ScenarioRunner x OPENSCENARIO 1.0

Scenario Modelling

ScenarioRunner: OpenSCENARIO Support

The scenario_runner provides support for the **OpenSCENARIO 1.0** standard. The current implementation covers initial support for maneuver **Actions**, **Conditions**, **Stories** and the **Storyboard**.

If you would like to use evaluation criteria for a scenario to evaluate pass/fail results, these can be implemented as **StopTriggers.**

However, not all features for these elements are yet available. If in doubt, please see the module documentation in srunner/tools/openscenario_parser.py

!!! Note In the OpenSCENARIO 1.0 standard, a definition of test / evaluation criteria is not defined. For this purpose, you can re-use StopTrigger conditions with CARLA. The following StopTrigger conditions for evaluation criteria are supported through ParameterConditions by providing the criteria name for the condition:

```
* criteria_RunningStopTest
* criteria_RunningRedLightTest
* criteria_WrongLaneTest
* criteria_OnSideWalkTest
* criteria_KeepLaneTest
* criteria_CollisionTest
* criteria_DrivenDistanceTest
```

https://github.com/carla-simulator/scenario_runner/blob/master/Docs/openscenario_support.md

OpenSCENARIO 1.0 User Guide: https://www.asam.net/index.php?eID=dumpFile&t=f&f=4092&token=d3b6a55e911b22179e3c0895fe2caae8f5492467

OpenSCENARIO 1.0 Structure

```
<?xml version="1.0" encoding="utf-8"?>
<OpenSCENARIO
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance  "
    xsi:noNamespaceSchemaLocation="../Schema/OpenSCENARIO.xsd ">

  <FileHeader
      revMajor="1 revMinor="3 date="2020-02-21T10:00:00"
      description="Cut-In example" author="ASAM e.V." />

  <ParameterDeclarations> ... </ParameterDeclarations>

  <CatalogLocations />

  <RoadNetwork> ... </RoadNetwork>

  <Entities> ...  </Entities>

  <Storyboard> ... </Storyboard>

</OpenSCENARIO>
```

Structure of File

- **File Header**: Metadata about Scenario (name, description, author, version)
- **Parameter Declaration**: Parameter that can be used throughout scenario (makes scenario more flexible)
- **Road Network**: Road Infrastructure - Reference to road network file (i.e. OpenDRIVE files)
- **Entities**: Define Actors of scenario (vehicles, pedestrians, other dynamic object)
- **Storyboard**: Describes temporal Sequence of events and action that occur in the scenario
    - **Init**: Initial Actions to setup the scenario
    - **Story**: Sequence of action that events organized into stories and acts
    - **StopCondition**: Condition under which the scenario should end

OpenSCENARIO 1.0 Structure: RoadNetwork

```
<OpenSCENARIO ...>
 ...
  <RoadNetwork>
   <LogicFile
     filepath="Databases/AB_RQ31_Straight.xodr" />
  </RoadNetwork>
 ...
</OpenSCENARIO>
```

RoadNetwork

- Contains LogicFile - Tags with filepath to OpenDRIVE file (= .xodr)
- Pre-build CARLA Maps (Town01, Town02, …) are based on OpenDRIVE files
- Example Code for using Town01 (FollowLeadingVehicle.xosc):

```
<RoadNetwork>
   <LogicFile filepath="Town01"/>
   <SceneGraphFile filepath=""/>
 </RoadNetwork>
```

OpenSCENARIO 1.0 Structure: Entities

```
<OpenSCENARIO ...>
...
  <Entities>
    <ScenarioObject name="Default_Car">
        <Vehicle vehicleCategory="car"> ... </Vehicle>
    </ScenarioObject>

    <ScenarioObject name="Pedestrian1">
        <Pedestrian pedestrianCategory="pedestrian"> ... </Pedestrian>
    </ScenarioObject>

    <EntitySelection name="MySelection">
      <Members>
        <EntityRef entityRef="Default_Car"/>
        <EntityRef entityRef="Pedestrian1"/>
      </Members>
    </EntitySelection>
  </Entities>
...
</OpenSCENARIO>
```

Entities can contain …

- **ScenarioObject**: For declaring entities, i.e
  - Vehicle
  - Pedestrian
- **EntitySelection**: Groups different previously defined entities together (useful as you can now reference all objects with one identifier)

ScenarioObject

- Attribute name: Global identifier for this entity
- Typically, one ScenarioObject is called “Ego” / “Ego_Vehicle” / “hero”

EntitySelection

- Comparable to in HTML
- Attribute entityRef @ EntityRef: Unique ID of ScenarioObject

OpenSCENARIO 1.0 Structure: Entities (II)

```
<ScenarioObject name="hero">
  <Vehicle name="vehicle.lincoln.mkz_2017" vehicleCategory="car">
    <ParameterDeclarations/>
    <Performance maxSpeed="69.444" maxAcceleration="200" maxDeceleration="10.0"/>
    <BoundingBox>
      <Center x="1.5" y="0.0" z="0.9"/>
      <Dimensions width="2.1" length="4.5" height="1.8"/>
    </BoundingBox>
    <Axles>
      <FrontAxle maxSteering="0.5" wheelDiameter="0.6"
                 trackWidth="1.8" positionX="3.1"
                 positionZ="0.3"/>
      <RearAxle maxSteering="0.0" wheelDiameter="0.6"
                trackWidth="1.8" positionX="0.0" positionZ="0.3"/>
    </Axles>
    <Properties>
      <Property name="type" value="ego_vehicle"/>
      <Property name="color" value="0,0,255"/>
    </Properties>
  </Vehicle>
</ScenarioObject>
```

- Technical Properties
- Instance Property based on Blueprint

ScenarioObject: Vehicle

- Attribute `name`: Represents blueprint reference
- Starts with `<ParameterDecleration/>`
- Technical Properties (•)
  - Performance
  - BoundingBox
  - Axle
- CARLA Blueprint Properties (•)

Open Issue

- How to fetch the exact values for a vehicle (Performance, FrontAxle, RearAxle, BoundingBox)
- ChatGPT suggest it's directly possible to read values with UnrealEngine inspecting (TODO: Download UnrealPak / UnrealUEViewer / UnrealEngine to inspect …)

OpenSCENARIO 1.0 Structure: Storyboard

```
<Storyboard>
    <Init>
      <Actions>
            ...
      </Actions>
    </Init>
    <Story name="MyStory">
      <ParameterDeclarations>
         ...
      </ParameterDeclarations>
      <Act name="Act1">
            ...
      </Act>
    </Story>
    <StopTrigger> ... </StopTrigger>
</Storyboard>
```

Storyboard Key Components:

- **Init**: Specifies Initial Conditions (Positions, Speed, State of Entities)
- **Story**: Wrapper of Sequence of Acts
- **Act**: Contain Sequence of ManeuverGroups, which are collections of Maneuvers assigned to specific entity)
- **Event**: Defines single Action or set of Action triggered by a specific condition
- **Actions**: Are the smallest Unit and include Movements, Speed Changes, Lane Changes, etc..
    - **PrivateAction**: relate to specific, individual entity
    - **GlobalAction**: affect entire scenario or multiple entities (i.e. Environmental Condition, Traffic Signal Control, Scneario-Wide Trigger)

OpenSCENARIO 1.0 Structure: Storyboard (II)

```
<Storyboard>
  <Init>
    <Actions>
        ...
    </Actions>
  </Init>
  <Story name="MyStory">
    <ParameterDeclarations >
      ...
    </ParameterDeclarations >
    <Act name="Act1">
        ...
    </Act>
  </Story>
  <StopTrigger> ... </StopTrigger>
</Storyboard>
```

Storyboard Component Hierarchy

Storyboard: The main container for all the scenario's actions and events. It orchestrates the timeline and sequence of different stories.

- **Init:** Specifies initial Conditions
- **Story**: A story is a high-level sequence of activities involving multiple entities. It can be seen as a collection of acts.
  - **Act**: An act is a part of a story that groups maneuvers. Acts can be repeated and have conditions for their execution.
    - **Maneuver**: A maneuver contains a sequence of events for a specific entity or a group of entities.
      - **Event**: The event is the basic unit within a maneuver. It defines specific actions that occur when certain conditions are met.
        - **Action**: Smallest Unit
        - **StartTrigger**: Condition for starting event
  - **StartTrigger**: Condition when act begins
    - **Condition**
  - **StopTigger**: Condition when act ends
    - **Condition**
- **StopTrigger**
  - **Condition**

OpenSCENARIO 1.0 Structure: Storyboard (III)

StopTrigger @ Storyboard

- Criteria that will be evaluated in the final Report (either FAILURE or SUCCESS)

!!! Note In the OpenSCENARIO 1.0 standard, a definition of test / evaluation criteria is not defined. For this purpose, you can re-use StopTrigger conditions with CARLA. The following StopTrigger conditions for evaluation criteria are supported through ParameterConditions by providing the criteria name for the condition:

```
* criteria_RunningStopTest
* criteria_RunningRedLightTest
* criteria_WrongLaneTest
* criteria_OnSideWalkTest
* criteria_KeepLaneTest
* criteria_CollisionTest
* criteria_DrivenDistanceTest
```

See example FollowLeadingVehicle.xosc

Storyboard Component Hierarchy

Storyboard: The main container for all the scenario's actions and events. It orchestrates the timeline and sequence of different stories.

- **Init:** Specifies initial Conditions
- **Story**: A story is a high-level sequence of activities involving multiple entities. It can be seen as a collection of acts.
  - **Act**: An act is a part of a story that groups maneuvers. Acts can be repeated and have conditions for their execution.
    - **Maneuver**: A maneuver contains a sequence of events for a specific entity or a group of entities.
      - **Event**: The event is the basic unit within a maneuver. It defines specific actions that occur when certain conditions are met.
        - **Action**: Smallest Unit
        - **StartTrigger**: Condition for starting event
  - **StartTrigger**: Condition when act begins
    - **Condition**
  - **StopTigger**: Condition when act ends
    - **Condition**
- **StopTrigger**
  - **Condition**

OpenSCENARIO 1.0 Structure: Storyboard :: Init (IV)

```
<Storyboard>
  <Init>
    <Actions>
          ...
    </Actions>
  </Init>
  <Story name="MyStory">
    <ParameterDeclarations>
       ...
    </ParameterDeclarations>
    <Act name="Act1">
          ...
    </Act>
  </Story>
  <StopTrigger> ... </StopTrigger>
</Storyboard>
```

Environment Setup

- Time
- Weather (Fog, Sun, Precipitation)
- RoadConditions

Entity Initialization

- Location (impl. with “TeleportAction”)
- Controller and initial start values

OpenSCENARIO 1.0 Structure: Storyboard :: Init (V)

```
<Storyboard>
  <Init>
    <Actions>
          ...
    </Actions>
  </Init>
  <Story name="MyStory">
    <ParameterDeclarations>
        ...
    </ParameterDeclarations>
    <Act name="Act1">
          ...
    </Act>
  </Story>
  <StopTrigger> ... </StopTrigger>
</Storyboard>
```

Environment Setup

Entity Setup

```
<GlobalAction>
  <EnvironmentAction>
    <Environment name="Environment1" >
      <TimeOfDay animation="false" dateTime="2020-02-21T12:00:00" />
            <Weather fractionalCloudCover ="zeroOktas" >
              <Sun illuminance="100000.0" azimuth="0.0" elevation="1.571" />
              <Fog visualRange="100000.0" />
              <Precipitation precipitationType="dry" precipitationIntensity="0.0" />
            </Weather>
            <RoadCondition frictionScaleFactor="1.0" />
    </Environment>
  </EnvironmentAction>
</GlobalAction>
```

```
<Private entityRef="Ego">
  <PrivateAction>
    ...
  </PrivateAction>
  <PrivateAction>
    ...
  </PrivateAction>
</Private>
```

```
<Private entityRef="A1">
  <PrivateAction>
    ...
  </PrivateAction>
  <PrivateAction>
    ...
  </PrivateAction>
</Private>
```

OpenSCENARIO 1.0 Structure: Storyboard (VI)

```
<Storyboard>
  <Init>
    <Actions>
          ...
    </Actions>
  </Init>
  <Story name="MyStory">
    <ParameterDeclarations>
       ...
    </ParameterDeclarations>
    <Act name="Act1">
          ...
    </Act>
  </Story>
  <StopTrigger> ... </StopTrigger>
</Storyboard>
```

Action Wrapper of Private Actions

- **LongitudinalAction**
  - SpeedAction
- **TeleportAction**: Entity gets teleport to position
- **LeteralAction:** Lateral Movement of Entity
  - LangeChangeAction
  - LangeOffsetAction
- **RoutingAction**
  - FollowRouteAction
  - AcquirePositionAction
- **ActiveControllerAction**: Can be used to set CARLA autopilot
- **OverrideControllerAction**
- **UserDefinedAction**
- *VisibilityAction: Not Supported*

ScenarioRunner: OpenSCENARIO$^{1.0}$ Examples

All Examples: srunner/examples/

- CatalogExample.xosc
- ChangingWeather.xosc
- CyclistCrossing.xosc
- FollowLeadingVehicle.xosc
- InitAddEntityAction.xosc
- InitDeleteEntityAction.xosc
- IntersectionCollisionAvoidance.xosc
- LaneChangeSimple.xosc
- LaneOffsetActionExample.xosc
- OxcControllerExample.xosc
- PedestrianCrossingFront.xosc
- Slalom.xosc
- StoryAddEntityAction.xosc
- StoryDeleteEntityAction.xosc
- SyncArricalIntersection.xosc
- VehicleLateralDistance.xosc

ScenarioRunner: OpenSCENARIO$^{1.0}$ Examples

srunner/examples/

CatalogExample.xosc

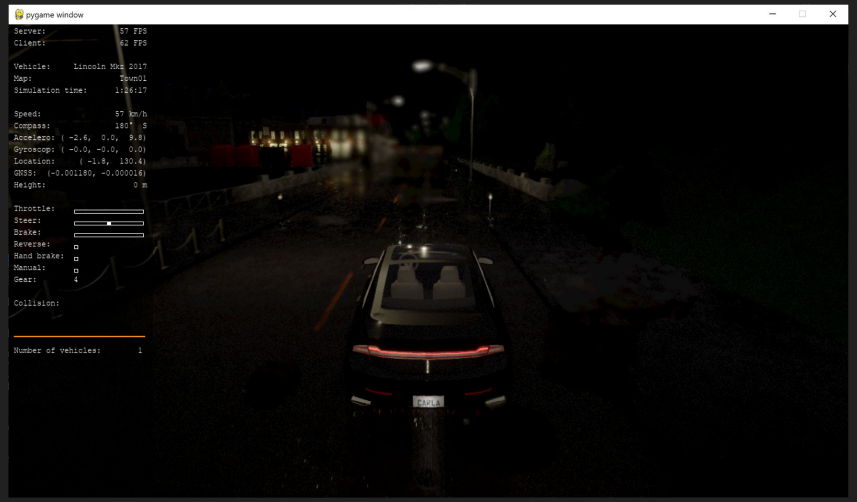
Description

- 2 Vehicles are in front (not moving)
- Scenario ends once cars are approached

ScenarioRunner: OpenSCENARIO$^{1.0}$ Examples

srunner/examples/

ChangingWeather.xosc



Description

- After a vehicle, Weather changes to low-light rainy environment
- No Stopping Condition defined (runs infinite)

ScenarioRunner: OpenSCENARIO$^{1.0}$ Examples

srunner/examples/

InitAddEntityAction.xosc

Description

- Very Similar to FollowLeadingVehicle, but with 2 more participants
- Overall: 3 Vehicles that create traffic jam on right lane (due to red traffic light)

ScenarioRunner: OpenSCENARIO$^{1.0}$ Examples

srunner/examples/

InitDeleteEntityAction.xosc

Description

- Basically the same as FollowLeadingVehicle
- Difference ??

ScenarioRunner: OpenSCENARIO$^{1.0}$ Examples

srunner/examples/

LaneChangeSimple.xosc

Description

- Setup on a 3 lane highway
- One car stands, accelerates and moves lane to left
- For some reason, the effect for steering is like on a drift track with drift tires ????

ScenarioRunner: OpenSCENARIO$^{1.0}$ Examples

srunner/examples/

LaneOffsetActionExample.xosc

ScenarioRunner: OpenSCENARIO$^{1.0}$ Examples

srunner/examples/

OscControllerExample.osxc

Description

- Setup similar to FollowLeadingVehicle, but leading vehicle is not breaking but keeps on driving
- No stop
- For some reason, the leading vehicle ignores all red lights though

ScenarioRunner: OpenSCENARIO$^{1.0}$ Examples

srunner/examples/

PedestrianCrossingFront.xosc

Description

- Pedestrian crosses crossing in front of us, remains on the middle of the lane for a few seconds, and keeps on going

ScenarioRunner: OpenSCENARIO$^{1.0}$ Examples

srunner/examples/

Slalom.xosc

Description

- Multiple static assets (Misc Objects) are on the lane

ScenarioRunner: OpenSCENARIO$^{1.0}$ Examples

srunner/examples/

StoryAddEntityAction.xosc

Description

- Basically FollowLeadingVehicle, that ends quite early
- Only Difference: ???

ScenarioRunner: OpenSCENARIO$^{1.0}$ Examples

srunner/examples/

StoryDeleteEntityActionx.xosc

Description

- Similar to FollowLeadingVehicle.xosc
- Before the vehicle, 3 other vehicle exist that gets then (at runtime) deleted (they disappear)
- Rest of behaviour is similar to FollowLeadingVehicle.xosc

ScenarioRunner: OpenSCENARIO$^{1.0}$ Examples

srunner/examples/

SyncArrivalIntersection.xosc

Description

- Vehicle Left arrives when it's green for us but drives (even though it should be red for him!)

ScenarioRunner: OpenSCENARIO$^{1.0}$ Examples

srunner/examples/

VehicleLateralDistance.xosc

Description

- Car behind us spawns
- Once we achieved a certain distance, the car starts to drive
- The car overtakes us if we are too slow

OpenSCENARIO x ScenarioRunner: Modelling

Example Analaysis: PedestrianCrossingFront

Basic Setup

- Meta Data
- Define Location
- Define Entities
- Define Storyboard
  - Initial State
  - Story
  - Testing Success/Failure Criteria

Description

- Pedestrian crosses crossing in front of us, remains on the middle of the lane for a few seconds, and keeps on going

OpenSCENARIO x ScenarioRunner: Modelling (II)

Example Analaysis: PedestrianCrossingFront

Basic Setup

- Meta Data
  - Fileheader description must start with “**CARLA:**” in order to use CARLA - coordinate system
- Define Location
  - Town01
- Define Entities
  - Define Entity Blueprint
  - Define Technical Properties
  - Define CARLA Properties (i.e. type,color, ..)
- Define Storyboard
  - Initial State
  - Story
  - Testing Success/Failure Criteria

```
<FileHeader revMajor="1"
        revMinor="0"
        date="2020-03-24T12:00:00"
        description="CARLA:PedestrianCrossing"
        author=""/>
  <ParameterDeclarations />
  <CatalogLocations />

  <RoadNetwork>
    <LogicFile filepath="Town01"/>
    <SceneGraphFile filepath=""/>
  </RoadNetwork>
```

OpenSCENARIO x ScenarioRunner: Modelling (III)

Example Analaysis: PedestrianCrossingFront

Basic Setup

- Meta Data
- Define Location
- Define Entities
  - Define Entity Blueprint
  - Define Technical Properties
    - Performance
    - BoundingBox
    - Axles
  - Define CARLA Properties (i.e. type,color, ..)
- Define Storyboard
  - Initial State
  - Story
  - Testing Success/Failure Criteria

Involved Entities

- Ego Vehicle
- NPC Pedestrian

OpenSCENARIO x ScenarioRunner: Modelling (IIIb)

Define Entities

```
<ScenarioObject name="hero">
  <Vehicle name="vehicle.volkswagen.t2" vehicleCategory="car">
    <ParameterDeclarations />
    <Performance maxSpeed="69.444"
                 maxAcceleration="200"
                 maxDeceleration="10.0" />
    <BoundingBox>
      <Center x="1.5" y="0.0" z="0.9" />
      <Dimensions width="2.1" length="4.5" height="1.8" />
    </BoundingBox>
    <Axles>
      <FrontAxle maxSteering="0.5"
                 wheelDiameter="0.6"
                 trackWidth="1.8"
                 positionX="3.1"
                 positionZ="0.3" />
      <RearAxle maxSteering="0.0"
                wheelDiameter="0.6"
                trackWidth="1.8"
                positionX="0.0"
                positionZ="0.3" />
    </Axles>
    <Properties>
      <Property name="type" value="ego_vehicle"/>
    </Properties>
  </Vehicle>
</ScenarioObject>
```

```
<ScenarioObject name="adversary">
  <Pedestrian model="walker.pedestrian.0001"
              mass="90.0"
              name="walker.pedestrian.0001"
              pedestrianCategory="pedestrian">
    <ParameterDeclarations />
    <BoundingBox>
      <Center x="1.5" y="0.0" z="0.9"/>
      <Dimensions width="2.1" length="4.5" height="1.8"/>
    </BoundingBox>
    <Properties>
      <Property name="type" value="simulation"/>
    </Properties>
  </Pedestrian>
</ScenarioObject>
```

OpenSCENARIO x ScenarioRunner: Modelling (IV)

Example Analaysis: PedestrianCrossingFront

Basic Setup

- Meta Data
- Define Location
- Define Entities
- Define Storyboard
  - Initial State
    - Init Environment
      - Time of Day
      - Weather
      - Road Condition
    - Init Ego Vehicle
    - Init NPC Pedestrian
  - Story
    - Event: PedestrianStartsWalking
    - Event: PedestrianStopsAndWaits
    - Event: PedestrianWalksAway
    - Event: PedestrianWaits
  - Testing Success/Failure Criteria

Scenario Description (Temporal Sequence)

- Pedestrian spawns at Sidewalk
- Pedestrians walks on street as soon as ego vehicle is near enough
- Pedestrian stops walking and waits for a few seconds
- Pedestrian continues walking to other street side

OpenSCENARIO x ScenarioRunner: Modelling (IVb)

Example Analaysis: PedestrianCrossingFront

Basic Setup

- Meta Data
- Define Location
- Define Entities
- Define Storyboard
  - Initial State
  - Story
    - Event: PedestrianStartsWalking
    - Event: PedestrianStopsAndWaits
    - Event: PedestrianWalksAway
    - Event: PedestrianWaits
  - Testing Success/Failure Criteria

Scenario Description

- Pedestrian spawns at Sidewalk
- Pedestrians walks on street as soon as ego vehicle is near enough
- Pedestrian stops walking and waits for a few seconds
- Pedestrian continues walking to other street side

Modelling

- States relates to Action [or an Event]
- State is “completed” if action(s) were executed completely
- Each State defines when it’s activated (in Event Wrapper Context via *StartTrigger*)

OpenSCENARIO x ScenarioRunner: Modelling (IVc)

Example Analaysis: PedestrianCrossingFront

Basic Setup

- Meta Data
- Define Location
- Define Entities
- Define Storyboard
  - Initial State
  - Story
    - Event: PedestrianStartsWalking
    - Event: PedestrianStopsAndWaits
    - Event: PedestrianWalksAway
    - Event: PedestrianWaits
  - Testing Success/Failure Criteria

```
<Event name="PedestrianStopsAndWaits" priority="overwrite">

  <Action name="PedestrianStopsAndWaits">
  ...
  </Action>

  <StartTrigger>
    <ConditionGroup>
      <Condition name="AfterPedestrianWalks"
                 delay="0"
                 conditionEdge="rising">
        <ByValueCondition>
          <StoryboardElementStateCondition
                storyboardElementType="action"
                storyboardElementRef="PedestrianStartsWalking"
                state="completeState"/>
        </ByValueCondition>
      </Condition>
    </ConditionGroup>
  </StartTrigger>

</Event>
```

When Action called “**PedestrianStartsWalking**” yields in state “**completeState**”, the Condition is fulfilled → Trigger is activated
→ All Actions of Event are executed

Summary: ScenarioRunner Modelling with OpenSCENARIO

Summary of Modelling Approach

- Very similar as Modelling in typical Model-based Testing
- Define Basic Components
  - Define which Map is used
  - Define Entities that are involved
  - Define Environment and Initial Setup in <init>
  - Define what the overall Success / Failure Conditions that will be shown in the final Test Report
- Define FSM
  - Identify States (where something is performed from an entity)
  - Identify State Transitions between States
- Translation of FSM
  - Each State is modelled in an "**Event**" Wrapper
  - The "thing" that done during the state is modelled as **Action**
  - *Incoming* Transitions from other states are modelled in **StartTrigger** (Core: StoryboardElementStateCondition that waits for *completeState* of another *ACTION* of another state)
- Define Exit Criteria for Success / Failure
- Consider further Aspects [*]

Notes

- There are MANY ways of modelling, the modelling approach left is quite restrictive but keeps it simple (but does not exploit all possibilities that OpenSCENARIO provides)
- A "*state*" in context of OpenSCENARIO usually refers to either *Event*, or an *Action*, in the Summary left in relates to the concept of a "state" in context of FSM
- "*Define FSM*" + "*Translation of FSM*" is the non-trivial aspects (everything else is usually easy)

Further Aspects

- Define when Scenario should terminate (in case that specific events are not triggered)
- If necessary, wait for scenario start a bit (i.e. until controller is successfully connected and actively interacts)

Darko @ AVL Prior Mail Summary

I think it would be best if they start working on different parking scenarios in Carla

Just to reiterate what is important for us regarding parking scenarios:

- we would like to be able to populate Carla's parking slots in a predefined way and
- have a representative OpenSCENARIO dataset of scenarios/movements of actors (pedestrians or other vehicles) at the parking lots.

I think, parking lots in Carla doesn't have lanes, so you might need a scenariorunner to move actors, which is available in AVL. In the follow-up MT, we could integrate an ADAS parking function into the simulation and do testing and validation by using the method we discussed.

Master Theses on the "**Scenario-based testing of ADAS parking functions**" topic where we will explore the test setup built in AVL and the methodologies you proposed in the meeting.

Definition and preparation of a project proposal for March which should be dedicated to **functional testing**. It is preferable to target national calls (Digital, Bridge, …).