Connected Places Catapult

Design considerations for ODD ontology

March 2020

Executive Summary

The MUSICC (Multi-User Scenario Catalogue for CAVs) project has produced a proof of concept system to store scenarios for use in CAV (Connected and Autonomous Vehicle) safety tests. For the foreseeable future, CAVs will not be 'go anywhere' vehicles: they will have a defined Operational Design Domain (ODD) which defines the conditions under which they can operate. When testing these vehicles, it is important to be able to identify which scenarios fall within their ODD. MUSICC supports a basic form of ODD description which can be written in the form of a search query. However, it has become clear that this simple approach will not support all likely use cases, and we believe that development of a more sophisticated language needs to be tackled by the community. This document sets out some of the design considerations for this language and is intended to help inform development efforts.

Firstly, we identify the purpose and high-level requirements of the language. These include the need for it to be machine readable (for use in databases and monitoring systems) and to define an ODD which is understandable to a human (for use by regulators, buyers and safety drivers, where applicable).

Secondly, we explore the types of content which an ODD should describe. There are two broad types: geographical information (e.g. a map of roads where the vehicle may drive) and a description of the operating environment. We expect that the operating environment description will be best defined using categoric variables, which can be both easy to understand and technically well defined. Special care needs to be taken when defining categoric variables relating to weather and other road user behaviour. For each of these, two options are proposed. Weather conditions may be described using physical properties (e.g. droplet size, intensity) or their effect (e.g. visibility). The former relates more directly to the effect on vehicle sensors, while the latter is easier to understand are more likely to be available in historical records/current forecasts. Similarly, road user behaviour can be inferred from the rules of the road (e.g. "cycling not permitted") or defined directly (e.g. "cannot operate on roads with cyclists"). It may be necessary to specify both, as one cannot be reliably inferred from the other.

Finally, the requirements for the language semantics are explored. A key observation is that safety-related decisions will be made on the basis of ODD descriptions: careful language design can help reduce the risk of dangerous errors. The risk is especially high when, inevitably, the language is altered in the future. To mitigate this, we propose that every ODD description should be explicitly tied to a single version of the language. Tools reading an ODD description can then check that it is valid for the declared version. When a description is moved from one version of a language to another, a migration process should make any changes in meaning clear to the user.

It may be convenient if categoric variables are organised in a hierarchy. This would help to keep descriptions concise, by allowing expressions such as "all urban roads" (where many different types of urban road can be defined at lower levels). However, this is slightly inflexible, as it does not allow for more than one type of grouping (if roads are grouped by urban/rural, "all dual carriageways" is not an allowable expression). Allowing multiple groupings is possible by relaxing the requirement for strict hierarchy, but this could make the language harder to manage.

An issue identified, but not solved, is the need to express relatively complex conditions. Real ODDs may need to refer to combinations of features (e.g. "priority junction where speed limit > 40mph"). Searching scenario databases for combinations like this may require scenario and road network description languages which can be searched directly.

Contents

Notice

This document has 20 pages including the cover.

AUTHORISATION:

ACTION	SIGNATURE BLOCK	NAME AND POSITION WITHIN CPC
Written by:	Robert Myers	Robert Myers Technologist
Reviewed by:	Zeyn Saigol	Zeyn Saigol Principal Technologist

RECORD OF CHANGES:

REALEASED TO	VERSION	REASON FOR CHANGE	DATE
BSI PAS 1883 SG, MUSICC IAG	0.1	First draft	02/03/2020

1. Context

The MUSICC (Multi-User Scenario Catalogue for CAVs) project has produced a proof of concept system to store scenarios for use in CAV (Connected and Autonomous Vehicle) safety tests[1]. As well as storing the scenarios themselves, the MUSICC system also stores metadata: this is searchable and is designed for use in scenario selection.

Conceptually, MUSICC metadata is separated into four groups:

- General properties: information about the test case which the scenario describes. Includes information likely to be useful when selecting scenarios to test a particular feature, as well as contextual information (e.g. exposure and expected outcome).
- ODD specification: metadata used to identify scenarios which match the specified Operational Design Domain.
- Admin data: information relating to ownership, version control and any external sources which reference this scenario (e.g. specifications and regulations).
- Custom data: user extensible fields.

Appendix A gives a summary of the current MUSICC metadata fields.

As MUSICC has progressed, it has become clear that the simple ODD specification currently used will not be enough to cover all likely use cases. This document sets out some of the design considerations for a future, more comprehensive approach. There are likely to be others: the considerations here have become clear through our work on the MUSICC project, but it is likely that other members of the CAV development community will be able to add to them based on their own experience. We hope that publishing these design considerations will make a useful contribution to ODD language development efforts, ideally as part of a standardisation process.

2. Purpose and high-level requirements

What can legitimately form part of an ODD?

An Operational Design Domain is formally defined (by SAE J3016) as:

> Operating conditions under which a given driving automation system or feature thereof is specifically designed to function, including, but not limited to, environmental, geographical, and time-of-day restrictions, and/or the requisite presence or absence of certain traffic or roadway characteristics (Society of Automobile Engineers, 2018).

An ODD defines the conditions under which an Automated Driving System (ADS) may operate. This includes featues of the world which are static (e.g. road layouts) and dynamic (traffic behaviours).

The SAE definition above allows any set of conditions to be defined as an ODD. However, for vehicles at SAE level 4, leaving the ODD could be dangerous. We suggest that any ODD defined for these vehicles

[1] A general description of the project can be found at https://cp.catapult.org.uk/case-studies/musicc/

should be ‘**reasonable’**: contain only conditions which can be predicted before a vehicle commits to entering them. For example:

- An ODD which contains only urban roads in daylight is reasonable, since the location of urban roads and the hours of daylight can be predicted in advance.
- An ODD which includes multi-lane highways but excludes other vehicles overtaking the ego vehicle is not reasonable: other road user behaviour of this type could occur at any point.
- An ODD which does not include cyclists may be reasonable, provided that the vehicle only operates on restricted access roads.

It is possible that regulators will impose further restrictions on the content of an ODD. These might be required to address concerns about traffic flow, consumer understanding, or safety margins. Regulators may also require the vehicle to demonstrate a certain level of performance under conditions outside of its ODD.

Possible use cases

For ADS developers

Developers need to design, verify and validate a system based on its expected use. An ODD ontology provides a formal way of defining the conditions under which a certain level of ADS performance is expected.

An ODD ontology could assist with identifying any features which have been omitted from a system’s testing. It may need to be supported by some additional metadata on scenario content, which would be used to represent things which cannot form part of a reasonable ODD.

For regulators

Regulators need to have a good understanding of where and when a vehicle may operate. This is important for them to be able to assess and monitor its safety/performance, manage liability, and investigate incidents.

Regulators need to be able to check that the ODD is reasonable and that the risk of a vehicle exiting it is controlled. This will include ensuring that:

- Suitable testing has been carried out and shows that the vehicle performs safely within its ODD.
- There are suitable safeguards (whether technical or operational) to ensure that the vehicle is not operated outside of its ODD.

This introduces two requirements for the ODD ontology:

- If scenario-based testing is used as part of a regulatory process, it must be possible to search for scenarios which match an ODD specification. A vehicle should be tested against these (and only these) scenarios. In the MUSICC system, the ODD description forms part of the scenario metadata.
- The ODD definition needs to be understandable by a person not involved in developing the vehicle. This is essential for the regulator to have confidence that the vehicle will stay within the ODD in operation.

For drivers (SAE levels 3 and 4 only)

Drivers need to know where and when they can use an ADS. Systems currently under test use a safety driver: this person needs to have a detailed understanding of the vehicle's capabilities. Where SAE level 3 or 4 systems are used in consumer vehicles, it is important that the driver understands their capabilities.

For buyers (end users/ system integrators)

A well-defined ODD specification is essential to make sure that the system is appropriate for its intended use.

ODD excursions

Upon reaching ODD limits, one of two things may happen:

- The ADS detects limits of the ODD being reached and achieves a suitable fallback condition.
- The vehicle does not detect that it has left its ODD: as the ADS has not been designed for this eventuality, unexpected behaviour may occur.

Clearly, the first option is preferable if there is a realistic possibility of the vehicle leaving its ODD. This suggests a need for an 'ODD monitoring system', which will, for example:

- Make sure that weather conditions remain within an acceptable range.
- Ensure that the vehicle is operating within a geofenced area.

Inevitably, any system to monitor whether the vehicle is inside its ODD will itself have a design specification. If an event outside of this occurs, a vehicle may fail to detect that it has left its ODD. A level of residual risk will always remain and will have to be accepted. For example, there will always be a possibility of an asteroid landing on a highway: a reasonable regulator is unlikely to require that every vehicle can detect this as an out-of-ODD event. Regulators need to understand the level of residual risk (which could be presented to them as part of a safety case) and decide whether to accept it.

Applications

There are many important applications for ODD ontologies, and it is highly desirable that the same language can fulfil all of them. In this section we discuss three examples. For these, use of a single language would allow a direct link to be demonstrated between the tests a vehicle has passed, the routes it may plan and the conditions under which it will revert to a minimal risk state.

Test case selection

Verification and validation efforts will assess the performance of an ADS in a defined set of test cases. It needs to be possible to identify which of these test cases are within a vehicle's ODD.

The use case for developers is immediately obvious. For regulators it is less clear, as the type approval process for L3+ vehicles is yet to be defined. However, due to the vast complexity and scale involved in the V&V of a L3+ system in all but the most restricted ODD it is clear that even a 'spot check' including a minimal proportion of the total set of V&V test cases would be of a scale that is impractical to manage manually. Therefore, agnostic the details of how regulation of CAV transpires, the automated management of the ODD is clearly essential. This means that a machine readable language is essential.

MUSICC allows test cases to be selected by defining the ODD description as a search query. Each scenario in the database has a metadata file associated with it: the search will return results where the metadata matches the ODD query. For this to work, it needs to be possible to define all aspects of the ODD in the form of test case metadata.

It is also possible to imagine an ODD language which can be used to directly search test case, scenario and road network descriptions. This is potentially powerful (see Section 4) but would be difficult or impossible to achieve with existing file formats.

ODD monitoring

An ADS may also rely on an ODD description in order to identify ODD excursions. This means comparing an ODD description file to real outputs from sensors.

Route planning/ vehicle selection

ODD descriptions could be useful for route planning, to choose a route which the vehicle can reasonably expect to complete. If an organisation or person has a choice of vehicles for a task, the ODD description may also be useful for selecting an appropriate one. Both of these applications require the ODD description to be compared to known or forecast conditions on a road network.

Summary of requirements

Requirement	Description
Reasonable	A reasonable ODD includes only conditions which can predicted before a vehicle commits to enter them. Ideally, the language should be able to specify any reasonable ODD.
Machine readable	The ODD description must be readable by a machine. It should be possible to use this to: • Select test cases which match an ODD description. This requires the ODD language to contain only elements which can be expressed in the form of metadata (or alternatively, for it to be possible to query the scenario description itself). • Determine whether a vehicle is inside its ODD, by comparing the ODD description to real sensor outputs. • Plan a route which is expected to remain inside the ODD, based on known and forecast data about the road network.
Unambiguous and exchangeable	The same description will need to be used by several stakeholders, including developers and regulators. It is vital that the language does not contain ambiguities, as these could make the division of safety responsibility unclear.
Understandable boundaries	The regulator must be able to understand the boundaries of a vehicle's ODD. This does not necessarily mean that the process to create the ODD must be understandable – for example, a map of roads created using a machine learning algorithm might be acceptable. If a system relies on a human driver to identify situations outside of a vehicle's ODD, the ODD boundaries must be easily understood by that driver.
Use cases	A single language must be useable for a variety of use cases, including test case selection, monitoring by an ADS (to allow it to revert to a minimal risk condition if required), and route selection.

3. ODD content

Geofences

ADS developers expect that early level 4 vehicles will be restricted to operating in a defined area. The process of defining this area may be complex, take a large number of variables into account, and/or require significant human effort. The result could be a defined route, a list of roads which have been specifically assessed, or an entire city. As a result, it is clear that geofences must be allowed as part of any ODD definition: a road feature which does not occur within the defined area is automatically outside of the ODD.

Geofences are not enough to define an ODD on their own, because of the following limitations:

- They only define the properties of physical road infrastructure: they do not give any information about the dynamic conditions which a vehicle may experience (e.g. weather, roadworks).
- It is difficult to express conditions which combine physical infrastructure criteria with other types of requirement. For example, the requirement "cannot use rural roads at night" would require separate geofences to be specified for day and night conditions. Section 4 - Expressing complex criteria discusses this type of restriction in more detail.
- Real world road networks change over time: an ODD defined purely using a geofence will also have its features change over time. Where a geofence is used, the ODD description should define what changes in the road network are allowed. Any changes other than these should be treated as ODD excursions (see Section 2).

Categoric variables

Several organisations have proposed lists of variables to be used when defining an ODD. These are generally designed to be both human readable and technically precise – for example, light intensity might be described as 'bright sunlight', which would correspond to a defined range of luminance values. BSI PAS 1883 (British Standards Insitution, 2019) aims to standardise the content of these lists.

Categoric lists could be used in combination with a geofence. Compared to using a geofence alone, this will make the ODD easier to understand and allow more complex conditions to be expressed. Geofences could be used to narrow a categoric list, for example, a vehicle restricted to use on specific motorways might specify that the road type must be 'motorway' AND the road must be within a specified geographic area.

Approaches to defining values

Weather

Weather can be described in two ways:

- By its physical characteristics (e.g. droplet size distribution, quantity falling per unit area).
- By its effect on sensors (e.g. visibility distance).

Weather is important for automated vehicles because it can affect sensors and the dynamics of the vehicle. However, the same type of weather may have different effects on different sensors.

The physical characteristics of weather relate more directly to its effect on the vehicle, and are more likely to be useful inputs for a models. However, describing weather in this way does have some important disadvantages:

- There is an absence of historical data with sufficient granularity.
- Currently available forecasts to not provide the required information.
- Non-experts may find weather described in this way hard to interpret (though it should be possible for tools to translate it to a more recognisable format).

Road user behaviour

An ADS must make certain assumptions about what other road users may be present and how they will behave. This can form part of a reasonable ODD, since predictions can be made in advance (e.g. it is unlikely that a vehicle operating only in pedestrianised areas will encounter road users travelling at motorway speeds). There are at least two ways to approach this problem:

- The ODD definition could specify the behaviours which a vehicle can handle. A separate process will be required to relate this to the roads which the vehicle may use.
- The ODD definition could specify only road conditions and rules. This implies, but does not guarantee, certain road user behaviours.

If the first approach is taken, test cases will still need to specify rules of the road to enable pass/fail criteria to be defined.

It may be necessary for a regulator to specify some assumptions which an ADS developer may make about other road user's behaviour. This is related to the concept of 'Reasonable Behaviour' introduced by Intel Mobileye in their Responsibility Sensitive Safety paper (Shalev-Shwartz, Shammah, & Shashua, 2017).

Summary of requirements

Requirement	Description
Geofences and categoric lists	The ability to specify a geofenced ODD is necessary, but not sufficient. It should be combined with a categoric list to specify dynamic behavior, to make the ODD more understandable and (possibly) allow more complex conditions to be specified.
Weather	Two options identified: • Define in terms of physical weather properties • Define in terms of the effect on sensors
Road user behavior	Two options identified: • Specify rules of the road • Specify actual actor behavior It may also be required to do both – one cannot be reliably inferred from the other

4. ODD semantics

Hierarchical structure

There are advantages to using a hierarchical structure of metadata to describe an ODD. This would make it possible to specify the ODD in whatever level of detail is required for a particular application – with the ability to ignore irrelevant levels. For example:

- A vehicle designed to operate in the desert might define 'precipitation' as outside of its ODD.
- Another vehicle might allow Light-Moderate rain but disallow any snow or other precipitation types.

Proposed terminology

We suggest using the following terms are used to describe hierarchical lists:

Term	Definition
Item	General term, used to refer to any single item in the list
Hierarchical list	A list where some elements have more than one child element associated with them. A reference to a parent element is equivalent to referencing all of its children.
Category	An item in a hierarchical list which has one or more child elements, but cannot be used as a value in itself – e.g. "weather"
Collection	An item which has children but can also be used as a value in itself – e.g. "rain"
Heading	A category at the top level of a hierarchical list
Leaf item	An element with no children which can be used directly
Field	A leaf node which has a value – e.g. "lane width"

Need for 'other'

It may be helpful to introduce an 'other' attribute at each level of the hierarchy, representing an unknown element. This would be used to represent any item which does not have an explicit category in the ODD ontology and would remove a source of ambiguity. Any item which does not have its own term in the language would fall into 'other': therefore, it can be explicitly included or excluded from the ODD. For example, it could be used to exclude unusual junction types from the ODD without requiring the language to have a value for all of them. Including 'other' may make migration between versions of the standard easier to understand.

Advantages and disadvantages

A hierarchical structure can make an ODD definition much easier to read by allowing unnecessary levels of detail to be hidden. Compare the two examples describing “major roads” in Figure 1 below. All of the items on the left could be grouped into a collection called “Arterial”: if this is done, there is no need to rewrite every item from the list. This makes the ontology more readable and therefore easier to understand (see requirements in section 1).

Description using non-hierarchical ontology

```
Road types:

        Urban motorway

        Urban non-motorway dual
        arterial

        Urban single arterial

        High street (significant
        through traffic)

        Rural motorway

        Rural non-motorway dual trunk
        road

        Other
```

Description using hierarchical ontology

```
Road types: Arterial (all)
```

Figure 1: Hierarchical and non-hierarchical definitions of “major roads”

Using a hierarchy can be problematic, since some items may naturally belong to more than one category. For example, the animal ‘dog’ could sensibly be included in any of the collections below:

- Small animals
- Medium animals (dogs vary in size)
- Domestic animals
- Animals commonly found on UK roads
- Animals commonly found on Swedish roads
- Animals listed in the UK road traffic act 1988

Avoiding overlapping collections means that some useful ODD restrictions may be difficult to specify concisely.

Alternative: non-strict hierarchy (defined lists)

Rather than following a strict hierarchy, a similar outcome could be achieved be defining lists of items which are likely to be commonly referenced together. Referencing a list would be equivalent to specifying its members individually, so still results in a well-defined ODD. However, this approach does have some disadvantages:

- A large number of lists may be required to cover all applications:
    - If these are maintained by a standards body or regulator, a significant amount of work may be required.

    - If ODD specification authors define their own lists, many of the readability benefits are lost. For example, a third party reviewing an ODD would not be able to rely on a list named “animals commonly found on UK roads” truly containing all relevant animals.
- Overlapping lists may make it easier to include or exclude an element by mistake. However, a language using them can still produce a well-defined ODD, provided that the semantics are defined carefully.

Restrictive and permissive ODD definition

We have identified two ways of specifying an ODD: restrictive and permissive. In both cases, an ODD ontology is required which lists the variables which may be included.

A restrictive ODD definition means that, if an item is not specifically listed, it is outside of the ODD. In other words, it lists the items which are allowed. In contrast, a permissive ODD lists the items which are not allowed.

Conditions may exist in the real world which are not modelled in the ODD ontology, and therefore cannot be represented in an ODD specification. With a restrictive ODD definition, it is clear that these are outside of the ODD, which is consistent with the SAE J3016 definition (since the ADS is unlikely to have been “specifically designed to function” with an unforeseen feature). If the ODD definition is permissive, it is unclear whether the element is allowed for the vehicle or not.

Element specifically listed?	Restrictive	Permissive
Yes	Element is inside ODD	Element is outside ODD
No, but element is represented in ontology	Element is outside ODD	Element is inside ODD
No, and element is not represented in ontology	Element is outside ODD	Undefined

Mixed permissive/restrictive ODDs

Wide ODDs are most conveniently expressed using restrictive language (e.g. “not dirt roads”), while narrow ODDs are best expressed with permissive language (e.g. “Motorways only”). For a hierarchical structure to be useful, it should be possible to use combined expressions (e.g. “Rural roads except motorways”): otherwise, all but the simplest of ODDs require every leaf item to be specified.

Searching test case metadata for a restrictive ODD

When searching for scenarios to match a restrictive ODD, the aim is to return scenarios which only contain elements permitted by the ODD. The search process should be:

- Take the complement, C_{ODD}, of the ODD definition to be searched (i.e. find every element which can be represented by the ODD ontology but is not listed in this definition)
- Construct a query: NOT the OR of all elements in C_{ODD}

Example
An ODD ontology allows the features [A … Z] to be expressed.

The ODD definition for an ADS allows the elements [C … Y]

This means that C_{ODD} = [A, B, Z]

The scenario database should be queried with:

NOT (**A** OR **B** OR **Z**)

Searching test case metadata for a permissive ODD

Searching a permissive ODD is slightly simpler, since the elements to be excluded are already defined. All elements in the ODD can directly be combined using AND and NOT.

Example
An ODD ontology allows the features [A … Z] to be expressed.

The ODD definition for an ADS prohibits the elements A, B and Z

The scenario database should again be queried with:

NOT (**A** OR **B** OR **Z**)

Searching test case metadata for a mixed ODD

Searching for a mixed ODD can be achieved by converting it to either permissive or restrictive first. The mixed ODD is still well defined, so the conversion process should be straightforward.

Safe versioning

An ODD ontology is likely to change over time. New versions may add, remove, move or modify elements. This introduces the possibility of several types of error:

1. An ODD definition no longer makes sense when applied to the new ontology. For example, it references a category which no longer exists. Input validation should be used to detect errors of this type.
2. An ODD definition may incorrectly include something which should be excluded. For example, if a new element is added to an ODD ontology, a permissive ODD definition allows it by default. This could also occur if the meaning of an element is changed (e.g. the scope of 'heavy rain' is expanded to include higher rates of rainfall).
3. An ODD definition may incorrectly exclude something which should be included. For example, a new element being added to an ODD ontology will cause it to be disallowed by a restrictive ODD definition.

Errors 2 and 3 will result in a valid (but wrong) ODD definition and have the potential to lead to a hazardous situation:

- Error 2 is problematic if the ODD definition is used to determine where and when a vehicle may be used. The ADS is will not have been designed to handle the additional situation which has accidently been permitted.
- Error 3 is problematic if the ODD definition is used to select test scenarios. The vehicle will not be tested against any scenario which contains the new feature, even if it may encounter it in reality.

This leads to the following suggestions:

- Any ODD description should be tied to a single version of the ODD ontology.

- Any systems which use the ODD description should check that it is valid for the ODD ontology version being used.
- A tool should be created to assist with migration between ODD ontology versions. It should make any changes in meaning clear to the user.

Error 2 may more easily occur in a permissive language, and error 3 in a restrictive one. Since either error can be problematic, choosing one over the other is unlikely to help prevent versioning errors.

Expressing complex criteria

So far, this document has focused on elements which are unconditionally inside or outside an ODD. However, in some cases, there might be a need to specify more complex conditions. There might use a combination of elements, or go beyond simply requiring that a feature is present or absent. Plausible examples of ADS restrictions include:

- Cannot make right turns at roundabouts
- Cannot overtake unless high-quality mapping data is available
- Cannot operate on roads with pedestrians or cyclists at night, unless the speed limit is less than 30 mph or street lighting is provided.
- Can only operate if at least one lane is greater than 3m wide

The more conditions of this type which are included, and the more complex they become, the harder the ODD is to understand. This could create a problem for regulators and safety drivers. However, this is not necessarily a reason to restrict the language: complex criteria may be essential to represent the capabilities of real systems, especially those still under development. We suggest that the language should (as far as possible) allow all reasonable ODDs to be expressed. Users are free to impose their own restrictions if required.

Searching test case metadata using complex criteria

Searching for combinations of elements requires an AND operator to be introduced into an ODD search query. For example, the first restriction above could be represented as:

NOT (Ego maneuver = “right turn” AND Junction type = “roundabout”)

This would allow scenarios which contain a right turn and scenarios which contain a roundabout but disallow those which contain both.

The example above works well for cases where scenarios (and associated road networks) are small and simple. However, it can result in false inclusions or rejections. In the example above, a scenario where the ego vehicle turns right at T-junction then left at a roundabout would be rejected, even though it was intended to be within the ODD. This is concerning, as it could lead to important tests being missed.

There are a few possible solutions to this problem, which include:

- Disallowing ODDs which are defined using a combination of features. ADS developers would probably see this as an unreasonable restriction.
- Allowing only a small number of combination conditions to be specified and giving these specific metadata labels. For example, “right turn at a roundabout” could be a specific maneuver. This is

slightly less restrictive than the approach above but could add complexity and still severely limits the flexibility of the language.

- Using metadata to describe only simple scenes from a scenario (e.g. a maximum of one junction per scene). It may be possible to automatically associate several of these to describe a complete scenario.
- Developing a searchable scenario and road network description language. This would associate relevant ODD ontology elements with specific scenario features. Current scenario languages are not designed to surface semantic concepts relevant to ODD descriptions. For example, an OpenDRIVE file could describe the layout of an urban road with a low speed limit but would not label it as a high street.

Summary of requirements

The requirements we have identified are driven by several high-level aims, balancing human readability and understanding with technical detail and accuracy. These aims are for a language that:

- Can specify simple ODDs concisely.
- Is powerful enough to express complex ODDs.
- Can cope with the underlying ontology of items not including everything in the real world (because it is not possible or desirable to model everything).
- Helps users to easily identify errors associated with version changes.

Requirement	Description
Tied to single ODD ontology revision	An ODD description needs to be tied to a single revision of the ODD ontology. Which revision is used may have implications for what should be tested, or is assumed to be tested.
Validity check	Any systems making use of an ODD description should check that it is valid and of the correct version
Migration tool	A migration tool should be created to help with converting ODD descriptions between ontology revisions, to reduce the risk of human error.
Hierarchy or defined lists	It should be possible to reference a group of elements without including them all individually. This may be achieved through a hierarchical structure or using predefined lists of elements (each has advantages and disadvantages)
Mixing restrictive and permissive elements	In order to make the definition readable, it will be necessary to use a combination of restrictive and permissive language elements (e.g. all roads except…).
Complex conditions and rich metadata	Real ODDs are likely to require ODD definitions which include complex conditions. Adding information to OpenSCENARIO/OpenDRIVE to make this searchable could be a feasible option. However, these conditions have the potential to make the ODD hard to understand.

5. Acknowledgements

Input from the MUSICC advisory group has greatly improved our understanding of the ODD description challenge. In particular, we would like to thank the Met Office, Jaguar Land Rover, FiveAI, and Oxbotica for their willingness to discuss, comment on and provide feedback about our proposals.

6. Bibliography

Shalev-Shwartz, S., Shammah, S., & Shashua, A. (2017, August 21). *On a Formal Model of Safe and Scalable Self-driving Cars*. (Mobileye) doi:arXiv:1708.06374 [cs.RO]

Society of Automobile Engineers. (2018, June 6). Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles. *J3016*. Society of Automobile Engineers.

Appendix A: MUSICC Metadata

General properties

- Description: free-text description of the scenario
- CountryCode
- DriveOnRightOrLeft
- UseCase: Highway, Urban, Inter-Urban or Parking
- ScenarioType: Logical or Concrete
- Exposure, which represents how frequently this scenario is expected to occur
- SituationDemand, which is an indication of the complexity of a scenario, and therefore the demand it places on the driver (whether ADS or human; although SituationDemand may be slightly different between the two)
- CollisionCategory: NormalDriving, Collision or NearCollision
- InitialSpeedLimit, the speed limit at the start location for the ego vehicle (noting the speed limit may change as the scenario unfolds): SlowUrban, Urban, FastUrban, TrunkRoad, or Highway.
- RealWorldMap, true if the OpenDRIVE file corresponds to a real-world location
- RealWorldCoordinate, latitude and longitude for the origin of the OpenDRIVE map
- RepresentsADASTest, true if this scenario is a tightly scripted ADAS test procedure (c.f. NCAP testing)
- ADASFeaturesTested, a list of the ADAS feature(s) being tested (if RepresentsADASTest is true)
- SceneGraphModelDetail, which provides metadata for scenery files.
- SceneGraphModelSensorRealism, which allows labelling with intended use cases of the scenery files (testing of camera, LIDAR and RADAR perception).

For speed limits (and traffic density and average speed, defined in the following section), we have collected values into named bins, rather than using the raw numerical value. This ensures clarity on membership of scenario sets, makes it easier for end-users to select scenarios for a use case, and also avoids ambiguities with speed limits in mph in some territories and km/h in others.

Core ODD specification

The core ODD is summarised in seven variables:

- Features of the road (e.g. 2-lane, divided carriageway)
- Environmental conditions (weather, time-of-day)
- Intended ego vehicle category (e.g. M1 - passenger car)
- Primary ego vehicle maneuver (e.g. turn right)
- Any other key actors (e.g. a pedestrian or an emergency vehicle)
- The action performed by the other actor (e.g. cross the road, emergency stop)
- Whether any of the key actors break rules, such as road traffic laws or advice to drivers
- The traffic density and average speed, for the busiest section of road the ego vehicle must navigate during the scenario

The intention is to keep these as simple and generic as possible, while still allowing queries that correspond at a high level to an ODD. Multiple values can be specified for each of the variables to increase the expressive power.

Note that, for the road features in particular but also for others, the intention is *not* to simply record every feature present in the scenario's OpenSCENARIO/DRIVE record. While it may be possible in the future to automate the extraction of such information, it will not capture which of these features are important to the scenario, and which are incidental background. For example, a scenario may incorporate a bicycle, but the key aspect is to test the SUT when a car in front performs an unexpected emergency stop. In that case, the KeyActorTypes field would be expected to include PassengerCar, but not PedalCycle.

Scenario admin data

- SourceOrganization, the organisation that contributed the scenario
- OwningOrganization, the organisation that owns the IP of the scenario
- Information from the FileHeader of the MUSICC record:
    - label, a short text description
    - version, a simple integer. Label+version+revision form a unique key for a MUSICC scenario
    - updateDateTime
    - updateUsername
    - revision, the MUSICC SDL revision that the scenario conforms to
- Global tags, which allow arbitrary labels (e.g. test set identity) to be assigned to scenarios
- Regulation tags, which will label any scenarios referred to by a regulation.

Custom data

A CustomMetadata element allows arbitrary name-value pairs to be included in the metadata. The hope is that, if certain custom metadata fields are found frequently in incoming scenarios, they can be migrated to a properly validated schema element in a future revision of the MUSICC SDL.

1 Sekforde Street
Clerkenwell
London
EC1R 0BE
Tel: 020 7952 5111

The Pinnacle
170 Midsummer Boulevard
Milton Keynes
MK9 1BP
Tel: 01908 359 999

CATAPULT
Connected Places