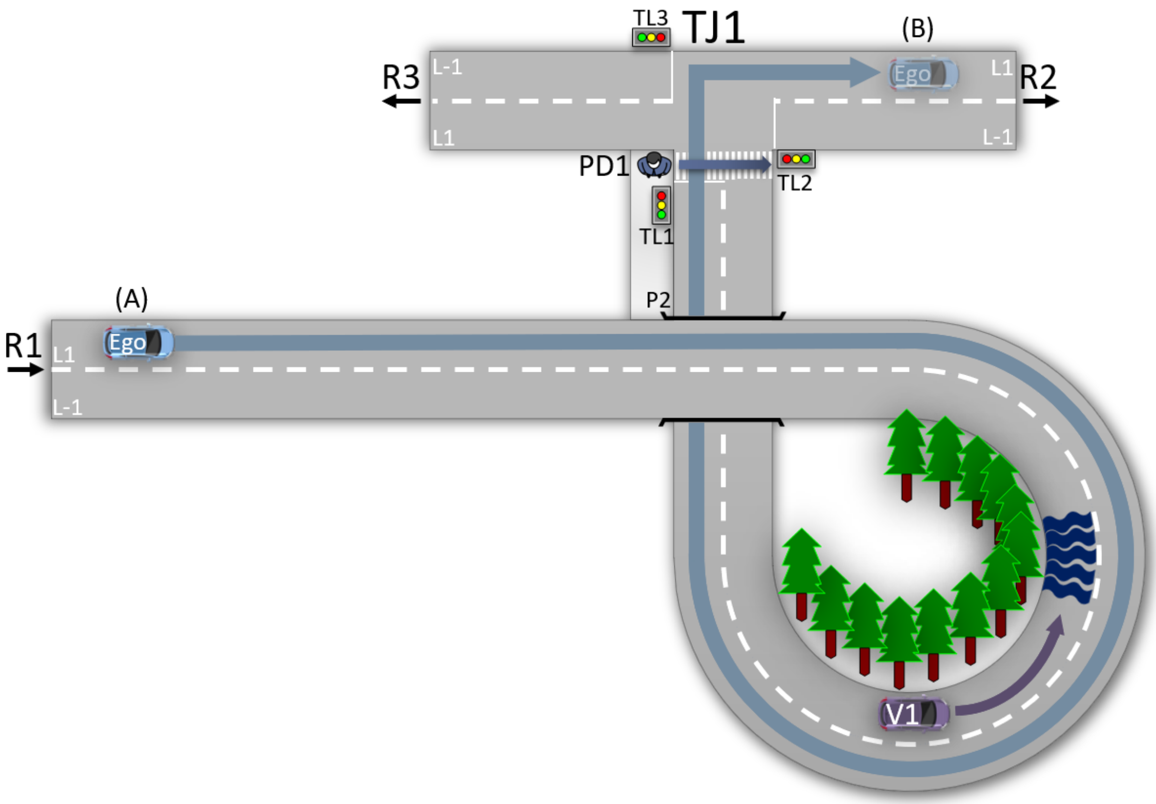


Figure 39 - Multi-Phases Example with Road Surface Feature

Scenario summary: Ego vehicle is going from point (A) to point (B), and the dynamic actor (V1) is driving on the opposite lane towards the Ego whilst trying to avoid flood water, and a pedestrian (PD1) is crossing at traffic amber light next to a T-Junction (TJ1).

SDL Level 1:

VERSION: 8.2
AUTHOR: "Patrick Irvine"

During the day, there is a light, cloud covered sky, with fog, heavy rain and a moderate breeze.

There is a T-Junction, Junction1, which has connections with Road1, Road2 and Road3. Road1 to Road2 is turning left. Road 1 to Road 3 is turning right. Road3 to Road2 is straight ahead. Road1 is a curved, distributor road. Road2 and Road3 are straight, distributor roads. Also, Road1, Road2 and Road3 have broken lane markings.

There are 2 vehicles, Ego and Vehicle1. There is 1 pedestrian, Pedestrian1. Ego is in Road1, Vehicle1 is in Road1 and Pedestrian1 is in Road1.
When Ego is driving ahead, Vehicle1 drives towards Ego, and changes lane right, speeding up, cutting-in ahead of Ego. Vehicle1 then changes lane left away from Ego, and then drives away. Next, when Ego is driving ahead on Road1 and near Junction1, Pedestrian1 walks towards Ego at its right, walks across in front of Ego, and then walks away from Ego at its left.

SDL Level 2:

VERSION: 8.2
EXTENSION: None
AUTHOR: 'Patrick Irvine'

SCENERY ELEMENTS:
DO: Map - roads and junctions network [Network1] as:
Junctions:
TJ1: Junction type [T-Junction] as [TJ1] which has [Traffic light] connection control and [3] connections with [R1, R2, R3]
Angles between roads [R1 -> R2, R1 -> R3, R2 -> R3] [85 to 95, 265 to 275, 175 to 185]
Road lane connections [R1.L1 -> R2.L1 + R3.L1,
R2.L-1 -> R1.L-1 + R3.L1,
R3.L-1 -> R1.L-1 + R2.L1]
Dimensions [Width: 5 to 7, Depth: 7 to 10]
Traffic light [With pedestrian crossing] as [TL1 -> R1.L1, TL2 -> R2.L-1, TL3 -> R3.L-1]

Roads:
R1: START
Road type [Distributor road] as [R1] with zone as [N/A] AND speed limit of [30] in an [Urban] environment with
Number of lanes [2] as [R1.L1, R1.L-1]
Road traffic direction [Left-handed]
Lane type [Traffic lane]
Lane markings [Broken line]
Road surface type [Uniform] with surface condition [Dry]
Horizontal road geometry [S1:Straight, CR1:Curved, CR2:Curved, CR3:Curved, S2:Straight] with curvature radius of [S1:N/A, CR1:90 to 90, CR2:90 to 90, CR3:90 to 90, S2:N/A]
Vertical road geometry [Level plane]
Transverse road geometry [Divided] with [No] roadside feature
Fixed road structures [Trees] at [CR1, CR2, CR3], [Bridge] at [S1], [Tunnel] at [S2]
Temporary road structure [Flood] at [CR1, CR2]
Length [S1: 200 to 500, CR1: 100 to 200, CR2: 100 to 200, CR3: 100 to 200, S2: 200 to 400] AND Lane width [4 to 4.2]
END: [TJ1]

R2: START [TJ1]
Road type [Distributor road] as [R2] with zone as [N/A] AND speed limit of [30] in an [Urban] environment with
Number of lanes [2] as [R2.L1, R2.L-1]
Road traffic direction [Left-handed]
Lane type [Traffic lane]
Lane markings [Broken line]
Road surface type [Uniform] with surface condition [Wet]
Horizontal road geometry [Straight]
Vertical road geometry [Level plane]
Transverse road geometry [Divided] with [No] roadside feature
Length [100 to 200] AND Lane width [4 to 4.2]
END

R3: START [TJ1]
Road type [Distributor road] as [R3] with zone as [N/A] AND speed limit of [30] in an [Urban] environment with
Number of lanes [2] as [R3.L1, R3.L-1]
Road traffic direction [Left-handed]

Lane type [Traffic lane]
Lane markings [Broken line]
Road surface type [Uniform] with surface condition [Wet]
Horizontal road geometry [Straight]
Vertical road geometry [Level plane]
Transverse road geometry [Divided] with [No] roadside feature
Length [100 to 200] AND Lane width [4 to 4.2]
END

DYNAMIC ELEMENTS:
INITIAL: Vehicle [Ego] in [R1.L1]
AND Vehicle [V1] in [R1.L-1] AND at relative position [FSR] with relative heading angle [175 to 185] to [Ego]
AND Pedestrian [PD1] in [R1.P2] AND at relative position [FSL] with relative heading angle [265 to 275] to [Ego]

WHEN: [Ego] is [Going_Ahead] in [R1.L1]
DO: [V1]
PHASE 1: [Drive_Towards] [-, 25 to 35, 2 to 3] [Ego: -5 to 5, FSR]
WHILE: [V1] [longitudinal] offset to [CR1] > [7]
PHASE 2: [LaneChangeRight_CutIn] [-, 25 to 35, 1 to 2] [Ego: 0 to 10, FSR]
PHASE 3: [Drive_Towards] [-, 20 to 30, 0 to 1] [Ego: -5 to 5, F]
WHILE: [V1] [longitudinal] offset to [CR1] < [7]
PHASE 4: [LaneChangeLeft_CutOut] [-, 25 to 35, 0 to 1] [Ego: 0 to 10, FSR]
WHILE: [V1] [lateral] offset to [CR1] > [0.5]
PHASE 5: [Drive_Away] [-, 30 to 40, 1 to 2] [Ego: 5 to 15, RSR]

WHEN: [Ego] is [Going_Ahead] in [R1.L1] AND [TL1] is [Amber]
DO: [PD1]
PHASE 1: [Walk_Towards] [TJ1, 1 to 5, 0 to 1] [Ego: 1 to 5, FSL]
PHASE 2: [Walk_Cross] [TJ1, 1 to 5, -1 to 1] [Ego: 1 to 5, F]
PHASE 3: [Walk_Away] [TJ1, 1 to 5, -1 to 0] [Ego: 1 to 5, FSR]

END: [Ego] in [R2.L1]

ENVIRONMENT ELEMENTS:
DO: [ENV1]
Wind [10 to 15]
Cloudiness [5 to 6]
Particulates [None]
Rainfall [None:N/A]
Time of the day [10:00 to 13:00]
Illumination [Day] with [Sun] as light source at [30 to 35] degree elevation AND [FSR] position

7.3. Example 3 (Euro NCAP: AEB system)

Scenario summary: These set of scenarios focuses on testing the Autonomous emergency braking (AEB) system, it applies braking is to the vehicle in response to any detections of likely collisions along its path [16].

7.3.1. Example 3.1: Car-to-Car Rear Stationary (CCRs)

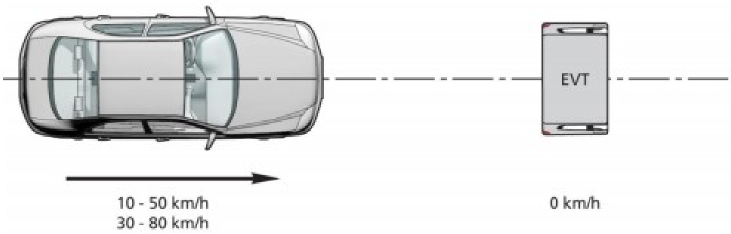


Figure 40 - EuroNCAP CCRs Example

Scenario description: Vehicle EVT as controlled agent is stopped at the front position to ego vehicle.

SDL Level 1:

VERSION: 8.2
AUTHOR: "Jason Zhang"

During the day, there is a light, cloud covered sky, and a moderate breeze.

There is no junction present. Road1 is a straight, "Test track".

There are 2 vehicles, Ego and EVT. Ego is in Road1 and EVT is in Road1. When Ego is driving ahead, EVT stops ahead of Ego.

SDL Level 2:

VERSION: 8.2
EXTENSION: None
AUTHOR: 'Jason Zhang'

SCENERY ELEMENTS:
DO: Map - roads and junctions network [Network1] as:
Junctions: None

Roads:
R1: START
Road type ["Test_track"] as [R1] with zone as [N/A] AND speed limit of [N/A] in an [Urban] environment with
Number of lanes [1] as [R1.L1]
Road traffic direction [N/A]
Lane type [Traffic lane]
Lane markings [None]
Road surface type [Uniform] with surface condition [Dry]
Horizontal road geometry [Straight]
Vertical road geometry [Level plane]
Transverse road geometry [Undivided] with [No] roadside feature
Length [500 to 700] AND Lane width [4 to 4.2]
END

DYNAMIC ELEMENTS:
INITIAL: Vehicle [Ego] in [R1.L1]
AND Vehicle [EVT] in [R1.L1] with a [Longitudinal] offset of [100 to 100]
AND at relative position [F] with relative heading angle [0 to 5] to [Ego]
AND Global timer [T1] = [0]
WHEN: [Ego] is [Going_Ahead]
DO: [EVT]

PHASE 1: [Stop] [-, 0 to 0, 0 to 0] [Ego: -22.2 to -2.7, F]

END: [T1] == [40]

ENVIRONMENT ELEMENTS:
DO: [ENV1]
Wind [0 to 1]
Cloudiness [5 to 6]
Particulates [None]
Rainfall [None:N/A]
Time of the day [10:00 to 13:00]
Illumination [Day] with [Sun] as light source at [30 to 35] degree elevation AND [FSR] position

7.3.2. Example 3.2: Car-to-Car Rear Moving (CCRm)

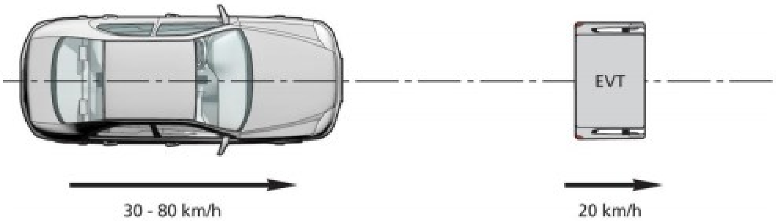


Figure 41 - EuroNCAP CCRm Example

Scenario description: Vehicle EVT as controlled agent is cruising at a much slower speed and at the front position to ego vehicle.

SDL Level 1:
VERSION: 8.0
AUTHOR: "Jason Zhang"

During the day, there is a light, cloud covered sky, and a moderate breeze.

There is no junction present. Road1 is a straight, "Test track".

There are 2 vehicles called Ego and EVT. Ego is in Road1 and EVT is in Road1. When Ego is going ahead, EVT drives ahead of Ego at its front with constant speed.

SDL Level 2:
VERSION: 8.0
EXTENSION: None
AUTHOR: 'Jason Zhang'

SCENERY ELEMENTS:
DO: Map - roads and junctions network [Network1] as:
Junctions: None

Roads:
R1: START
Road type ['Test_track'] as [R1] with zone as [N/A] AND speed limit of [N/A] in an [Urban] environment with
Number of lanes [1] as [R1.L1]
Road traffic direction [N/A]
Lane type [Traffic lane]
Lane markings [None]
Road surface type [Uniform] with surface condition [Dry]
Horizontal road geometry [Straight]
Vertical road geometry [Level plane]

Transverse road geometry [Undivided] with [No] roadside feature
Length [500 to 700] AND Lane width [4 to 4.2]
END

DYNAMIC ELEMENTS:
INITIAL: Vehicle [Ego] in [R1.L1]
AND Vehicle [EVT] in [R1.L1] with a [Longitudinal] offset of [300 to 300]
AND at relative position [F] with relative heading angle [0 to 5] to [Ego]
AND Global timer [T1] = [0]
AND Local timer [t1] = [0]

WHEN: [Ego] is [Going_Ahead]
DO: [EVT]
PHASE 1: [Drive_Away] [-, 5.56 to 5.56, 0 to 0] [Ego: -16.67 to -2.78, F]

END: [T1] == [40]

ENVIRONMENT ELEMENTS:
DO: [ENV1]
Wind [0 to 1]
Cloudiness [5 to 6]
Particulates [None]
Rainfall [None:N/A]
Time of the day [10:00 to 13:00]
Illumination [Day] with [Sun] as light source at [30 to 35] degree elevation AND [FSR] position

7.3.3. Example 3.3: Car-to-Car Rear Braking (CCRb)

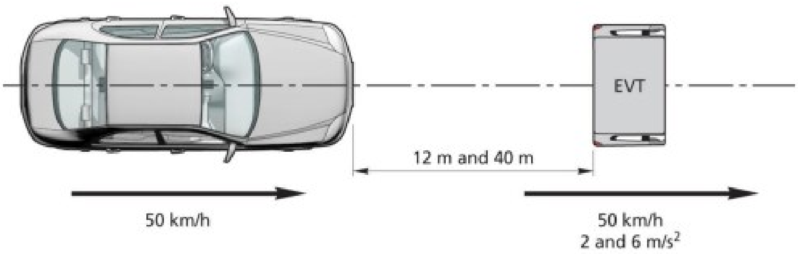


Figure 42 - EuroNCAP CCRb Example

Scenario description: Vehicle EVT as controlled agent starts with driving at the same speed as ego vehicle, after certain time EVT perform braking and cruising at a low speed.

SDL Level 1:

VERSION: 8.0
AUTHOR: "Jason Zhang"

During the day, there is a light, cloud covered sky, and a moderate breeze.

There is no junction present. Road1 is a straight, "Test track".

There are 2 vehicles called Ego and EVT. Ego is in Road1 and EVT is in Road1. When Ego is going ahead, EVT drives, at constant speed, ahead of Ego, and then EVT drives, slowing down, ahead of Ego.

SDL Level 2:

VERSION: 8.0
EXTENSION: None
AUTHOR: 'Jason Zhang'

SCENERY ELEMENTS:
DO: Map - roads and junctions network [Network2] as:
Junctions: None

Roads:
R1: START
Road type ['Test track'] as [R1] with zone as [N/A] AND speed limit of [N/A] in an [Urban] environment with
Number of lanes [1] as [R1.L1]
Road traffic direction [N/A]
Lane type [Traffic lane]
Lane markings [None]
Road surface type [Uniform] with surface condition [Dry]
Horizontal road geometry [Straight]
Vertical road geometry [Level plane]
Transverse road geometry [Undivided] with [No] roadside feature
Length [500 to 700] AND Lane width [4 to 4.2]
END

DYNAMIC ELEMENTS:
INITIAL: Vehicle [Ego] in [R1.L1]
AND Vehicle [VUT] in [R1.L1] with a [Longitudinal] offset of [12 to 40]
AND at relative position [F] with relative heading angle [0 to 5] to [Ego]
AND Global timer [T1] = [0]
AND Local timer [t1] = [0]

WHEN: [Ego] is [Going_Ahead]
DO: [VUT]
PHASE 1: [Drive_Away] [-, 25 to 35, 0 to 0] [Ego: 0 to 0, F]
WHILE: [t1] < [10]
PHASE 2: [Drive_Away] [-, 10 to 15, -6 to -2] [Ego: -20 to -15, F]

END: [T1] == [120]

ENVIRONMENT ELEMENTS:
DO: [ENV1]
Wind [0 to 1]
Cloudiness [5 to 6]
Particulates [None]
Rainfall [None:N/A]
Time of the day [10:00 to 13:00]
Illumination [Day] with [Sun] as light source at [30 to 35] degree elevation AND [FSR] position

7.4. Example 4

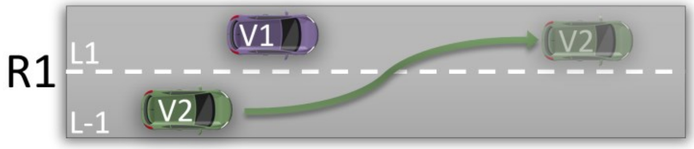


Figure 43 - Overtaking Scenario

Scenario description: Vehicle V2 as controlled agent overtakes and changes lane left cut-in into V1's lane. Forcing V1 to slow down to avoid collision.

SDL Level 1:

VERSION: 8.0
AUTHOR: "Jason Zhang"

During the day, there is a light, cloud covered sky, with heavy rain, and a light breeze.

There is no junction present. Road1 is a straight, radial road.

There are 2 vehicles called Vehicle1 and Vehicle2. Vehicle 1 is in Road1 and Vehicle2 is in Road1. When Vehicle1 is going ahead, Vehicle2 drives, accelerating towards Vehicle1 at its rear right, and then Vehicle2 drives, accelerating away from Vehicle1 at its front right. Vehicle2 then changes lane left in front of Vehicle1 and drives away on Road1.

SDL Level 2:

VERSION: 8.0
EXTENSION: None
AUTHOR: 'Jason Zhang'

SCENERY ELEMENTS:
DO: Map - roads and junctions network [Network1] as:
Junctions: None

Roads:
R1: START
Road type [Radial road] as [R1] with zone as [N/A] AND speed limit of [N/A] in an [Urban] environment with
  Number of lanes [2] as [R1.L1, R1.L-1]
  Road traffic direction [Left-handed]
  Lane type [Traffic lane]
  Lane markings [Broken line]
  Road surface type [Uniform] with surface condition [Dry]
  Horizontal road geometry [Straight]
  Vertical road geometry [Level plane]
  Transverse road geometry [Undivided] with [No] roadside feature
  Length [500 to 700] AND Lane width [4 to 4.2]
END

DYNAMIC ELEMENTS:
INITIAL: Vehicle [V1] in [R1.L1]
  AND Vehicle [V2] in [R1.L-1] with a [Longitudinal] offset of [12 to 40]
  AND at relative position [RSR] with relative heading angle [0 to 5] to [V1]

WHEN: [V1] is [Going_Ahead]
 DO: [V2]
   PHASE 1: [Drive_Towards] [-, 35 to 45, 4 to 5] [V1: 10 to 20, RSR]
   PHASE 2: [Drive_Away] [-, 35 to 40, -5 to 0] [V1: 5 to 15, FSR]
  WHILE: [V2] [Longitudinal] offset to [V1] < [7]
   PHASE 3: [LaneChangeLeft_CutIn] [-, 25 to 35, -5 to 0] [V1: 0 to 10, FSR]
   PHASE 4: [Drive_Away] [-, 25 to 35, -1 to 1] [V1: 0 to 10, F]

END: [V1] in [R1.L1]

ENVIRONMENT ELEMENTS:

DO: [ENV1]

Wind [0 to 1]

Cloudiness [5 to 6]

Particulates [None]

Rainfall [None:N/A]

Time of the day[10:00 to 13:00]

Illumination [Day] with [Sun] as light source at [30 to 35] degree elevation AND [FSR] position

8. References

[1] N. Kalra and S. M. Paddock, “Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability?,” *Transp. Res. Part A Policy Pr.*, vol. 94, pp. 182–193, 2016.

[2] T. Menzel, G. Bagschik, and M. Maurer, “Scenarios for development, test and validation of automated vehicles,” *arXiv*, vol. 2018-June, no. Iv, pp. 1821–1827, 2018.

[3] Euro NCAP, “Euro NCAP 2025 Roadmap: In pursuit of Vision Zero,” 2017.

[4] S. Khastgir, S. Birrell, G. Dhadyalla, and P. Jennings, “The Science of Testing: An Automotive Perspective,” in *SAE Technical Papers*, 2018, vol. 2018-April.

[5] K. Go and J. Carroll, “The blind men and the elephant: views of scenario-based system design,” *Interactions*, vol. 11, no. 6, pp. 44–53, 2004.

[6] S. Geyer *et al.*, “Concept and development of a unified ontology for generating test and use-case catalogues for assisted and automated vehicle guidance,” *IET Intell. Transp. Syst.*, vol. 8, no. 3, pp. 183–189, 2014.

[7] S. Ulbrich, T. Menzel, A. Reschka, F. Schuldt, and M. Maurer, “Defining and Substantiating the Terms Scene , Situation , and Scenario for Automated Driving,” *Proc. 2015 IEEE 18th 18th Int. Conf. Intell. Transp. Syst.*, pp. 982–988, 2015.

[8] E. de Gelder *et al.*, “Ontology for Scenarios for the Assessment of Automated Vehicles,” 2020.

[9] S. Khastgir, G. Dhadyalla, S. Birrell, S. Redmond, R. Addinall, and P. Jennings, “Test Scenario Generation for Driving Simulators Using Constrained Randomization Technique,” *SAE Tech. Pap.*, vol. 2017-March, no. March, 2017, doi: 10.4271/2017-01-1672.

[10] C. Neurohr, L. Westhofen, M. Butz, M. Bollmann, U. Eberle, and R. Galbas, “Criticality Analysis for the Verification and Validation of Automated Vehicles,” *IEEE Access*, vol. 9, no. i, 2021, doi: 10.1109/ACCESS.2021.3053159.

[11] M. Brackstone *et al.*, “OmniCAV: A Simulation and Modelling System that enables CAVs for AI,” *IEEE Intell. Transp. Syst. Conf.*, vol. 2020, pp. 2190–2195, 2019.

[12] E. de Gelder and J. P. Paardekooper, “Assessment of Automated Driving Systems using real-life scenarios,” *IEEE Intell. Veh. Symp. Proc.*, no. Iv, pp. 589–594, 2017.

[13] A. Pütz, A. Zlocki, J. Bock, and L. Eckstein, “System validation of highly automated vehicles with a database of relevant traffic scenarios,” *12th ITS Eur. Congr.*, no. June, pp. 1–8, 2017.

[14] G. Bagschik, T. Menzel, and M. Maurer, “Ontology based Scene Creation for the Development of Automated Vehicles,” *2018 IEEE Intell. Veh. Symp.*, no. Iv, pp. 1813–1820, 2018.

[15] “Road Safety Data - STATS19,” *UK Department for Transport*, 2020.

[16] E. Esenturk, S. Khastgir, A. Wallace, and P. Jennings, “Analyzing Real-world Accidents for Test Scenario Generation for Automated Vehicles *.”

[17] S. Chen, S. Khastgir, I. Babaev, and P. Jennings, “Identifying Accident Causes of Driver-Vehicle Interactions Using System Theoretic Process Analysis (STPA),” *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 2020-Octob, pp. 3247–3253, 2020, doi: 10.1109/SMC42975.2020.9282848.

[18] I. Information, “Operational Design Domain (ODD) taxonomy for an automated driving system (ADS) – Specification,” no. April, 2020.

[19] T. Menzel, G. Bagschik, L. Isensee, A. Schomburg, and M. Maurer, “From functional to logical scenarios: Detailing a keyword-based scenario description for execution in a simulation environment,” *IEEE Intell. Veh. Symp. Proc.*, vol. 2019-June, pp. 2383–2390, 2019.

[20] ISO, “Intelligent transport systems — Low-Speed Automated Driving (LSAD) Systems for Predefined routes — Performance requirements, system requirements and performance test procedures - ISO 22737,” 2021. .

[21] X. Zhang, S. Khastgir, and P. Jennings, “Scenario Description Language for Automated Driving Systems: A Two Level Abstraction Approach,” *IEEE Int. Conf. Syst. Man Cybern.*, 2020.

[22] ASAM, “ASAM OpenSCENARIO - User Guide,” pp. 0–61, 2020.

[23] SAE International, “Surface Vehicle Recommended Practice - J3016,” 2018.