SIMULATION SCENARIOS

Public report

Project within FFI - Elektronik, mjukvara och kommunikation 20161213

Authors
Emil Knabe - emil.knabe@volvocars.com
Bruno Augusto - bruno.augusto@vti.se
Viktor Johansson - viktor.johansson@astazero.com
Henrik Lönn - henrik.lonn@volvo.com
Anders Bengtsson - anders.b.bengtsson@hiq.se
Frida Reichenberg - frida.reichenberg@ri.se
Niklas Mellegård - niklas.mellegard@ri.se
Benjamin Vedder - benjamin.vedder@ri.se

Date 2019-05-20

FFI Fordonsstrategisk Forskning och Innovation

Content

1. Summary

To meet the intense development of active driving support functions and ultimately autonomous vehicles, new approaches are needed for representing the environment in test applications. We need to perform tests on many kinds of roads and in MANY different traffic situations. Since the physical world is limited in time and space we need to seek complementing solutions in virtual test systems.

Even though simulators have been capable to represent traffic scenarios in decades there are a few great challenges: 1. The creation of scenarios and environment models needs to be much more efficient. 2. Reuse of scenarios and models requires standardized formats. 3. Test environments need to adapt to the standard formats.

This project addresses exactly these three challenges. The chosen approach is to start out from a set of carefully chosen key scenarios and go all the way to implementation and demonstration in a set of strategic test environments of various kinds. We believe this to be a good way to learn, evaluate and develop the data formats. If the result is successful, the actual harmonization of test environments has already started, not only being theorized about. There is no time to lose.

Three key scenarios have been chosen: Cut-in, Left Turn Across Path - Opposite Direction (LTAP-OD) and Highway-merge. The criteria for selecting these were a combination of their relevance for the project stakeholders and their broad and challenging requirements on the formats being used. The key scenarios were broken down and described in the XML-based and open data format OpenSCENARIO.

Among the appointed set of test environments are full scale driving simulators - for example VTI SIM4, small desktop solutions - for example based on the Unity3D game engine, but also test infrastructure at the AstaZero physical test track and the Adapt continuous integration platform. This is a very broad and representative selection that spans over different technologies, cost levels, purposes and use cases.

To be able to use the OpenSCENARIO format the test environments have been adapted in different ways. Multiple different software libraries, partly based on common modules, for executing scenarios have been implemented and integrated.

A tool chain for the creation of road models based on map data have been established. The process is based on commercial tools complemented by automated scripts and step-by-step guides to maximize efficiency. The goal being that a typical road model should not take more than an hour to create. This goal has been achieved for the kind of road models needed for the three key scenarios.

Finally the project demonstrated the key scenarios in all appointed test environments, validating that harmonization of test environments with regard to environment simulation and traffic simulation in particular is possible using OpenSCENARIO. At least for the kinds of scenarios and test environments covered by the project.

The most significant critique of the OpenSCENARIO format is the lack of a high level maneuver representation. It has been proposed to complement the format with a DSL (Domain Specific Language) offering a much more condensed way of describing scenarios. This is a work already started within the ASAM OpenSCENARIO standardization project.

Observing the growing interest from both OEMs and tool providers we believe the establishment of OpenSCENARIO having good chances. Both for simulation and additional application areas still to be discovered.

2. Sammanfattning på svenska

För att möta den intensiva och snabba utvecklingen av automatiska körstödsfunktioner, i förlängningen självkörande fordon, behövs helt nya grepp för hur omvärlden hanteras på testsidan. Vi behöver utföra prov på många typer av vägar och i MÅNGA olika trafiksituationer. Och eftersom inte den fysiska världen räcker till - rum och tid - är vi hänvisade att söka kompletterande lösningar i virtuella provmiljöer.

Även om simulatorer har kunnat representera trafikscenarier i årtionden finns några stora utmaningar: 1. Framtagning av scenarier och omvärldsmodeller behöver effektiviseras. 2. Återanvändning av scenarier och modeller kräver standardiserade format. 3. Testmiljöer behöver anpassas till standardformaten. Det är precis dessa tre utmaningar detta projekt adresserar. Vald taktik är att utifrån några väl utvalda nyckelscenarier gå hela vägen till implementation och demonstration i utvalda strategiska testmiljöer av olika slag. Detta är ett bra sätt att lära, utvärdera och utveckla dataformaten. Blir resultatet lyckat har vi dessutom redan påbörjat harmoniseringen av testmiljöer, inte bara teoretiserat. Här finns ingen tid att förlora.

Tre nyckelscenarier valdes ut: Cut-in, Left Turn Across Path - Opposite Direction (LTAP-OD) samt Highway-merge. Kriterierna var en kombination av att vara relevanta för projektets intressenter och samtidigt spänna upp en bred och utmanande kravsättning av formatet de ska beskrivas i. Nyckelscenarier har sedan brutits ner och beskrivits i det XML-baserade öppna dataformatet OpenSCENARIO.

Bland utvalda testmiljöer finns stora körsimulatorer, t.ex. VTI SIM4, små desktoplösningar, t.ex. baserade på spelmotorn Unity3D men också testinfrastrukturen på provbanan AstaZero och ramverket Adapt för continuous integration. Det är ett mycket brett och representativt urval som spänner över olika tekniker, kostnader, ändamål och användning.

För att kunna använda OpenSCENARIO har testmiljöerna anpassats på olika sätt. Flera olika mjukvaror, delvis baserade på gemensamma moduler, för exekvering av scenario har implementerats och integrerats.

En verktygskedja för skapande av vägmodeller utifrån kartdata har etablerats. Den är baserad på kommersiella verktyg som har anpassats och kompletterats med egenutvecklade verktyg och steg-för-steg guider för att effektivisera så långt som möjligt. Målet är att en typisk vägmodell ska inte ta mer än en timme att konstruera. Detta mål är uppfyllt för den typen av vägar som tagits fram för nyckelscenarierna.

Slutligen har projektet demonstrerat nyckelscenarierna i samtliga utvalda testmiljöer. Vi har därmed visat att harmonisering av testmiljöer med avseende på omvärldssimulering och trafikscenarier i synnerhet är fullt möjligt med hjälp av OpenSCENARIO. Åtminstone för de typer av scenarier vi använt oss av.

Den viktigaste kritiken mot OpenSCENARIO-formatet är avsaknad av en representation av manövrer på högre nivå. Det har föreslagits att komplettera formatet med ett DSL (Domain Specific Language) som kan beskriva scenarier på ett betydligt mer kortfattat sätt. Detta är något som redan börjat jobbas med inom OpenSCENARIO.

Av det växande engagemanget att döma, både hos OEM:er och verktygsleverantörer, bedömer vi att OpenSCENARIO har goda utsikter att etableras. Användningsområdet kan mycket väl komma att utvidgas bortom simulering.

3. Background

Driver support systems are evolving rapidly. Many challenges must be dealt with while moving towards the ultimate self-driving vehicle, like legal frameworks, insurance policies, certification, and regulations. But toughest of them all are the technical challenges. In a distant future autonomous driving may be easy - all vehicles communicating seamlessly via standard protocol, seeking and negotiating their way towards the desired destinations. To achieve this, these systems must first monitor, understand and react to unpredictable surrounding traffic, either controlled by humans or incompatible autonomous systems.

As the range of scenarios the vehicle is expected to handle increases exponentially, the test scope grows and the test systems need to evolve accordingly.

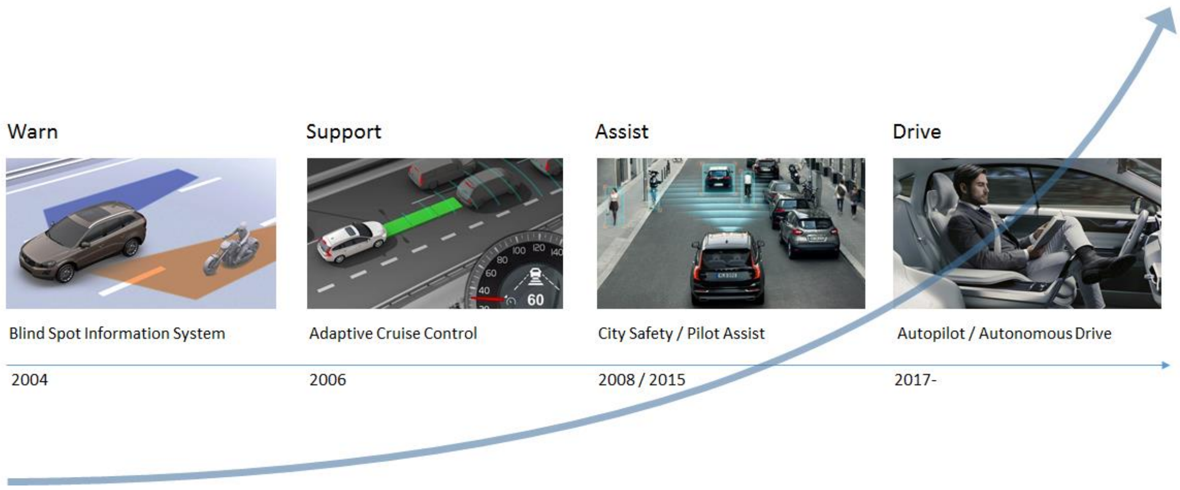

Figure 1: Increasing test scope driven by vehicle automation

There is no single silver bullet test approach for development, training and verification of self-driving systems. Instead a range of test strategies must be applied. Traffic situations captured by sensors from in-real-life driving is a valuable source for statistically based knowledge and testing. To increase coverage and find unlikely but critical situations we can complement with exploratory methods generating scenarios more or less randomly. Declarative, formally described scenarios will also be needed for reducing degrees of freedom and pinpointing purpose of tests, for example in basic function and regression testing to ensure expected system behavior and making sure previously solved problems do not reappear. Future certification and assessment programs will probably be based on this latter category of deterministic, repeatable and comparable scenarios.

Regardless of the nature of traffic scenarios, or how they are created and used, the assumption behind this project is that it should be possible to describe them using a common standardized data format.

The trend is not to use one tool for all testing. Instead we see varying needs for different purposes, domains, e.g. UX (user experience) and Active Safety, as well as different integration levels, from unit test to complete electrical integration. OEMs also depend on external test facilities like VTI SIM4 and AstaZero proving ground. The need to run the same test in different environments motivates the development of standards that facilitate interaction and exchange of knowledge, between the various players involved in the development or testing chain.

When it comes to testing of road vehicles, creating virtual road networks, 3D models, and traffic scenarios is usually considered a bottleneck. Current workflows involve tedious manual work and usability of the outcome is typically limited to a single tool. Lack of standardized formats prevents test cases and scenarios to be shared between test environments and organizations.

So the background for this project is, simply put, the need for harmonization of test descriptions, including traffic scenarios and environment models. Common formats for scenarios and 3D models are called for and they should span over different purposes, tools and test environments.

4. Purpose, research questions and method

The main purpose of this work is to contribute to standardization of traffic scenario descriptions. The project will contribute by evaluating, influencing and boosting OpenSCENARIO, the single known initiative to establish a standardized description format for driving scenarios. In particular, the project addresses the question of suitability of OpenSCENARIO for all relevant use cases and test facilities.

To investigate this, the applied method consists of using OpenSCENARIO for a representative subset of key scenarios in a relevant selection of test environments. This implies 1. Describe the key scenarios in the OpenSCENARIO format, 2. Implement software that can read and execute the scenarios (scenario engine) and finally 3, Adapt the test environments to OpenSCENARIO by integrating the scenario engine. To control and prove successful harmonization, the set of key scenarios will be demonstrated in each and one of the test environments. In this way OpenSCENARIO is evaluated not only on a theoretical level but also in a practical context and in technical detail.

Most of all, this project aims to evaluate the suitability of OpenSCENARIO format. To avoid any biases introduced by the features and semantics offered in the standard, a set of relevant test scenarios was devised and described in a traditional way: In plain English language and in plain text format. From there, these descriptions were translated into the OpenSCENARIO format. Suitability is then judged based on how well OpenSCENARIO can represent the scenarios.

5. Objective

Project goal

Driving support systems, ultimately self-driving vehicles, depend on highly complex software. To stay as safe as possible the systems must be improved and evaluated continuously. The majority of verification activities needs to be performed in virtual environments for test coverage and evaluation of tricky or dangerous situations. Establishing a toolchain that efficiently can provide road environments and scenarios for virtual testing will generate key competence and gains which will push Volvo Cars, Volvo Trucks and partners into a better position for the competition of self-driving vehicle development, and vehicle development in general. Furthermore, in the long term, standardization is also a first step towards scenario based certifications within NHTSA and EuroNCAP, on physical test tracks and in simulated environments.

The collaboration model is necessary to ensure compatibility between test tools and facilities. The partner lineup covers important aspects: VTI represents driver-in-the-loop simulators, SP/RISE and AstaZero represent physical tests, the OEMs (VCC and VGTT) represent developers of active driving support functions and autonomous vehicles. Viktoria/RISE provides the link to a platform for 3rd party innovation. HiQ provides competence and experience of developing simulator software for both aerospace and automotive use cases.

With the purpose of ensuring safe realization of all customer functions and driver interaction the EMK goal, within verification and validation, is to reach a higher simulation based verification level. This is achieved by placing the complete vehicle in the context of infrastructure and traffic situations, which matches this project’s goal in terms of establishing a generic and standardized way of simulating traffic scenarios. This will facilitate increased research and innovation within autonomous driving while bolstering the competitiveness of Swedish automotive industry; a high-level goal of the FFI program.

In summary, the overall goal of this project is to facilitate faster verification, including transfer of comparable test cases between test sites. More specifically, the project aims to contribute to the ecosystem with a common scenario format and related tools which is an important step towards the vision of sharing scenarios from various sources for test environments with various purposes.

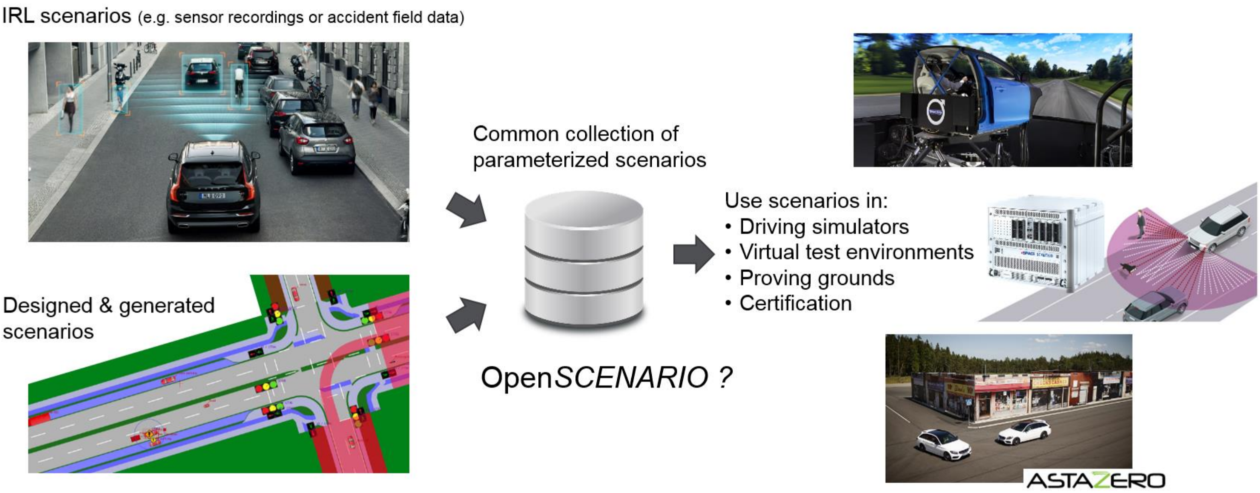

Figure 2: Vision of sharing scenarios

Project scope

The figure below shows a schematic overview of the complete tool chain related to traffic simulation including traffic scenarios, road modeling and visualization of the environment. The presented scope is tailored in accordance to the project goal of having the same scenarios run across different facilities and their diverse software tools.

The initial plan was to exclude a scenario editor from the scope. A few key scenarios would anyway be feasible to write in the OpenSCENARIO format using a basic text editor, much less effort compared to the estimated effort of developing an editor. Nevertheless, in the final phase of the project an editor could be implemented with limited effort by making use of the implemented scenario engine.

Initially it was part of the scope to develop method for converting sensor data to scenario description. After project start this work package was down-prioritized by the initial stakeholders. And since there were no dependencies to other work packages it was agreed to remove it from the project scope at a later stage.

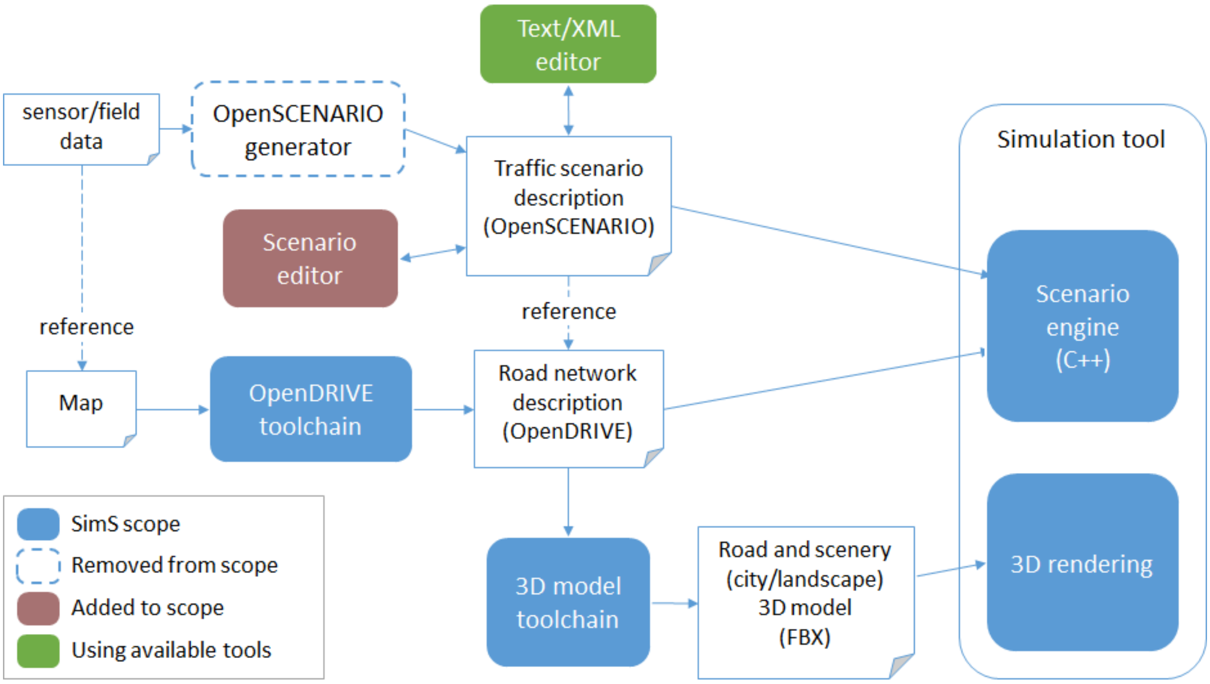

Figure 3: Tool chain overview and project scope

Part of project scope is also the establishment of an efficient tool chain for constructing road network models for use by the scenarios. This toolchain should cover all the steps in the process from map data to generation of road descriptions and even creation of 3D models for visualization.

Finally, the scope of the project further extends into the development of a software component, the scenario engine which will parse and execute scenario descriptions. This component is to be integrated into selected test environments, in effect adding support for OpenSCENARIO. The ambition is to release this software as open source with a permissive license to stimulate the use and spreading of the OpenSCENARIO format.

Test environments

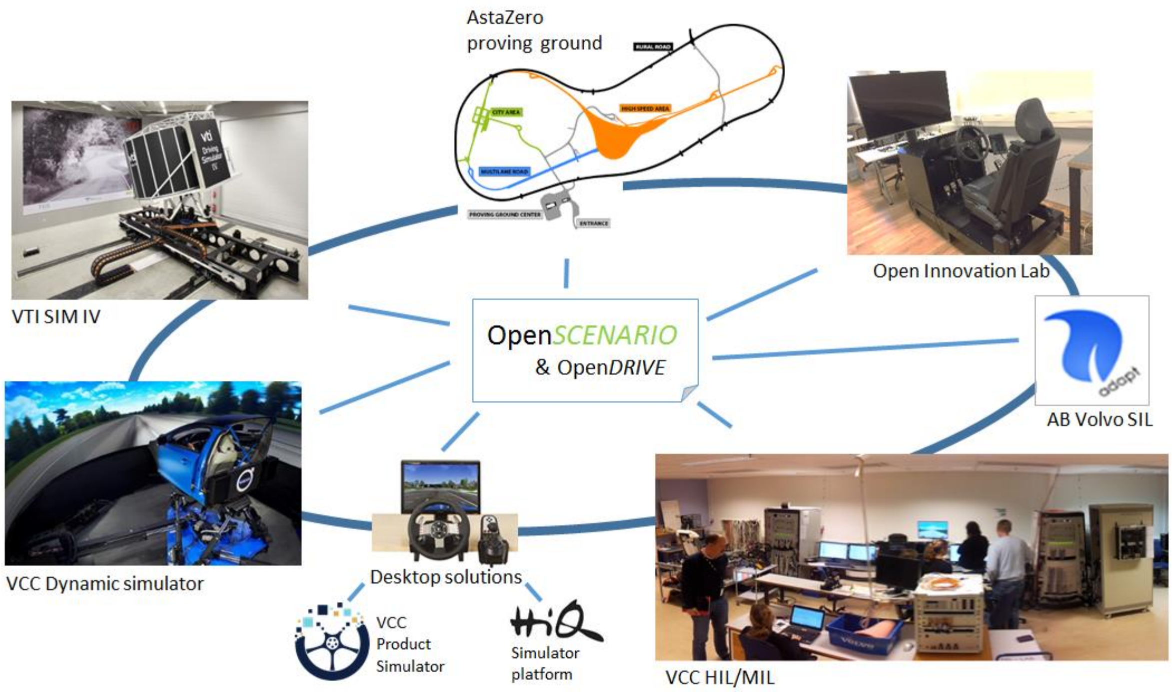

Figure 4: Selection of test environments appointed by the project to be used for proof of concept

The mix of test environments spans a wide range of technical platforms and use cases. Open Innovation Lab is an open simulation platform developed in another FFI project with the same name, aiming to involve 3rd party innovation of vehicle functions and transportation solutions. The ADAPT simulator from the FFI project HeavyRoad, now managed by the Adapt consortium[1] with Volvo Trucks and Volvo Cars as founding members, is a lightweight Windows based solution for integrating and testing components on all levels, up to the complete vehicle.
The Volvo Cars Hardware-In-the-Loop (HIL) and Model-In-the-Loop (MIL) simulators represent a collection of Simulink based test environments, some of them using the commercial tool VIRES[2] Virtual Test Drive (VDT) as environment simulator. VIRES will introduce OpenSCENARIO support during the project time span - which makes VTD highly interesting for OpenSCENARIO compatibility benchmarking. Volvo Cars' setup also includes a MIL/SIL hybrid environment named SPAS (Simulator Platform for Active Safety) whose environment simulation is not based on VIRES VTD. Volvo Cars Product Simulator is based on the Unity[3] game engine. HiQ simulator platform is a generic simulator software platform that will be extended to support OpenSCENARIO for automotive applications.
Volvo Cars Dynamic Simulator and VTI SIM 4 are both advanced full immersive motion platform based simulators. VCC Dynamic Simulator is mostly used for vehicle dynamics development and active safety research. In this simulator, VIRES VTD is used for environment simulation in scenario based studies. VTI SIM4 is mostly used for HMI and human perception related studies and is integrated with in-house developed software for simulation and simulation management.
Finally, AstaZero[4] is a physical proving ground. With focus on active safety and advanced centralized traffic control using robot road users of various kinds. It makes an interesting scenario format stakeholder with potentially different requirements when compared with other, more software based, test environments.

---

[1] http://github.com/orgs/VolvoGroup/teams/adapt/ (access required)
[2] https://vires.com/
[3] https://unity3d.com/
[4] http://www.astazero.com/

6. Results and deliverables

This chapter described the project results, how they relate to the original plans introduced in the project application and their coupling to the overall objectives of the FFI program.

Key scenarios

The set of strategic key scenarios should be relevant for the potential users. These scenarios need to cover features which are considered essential with respect to road models and traffic situations. In addition, they should be comparable to existing test conditions of interest, for the industry and research worlds. As such, these scenarios will indirectly shape the scope of the work in the project, in terms of the detail built into the execution engines, and how deep the OpenSCENARIO standard is scrutinized.

From an initial list of fourteen discussed candidates the following three key scenarios were finally chosen:

1. Cut-in
Two lane highway. Ego traveling in right lane. Overtaking vehicle cuts in and brakes to a stop. Step by step:

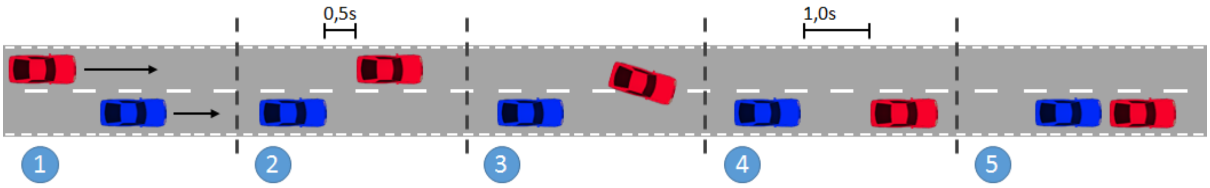

Figure 5: Cut-in

1. Ego (blue) car being overtaken by red car (red car speed = 1.2 x Ego speed)
2. When time-headway reaches 0.5 seconds, trig lane change
3. Red car changes lane, cutting in front of Ego car
4. When time-headway reaches 1.0 seconds, trig brake
5. Red car brakes to a stop

This scenario covers several useful mechanisms. It should be possible to trigger actions based on conditions involving relations between cars, e.g. time-to-collision, headway-time or simply relative distance. In this case the time-headway is used. Next, it should be possible to perform multiple actions in parallel. For this specific scenario, the brake maneuver may start before the lane change action is finished. Positioning should be relative to road, so the same scenario should work on any given road, independent of its geometry, i.e. straight or curved, flat or hilly.

2. Left Turn Across Path - Opposite Direction (LTAP-OD)
LTAP-OD commonly refers to a scenario where the Ego vehicle is about to turn left when a vehicle from opposite direction crosses its path. To challenge the scenario engine we changed roles so that the scenario controlled car makes the left turn. The scenario is defined as follows:

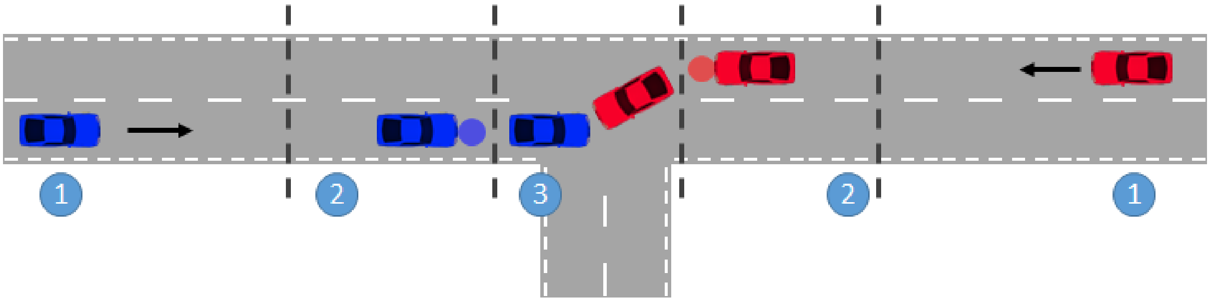

Figure 6: LTAP-OD

1. Ego (blue) and red car approaching the intersection from opposite directions
2. The red car arrives at specified destination (red point) exactly whenever Ego arrives at another specified destination (blue point).
3. Ego continues straight through the intersection, red turns left.

The key mechanism in this scenario is synchronization of vehicles. The critical situation in the intersection should always arise, regardless of the Ego vehicle speed envelope. To support use cases where the Ego vehicle is controlled externally, e.g. by a human-in-the loop or an external test framework, the red car must adapt its speed continuously in order fulfil the synchronization task. Additionally, this scenario involves routing through junctions. Hence it should be possible to specify vehicle routes on top of the underlying road network.

3. Highway-merge

In this scenario, the Ego vehicle merges into a two lane highway, from a separate road, via an onramp. In the basic version of this scenario the highway vehicles are separated by constant time gaps (time-headway), making it easy for the Ego car to find a gap and merge. The scenario evolves in the following fashion:

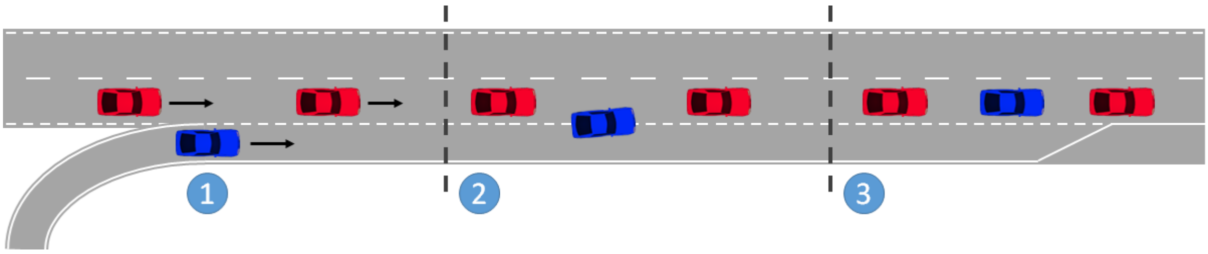

Figure 7: Highway-merge, basic version

1. Ego vehicle enters the onramp from separate road
2. Ego finds a gap and starts the lane change
3. The merge is complete

In the advanced version the scenario vehicles are capable of interacting with Ego vehicle by regulating the distance to the vehicle ahead. Whenever Ego starts a lane change, the gap is immediately being closed by the vehicle behind. Step by step:

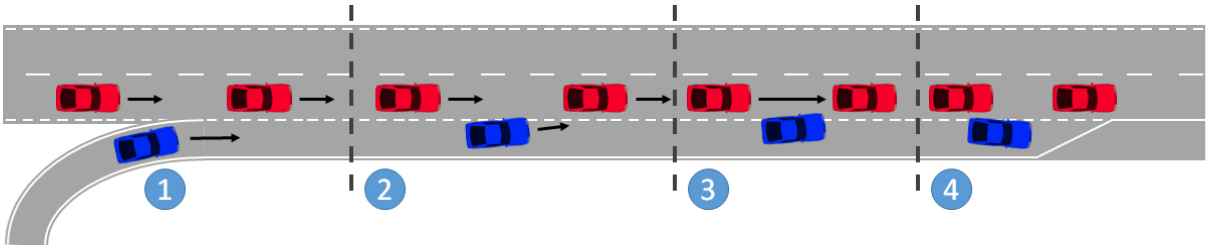

Figure 8: Highway-merge, advanced version

1. Ego vehicle enters the onramp from separate road
2. Ego finds a gap and starts lane change
3. The highway car behind increases speed, closing the gap
4. The merge should be aborted to avoid contact

This scenario adds another level of interaction where the target vehicles respond to Ego lateral and longitudinal displacement. It also involves road modelling challenges since onramps typically are not represented in low-fidelity map data[5], and hence needs to be designed in the modeling process. The onramp also exercises the OpenDRIVE feature of lane sections, allowing flexible number of lanes within the same road segment.

All three scenarios should work for both internally and externally controlled Ego vehicles. Internal meaning that vehicle is controlled via the scenario, hence the maneuvers need to be fully defined within the scenario description. External control means that the vehicle is controlled in any other way, out of scenario control, for example by a human in the loop or a separate test framework.

All key scenario descriptions are available as part of a public file package[6].

Formal scenario description

During the project planning stages and under the project span, the only widespread open data format for traffic scenarios has been OpenSCENARIO[7]. Hence it was the chosen format for traffic scenario representation. The format was initiated 2014 as a collaboration project by a group of German tool providers and OEMs. In 2016 a draft was finished and the project opened up for international partners. Volvo Cars and VTI have since joined the project. Initially the project was led by simulation tool provider VIRES. In November 2018 the project was formally transferred to ASAM[8], where the standardization process is now ongoing.

The foundation for traffic simulation is the road network. OpenSCENARIO relies on OpenDRIVE[9] for this part which covers road geometry, lanes, junctions, road markings, road signs, and other static road related information. OpenDRIVE could be regarded as the static component of a scenario, the stage where the action will be played. OpenDRIVE can optionally include OpenCRG data describing the road surface in more detail. OpenSCENARIO adds the dynamic objects and events thus defining the actions of all elements, basically a storyboard.

[5] Such as OpenStreetMap and HERE. Emerging HD map formats will contain this kind of information.
[6] Demo application and scenarios available at https://sites.google.com/view/simulationscenarios
[7] http://www.openscenario.org/
[8] https://www.asam.net/
[9] http://www.opendrive.org/

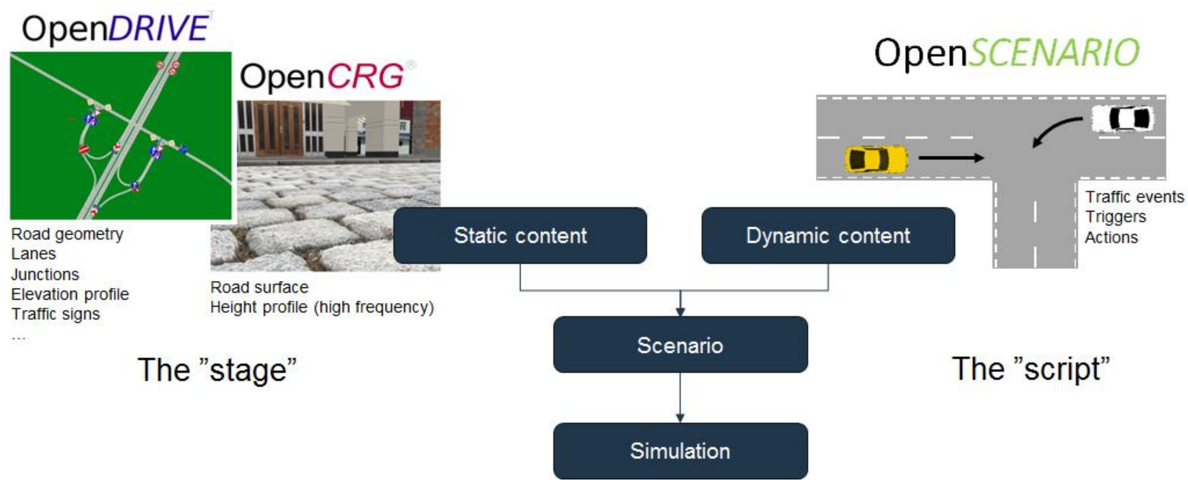

Figure 9: The OpenDRIVE, OpenCRG and OpenSCENARIO formats integrates into a complete scenario

Both OpenDRIVE and OpenSCENARIO are XML-based formats. XML is a text format, with the advantage of being readable for humans in plain text viewers and editors. XML is well spread and supported by most toolkits and libraries. So, as a programmer, parsing any XML file in Python, C++ or C# is very simple. XML is based on a hierarchical structure of elements and associated attributes. The actual supported elements and attributes of the formats are defined in a schema file. In effect this is a low level specification of the format which can be used for machine and user validation of any XML-file to check how it complies with the format. Additionally, the schema file can also be read by users as a reference documentation of the format under consideration.

A downside of being a text format is size since text based formats are typically much bigger than binary ones. This can often be mitigated by compressing files for storage. Another disadvantage of XML formats in general, and OpenSCENARIO specifically, is that the element structure adds significant overhead to the actual content. So for example, a simple cut-in scenario takes more than 100 lines to describe in this format, while expectations might be a few lines corresponding to the number of objects and actual maneuvers.

XML formats are not primarily meant to be edited by users directly. Typically an XML file is created or edited from an application providing a user friendly representation of the information. Still, it is possible to open any XML file in a text editor, which makes it easy to do at least minor modifications. Writing XML files manually from scratch is of course also possible, effort depending on extent and complexity of the content. Anyway, it can be a good way to get familiar with a specific XML format.

OpenSCENARIO is an open format, which means that its documentation is publicly available. Furthermore, specification and schema file will be distributed free of charge[10], even after the transfer from VIRES to ASAM for standardization. As of the time of writing this report, March 2019, there are no public specification documents available, yet. However, the schema files are already accessible for the current version 0.91[11]. ASAM plan to release a fully documented version 1.0 by December 2019.

OpenSCENARIO basics

The top level elements of OpenSCENARIO are:

- Road network: This element basically refers to a separate OpenDRIVE file.
- Entities: List of scenario participants. Can be of various types, e.g. car, truck, tram, pedestrian.
- Storyboard: The actual events, actions and triggering conditions.

[10] https://www.asam.net/standards/detail/openscenario/
[11] http://www.openscenario.org/download.html

Events consist of one or several actions. An action is either private to a specific object, for example speed or lane change, or global, like weather conditions or traffic light signal states.

Positions can be specified in different ways, either in “world” coordinates (cartesian x, y, z, head, pitch, roll) or road lane coordinates (distance along the road, what lane and the lateral offset within the lane). A position can be either absolute or relative to any other object.

Actions are started, and in some cases terminated, by condition fulfilment. There are three types of conditions:

- ByEntity: Conditions that make use of entity parameters, e.g. position, time headway etc.
- ByState: Conditions that refer to non-player referred states, e.g. termination of another action
- ByValue: Conditions referring values, e.g. simulation time

Conditions can be combined in hierarchical levels and also in groups for logical AND/OR checking.

Fidelity of the maneuvers specified in actions is controlled by their attributes. For example, a lane change maneuver can be defined as in any of the following ways: Step (immediate jump to target lane), Linear, Sinusoidal (S-shaped) or any cubic curve (according to user specified coefficients). Distance can be measured either along the road or a straight line between objects. Furthermore distance can be either between center points, or actual free space between objects taking their bounding geometry into account.

Any value can be defined as a user parameter to be reused within the scenario file.
The storyboard is hierarchically subdivided in Acts, Maneuvers, Events and Actions.

The structure of a typical scenario is illustrated in the diagram below, in this case a lane change action. “Vehicle1” will perform a lane change triggered when the time-to-collision from the “Ego” vehicle is 10 seconds. The lane change should follow a S-shaped trajectory, take 3 seconds and go one lane to the right (-1).

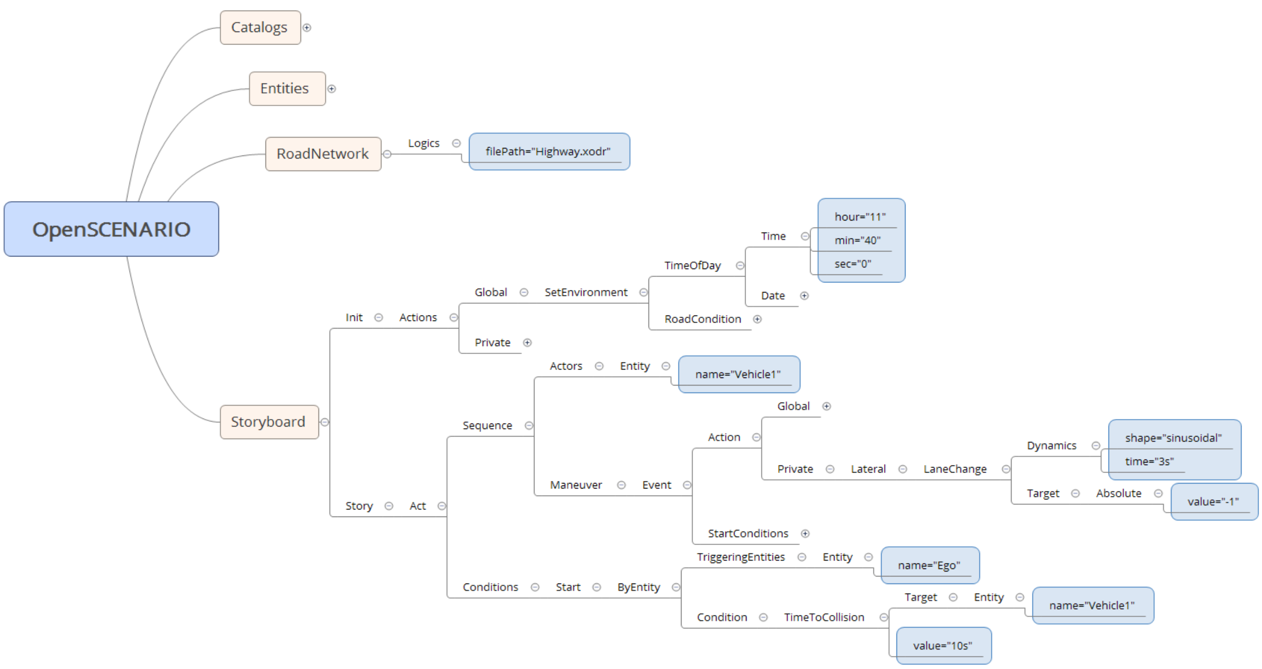

Figure 10: The structure of OpenSCENARIO indicated by a lane change scenario example

This was a brief introduction to OpenSCENARIO. For a complete listing of content, see the XML schema and any other documentation made available at above specified web sites. Full documentation of the standard format will be available along with the release of version 1.0, scheduled for December 2019. The documentation will target both scenario users and software tool developers.

For a detailed documentation of what OpenSCENARIO features have been used for the key scenario of this project please see Appendix A - Key scenario mechanisms.

Road modeling tool chain

The road network is the foundation for most scenarios, sometimes referred to as the static part of a scenario while the traffic and conditions belong to the dynamic content. While it is easy to produce single non-complex road models the complexity escalate when junctions, on-ramps and similar features are needed. As the efforts to produce these assets increases it makes sense to strive for reusability and sharing. Common formats address these needs.

One project goal is to establish a tool chain for efficient production of road models, in a format useable by the project selection of test environments. The resulting models should be easy and free to use from both license and cost point of views. However, the actual creation of models must not be free of cost, thus commercial tools have been considered as potential elements of the tool chain.

Since maps are increasingly involved in active safety functionality, Autonomous Drive in particular, a main requirement on the road modeling tool chain is to support map data input. The expected output of the toolchain consists of both a logical description and a 3D visual model of the road network.

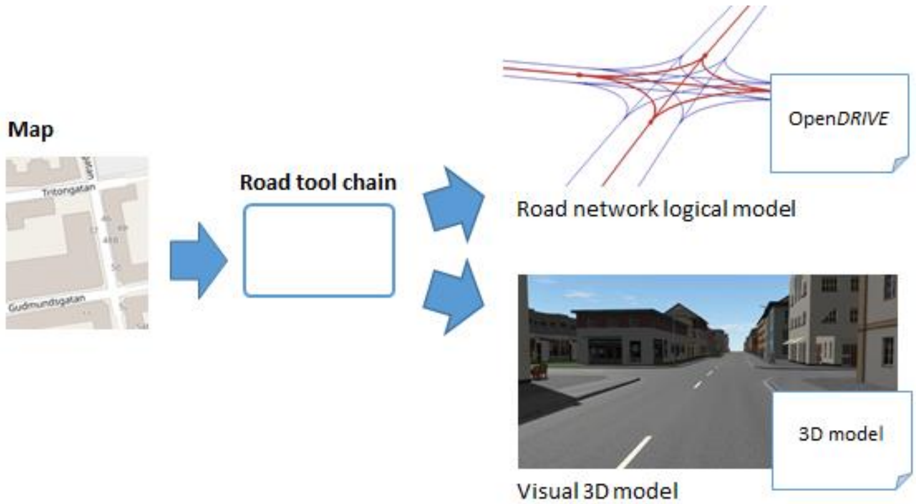

Figure 11: Road tool chain overview

The logical description of the road network should be in the OpenDRIVE format. Firstly because it is the most established road format for simulation applications, secondly because as of today the OpenSCENARIO format provides seamless integration with OpenDRIVE.

While selection of the road logics format was easy, the selection of the 3D data format was not trivial. A separate work-package was defined to investigate alternatives and find a suitable common format or tool to translate between relevant formats. The goal was to enable visualization of the road and its surroundings in all target test environments using the same 3D model file.

Tool survey

The first task was to find and evaluate existing potentially useful tools which resulted in the following list:

Tool	map input	OpenDRIVE output	3D model output
SUMO netconvert	OSM	Yes	No
HLRS OddLOT	OSM	Yes	No
Trian3DBuilder	HERE/OSM	Yes	Yes
VIRES Road Designer (ROD)	OSM	Yes	Yes
ESRI CityEngine	OSM/ArcGIS[12]	No	Yes
CarMaker / ADASRP	HERE	No (not yet)	Yes
EasyRoads3D	OSM (limited [13])	No	Yes
Unity Road Architect	No	No	No

Figure 12: Table listing evaluated tools

The tools considered in the evaluation process have a vast set of features that could not reasonably be assessed by the project. This motivated a focused evaluation which should not be mistaken with a thorough and general review. Our conclusions should therefore be seen from the project's perspective and its limited scope. What follows is a short summary of the project findings.

Sumo is a free tool, but does not fit our needs since individual lanes ended up as separate roads, hence the road network structure is lost. No further efforts were spent on this tool. Sumo supports OSM and OpenDRIVE.

In addition to generating highly detailed buildings, CityEngine offers a high-level rule based modeling technology that the user can control by modifying the underlying scripts. For example, a single lane road connecting to a highway is assumed to be an on-ramp merging via a parallel lane - but only if the join angle is within defined constraints, otherwise an intersection is created. The resulting models are too simple for our purposes and lack many road features that are expected for scenario control.

Trian3DBuilder is the most advanced of the evaluated tools. It offers a detailed user interface to edit the road network. On the other hand it lacks the high-level logics provided in CityEngine. So, given the lack of details in the tested map data sources, HERE in this case, the initial generated model is incorrect or incomplete and needs a significant amount of manual work in the editor. Trian3DBuilder will support high fidelity (HD) maps containing relevant information for generating 3D models. Using HD maps there is no need for high-level logics, just the capability to represent all features.

The two Unity based road tools Easy Roads and Road Architect are both lacking support for OpenDRIVE why their usefulness is limited from this project standpoint.

VIRES Road Designer (ROD) generates road models of visually acceptable quality. The user can start out either from scratch or from map data (OpenStreetMap). It provides a graphical road editor where adjustments can be done in a fairly straightforward manner, even though there is a learning threshold.

[12] OSM support is included in all versions, while ArcGIS requires a separate license at additional cost
[13] Intersections and junctions not supported, yet

Based on OpenSceneGraph[14] (OSG) the resulting 3D files (.osgb) can easily be converted into common formats like Alias/WaveFront .obj[15] or Autodesk Filmbox[16] .fbx.

VIRES ROD is not the perfect tool but, among the evaluated ones and within this project's context and time frame, it provided the best balance between automation and editing possibilities to produce 3D models and correlating OpenDRIVE road descriptions out of map data (OSM). It was ultimately chosen for creation of the road models (OpenDRIVE + 3D) for the key scenarios of this project.

3D file format

Since rendering technology in the majority of the target test environments are based on the OpenSceneGraph (OSG) library, we have focused on formats supported by OSG.

Filmbox .fbx is a 3D format owned and maintained by Autodesk. It is supported by OSG and Unity. An interesting feature of FBX is animation support. Even though not tested in this project, that feature could be useful in a scenario simulation context, e.g. for animated pedestrians or animals. Animation would apply only to the limbs while movement of the complete body is handled by the scenario.

Collada/DAE[17] is an ISO standard for interchange 3D model format. It is supported by many tools, including OSG. For example it was early used to populate Google Earth with 3D features, mainly buildings. Based on XML (ASCII) the file size is significantly bigger than binary formats. The upside with XML is simple reading by both humans, since it can be inspected in any text editor, and machines, since all major SDKs includes XML parser. Unfortunately, we found the OSG Collada exporter producing incorrect and unpredictable results.

Alembic[18] is an open source 3D model interchange file format hosted by Sony Pictures Imageworks, Lucasfilm. It was announced at SIGGRAPH 2011 but is not yet supported by OSG.

Wavefront .obj[19] was initially developed for Wavefront technologies products. Mostly known as an ASCII format but there is also a binary variant. The format is well documented on several online sources, see external links on Wikipedia. Its simple structure contributes to the wide and stable support by many libraries and tools.

Of the above formats, the .obj and .fbx turned out to work best between OSG, CityEngine and Unity. All translations performed during the creation of the key scenario road models met all of the project's expectations. Ultimately, the scales were tipped towards FBX given that it is a binary format and the resulting files were smaller when compared to the alternative. FBX was chosen as the preferred format used for the key scenario road models of this project.

Map data

Various map data sources have been considered, of which HERE and OpenStreetMap (OSM) have been available to the project and investigated in more detail. In summary HERE and OSM are similar with regard to features and fidelity. A general difference is that OSM, based on crowdsourcing, has more variation in quality compared to HERE. On the other hand, OSM is a widely supported map data source and free of charge, which is not the case for HERE.

---

14 http://www.openscenegraph.org/
15 https://en.wikipedia.org/wiki/Wavefront_.obj_file
16 https://www.autodesk.com/products/fbx/overview
17 https://www.khronos.org/collada/
18 http://www.alembic.io/
19 https://en.wikipedia.org/wiki/Wavefront_.obj_file

Since the preferred road modeling tool, VIRES ROD, supports only OSM it was chosen as a the map source for the key scenario models.

Neither OSM nor HERE can be used for creation of high fidelity road models. HD maps, including detailed information like lane widths, road markings and junction connectivity will be much more useful for this purpose. At the time we carried out this work, spring 2018, the HD map coverage was still limited as was the tool support of HD maps. Hence further investigations was not feasible.

The Road Modeling Process

Unfortunately no tool solves it all. To generate useful road models additional steps are needed: The map data requires processing to make sure road types are correct. As an example, without such steps a two lane road might turn up as a one lane. The road network structure typically needs some adjustments, corrections and additional features. For example adding the on-ramp and additional file sections to a highway. 3D model features like buildings, trees and similar are not a part of the road network, yet need to be added. Finally the 3D model typically need some processing, e.g. with regard to level of detail and texture filtering.
All above taken into account, with a mix of automated scripting and systematic use of the chosen road modeling editor it takes typically less than an hour to produce a model like the ones created in this project.

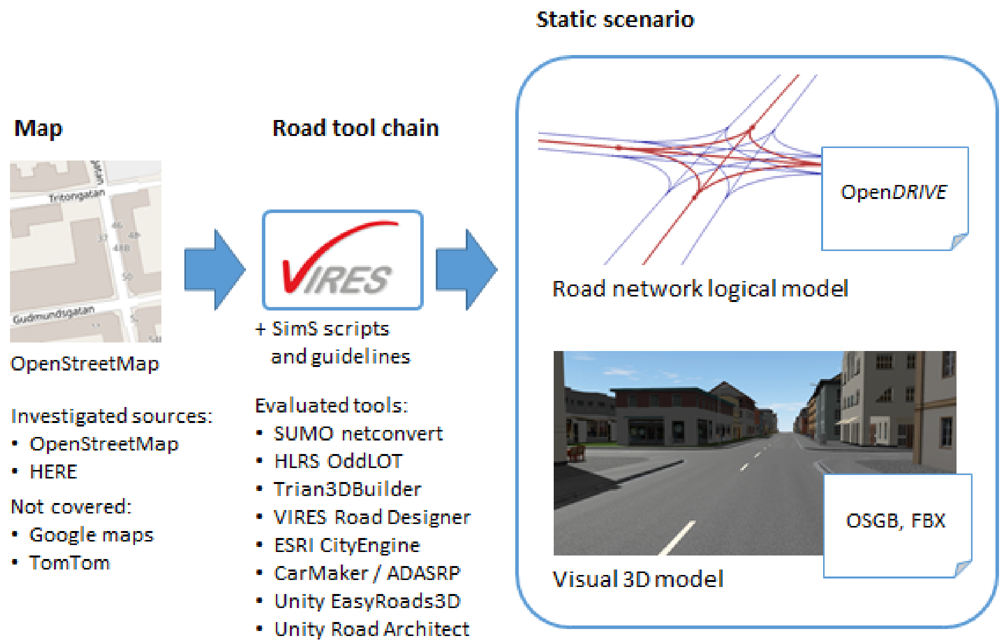

Figure 13: Resulting road modeling tool chain indicating selected formats and tools

Created models

Using the tool chain described above we have created a set of 3D environment models, one for each key-scenario. To support cut-in on different road types, one country road and one highway model were created for that scenario.

Selection of locations

For the cut-in scenario, a basic country road and a highway segment were selected, containing no junctions. For the LTAP-OD scenario, a simple four way intersection with no unnecessary elements like bike lanes or traffic lights was selected. Finally, for the highway-merge scenario, a basic on-ramp with a separate lane coming to an end was chosen. Suitable locations were identified simply by browsing Google maps with satellite mode and some street view touchdowns. The final locations are depicted in the image below.

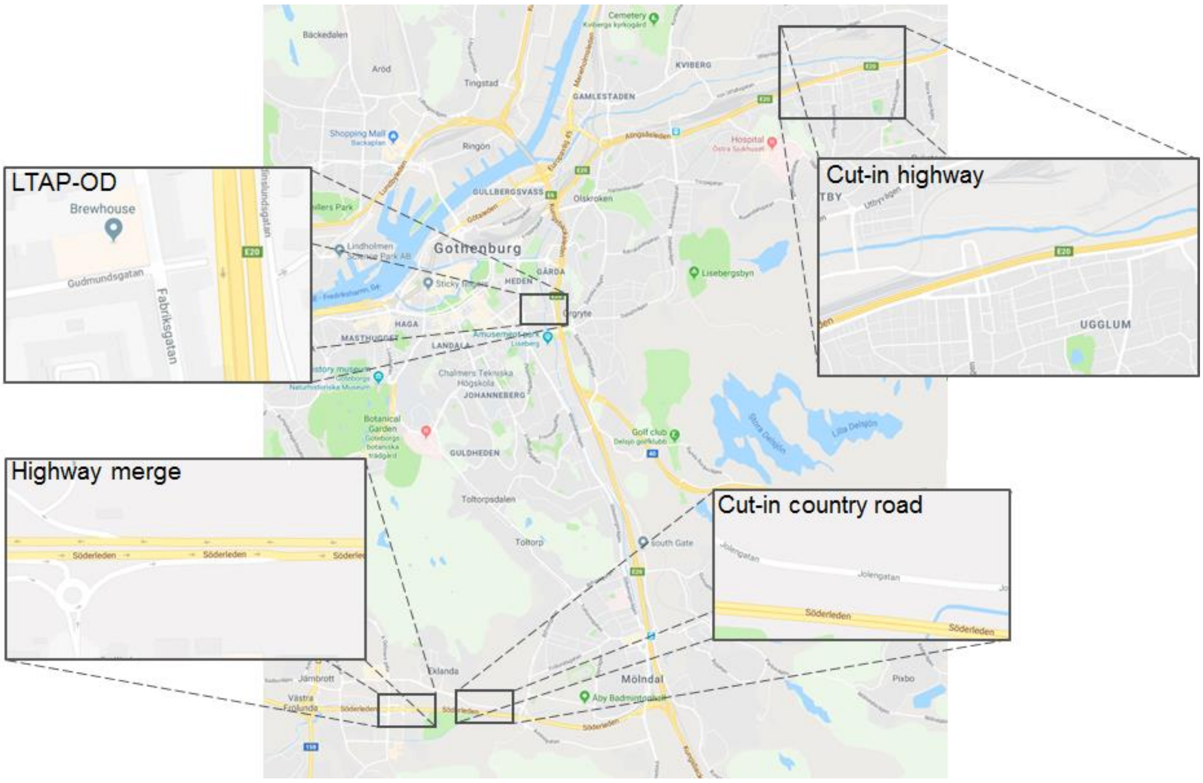

Figure 14: Key scenario locations

Resulting 3D models

Figure 15: Cut-in on highway: E20

Figure 16: Cut-in on country road: Jolengatan

Figure 17: LTAP-OD: Fabriksgatan

Figure 18: Highway-merge: Söderleden

Adding support for key scenarios in target test environments

All target test environments presented in the section “Test environments” have been adapted in various ways to support OpenSCENARIO, at least the subset required by the key scenarios. Several different solutions were developed within the project. VTI adapted existing in-house developed simulation software to support OpenSCENARIO. HiQ extended their existing simulation software platform with an environment simulator module supporting OpenSCENARIO. Volvo Cars developed a separate environment simulator package consisting of a road manager (supporting OpenDRIVE), 3D graphics rendering (based on OpenSceneGraph[20]) and a scenario engine supporting OpenSCENARIO.

The purpose and ambition of the VCC implementation is portability as it should be easy to integrate this environment simulator component into various platforms and operating systems. In the end, the component was successfully integrated into MATLAB, Simulink, Unity (C#), ADAPT framework (C++), and other native C++ applications. It works at least on Windows, MAC, Linux and Android. It can be linked as a static or shared library. The library itself can link sub-modules statically or dynamically. Statically linked the component becomes a handy 5 MByte all-inclusive monolith. The image below shows a high level diagram of the VCC environment simulator (ES mini) library and what targets it has been integrated into.

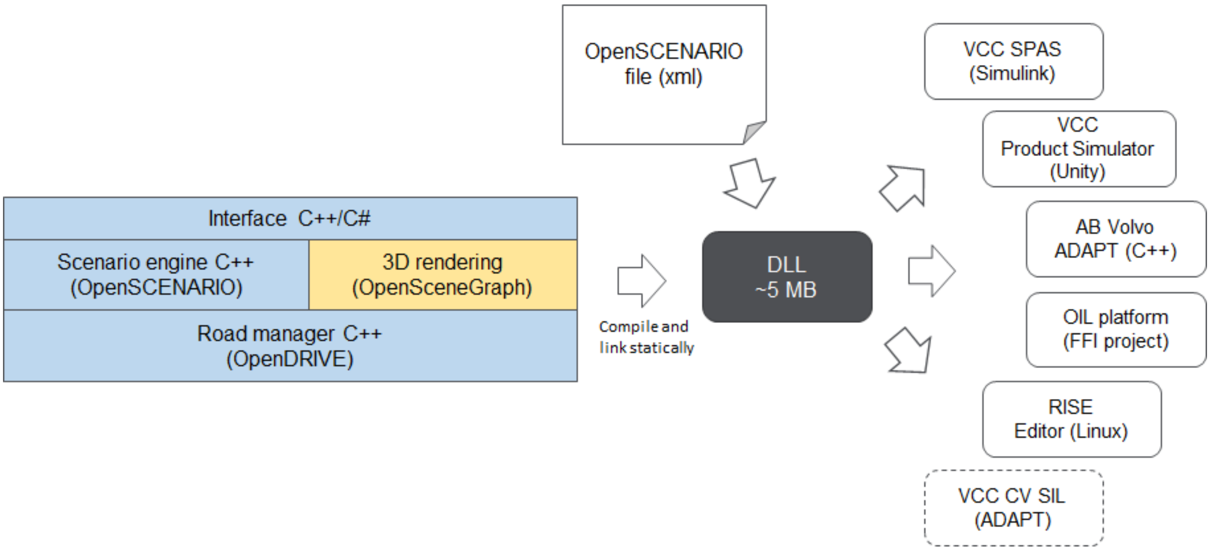

Figure 19: ES mini block diagram and target integrations

For AstaZero, road user displacement needs to be described in term of a trajectory in space and anchored in time. This was achieved by a custom tool, developed within this project, which translated valid trajectories into OpenSCENARIO format to be provided as references to the test vehicles.

The image below shows an overview of the various scenario engine implementations and interrelations.

[20] http://www.openscenegraph.org/

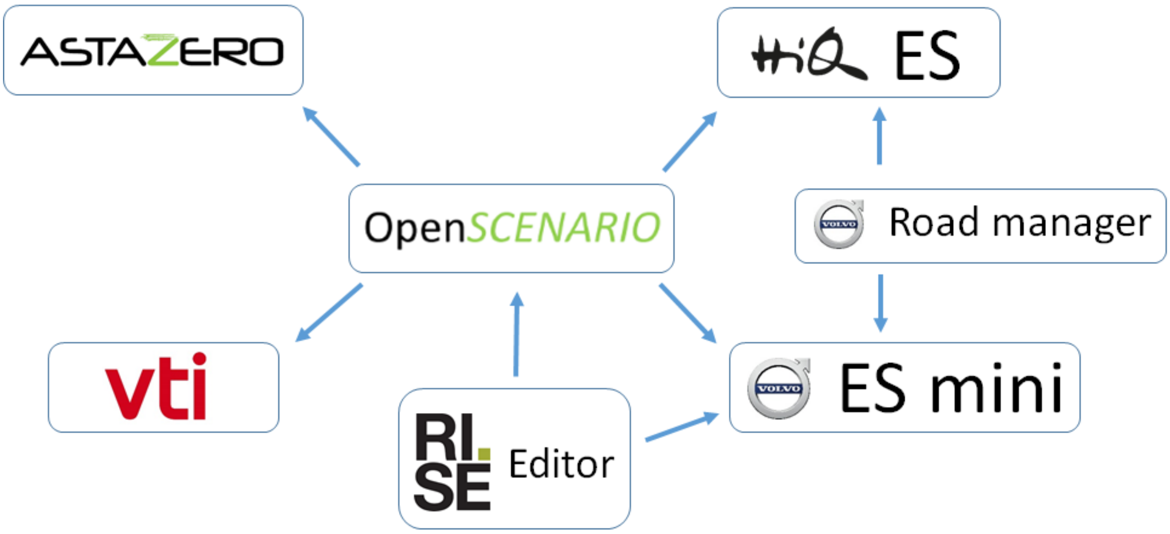

Figure 20: Overview of scenario engine implementations of this project

The scenario editor (RISE Editor) being a tool, not a test environment, relies on ES mini for road information and scenario execution.

Contribution to an open standardized format for traffic scenarios

Simulation Scenarios has contributed to the OpenSCENARIO standardization in several ways. By proposing improvements, synchronization of objects in particular, this feature can be addressed early and potentially be introduced already with the first version of the standard, 1.0. Testimony of usefulness on practical examples is valuable to create confidence and interest in the format. The implementation of scenario engines and adaptation of test environments have created a technical readiness to start use the format. Finally, releasing the source code as open source will hopefully inspire others to investigate, try out and then join the movement. The actual release is being prepared for while this document is written, but according to the plan it will be released spring 2019 on GitHub under license MPL 2.0.

Building a platform for future research involving traffic scenarios

By adapting this broad range of test environments to a single portable scenario format we have shown the potential of sharing scenarios using the OpenSCENARIO standard format. Project partners have now the opportunity to continue this platform collaboration. And by releasing selected software as open source also other partners can join the collaboration and take advantage of the results. Anyway, the platform is a good start supporting research and development of autonomous vehicles with standardized traffic scenario simulation.

Strengthening the competitiveness of Swedish automotive industry

Since vehicle systems increasingly need to be validated in the context of traffic scenarios we believe that several components and competence resulting from this project will contribute to Sweden's ability to stay competitive, even in this high profile area of autonomous driving.

By improving traffic scenario handling in strategic test environments the expected effects are:

- **Faster development of vehicles in general** by boosting the use of scenarios in automatic verification which is a central part of continuous integration.

- **Safer autonomous vehicles** by extended training and testing of the vehicle's brain in traffic situations. Being a parametric format OpenSCENARIO scenarios can easily be varied to increase coverage. Portability ensures consistency and comparability, as far as the format allows.

- **Elevated innovation.** Including traffic scenarios, in effect raising the context of simulations to traffic and transportation levels, opens up for innovations beyond single functions and subsystems.

- **Increased cooperation with third parties.** The possibility to share competence and resources will lower the threshold for cooperation in development and research, consequently opening up new cooperation opportunities.

- **Increased transparency**. A sharable and standard way to define scenarios can increase transparency in the how vehicles are developed or tested. This can also boost harmonization, certification, and validation efforts.

7. Dissemination and publications

Demonstrations of achievements in the target test environments

During January and February 2019 a range of test runs and open demonstrations were carried out. Video clips have been produced for many of them, including VCC SPAS (Simulink), VCC Product Simulator (Unity), VCC Dynamic Simulator (VIRES VTD), VTI SIM4, AstaZero proving ground, HiQ simulator, ADAPT, VCC HIL/MIL (VIRES VTD), RISE editor (Linux) and a few Windows POC applications. All planned integrations and demonstrations were completed successfully. This means that all target test environments have been aligned to support the subset of OpenSCENARIO covered by the key scenarios. A great achievement which will be elaborated upon in the conclusions section.

What follows are some photos and screenshots from selected demonstrations and applications:

Figure 21: Demonstration at VTI SIM4 - control room

Figure 22: Demonstration at VTI SIM4 - in simulator platform

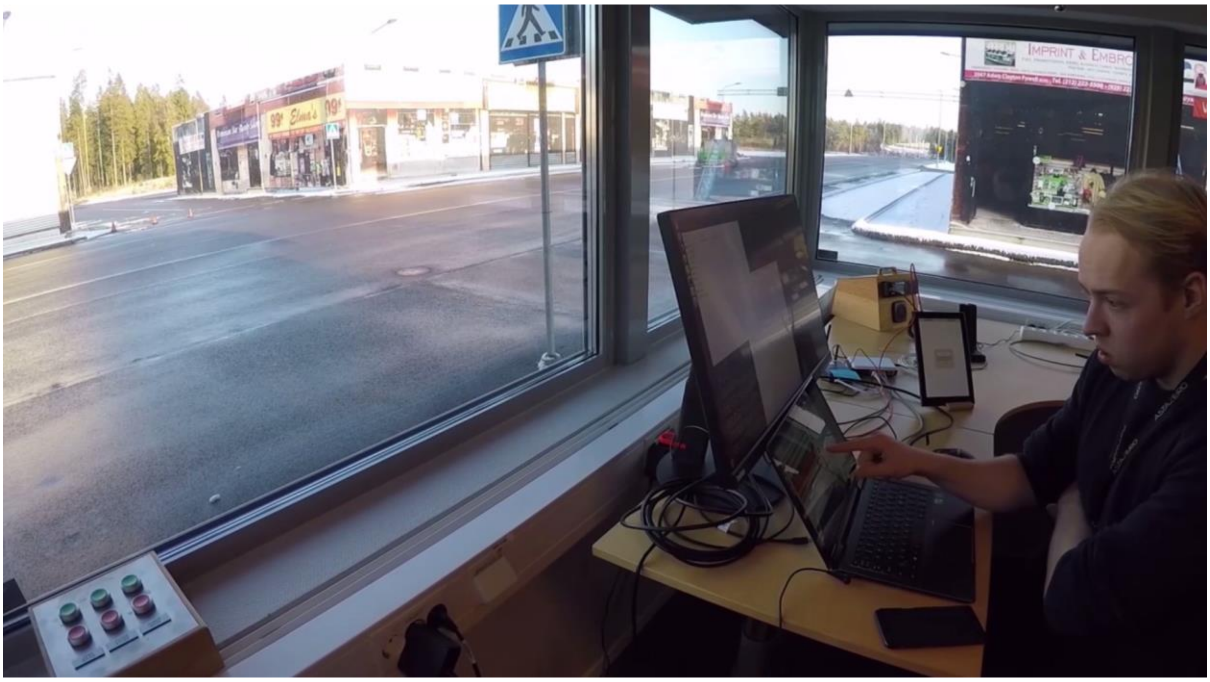

Figure 23: Demonstration at AstaZero - in control room

Figure 24: Demonstration at AstaZero - scene overview

Figure 25: VCC Dynamic Simulator demonstration

Figure 26: Unity based demo application

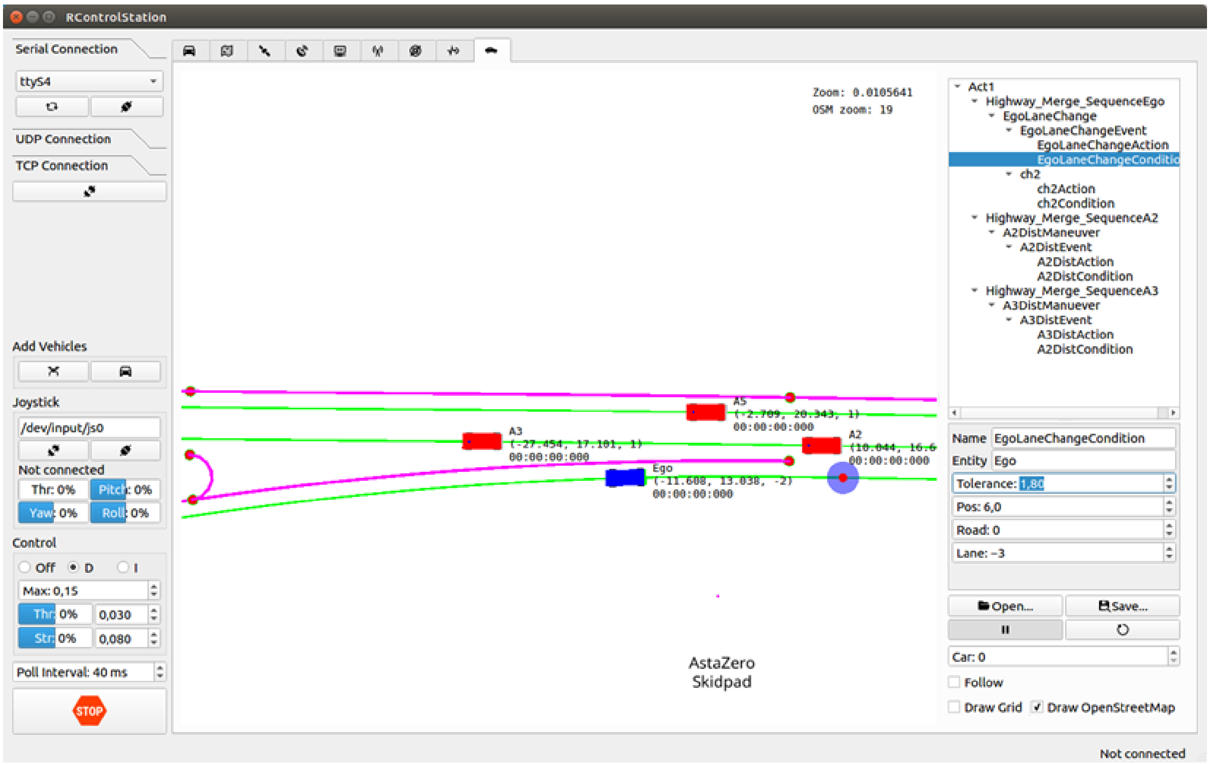

Figure 27: RISE Editor and highway-merge scenario

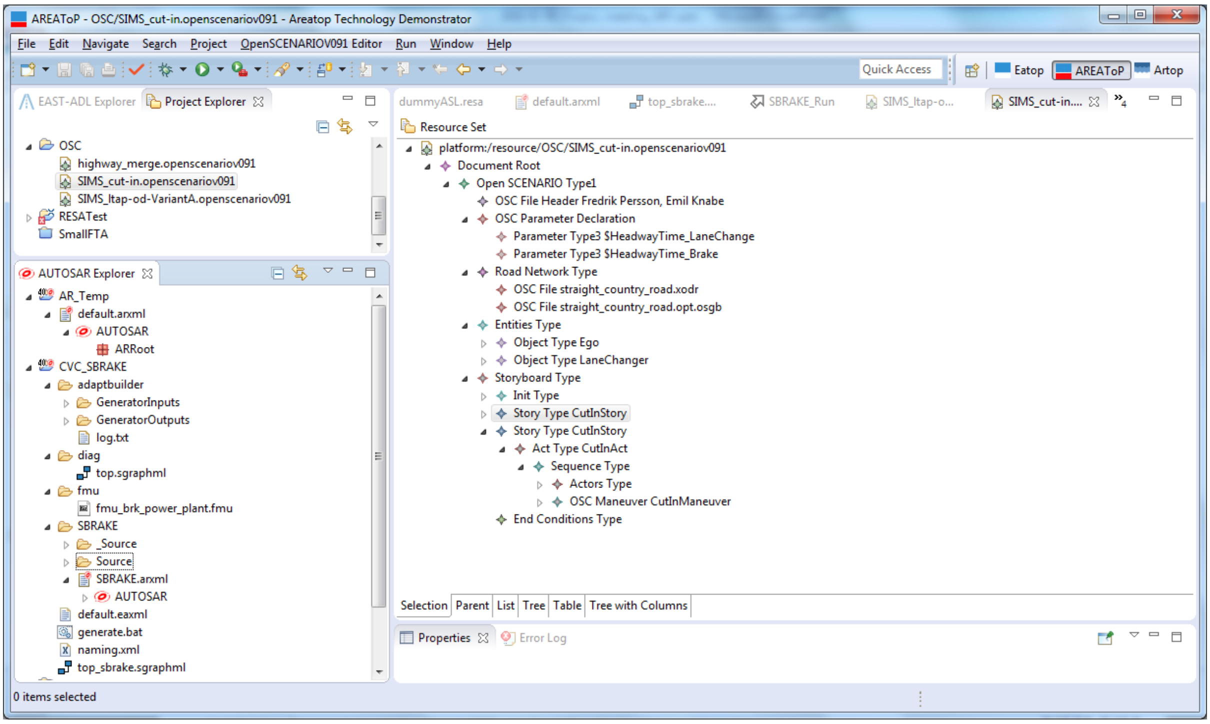

Figure 28: Eclipse OpenScenario Editor with cut-in scenario

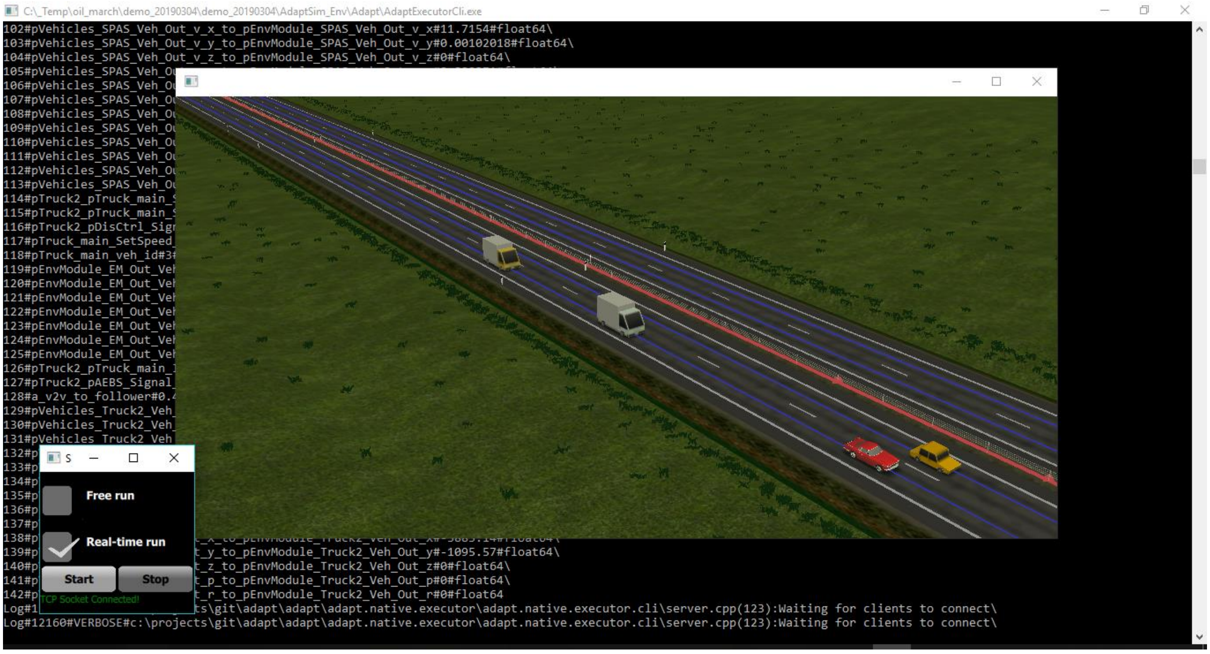

Figure 29: Adapt simulator with ES Mini, executing the cut-in scenario with the platooning setup from FFI OIL

Dissemination

How are the project results planned to be used and disseminated?	Mark with X	Comment
Increase knowledge in the field	X	Key scenarios and software source code will be released as Open Source with the permissive MPL 2.0 license. This is the ultimate action to share results and boost learning and further development.
Be passed on to other advanced technological development projects	X	We have brought a technology to the table, at proof of concept level. There is now an opportunity for Volvo Cars and other partners to take next step and try out OpenSCENARIO for real use cases, which might involve further development and customization.
Be passed on to product development projects		
Introduced on the market		
Used in investigations / regulatory / licensing / political decisions	X	The project has contributed to the establishment of the OpenSCENARIO standard. The format is frequently mentioned in the context of Autonomous Drive certification and regulation. It too early to see what impact and presence OpenSCENARIO will have in this field.

Collaborations

In addition to the OpenSCENARIO standardization project the following collaborations should be mentioned:

FFI Project CHRONOS2
CHRONOS2 has provided input and influenced the selection of key scenarios. As soon as the OpenSCENARIO descriptions of the key scenarios were created they were shared with CHRONOS2. CHRONOS2 scenarios described in OpenSCENARIO format have been shared with this project for review and validation to ensure compatibility between the projects.

FFI project Open Innovation Lab
Close collaboration with the FFI project Open Innovation Lab has resulted in 1. A common network protocol for integration of environment simulations with vehicle simulators. 2. Environment simulation implementations from Simulation Scenarios have been integrated into the Open Innovation Lab software platform. So the win-win relation is that Simulation Scenarios provides a ready-to-use environment simulation component while Open Innovation Lab provides a complete simulator platform supporting the OpenSCENARIO data format.

Publications

Master Thesis: Multi-user Driving Simulation, Jiahui Liu and Yanni Xie, Chalmers (to be published)

Additional public presence:
Presentation at Vehicle Electronics and Connected Services (VECS)[21] 2018: "Sharing Traffic Scenarios for Continuous Development Using Open Formats".
Presentation at Smart Automotive Variant.CON, November 2018, Berlin
Presentations on multiple OpenSCENARIO meetings, 2017-2018.

8. Conclusions and future research

The project has shown that OpenSCENARIO is useful for the kind of scenarios and test environments represented. The two most important aspects are 1. Coverage: Can the key scenarios be fully described by the format? 2. Implementability: Is the format easy and feasible to implement support for.

Coverage
With one exception the project identified suitable elements for all three key scenarios within the existing OpenSCENARIO version 0.91. It was a straightforward, though tedious, process to write the descriptions in a text editor. The missing feature, a way to synchronize the arrival of two vehicles, was easily added as an extension to an existing action element. The extension has been filed as a proposal to the OpenSCENARIO project. Anyway it demonstrates the flexibility of adding custom features on top of the standard.

Implementability
The fact that the project has produced three separate implementations of scenario engines supporting (a subset of) OpenSCENARIO is evidence enough that the format is easy to parse, interpret and execute. In that sense the format is well designed. Though it should be said that in current status, version 0.91, a complete specification document is lacking. The XML schema together with a brief beta user manual and a few examples offer a good base. But more detailed information is needed in order to achieve compatible implementations of the format. For version 1.0, expected December 2019,

[21] https://insightevents.se/vehicle-electronics-connected-services/

there will be a UML model and full specification document to present and clarify information at high level as well as the details.

Further, given the number of successful integrations it is obvious that harmonization of environment simulation, including 3D models and traffic scenarios, is achievable.

Criticism
OpenSCENARIO is a low level format in the sense that it specifies the motion or kinematics of road user objects on a detailed level. It also contains many pure structural elements useful not only for grouping content but also to support story control, for example whether a brake action should interrupt, follow or execute in parallel with an ongoing lane change action.
The fact that it takes about 100 lines of XML code to describe a cut-in scenario may seem ridiculous. And it would be if the format was meant for users to write and read directly. However, the format is primarily targeting simulators and related tools. In other words it is a machine language. For that purpose it is an advantage with clear unambiguous structure and details, avoiding diverging interpretations and instead making implementation straight forward. XML, as a ASCII format, is not as compressed as binary alternatives. On the other hand it allows for any user to open the file for reading and even editing in any text editor.
So instead of introducing a totally new incompatible high level language, which has been suggested, we support the idea of high level representations one-to-one mapping to the existing OpenSCENARIO format. The two most obvious use cases are: 1. Graphical editor providing an intuitive and easy to understand illustration of the scenario for reading, demonstrating and manipulating scenarios - which then are saved by the editor in the OpenSCENARIO XML format. 2. High level (domain specific language) text language, which presents and manipulates scenarios on a higher level. Via translation the scenario can be stored in OpenSCENARIO XML format. Compare with scripting languages which typically provide a limited set of relevant shortcut commands for accessing more complex lower level execution engines.
Another common early critique pointed out that OpenSCENARIO is not a real standard, just an open format potentially an ad hoc standard. However since OpenSCENARIO and OpenDRIVE were transferred from VIRES to ASAM in 2018, that critique is obsolete.

Alternatives
There are no, to our knowledge, today ongoing alternative initiatives to establish a standard for scenario descriptions for simulation context.

Conclusion
The results of this project show that OpenSCENARIO is a legitimate format in terms of usability and ease of implementation for a wide range of test environment types and use cases. However, it is far from ideal for direct editing by human users. By complementing the low level XML format with a higher level representation, preferably a standardized DSL language with one-to-one mapping to OpenSCENARIO XML, we believe that readability would be greatly increased. If a customized scenario format is needed for any reasons, we encourage to consider OpenSCENARIO compatibility to make possible translation between the formats.

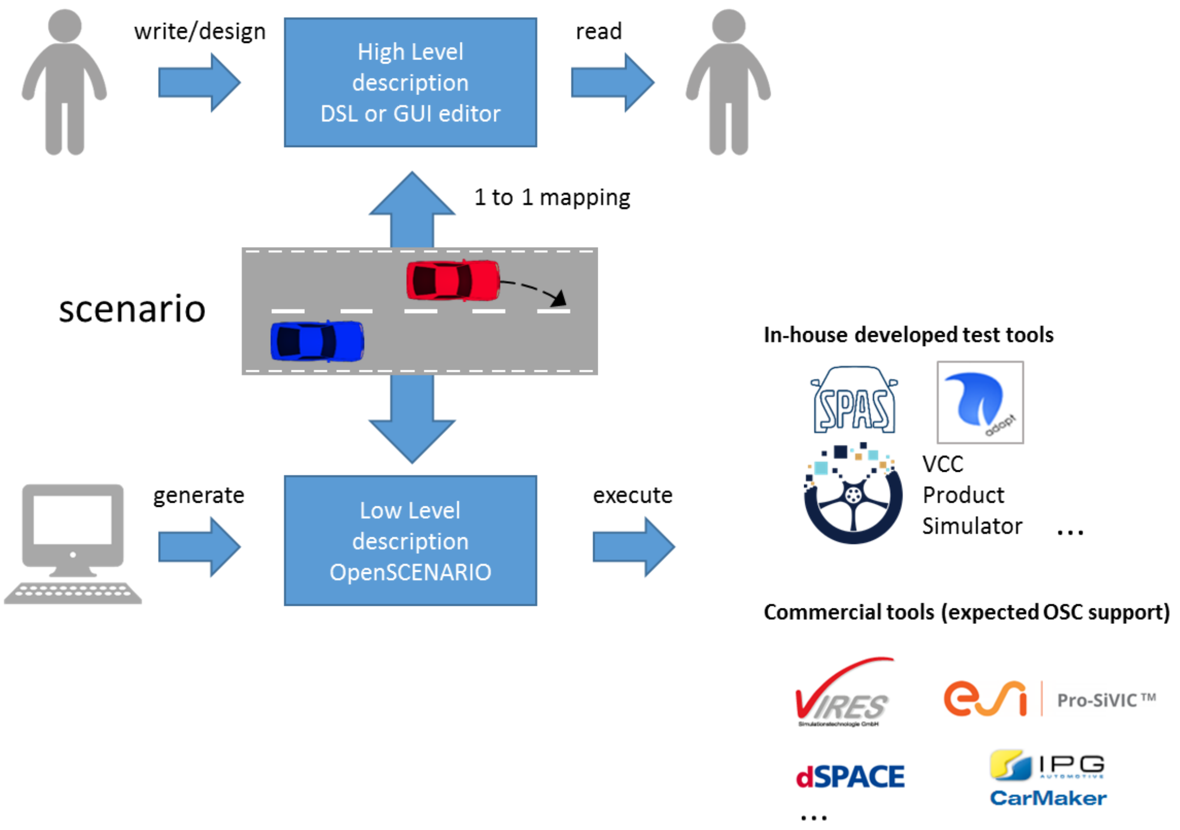

Figure 30: Updated vision taking into account the need of a high level scenario representation

9. Participating parties and contact persons

- Volvo Car Corporation, coordinator, Emil Knabe, emil.knabe@volvocars.com (project leader)
- AstaZero, Viktor Johansson, viktor.johansson@astazero.com
- HiQ, Anders Bengtsson, anders.b.bengtsson@hiq.se
- RISE (former SP), Jonny Vinter, jonny.vinter@ri.se
- RISE (former Viktoria Institute), Niklas Mellegård, niklas.mellegard@ri.se
- Volvo AB, Henrik Lönn, henrik.lonn@volvo.com
- VTI, Arne Nåbo, arne.nabo@vti.se

10. Abbreviations

ECU	Electronic Control Unit. In automotive referring to an embedded component including processing hardware and software.
HIL	Hardware In the Loop. A simulation environment where one or several physical ECU's are part of the system under test.
SIL	Software In the Loop. A simulation environment including actual production software code, cross-compiled for the simulator platform.
VTI	Swedish National Road and Transport Research Institute

Appendix A Key scenario mechanisms

As already mentioned, the actual scenario files in the OpenSCENARIO format are available as part of a project public file package available at: https://sites.google.com/view/simulationscenarios

In this section we list, on high level, the major elements used for each of the key scenarios. For sake of readability some details have been omitted and some names have been replaced.

Cut-in	OpenSCENARIO elements
Declare actors	Ego (vehicle under test) Overtaker (target vehicle which will make the cut-in and brake maneuver)
Initial position	Using road coordinates, e.g. Lane position (roadId=0, laneId=-3, offset=0, s=50) offset = lateral offset from lane center s = distance along road Ego is positioned in a lane -1 from Overtaker, and at higher s value (in front of Overtaker) Position can be world/cartesia or road coordinates, and either absolute or relative other objects
Initial speed Ego	Using private action Longitudinal/Speed dynamics = “step” speed = 30 (m/s) Dynamics can be “step”, “linear”, “sinusoidal” or “cubic” Speed value can be either absolute or relative another vehicle
Initial speed Overtaker	Using private action Longitudinal/Speed dynamics = “step” target value = relative(object=”Ego”, value=1.2) valueType = “factor” continuous = “true” Overtaker vehicle is set to a higher speed than Ego, in effect starting the overtaking maneuver Continuous = “true” will keep speed updated to 1.2 x Ego speed
Action lane change for Overtaker	Using private action Lateral/LaneChange dynamics = "sinusoidal" time = ”3” target = relative(object=”Ego”, value=0) Target lane is 0 relative Ego, meaning it will change to the same lane as Ego. Duration of lane change = 3 seconds
Condition lane change	Using condition ByEntity/TimeHeadway TriggeringEntity = “Ego” Entity = “Overtaker” value = “0.5” rule = “greater_than” edge=”rising” Headway time measured from Ego to Overtaker edge rising = trig when condition goes from False to True In summary: Condition will trig when Overtaker reaches a time gap of 0,5 seconds in front of Ego
Action brake OverTaker	Using private action Longitudinal/Speed dynamics=”linear” rate=-4 target value = 0 Change speed linearly at rate (acceleration/deceleration) -4 m/s2, to 0 m/s

Condition brake	Using condition ByEntity/TimeHeadway TriggeringEntity = “Ego” Entity = “Overtaker” value = “1.0” rule = “greater_than” edge=”rising” Condition will trig when Overtaker reaches a time gap of 1.0 seconds in front of Ego

LTAP-OD	OpenSCENARIO elements and attributes
Declare actors	Ego (vehicle under test) Target (target vehicle which will make the left turn in the intersection)
Assign routes	Using private action FollowRoute catalogName = “RouteCatalog” entryName = “StraightRoute” for Ego, “LeftTurnRoute” for Target Routes are defined in separate catalog file. A route is basically a set of waypoints (road positions) through the road network.
Synchronize target vehicle with Ego at intersection	Using private action Meeting* Sync position = Lane(roadId="2" laneId="-1" s="300”) Ego sync position = Lane (roadId="0" laneId="1" s="8") offset = 0 Target vehicle will arrive at its sync position whenever Ego arrives at its sync position. Sync positions are located on corresponding incoming road and close to the intersection Offset parameter can adjust the arrival time of Target compared to Ego
Condition for the sync action	Using condition ByState/AtStart type=”act” name="Act1" Action will trig whenever act “Act1” is started. Acts are the OpenSCENARIO storyboard top level element for grouping actions. A scenario file can contain multiple named acts. Type can be “act”, “scene”, “maneuver”, “event” or “action”.
Target vehicle speed after synchronization	Using private Longitudinal/Speed dynamics shape = ”linear” rate = ”1” target value = ”7” Change speed to 7 m/s, at rate 1 m/s2 .
Condition for Target speed action after sync	Using condition ByState/AfterTermination type=”action” name="Sync” Trig whenever the “Sync” action above has completed.

* No existing OpenSCENARIO element can solve the task to synchronize two vehicles as needed in this scenario. We have has developed and filed[22] a proposal to extend the Meeting action element to support synchronization with other objects. Meanwhile this extended version has been used within the project.

[22] See OpenSCENARIO improvement ticket #6714: https://redmine.vires.com/issues/6714 (site requires registration) or contact the author of this report.

Highway-merge basic	OpenSCENARIO elements and attributes
Declare actors	Ego (vehicle under test) TargetA (First vehicle on highway) TargetB (Second vehicle on highway)
Initial position	Using road positions Ego vehicle on road ID 1 and target vehicles on road ID 2 (highway)
Initial speed Ego	Using private action Longitudinal/Speed dynamics shape = ”step” target value=”25” Sets Ego vehicle speed = 25 m/s. “step” = instantly (no acceleration)
Initial speed TargetA	Using private action Longitudinal/Speed dynamics shape = ”step” target value=”25” Sets vehicle speed = 25 m/s. “step” = instantly (no acceleration)
Initial speed TargetB	Using private action Longitudinal/Speed dynamics shape = “step” target value = relative(object=”TargetA”, value=0) valueType = “delta” continuous = “false” This will set TargetB speed equal to the speed of TargetA Delta value 0, would be same, in this case, as factor 1.0 Continuous = “false” will set speed once, in this case enough since TargetA speed will not change
Action Ego merge (lane change)	Using private action Lateral/LaneChange dynamics shape = "sinusoidal" time=”2.8” target value=”-2” Lane change duration is 2.8 seconds. Change to lane ID -2.
Condition Ego merge (lane change)	Using condition ByEntity/ReachPosition TriggeringEntity = “Ego” position = Road coordinate (roadID=”0”, laneID=”-3”, s=”20”) Tolerance = “10” Condition will trig whenever Ego reaches a circle of radius 10m around specified position.

Highway-merge advanced	OpenSCENARIO elements and attributes
Declare actors	Ego (vehicle under test) TargetA (First vehicle on highway) TargetB (Second vehicle on highway)
Initial position	Using road positions Ego vehicle on road ID 1 and target vehicles on road ID 2 (highway)
Initial speed Ego	Using private action Longitudinal/Speed dynamics shape = ”step” target value=”12” Sets Ego vehicle speed = 12 m/s. “step” = instantly (no acceleration)
Initial speed TargetA	Using private action Longitudinal/Speed

	dynamics shape = ”step” target value=”12” Sets vehicle speed = 12 m/s. “step” = instantly (no acceleration)
Initial speed TargetB	Using private action Longitudinal/Speed dynamics shape = “step” target value = relative(object=”TargetA”, value=0) valueType = “delta” continuous = “false” This will set TargetB speed equal to the speed of TargetA Delta value 0, would be same, in this case, as factor 1.0 Continuous = “false” will set speed once, in this case enough since TargetA speed will not change
TargetB Action close gap	Using private action Longitudinal/Distance object=”TargetA” distance=”2” Dynamics/limited (maxAcceleration=”10”, maxDecceleration=”7”, maxSpeed=”50”) This action will move TargetB to a distance of 2 meters from TargetA. The movement will be limited by maximum values of acceleration, deceleration and speed.
Close gap action condition	Using condition ByEntity/RelativeDistance TriggeringEntity = ”TargetB” Entity = “Ego” Value = "1" Type = "lateral" Rule = "less_than" edge = “rising” Condition will be true when the lateral distance between Ego and TargetB is decreasing to less than 1 meter. Edge = “rising” makes the trig happen when condition switches from false to true.
TargetB Action open gap	As close gap above, with only difference that distance = 15 meters instead of 2.
Open gap action condition	Same as close gap action, with only difference that Rule = “greater_than” So, this will trig when lateral distance grows greater than 1 meter, as opposed to decreasing to less than 1 meter.
Activate Act condition	Using condition ByEntity/RelativeDistance TriggeringEntities (“TargetA”, “TargetB”, rule=”all”) entity="Ego" value = "30" type = "intertial" rule = "less_than" edge = “rising” TriggeringEntities rule = “all” means that the condition must be fulfilled for ALL triggering entities type “inertial” makes distance calculated in world (x, y, z) coordinates (not road coordinates) This condition will trig whenever Ego moves within 30 meters of both vehicles TagetA and TargetB. The Act serves as an overall group node for the open/close gap actions. When Act is not active, the open/close gap conditions will not even be evaluated.
Deactivate Act condition	Same as Activate Act condition above, with only difference that: edge = “falling” So, this will trig whenever condition (Ego within 30 meters of other target vehicles) becomes false from being true.

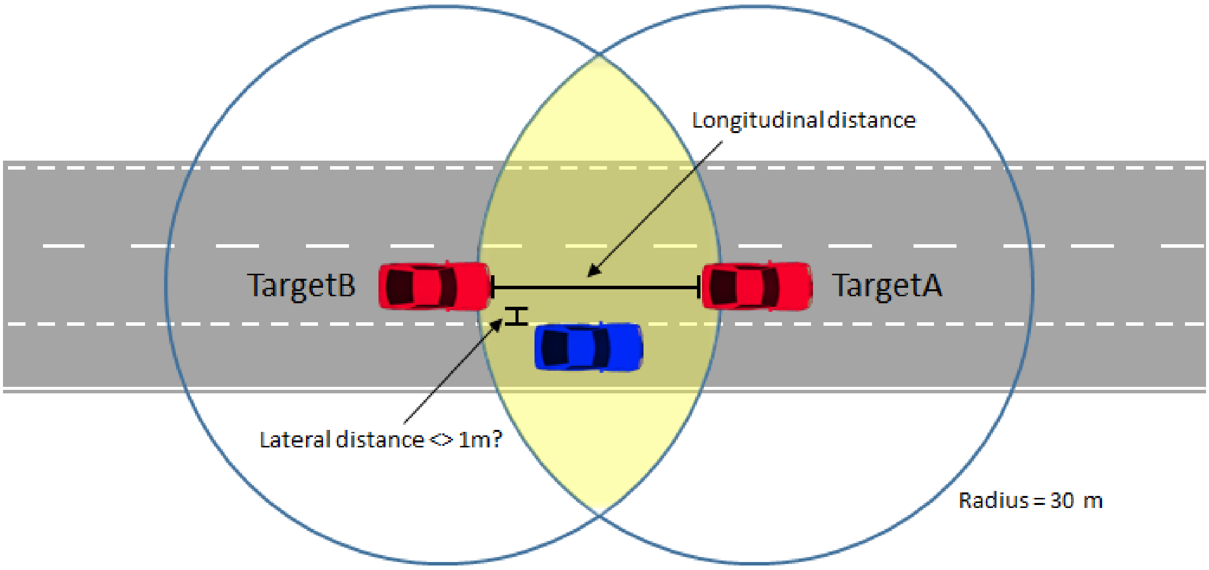

Image above: Relating to Highway-merge advanced, the yellow area shows where the overall act is activated. Distance actions for opening or closing the gap are triggered by lateral distance greater or less than 1 meter.