ASAM OpenSCENARIO

List of Features and Requirements

Authors:	Thomas Thomsen Global Technology Manager ASAM e.V. Pierre R. Mai PMSF IT Consulting
Date:	2019-01-09
Version:	1.8

Content

1 Purpose

The purpose of this document is to capture features and requirements for the further development of OpenSCENARIO. The input for this document originates from a series of meetings with industry-experts on the subject matter. Their statements from presentations and discussions has been compiled into concise and non-overlapping feature- and requirements descriptions. They are the foundation for further project planning and project proposals at ASAM.

A "feature" in the context of this document is tool functionality, which is perceivable by a user of a standard-compliant tool, with which he can interact, and which is clearly separated from other functions of the tool.

A "requirement" in the context of this document is a description of a necessity that shall be met by the standard or the standard-compliant tool, respectively. Requirements can also be used for describing details of a feature.

The list of features for OpenSCENARIO shall be as complete as possible in this document. Features typically describe major parts of a standard. One feature can correspond to one chapter in the later standard document. The specification effort for features is typically very high. This must be known prior to project start, so that the necessary resources for the project can be allocated.

The list of requirements does not have to be complete prior to project start. Requirements typically have a lesser impact on the efforts to create a standard. This document shall only include those requirements, which are deemed as important and require acceptance by the ASAM community prior to project start. All further requirements for standard development can be defined after the project start.

The chapter "Other Topics" includes those expert contributions, which can neither be classified as a feature nor as requirement, or which will certainly not be part of an OpenSCENARIO standardization project. Most of the topics would potentially be realized in a software implementation project, which would produce source code or tools that support the application of OpenSCENARIO.

2 General Terms of Reference

The following terms of reference provide general points of guidance to the continued evolution of OpenSCENARIO based on the input of industry experts:

1. In order to harmonize standardization work on the OpenX standards and to aid in the scope definition of each work project, a **common Glossary** should be drawn up, that clearly defines the meaning, role and scope of common parts and terms used to describe a test run, such as: Test case, scenario, vehicle under test, road network, etc. The glossary should focus discussions on where certain information should or should not reside (e.g. environmental data, road surface information, …).
2. OpenSCENARIO is going to be used in quite different constellations, leading to differing and potentially incompatible requirements. For example different information is considered mandatory for different use cases: Attributes that might be mandatory for one use case (e.g. eye distance of driver) are not needed for other use cases. In order to resolve those conflicting requirements, thought should be given to defining different **profiles/feature sets** that encompass the required features and attributes for a certain set of use cases.
3. For each use case the **degree** of cross-tool **comparability and reproducibility** of results that is needed should be clearly specified. Based on this information it will then be possible to define **profiles/feature sets** that ensure comparability/reproducibility while allowing for more flexible features that do not require the same degree of comparability and reproducibility across tools.
4. Given the speed of development in the whole area of autonomous systems and their development and testing, it should be possible to develop **experimental profiles** and **extensions** of OpenSCENARIO that enable the usage of OpenSCENARIO in new use cases without waiting for finalized solutions from the base standard. The mechanisms put in place to enable this should allow for easy integration of such extensions into the base standard once they have reached maturity and stability.
5. Different use cases will require **different levels of scenario definition** (different levels of detail, e.g. mathematical trajectory vs. logical route, different levels of complexity, e.g. simple overtaker vs. complex long-running city traffic scenarios). It should be examined whether the same scenario definition language can cater to all of those use cases, or whether **different languages or language subsets** are more suitable. The focus of this examination should be the ability to express relevant scenarios concisely and readably in order to allow maintainable flexible scenario definitions for those use cases.
6. Regardless of how different levels of scenario definition are handled, it should be possible to **map higher levels** of description to **lower levels** of description through **automatic translation** in such a way, that missing specificity is added through reference implementations. The resulting in low level descriptions should be more portable across implementations. This approach does not preclude implementations from directly supporting the higher levels of description.
7. In all instances where possible, the **reduction of complexity** in describing scenarios should be **a priority**. This can be achieved for example by providing useful defaults covering common cases, definition of intuitive primitives, or by providing information of best practice approaches for common goals.

3 Features

ID	F001	Priority	Normal
Title	Maneuver Model		
Description	Complete the specification of maneuver descriptions in OpenSCENARIO with advanced features: (a) time-based maneuver definition (b) distance-based speed profile (c) splines (d) general composition operators, such as 'serial', 'parallel', 'phase', 'mix' (e) activate and constrain sub-scenarios (f) explicit and implicit constraints (g) define lateral speed the same way as longitudinal speed (h) allow to disable lateral control (i) maneuver hints, e.g. suggest lane change, follow trajectory as long as possible, accelerate, decelerate, keep speed (j) clearly specify which direction a vehicle is travelling within a lane, e.g. overtaking in the "wrong" lane (k) meta data, e.g. risk, category/tags, expected execution time, max. acceleration, max. speed (l) driver actions, e.g. ignition on/off, ACC on/off, horn, indicate (m) multiple, simultaneous actions, e.g. brake and lane change in an evasive emergency maneuver (n) component failures, e.g. sensor failure, engine failure. (o) attach trailer to the vehicle (p) camera angles (q) embed simulation control messages (r) country of applicability (s) use-case (e.g. highway, urban, inter-urban, etc.) (t) ID (u) version (v) any (any other meta data) The maneuver model shall furthermore allow to specify generic maneuvers, from which groups of specific maneuvers can be generated. Note: R010 "Synchronize maneuvers and events" is related to this feature.		

Rational	Basic maneuver trajectories can already be defined in OpenSCENARIO, such as sinusoidal trajectories. More complex trajectories shall be available such as splines, piecewise polynomial parametric curves or speed profiles. The other maneuver model features make the definition dynamic and responsive to the environment.

ID	F002	Priority	Normal
Title	Driver Model		
Description	The standard shall contain a model for describing driver behavior. The model describes various aspects of the driver's behavior in traffic situations. Examples are: reaction times, distance to the ahead vehicle, longitudinal and lateral acceleration, speed of steering-angle change, etc. It might also be considered to define types of drivers, e.g. 'aggressive' or 'cautious', which bundle characteristic values of them. The model represents either a human driver or an AI driver. The model shall support the description of typical ADAS functions, such as ACC. The maneuver model can be linked to a driver model. The specification of models shall include, which part of the maneuver is exactly executed as specified, and which part of the maneuver model is just the reference for the driver model to be followed (but not necessarily executed in an exact way).		
Rational	The driver model is required to simulate complex traffic situations in an effective and quick way. Without a driver model, the drive behavior of each car in a multi-vehicle traffic simulation would have to be described manually, which is time consuming.		

ID	F003	Priority	Normal
Title	Traffic Model		
Description	The traffic model defines the movement of traffic participants in the surrounding of the ego vehicle. The model shall allow to automatically generate complex traffic scenarios that includes moving vehicles, pedestrians, bicycles, animals and others. The model shall include deterministic and stochastic traffic scenario definitions. The latter requires parameters for traffic densities, safe-distance rules and traffic-light rules.		
Rational	Traffic simulation is a fundamental prerequisite for testing ADAS and AD systems. Together with the static environment (roads, buildings, traffic signs, etc.), they provide the primary objects that the ego vehicle has to respond to.		

ID	F004	Priority	Normal
Title	Weather Model		
Description	The data model shall include elements for describing the weather, such as precipitation, fog, wind, lighting and other phenomena. This shall allow the simulation of: • road conditions and its effect on friction between tire and surface. • visual conditions and its effect on sensor perception.		
Rational	Weather has an impact on sensor performance, quality of signals, object detection and vehicle dynamics. This is an important aspect in simulation.		

ID	F005	Priority	Normal
Title	Environmental Event Model		
Description	A data model shall be added to the standard, which allows to describe information originating from the infrastructure to influence the drive behavior of the ego car.		
Rational	In an infrastructure-to-vehicle communication scenario, environmental sensors capture specific traffic situations and send information to the vehicle. The vehicle responses to the information, usually by changing the route or by changing the immediate drive behavior. Typical examples are report of a traffic jam, where the ego vehicle changes the routing to go around the jam, or reporting of an accident in addition to the traffic jam, where the ego vehicle creates a corridor for emergency vehicles.		

ID	F006	Priority	Normal
Title	Vehicle Dynamics		
Description	The current scope of OpenSCENARIO is primarily for drive and traffic simulation. The standard shall also be usable for vehicle dynamics simulation.		
Rational	ADAS and autonomous driving cars can be simulated without considering vehicle dynamics effects. However, simulation results are then limited to test and verify the pure drive logics such as trajectory calculation, etc. It can not be simulated, when calculated drive maneuvers produce unplanned movements of the ego vehicle, e.g. excessive positive or negative wheel slip, oversteering, understeering, skidding and others. Simulation-based testing would be incomplete and would have to be complemented by vehicle testing. If the scope of vehicle dynamics is included in simulations,		

	then this would significantly increase the simulation fidelity and the overall value of simulation.

ID	F007	Priority	Normal
Title	Parameter Stochastics		
Description	Instead of just describing fixed parameter values, the standard shall also allow methods for describing parameter distributions and variations: (a) Intervals (e.g. min, max) (b) Stochastic distributions (e.g. linear- or gauss-distribution) (c) Discretely defined distributions (e.g. histograms)		
Rational	When scenarios are used in tests, then scenarios shall be re-usable, i.e. test engineers shall not need to create one scenario for each test. "Pulk" scenario definitions shall be available based on density data, which are usable for multiple tests. This can be (partially) achieved by parameter variations based upon distribution descriptions. The variation details shall become part of the scenario description, and not the test description.		

ID	F008	Priority	Normal
Title	High-Level Maneuver Descriptions		
Description	The standard shall provide a method for maneuver descriptions on a higher level of abstraction, aka key-scenario descriptions. This shall contain only the logical description of scenarios with as few parameters as possible. The high-level description is then automatically transformed into the detailed description of the lower level. Two versions of translation are conceivable: 1. Defining a simple translation on a high level, which has a defined semantics on / translation to the detailed level. The detailed level will offer a higher expressive power. 2. Having an incomplete translation to the detailed level. This means that parts of the OpenSCENARIO description would have to be completed manually. This might include, that one part of the maneuver is undefined and shall be handled by the driver- or traffic- models during simulation. The lower level description is the current OpenSCENARIO data model. There are three alternative proposals for the method of high-level maneuver description:		

	a) Data model: Same method as current OpenSCENARIO data model, but on a higher, more abstract level. b) Language: Domain-specific language (DSL). c) Language: General-purpose language and domain-API. If a language is chosen (proposal b or c), then a possible standardization approach could utilize an object-oriented model: 1. Create UML-model with the top-level objects "Road", "Sensor", "Traffic" and "Driver" and clear semantics description for each object. 2. Define attributes for each object. Objects, attributes and their relations constitute the data content of the UML model. 3. Define operations for each object. This constitutes the methods (aka API or code extension) of the UML model. In order to support constraint-based concepts, the OO-model has to take state-handling problems into account. High-level descriptions shall support "trigger-action" type of expressions. Trigger conditions shall be combined with relational operators (and, or).
Rational	In principal, the high-level descriptions are created by humans, which are domain experts, not necessarily simulation experts. They contain key-scenarios such as "cut-in", "left turn across path" or "highway merge". Low-level descriptions shall be automatically generated from them and only be read and used by tools, such as simulators. A higher level of description shall lower the specification burden for the user. They can read, understand, review and correct scenarios from others, or write scenarios by themselves and carry out basic debugging tasks. It allows to build libraries of maneuver descriptions, which are then used to (auto-) generate specific maneuver-variants and tests. High-level maneuver descriptions produce many different, interesting instances, exposing unconsidered combinations. Writing everything explicitly is not manageable, as it does not scale. If a language is chosen (proposal b or c) and implemented as an object-oriented UML-model, then schema and API prototypes can be automatically generated from the model. The API provides programmatic and abstracted access to the data and methods. This allows to exchange implementations for specific objects without the need to adjust the test specifications that depend on them. For example, a test might call a method to calculate the driver behavior in a given test scenario. The driver behavior method may originate from different tool vendors. They can be changed for a given test, without the need to change the test specification itself. Furthermore, this approach is programming-language independent, i.e. it can be easily mapped to popular languages such as C++, C#, Java or Python.

	With the language-approach, specific triggers, actions and events can be specified with mathematical terms, which are particularly easy readable by humans.

ID	F009	Priority	Normal
Title	Replay of Recorded Scenarios		
Description	Scenarios may be defined from pre-recorded trajectories, which shall be replayed during simulation.		
Rational	This is one fundamental method of scenario description.		

ID	F010	Priority	Normal
Title	Automatic Parameter Calculation		
Description	Instead of manually setting each parameter in the data model, the standard shall allow to specify mathematical formulas to calculate parameters. Those parameters would be automatically calculated, once the input arguments of the formula are known.		
Rational	There are parameter dependencies in the OpenSCENARIO data model. Specific parameters depend on other parameters. With the definition of mathematical formulas for such dependent parameters, the work burden to populate the data model can be greatly reduced. End-users would only have to manually determine the independent parameters. All other parameters would be automatically calculated. The chance for parameter-inconsistency is greatly reduced. This feature should support the specification of multiple sensor platforms and versions.		

ID	F011	Priority	Normal
Title	Additional Meta Data for Parameters		
Description	Parameters shall have attributes for URI and name space.		
Rational	The URI would make parameters shareable. The name space allows to distinguish between standardized parameters and user-defined parameters. The name space also allows to define country-variants of parameters.		

4 Requirements

ID	R001	Priority	Normal
Title	Avoid multiple ways of defining the same maneuver.		
Description	In the current version of OpenSCENARIO, it is possible to define simple maneuvers, such as trajectories, in different ways. The standard shall be reviewed under this aspect. Definition alternatives shall be reduced to just one alternative, whenever possible.		
Rational	When given the choice between alternative, standard-compliant definition methods, such as a the vehicle maneuver trajectory, reality has shown, that different tool vendors just implement one alternative, and do not support the other alternatives offered by the standard. When tools from different vendors are integrated into one tool chain, this regularly causes interoperability problems and broken tool chains, because the definition method supported by one tool is not supported by another tool, which would then error-out. This effectively makes the standard useless. In principal, a standard shall be strict and shall define only one specification method. This is a fundamental guideline to ensure tool-interoperability. If we strive to standardize maneuver descriptions, it will be required to have a solid inventory of well known-maneuvers that clearly prescribe how it's defined. From this standardized template every user can add complexity and richness by adding more elements to scenario.		

ID	R002	Priority	Normal
Title	Define elements as 'mandatory' only when absolutely needed.		
Description	Define elements of the OpenSCENARIO data model as mandatory only when it is absolutely required to run the simulation, maintain tool interoperability and to obtain correct simulation results. All other elements shall be optional.		
Rational	Currently, too many elements of the OpenSCENARIO data model are declared as 'mandatory', which are not really required in many simulation cases. For example, the 'eye distance' in the driver model is 'mandatory', but many simulators do not use this parameter. It should be 'optional'.		

ID	R003	Priority	Normal
Title	Maintain independence of standards, open linking and default parameters between standards.		
Description	The standards OpenDRIVE, OpenCRG and OpenSCENARIO shall be independent from each other. OpenSCENARIO shall have a generic interface to road and 3D environment description standards such as glTF. OpenSCENARIO shall have default parameters for OpenDRIVE, such as traffic light settings and street lamp settings. Furthermore, OpenSCENARIO may provide a parameter for the definition or "rating" of the road surface.		
Rational	There are simulation cases, where only one of the OpenX-standard shall be used with references to data to non-OpenX-standards. OpenDRIVE does not allow named reference points, which are available in non-OpenX standards. OpenSCENARIO shall be able to reference 3D objects defined in an glTF-compliant format. Objects may be perceived and described in accordance with OSI. Some properties of road networks are of dynamic nature and may not be described in OpenDRIVE, e.g. the actual status of traffic lights or street lamps. It shall be possible to define such settings in OpenSCENARIO, which can then be taken in case they are missing in OpenDRIVE.		

ID	R004	Priority	Normal
Title	Define three levels of control for ego vehicles.		
Description	The standard offers three modes of specifying the control of an ego vehicle (aka vehicle-under-test). The ego vehicle is controlled: (a) completely by scenario description. (b) partially by driver model and partially by scenario description. (c) completely by an external vehicle controller, e.g. human driver or AD-system. It shall be possible to switch the mode within one scenario. In case of (b), it shall be possible to switch the driver model within one scenario. Case (c) requires an API in the simulator to input the control commands from the external driver. For case (C), the objective of the ego vehicle shall be included in the data model, e.g. drive to a specific position or follow the car ahead. There should be support for multiple regions that can be defined as goal regions as well as ‘fail’ regions.		
Rational	The three level of vehicle control occur in practice. They shall be supported in scenario descriptions.		

	The definition of goal regions allows user to express a scenario pass/fail criteria also through maneuvers or regions the ego vehicle needs to avoid to succeed the scenario (e.g. stopping at a crosswalk). The success criteria therefore defined as 'all goal regions were driven over, and none of the fail regions were driven over'.

ID	R005	Priority	Normal
Title	Allow tool-vendor specific extensions.		
Description	The standard shall allow a method to add tool-vendor specific elements and parameters to the data model, without breaking standard-compliance, schema validation and tool interoperability.		
Rational	An industry standard is always a definition of a minimum set of features and requirements. Particularly for tool-standards, tool-vendors will always add features and meet additional (typically more strict) requirements on top of the standard feature-set and requirements. Tool vendors do this to add USPs to their product and set themselves apart from their competitors. There must be a standard-compliant method to do this. Otherwise, tool vendors will likely create their own extensions and deviations from the standard, which are incompatible to all other tools and effectively making the standard useless. This has been observed with other standards in the past. Although vendor-specific options are not an ideal solution from a standardization-point-of-view (such extensions will not work with other tools), ignoring this requirement will jeopardize the overall success of the standard.		

ID	R006	Priority	Normal
Title	Allow definition of feature subsets.		
Description	Allow the definition of groups of features, aka subsets or profiles. This allows tool vendors to inform end-users, which groups of features are supported, and which are not supported.		
Rational	Complex data models and underlying features are not necessarily completely implemented in tools. Tool vendors typically choose on feature-level, which features they are going to support and implement in their tool, and which are not implemented. Feature subsets shall provide means of unambiguously documenting this choice and communicating it to end-users. End-users can better and much earlier identify tool-chain integration issues before they occur in live operation. Another reason for subsets is to define tests that are valid for testing specific components of an ADAS or AD system. For example,		

	when testing only planning and control components of the AD system, some configurations that are only relevant for sensor simulation or perception may not matter.

ID	R007	Priority	Normal
Title	Define simulation results reproducibility.		
Description	The standard shall define for each feature, if exact reproducibility of simulation results among all standard-compliant simulators is required. For all other features, different simulators are allowed to produce different simulation results. It might also be considered to define an inexact reproducibility, i.e. defining a tolerance interval for simulation results. Another suggested approach for this requirement is to define a dual interpretation: • Active: How to or try to cause this scenario. • Passive: How to monitor that this scenario has happened.		
Rational	Standardized features often leave the expectation at end-users, that the same definitions (e.g. maneuver descriptions) are exactly executed the same way at each tool with the exact same result. This is true for many standards, but it is not true and can technically not be achieved with drive and traffic simulators. For example, maneuver descriptions can be executed exactly the same way on each simulator in open-loop, if no vehicle dynamics effects are considered. The results will be different, when dynamics are simulated and in closed-loop. The inclusion of driver behavior models will typically always result in different simulation results in open- and closed-loop simulation. This expectation shall be defined in the standard to set the expectation of end-users right.		

ID	R008	Priority	Normal
Title	Maneuver descriptions shall be suitable for open-loop and closed-loop simulation.		
Description	The standard shall allow the definition of maneuvers for open-loop simulation and closed-loop simulation.		
Rational	Open-loop maneuvers describe trajectories, that the ego vehicle follows exactly as described. This can be expressed in OpenSCENARIO with the current version. Closed loop-maneuvers describe trajectories, that the ego vehicle shall follow as close as possible. The ego vehicle must also considering the complete traffic situation in its immediate surrounding and responds to it. The described trajectory is the reference for a closed-loop controller, such as an AI-driver or an AD-system. The		

	actual driven trajectory may deviate from the reference trajectory, e.g. when there are obstacles on the road, or surface conditions (ice) do not allow to drive narrow curves at the set-point speed. The differentiation of maneuver descriptions for either open-loop or closed-loop simulation is currently not clear in the standard.

ID	R009	Priority	Normal
Title	Define parameter boundaries.		
Description	Parameters shall have attributes, which define their upper and lower limits.		
Rational	Parameter boundaries shall avoid to populate the data model with invalid, meaningless or unrealistic values.		

ID	R010	Priority	Normal
Title	Synchronize maneuvers and events.		
Description	The movement of multiple vehicles in a maneuver description can be synchronized at specific points of time, specific coordinates on the road or at the occurrence of specific events. The description shall include logical constraints. Events might be maneuver events or events related to the ego vehicle. The latter might include driver-initiated events or component failures, as described in F001.l or F001.n.		
Rational	Maneuver synchronization is a fundamental description requirement. For example, in a "left turn across path" maneuver, the adversary car shall arrive at a specific road coordinate near an intersection, when the ego car starts the left turn maneuver.		

ID	R011	Priority	Normal
Title	Allow the definition of success criteria for maneuvers.		
Description	Maneuver descriptions shall include success criteria, i.e. conditions that evaluate to 'true' when the maneuver was executed as intended. Success criterial shall include: (a) primitives (e.g. distance to next object, contact occurred, position, speed, signals) (b) logical operators (e.g. and, or, not) (c) timing (e.g. all frames, periodically, at end) The success criteria may be directly included in the OpenSCENARIO data model, or become part of a separate test specification		

	with a reference to the corresponding maneuver description in OpenSCENARIO.
Rational	Success criteria are necessary to be able to express complex scenarios. They are needed for testing autonomous driving systems. They form an elementary part of the composition operators of scenarios. Success criteria are potentially complex and shall potentially be evaluated directly after execution or on recording. This requires that the criteria is included in the OpenSCENARIO data model.

ID	R012	Priority	Normal
Title	The description format shall be suitable for manual scenario creation in text editors.		
Description	Users shall be able to manually write scenario definitions in a text editor. This means that the OpenSCENARIO description format (currently XML) shall not just meet requirements for machine-readability.		
Rational	XML-files become hard to read and barely comprehensible for humans, if specific design restrictions of the schema are not followed. For example, the data model for OpenSCENARIO shall avoid abstract elements, keep a top-down hierarchical structure as much as possible and avoid cross-references between many elements. A clear separation of features and feature groups in the structure with preferably no references (dependencies) between them would keep the model understandable. Common elements, which are referenced by many other elements (e.g. Units, Physical Dimensions, etc) shall be put into a global data dictionary and use data model patterns that are already used in other ASAM and AUTOSAR standards. The standardization work group may agree on modeling style guidelines before starting the actual specification work.		

5 Other Topics

5.1 Checker Tool

A checker tool would read OpenSCENARIO XML-files and carry out rule-based checks. Typical checks are:

- Parameter plausibility: Is one or a combination of parameters plausible and realistic?
- Parameter consistency: Do logically required parameters exist and are they free of contradictions?
- Logical plausibility: Are descriptions (such as maneuvers) realistic in the sense that they could also be performed in real life?

Schema-validation (syntax and structure of the file) would be out-of-scope, as this is already covered by the XSD.

5.2 Parser

A common parser shall be available, which reads OpenSCENARIO XML-files into memory. The data is then accessible for tools, such as editors, simulators, post-processors and generators.

5.3 Data Access API

A common data access API shall be specified and made available as source-code, which allows to read from and write to the OpenSCENARIO data model in memory. The API would be primarily used in test automation systems. For example, the API could be used in test scrips to programmatically change parameters of the OpenSCENARIO data model in memory.

5.4 Test Specifications

Scenario descriptions and test descriptions shall be separated from each other. This allows to create scenario libraries and re-use them in tests. OpenSCENARIO does not cover the testing use-case. A separate standard for test-specifications used in drive simulators can be developed to cover this use-case. There seems to be a preference to use a scripting language for test definition. Two alternative approaches would be available:

a) Define a domain-specific language (DSL).

b) Use a general purpose language and define an API .

A DSL would include typical language elements for configuring and executing tests, and to evaluate and capture the results. It would probably also include language elements to create test-variations during run-time. A general purpose language (such as Python or LUA) would carry out the same functions via API calls.

5.5 Tool Qualification

To reach the objective of OpenSCENARIO that same scenario descriptions shall produce same simulation results on different simulators, a tool qualification suite is proposed that includes:

- A common library of scenario and maneuver descriptions, including variations.

- Definition of expected simulation results.
- Fail/pass criteria for tool qualification.

5.6 Traffic Simulation Driver Reference Models and Implementations

The data model of OpenSCENARIO shall be extended with models for drivers for traffic simulation. This is not the same than the driver model for the ego vehicle. Traffic driver models are required to setup and simulate complex traffic situations for the ego vehicle. The traffic drivers dynamically respond to the ego vehicles maneuvers. In order to do that, the traffic driver models shall have a defined set of standard maneuvers, such as 'ride in traffic', 'overtaking', 'lane change', etc, which defines their standard behavior in such situations, when they are triggered during simulation. The standard maneuvers may exist in different profiles, such as 'cautious driver' or 'aggressive driver'. The models allow to create highly complex and dynamic simulation scenarios, which are close to real traffic situations.

Driver models (for ego and traffic vehicles) are typically used in closed-loop simulations. As the simulation runs are highly dynamic, there is a high probability that different simulators produce different simulation results from the same scenario descriptions and test-cases. In order to mitigate this risk, it is proposed to develop reference implementations for driver models. The reference implementation reads the diver model description and test-case, and outputs the expected drive trajectory. This drive trajectory is the reference for simulators.