IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 21 November 2023, accepted 26 December 2023, date of publication 5 January 2024, date of current version 23 January 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3350512

APPLIED RESEARCH

ODD and Behavior Based Scenario Generation for Automated Driving Systems

XIZHE ZHANG[1], SIDDARTHA KHASTGIR[1], JUSTIN-KIYOSHI TIELE[2], KAZUHITO TAKENAKA[2], TASUKU HAYAKAWA[3], AND PAUL JENNINGS[1]
[1]WMG, University of Warwick, CV4 7AL Coventry, U.K.
[2]Corporate R&D Unit, DENSO AUTOMOTIVE Deutschland GmbH, 85386 Eching, Germany
[3]DENSO CORPORATION, Global R&D Tokyo Haneda, Ota-ku, Tokyo 144-0041, Japan

Corresponding author: Xizhe Zhang (Jason.Zhang@warwick.ac.uk)

This work was supported in part by the UKRI (UK Research and Innovation) Future Leaders Fellowship under Grant MR/S035176/1, and in part by the Department of Transport, U.K., and Transport Canada under Grant T8080-220112.

ABSTRACT Automated Driving Systems (ADSs) bring many benefits, however their safety assurance poses challenges. Scenario-based testing has been proposed. To provide strong safety evidence for their safety assurance, the scenario-based testing process needs to consider the Operational Design Domain (ODD) of the systems, ODD combined with the behavioural elements can provide a foundation for the scenario generation workflow due to common domain elements between ODDs and scenarios. Based on such background, this paper introduces a novel framework to generate scenarios specifically target on testing the system's claimed ODD. It includes the process going from the system's ODD and behaviour competency, to logical scenarios generation utilising scenario construct rule sets, and to the concretisation of the logical scenarios into concrete scenarios. This paper also draws link towards the part II of the paper series, which illustrates a novel approach for scenario coverage analysis.

INDEX TERMS Automated driving systems (ADSs), scenario-based testing, safety, safety assurance, V&V, operational design domain (ODD), behavior, scenario generation, coverage.

I. INTRODUCTION

A. BENEFITS AND CHALLENGES OF ADSS

Despite the various benefits ADSs bring [1], [2], [3], [4], the complexity associated with such systems has raised challenges for their safety assurance. Traditionally, distance-based metrics have been used to provide safety evidences for the systems, however it was suggested that ADSs need to be driven 11 billions miles to prove that they are 20% safer than human drivers [5]. Translating this into the time scale resulted into more than 500 years for a fleet of 100 ADSs driving 24 hours a day and 365 days a year at 25 miles per hour (mph). This has led to the recent move within the industry and academia towards a scenario-based testing approach [6], [7], [8], within such approach purposely designed scenarios are used to test various aspects of the system, which focuses on the 'quality' of the miles tested rather than the 'quantity'.

The associate editor coordinating the review of this manuscript and approving it for publication was Junho Hong.

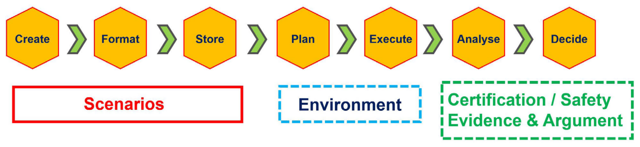


FIGURE 1. Scenario-based evaluation continuum with its sub components.

B. SCENARIO BASED TESTING WORKFLOW

A scenario-based safety assurance workflow was previously documented in [6], within such framework lays three key components: *Scenario*, *Environment*, and *Safety Evidence*, as illustrated in Figure 1. *Scenario* further covers *Create*, *Format*, and *Store*. During the *Create* stage, the relevant scenario content is generated using different scenario generation methods [9], [10], [11], which can be categorised into data-based approach and knowledge-based approach [6], [11]. *Format* then represents the scenario content in a specific formalised format, which needs to be both machine readable as well as human readable [12], [13], [14], [15] in order to cover the complete V model for system development and testing. *Store* includes the storage and organisation of the generated scenarios within an adequate scenario database, as well as the retrieval of them. For example, the Safety Pool^{TM} scenario database [16] has been incorporated within [6], it utilises an ODD & behaviour based tagging and query mechanism [17]. *Plan* and *Execute* determine the execution environment for the scenarios and perform the execution. The scenario execution data then feeds into the *Analyse* stage where a set of pass/fail criteria will be compared against. In addition, within the *Analyse* stage, the results are fed into optimisation algorithms where the concrete scenario values (correlated to the associated logical scenario) for the next iterations can be generated. Such optimisation algorithms optimise the scenario parameters towards potential system failure conditions, therefore it forms the hazard based testing concept [18]. The feedback loop formed between *Execute* and *Analyse* will continue explore the parameter spaces until a threshold of iteration number is reached. At the final step, the *Decide* will provide the outcome of the whole process.

C. UNDERPINNED BY ODD

Please note that the scenario-based testing workflow needs to be underpinned by the system's ODD during safety assurance processes [6]. A widely used definition of ODD was introduced in SAE J3016 as follow [19]:

> *"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"*

Based on the definition of an ODD, being able to navigate safely within its claimed ODD is crucial for the subject system. The ODD of a system is commonly visualised mentally as an 'ODD boundary', individual scenarios within, at, or outside the ODD boundary can then be explored through the scenario-based testing workflow. Since the scope of ODD does not cover the behavioural elements, the ODD elements need to be used in conjunction with the behavioural elements to cover the scope defined in a scenario (which will be discussed further in Section II). Behaviours include standard manoeuvres such as lane change, turn left/right, as well as other types of behaviours such as V2X communication, and signalling. In here, the word "scope" refers to the domain composition, i.e., those attributes that form the underlying domain model. To further illustrate, Figure 2 displays the domain decomposition for scenarios and ODDs. Please note, within the scope of ODD the *Dynamic element* covers the types of dynamic agent and the subject vehicle's maximum designed operating speed. Examples of the dynamic agents are pedestrian, cyclist, car, bus, etc. Based on the exact application, user may extend such attribute tree further.

Even with a fleet of vehicles driving 500 years using a distance based approach, if lacking the consideration of the system's ODD the miles are meaningless (i.e., testing system for residential usage on a motorway). Scenarios and ODD put the boundary of the testing scope into a much smaller, focused, while effective range. Within or on such ODD boundary, various coverage concepts can be used to provide arguments on the coverage of the scenario based testing process. Paper II of this paper series will introduce a novel and effective coverage concept. Combining scenario-based testing with ODD and virtual test environment offers testing efficiency and scalability. Based on the testing objective and target system/subsystem, the required level of fidelity from the simulation will be different, this allows the effective allocation of computational resource for the right level of testing.

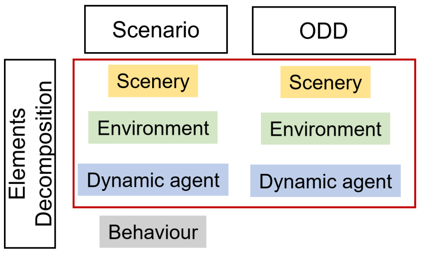


FIGURE 2. **Domain decomposition between scenario and ODD.**

D. KNOWN/UNKONWN, SAFE/UNSAFE

Based on this established relation between ODDs and scenarios, the main objective of scenario-based testing activities is to provide safety evidence within a system's claimed ODD by exploring relevant scenarios. The international homologation and public deployment of ADS are subject to regulatory frameworks that may explicitly reference several industry standards as acceptable practices of safety argumentation, such as [20] and [21]. One such standard is the ISO 21448 [22], which focuses on ensuring the safety of the intended functionality ("SOTIF"). The perspective of the known/unknown and the safe/unsafe scenarios is introduced as a way to demonstrate the absence of unreasonable risk due to the hazards resulting from the ADS (Figures 3).

As can be seen in the upper section of Figure 3, there are four distinctive quadrants indicating: 1) the known unsafe scenarios (Area 2), the known safe scenarios (Area 1), the known unknowns (Area 3), and the unknown unknowns (Area 4). Scenario-based testing allows us to increase Areas 1 and 2 by reducing Areas 3 and 4. Mitigation actions then convert Area 2 elements into Area 1 elements (examples include continuous improvement system development, or ODD exclusions). To argue the systematic minimization of the unknown areas, evidence towards the maximization of the known areas must be offered. Since these mechanisms can be described in terms of scenarios, the orthogonal ODD framework offers a way to express the relative increase and

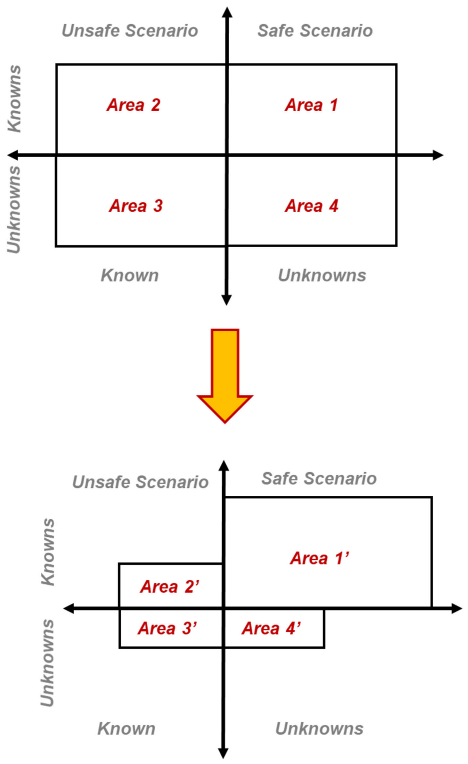


FIGURE 3. Changes of unknowns, knowns, unsafe and safe scenarios via safety assurance activities.

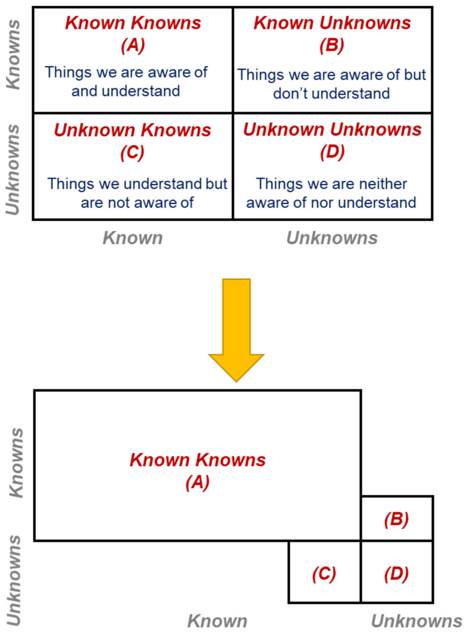

Known Knowns (A)
Things we are aware of and understand
Known Unknowns (B)
Things we are aware of but don't understand
Unknown Knowns (C)
Things we understand but are not aware of
Unknown Unknowns (D)
Things we are neither aware of nor understand
Knowns
Unknowns
Known
Unknowns
(B)
(C)
(D)

FIGURE 4. Changes of unknowns and knowns via safety assurance activities.

decrease of the areas in question (i.e., the "knowing the space").

Visualise Figure 3 from another perspective results into Figure 4, which focuses solely on the changes of the known and unknown quadrants via scenario-based testing activities underpinned by ODD. As can be seen, by increasing Area 1 and Area 2 through scenario-based testing process in Figure 3, the 'Known Knowns' area (A) in Figure 4 expands, whereas the 'Known Unknowns' (B), the 'Unknown Knowns' (C), and the 'Unknown Unknowns' (D) shrink. The 'Known Knowns' represents things we are aware of and understand, the 'Known Unknowns' represents things we are aware of but do not understand, the 'Unknown knowns' represents things we understand but are not aware of, and 'Unknowns Unknowns' represents things we are neither aware of nor understand.

E. OBJECTIVES OF THIS PAPER

A "quantity of miles" distance-based testing approach is unfeasible for tackling problems such as managing the long-tail distribution [23] of hazards that ADS can encounter in the real-world (rare events/surprise "black swans"), especially when considering that the entire effort must be at least partially repeated whenever changes are made to the ADS within development phases. The "quality" of miles offered by a scenario-based testing approach lies in the ability to systematically define, discover, and reduce the overall amount the test miles required to verify safety relevant variables. Possible strategies to achieve this are offered by relevant process frameworks, an example of which is introduced in Section I-D (SOTIF known-unknown/safe-unsafe scenarios). While the industry is increasingly adopting the approach that verification of scenario tests provides answers to the question of if the system complies with the intended specifications, the additional consideration that an underpinning ODD framework facilitates the validation that those specifications themselves are adequate (relevance of the question being answered), is less established. ODD is the key reference to relate the scenario-based test efforts to coverage goals regarding the SOTIF quadrants and ODD spaces (including boundary conditions; to be explored in Paper 2 of this series).

Based on the understanding of scenarios, ODDs, scenario-based testing, and safety assurance, the high-level objective of this work is to introduce an approach to scenario generation and coverage metrics that supports a SOTIF argumentation. Two separate papers will be introduced focusing on the two main areas: 1) scenario generation, and 2) evaluation framework focuses on coverage analysis. Figure 5 displays the relations between paper I (top section of the figure) and paper II (bottom boxed area of the figure).

It can be seen that this paper (paper I) covers a workflow starts with the Operational Domain (OD), the ODD specification of the system, and the behaviour competency at the design phase. At the scenario definition phase, different rule sets are used to construct logical scenarios. And at the scenario execution phase, concrete and executable scenarios are derived from the logical scenarios. Each of the phases carried out during paper I are then mapped to the corresponding evaluation pillars which will be introduced in paper II. At a

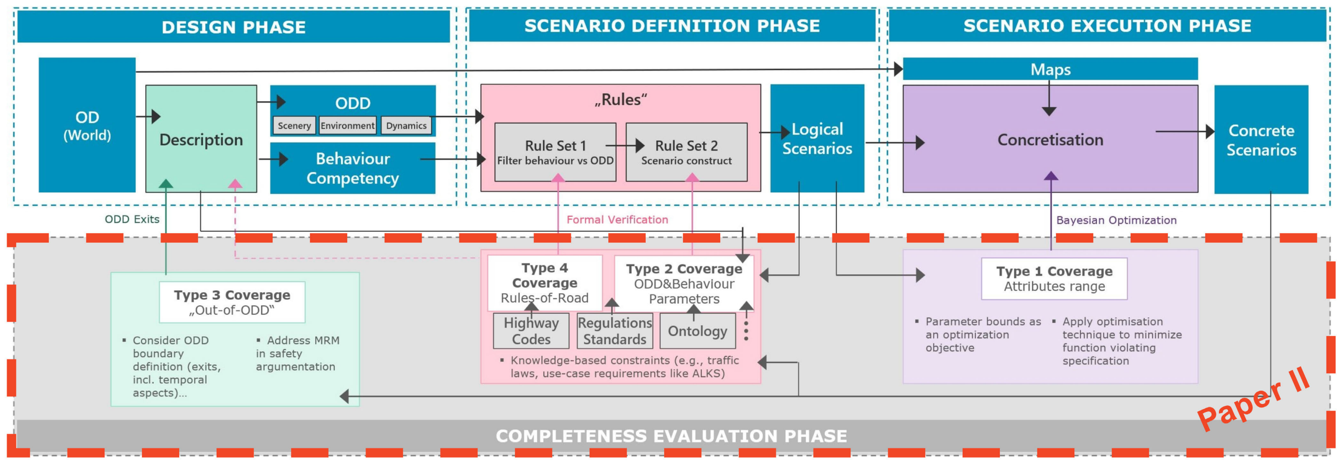


FIGURE 5. Correlation between scopes being covered in paper I and paper II of the series.

high level, they contain the *Type 1 coverage* focusing on the attributes range level, *Type 2 coverage* focusing on the ODD and behaviour, *Type 3 coverage* focusing on the out-of-ODD scenarios, and *Type 4 coverage* focusing on the rules of the road.

II. RELATED WORK

A. SCENARIO

Before diving into the scenario generation method, it is important to review the definition of scenario, and understand what is included in a scenario. One of the frequently cited definitions of a scenario can be found in a publication from Ulbrich et al. [24], it states:

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

In addition to the temporal nature of scenarios, convergence can also be seen across the industry and academia as to what is contained within a scenario. For example, the ASAM OpenDRIVE standard [15] describes the road and junction network, the road structures, and other objects. Whereas the ASAM OpenSCENARIO 1.x [14] describes the dynamic scenario agents and their corresponding behaviours, it references an OpenDRIVE file to locate the scenario location. The OpenDRIVE and OpenSCENARIO 1.x pair is commonly used to describe a scenario, and it enjoys a wide toolchain support across different simulators. Another recent publication [12] introduces a two level abstraction scenario description language, which is both human readable and machine readable. Underneath the syntax, it incorporates a domain model which covers the scenery aspects such as road topology, junction layout, and other road structures; as well as the environmental conditions and the dynamic behaviour of the scenario agents. The recent release of the BSI Flex 1889 [25] on the structured natural language based scenario description format, which received contributions from [12] and [13], also incorporates similar domain compositions, i.e., scenery, environmental conditions, and dynamic elements. This high level domain model breakdown of a scenario has already been summarised in Figure 2, this common domain compositions between scenarios and ODDs provides an opportunities for scenario generation based on ODD, and scenario coverage analysis.

B. SCENARIO CREATION

Several scenario creation methods have been summarised by Zhang et al. [6]. As shown in Figure 6, they include accident data, insurance claim/ telematics, system analytics, formal verification, operational design domain, ontology, standard/ regulation/ guidelines, real-world deployment and trials. However, at a high level they can all be categorised into data-based approach, and knowledge-based approach [6], [26], [27]. A knowledge-based scenario generation approach utilises domain specific knowledge to identify hazardous events systematically and creates scenarios. A data-based approach utilises the available data to identify and classify occurring scenarios. Please note that different scenario generation approaches focus on different perspectives, to gain a comprehensive scenario-based testing outcome a test suite of scenarios derived from multiple sources are needed.

From all the listed scenario generation methods, accident data derived scenarios focus on the causes of known accidents, such accident records are based on human drivers' behaviour rather than ADSs'. Esenturk et al. [10], [28] performed clustering analysis based on the UK's STATS 19 public accident database, which contains information about non-ADS incidents, to derive clusters that represent accident trends (certain combinations of scenario parameters). The three high level steps taken are: 1) data pre-processing, 2) data clustering, 3) understanding the clusters. The STATS 19 data in its raw form describes each accident using both common attributes (such as weather condition,

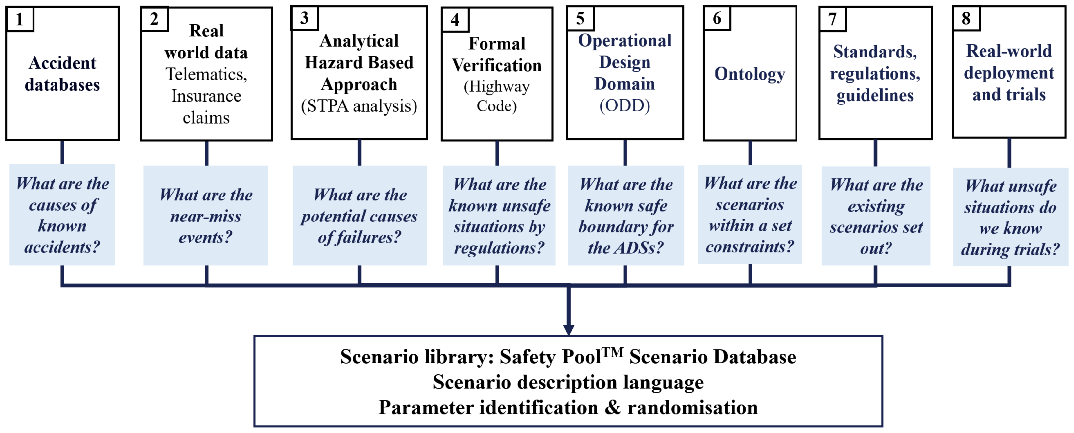

1
Accident databases
2
Real world data
Telematics, Insurance claims
3
Analytical Hazard Based Approach
(STPA analysis)
4
Formal Verification
(Highway Code)
5
Operational Design Domain
(ODD)
6
Ontology
7
Standards, regulations, guidelines
8
Real-world deployment and trials
What are the causes of known accidents?
What are the near-miss events?
What are the potential causes of failures?
What are the known unsafe situations by regulations?
What are the known safe boundary for the ADSs?
What are the scenarios within a set constraints?
What are the existing scenarios set out?
What unsafe situations do we know during trials?
Scenario library: Safety Pool™ Scenario Database
Scenario description language
Parameter identification & randomisation

FIGURE 6. Eight different scenario creation methods [6].

light condition), and specific attributes (such as gender of the driver, vehicle type). During the pre-processing, attributes that are considered unimportant for the scenario content are disregarded, such as local authority district, police attendance, effect of cultural origin. During the clustering stage, categorical clustering algorithm that seeks to minimise the overall entropy is used, where the entropy in a cluster is quantified using normalised frequencies of the attributes within the cluster, treating each attribute independently from each other. The algorithm begins by forming cluster seeds that contain the most different elements from each other in the dataset. The remaining data points are then assigned to the seed clusters one by one according to the reduction in the entropy. Upon the determination of the clusters, two steps are taken to understand the cluster meaning: 1) frequency analysis, and 2) market basket analysis. Frequency analysis is applied within each cluster and determines what the significantly frequent attributes within the cluster are in comparison to the rest of the data. To determine whether an attribute is significant, the expected frequency number of a particular attribute within a cluster is first calculated. The significance of an attribute is then determined by whether the frequency of an attribute observed within a cluster exceeds the expected frequency by a threshold. Market Basket Analysis, see (MBA) [29], is a methodology that has been mainly used in the analysis of business transactional data to identify relations between data (e.g., which products are often found together in the data), this is done by finding the association rules between attributes. The first step of the MBA is to identify frequent itemset using an itemset mining algorithm to produce itemset such as {motorbike, entering junction, turning left}. The association rules are then established to form the relation of antecedent (A) –> consequent (C) within the itemset. Two metrics, the lift and the confidence, are used when identifying the rules. Confidence is measuring how often A and C appear together as compared to only A appearing alone. Lift is a comparison between how often A and C actually appear together, with how often A and C would be expected to appear together if they were independent. By performing this list of steps, the accident clusters, as well as the association rules can be established to derive new scenarios that represent the underlying data. For the analysis of insurance claim data [6], [30], similar approach can also be used as the accident data analysis.

On the other hand, the System Theoretic Process Analysis (STPA) [9] is a top-down approach based on the conceptual accident causality model called System Theoretic Accident Model and Processes (STAMP). It treats accidents as a control problem instead of a failure problem and it prevents accidents by enforcing constraints on the behaviour of the system. STPA considers a diverse range of causal factors of hazardous interactions, including flawed control algorithms due to flaws in their requirements, communication errors, and delays, conflicted controls, processing delays, misinterpretations of the received data or signals, etc. A five-step process for deriving scenarios using STPA include: 1) define the purpose of the analysis, 2) model the control structure, 3) identify Unsafe Control Actions (UCAs), 4) identify loss scenarios, 5) identify testing parameters. As a deductive approach, STPA starts by defining the purposes of the analysis, including defining the losses, the system boundary and identifying the system-level hazards. The definition of the ODD of the system is required as an input to identify the possible interactions between the system and its outer world. Step 2 then creates a control structure based on the available information about the system, including the system description, system architecture, etc. A control structure consists of hierarchical functional blocks that link to each other as part of feedback control loops. Each control loop consists of a controller that provides control actions (CA) to control some processes and enforce constraints on the behaviour of the controlled process. After identifying CA in the control structure, each CA is further analysed to identify how the CA would manifest into a UCA. In certain circumstances, a correct CA could lead to one or multiple system-level hazards. To identify a UCA, the CA is usually considered together with a particular context and using guiding principles of the STPA method. Once all of the UCAs have been identified, the causes of each UCA are analysed - i.e., their loss scenarios. As an extension to the STPA analysis, Chen et al. [9] further added the step for identification of testing parameters based on the output loss scenarios.

Onto the ontology-based scenario generation, Bagschik et al. [26] initially used an ontology based approach to create a scene, as a snapshot of an underlying scenario. The ontology they used describes concepts and relations between driving context, task representation, (simulative) actions, simulation monitoring, and temporal representations between entities in the simulation. Providing a set of initial 'keywords' input, the ontology framework can construct valid scenes that satisfy the 'keywords', however this work does not extend into full scenarios (i.e., including temporal and behavioural aspects). The ASAM OpenXOntology [31] is a standardisation project that focuses on developing an ontology, covering both spatial and temporal aspects, for the automated driving domain. As part of the use case demonstration, it incorporated multiple inferences of scenario correctness based on a set of given input and the constructed ontology rules. Bruto da Costa et al. [32] leveraged ontologies to represent objects and their relationships in a scenario, while target a set of formalised rules of the road (e.g., the UK Highway Code [20]) for the scenario generation and testing of ADSs, this approach combines formal verification, ontology, as well as regulations as shown on Figure 6. The formalisation of the rules of the road, and the scenario generation against such formalised rules are two distinctive stages in their study. Within the formalisation of the rules of the road, a three-step approach was documented: 1) construct a concept and predicate vocabulary, 2) reduce the rule to its minimal form, and 3) express rule in first-order logic. Once the rules are formalised, they incorporated a scenario generation pipeline involving six sub elements, as shown in Figure 7. As can be seen, the scenario builder starts the initial set of instructions based on the selected rules. It then instructs the ontology & rule manager to prepare the ontology specifically for scenario generation (step 1) and specifies the rules for which scenarios need to be generated (step 2). Meanwhile, the ontology & rule manager configures the knowledge base (KB) according to the selected rules (step 3), this includes initialisation of objects, setting their attributes and relationship. The reasoner then synchronises the ontology with the newly added assertions (step 4) based on the rules, which consequently update the knowledge base (KB) in step 5. Once the KB is updated, the Scenario Builder proceeds to retrieve scenarios from the Scenario Attribute Manager (steps 6-11). Then, the Scenario Attribute Manager interacts with the Ontology to Attribute Mapping component (steps 7-10) to transform the current state of the ontology into abstract scenarios. The Scenario Attribute Manager further refines the abstract scenarios to create a set of logical scenarios (step 11).

In addition to the methods discussed so far, other sources of regulation, guidelines or standards can also be used for scenario generation. For example, the Safety Pool^{TM} scenario database [16] contains scenarios generated based on the Euro NCAP testing protocols [33], based on the low-speed automated driving standard (ISO22737) [34], as well as based on the UN regulation No. 157 on Automated Lane Keeping Systems [21]. Although various scenario generation methods have been discussed, each of them focuses on a different perspective. Accident data focuses on the cause of known accidents (e.g., from human driver perspective), insurance claim further contains the near-miss data which may not be classified as accidents. STPA based approach focuses on the potential causes of failures from the system architecture perspective, rules of the road combined with ontology focuses on the known unsafe situations from regulations under valid parameter constraints. Regulations, standards, guidelines provide existing available scenarios complies with the testing protocols, and real world deployment data focuses on the unforeseen data captured during deployment stage which can be used to further populated scenario test suite for future testing. The current gap within the scenario generation approaches is the lack of emphasis on the ODD, since ODD is the designed safe operating conditions for the system, it is crucial to obtain a set of scenarios that explore the system's ODD. This is also the main motivation of this research project.

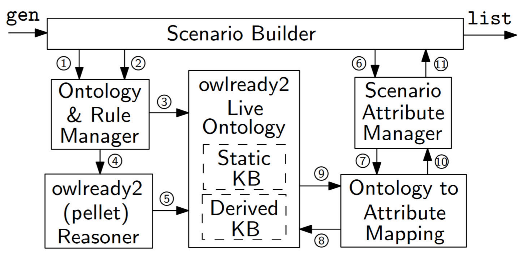


FIGURE 7. Scenario generation from Rules of the Road: Methodology Overview [32].

C. SCENARIO FORMATS TRANSLATION

A previous publication [35] documented an automation process to convert scenarios described at the logical scenario level using the two-level scenario description language (SDL) [12] to the corresponding ASAM OpenX format, i.e., OpenSCENARIO 1.x [14] & OpenDRIVE [15] at the concrete scenario level. The SDL level 1 aligns with the BSI Flex 1889 standard on natural language based scenario description format, and SDL level 2 is fitted at the more concrete scenario abstraction levels (logical & concrete), SDL level 1 & level 2 can be converted by adding or subtracting information. The importance of having different scenario formats target at different scenario abstraction levels, and having a seamless conversion process is that they allow a complete scenario coverage over the V model for system development and testing, and provide traceability between scenario abstraction levels. The scenario abstraction levels mapped to the V model is illustrated in Figure 8 [13].

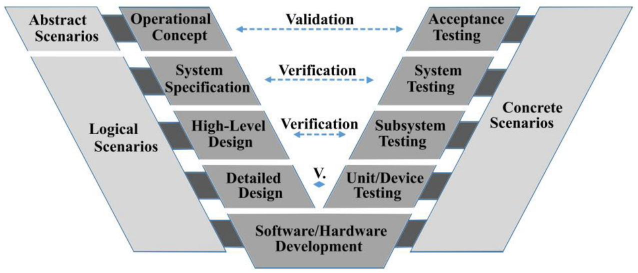


FIGURE 8. Scenario abstraction levels mapped to the V model for system development and testing [13].

III. ODD AND BEHAVIOR BASED SCENARIO GENERATION

The goal of the ODD & behaviour based scenario generation is to generate scenarios based on a given set of ODD input together with the behaviour capabilities of the system. For systems designed to operate on a fixed route but without explicitly defined ODDs, an additional step prior to the scenario generation process is needed to extract ODD features from the target route.

Figure 9 illustrates the overall process which will be expanded within this section. It can be seen that at the top of the workflow, there is an optional input (in dashed box) which is the route input. From the input route, the corresponding ODD features can be extracted and form the system's ODD specification; alternatively, if a system does not have a fixed route, the system's ODD specification can be directly used. The ODD specification defines the system's operating boundary, within such boundary lies all the ODD attributes the system is capable of handling, such as T-Junction, Motorway, Roundabout etc. The next step will select a set of non-competing (e.g., minor roundabout and motorway are competing attributes) ODD attributes within the boundary for creating the scenery aspect of a potential scenario. The behaviour library is another input into the workflow, it represents all the behaviours the system can perform. Based on the selected set of ODD attributes, the behaviours within the library can be filtered, for example minor road with a single lane can filter out lane change right/left behaviours. Upon this stage, a set of compatible ODD and behaviour 'keywords' (attributes) can be obtained, which at an abstract level represents a set of underlying logical and concrete scenarios. To ensure the correct instantiations of fully defined logical scenarios, including the correct scenery layout, dynamic behaviours and environmental conditions, a set of scenario construction rules are defined to detail the compatible ODD and behaviour keywords. Such detailing process adds the necessary missing information such as road connecting angles, traffic participants relative positions. All the keywords together with the added information are then formatted using different scenario description formats, and consequently stored within a database for further testing activities.

The above steps can be summarised as follows:

- Step 1 - Extract from route/ or obtain an ODD specification of the system
- Step 2 - Obtain the behaviour library for the actors
- Step 3 - Filter for applicable behaviours using the individual ODD elements
- Step 4 - Construct individual and valid logical scenarios using scenario construct rules
- Step 5 - Concretisation and conversion of the logical scenarios into the ASAM OpenX [14], [15] format

Section IV below will incorporate an example to demonstrate the various steps.

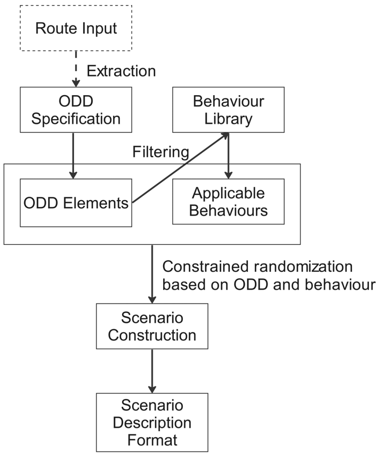


FIGURE 9. High-level workflow for logical scenario generation based on ODD.

IV. SCENARIO GENERATION USE CASE: ADS OPERATING IN A CONFINED AREA

A. PROBLEM STATEMENT

As part of the demonstration, the scenario generation approach was applied for an ADS operating in a confined area. In this section, a walk through will be provided to illustrate the process. For this particular use case, the input is the route detail of the confined area and the system's behaviour competency. The desired outputs are: 1) a set of logical scenarios, described in SDL level 2 format of the two-level abstraction language [12], generated based on the relevant ODD of the system; and 2) the auto-converted OpenSCENARIO 1.x [14] and OpenDRIVE [15] files of the logical scenarios. Please note for confidential reasons, the actual route details and system's behaviour competency have been changed for illustration purposes only.

B. STEP 1 - CONSTRUCTING THE ODD DESCRIPTION

Figure 10 illustrates the operating route of the ADS, as can be seen it consists of: a T-Junction (TJ1) connects Road 1 (R1), Road 2 (R2) and Road 3 (R3); a four-way roundabout (RA1) connects Road 3 (R3), Road 4 (R4), Road 5 (R5), and Road 6 (R6). Among the roads, R3 also has trees and buildings nearby.

All the extracted ODD elements can be directly mapped to the BSI PAS 1883 [36], and the ISO 34503 ODD standards [37]. In addition, for environmental conditions

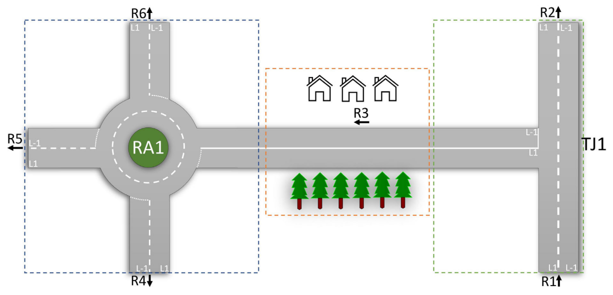


FIGURE 10. Example operating route of the ADS.

(which is not shown directly on Figure 10) such as rainfall, the system can only operates under non raining conditions. Using the ODD language format published in [37], [38], and [39], the system's ODD specification can be described using the ODD description as shown in Figure 11.

```
Include: PAS 1883:2020 Operational Design Domain (ODD) taxonomy
for an automated driving system (ADS) - Specification [2020,
https://www.bsigroup.com/en-GB/CAV/pas-1883/]
Base state: Restrictive
Extension: None

#ODD statements
Suitable Drivable area type is [Minor roads, Distributor roads]
Suitable Number of lanes is [less or equal than 2]
Suitable Lane dimension is [greater than 3.2]
Suitable Lane type is [Traffic lane]
Suitable Road surface is [Uniform]
Suitable Direction of travel is [Left-handed]
Suitable Fixed road structure is [Buildings, Trees]
Suitable Lane marking type is [Broken line, Solid line]
Suitable Intersection is [T-junction, Roundabout]
Unsuitable Weather is [Rainfall]
Suitable Illumination is [Day]
Suitable Dynamic agent type is [Vehicles]
```

FIGURE 11. ODD specification of the system under test.

From the "Include" statement, it can be seen that the baseline taxonomy for constructing this ODD description is based on the BSI PAS 1883 ODD [36] taxonomy for ADSs. Within the "Base state", it is divided into "Permissive" and "Restrictive" state. The "Permissive" state within the base state indicates that if none of the classes that belong to the same parent class attribute are mentioned in the ODD description, they are permitted by the ODD. On the contrary, under "Restrictive" state, if no statements are given for a parent class, then none of the children classes are allowed in the ODD. Both states are used to handle the ODD attributes that are not explicitly mentioned with the ODD description. For example, if an ODD only mentions the suitability of motorway and nothing else. Under the "permissive" state, it means the system can handle everything else, such as rain, snow, urban roads etc. Under the "restrictive" state, that would mean none of the other ODD elements are suitable. The rest of the ODD description is self explanatory as the format is designed to be human readable, for further information, please refer to the publications [37], [38], [39] of the ODD language format.

C. STEP 2 - OBTAINING THE BEHAVIOUR LIST

As mentioned earlier, the scope of ODD does not cover the behaviour aspect. To generate scenarios using ODDs, the complete behaviour list for the system and for the other traffic participants needs to be defined. The baseline for building the behaviour list took the foundation of the SDL level 2 manoeuvres [12], as well as the ASAM OpenXOntology concept project [31], which uses both absolute manoeuvres and relative manoeuvres as displayed in Table 1. Absolute manoeuvres are manoeuvres that can be performed by a single participant, without the need for a second reference participant; whereas relative manoeuvres reference towards a secondary participant. Absolute manoeuvres include *Drive*, *Lane change right*, *Lane change left*, *Turn left*, *Turn right*, *Stop*, *Reverse*, *Run*, *Slide*, *Walk*; and the relative manoeuvres include *Cut-in*, *Cut-out*, *Move towards*, *Move away*, *Cross*. Among the absolute manoeuvres, *Run*, *Slide*, and *Walk* are unique for describing the behaviours of human participants.

TABLE 1. Categorising behaviours into relative behaviours and absolute behaviours.

Absolute Behaviours	Relative Behaviours
Drive	Cut-in
Lane change right	Cut-out
Lane change left	Move towards
Turn left	Move away
Turn right	Cross
Stop	
Reverse	
Run	
Slide	
Walk	

D. STEP 3 - BEHAVIOUR FILTERING BASED ON ODD

Once the ODD description of the system is obtained based on the route input, together with additional input from the system's specification (e.g., ODD weather conditions), the next step is to perform sampling of the ODD attributes within the ODD boundary, and identify the suitable behaviours that can be performed over the sampled ODD attributes. Such ODD sampling can be divided into two different situations: 1) when there is a pre-defined fixed route, 2) when there is no fixed route defined, but the system ODD specification is available.

For situation 1, since the route for the system is pre-determined, the scenery aspect of the scenario is already available. Hence only the dynamic aspects and the environmental conditions of scenarios need to be generated, they can then be overlaid on top of the scenery. To achieve this, the route is divided into smaller subsections where ODD features within each subsection share similarities, this process forms three different scenery descriptions. As can be seen in Figure 10 indicated by the dashed boxes, *scenery 1* contains a four way roundabout, *scenery 2* contains a single straight road with houses and trees on both sides, and *scenery 3* contains a T-Junction.

For situation 2, auto-generation of the scenery aspect of scenarios is required as there is no pre-defined scenery. Such process uses similar keyword-based detailing process incorporating constraints and rules, which will be covered in 'Step 4'. For this current use case, the scenery generation is not required, hence only a brief overview will be provided

TABLE 2. ODD based behaviour filtering illustration.

Sampled ODD features	Absolute behaviour list
Straight	Drive ✓
Trees	Lane change right ✗
Buildings	Lane change left ✗
Solid line	Turn left ✗
Number of lane: 2	Turn right ✗
Agent type: vehicle	Stop✓
	Reverse✓
	Run ✗
	Slide ✗
	Walk ✗

in ‘Step 4’. The generation of scenery will be carried out using a subset of the ODD elements within the system’s ODD specification, obtained using a sampling process. This subset of ODD elements will then be used to constrain the allowed behaviour, and to create a scenario scenery.

The common part between both situation 1 & 2 is the ODD based behaviour filtering. Table 2 illustrate an example of filtering a behaviour list based on sampled ODD features. As shown, the sample ODD features represents the scenery 2 (single straight road) within Figure 10, they include straight geometry, trees, buildings, solid line marking, number of lane: 2, agent type: vehicle. On the other hand, all the behaviours within the behaviour list are displayed on the right side. Based on individual use cases and scenario criteria, such behaviour list can be filtered based on the ODD features. The reasoning below illustrates the filtering process displayed in Table 2.

- The absence of junction together with straight geometry filter out *Turn left* and *Turn right*
- Trees and buildings do not affect the filtering process
- Solid line will filter out *Lane change right* and *Lane change left* (assuming the current use case considers only scenarios following the rules of the road)
- Agent type vehicle filters out *Run*, *Slide*, and *Walk*

At the end of the ODD filtering process, for this sample ODD subset, only *Drive*, *Stop*, and *Reverse* are left. A selection function, can be either random or weighted, will then select a target behaviour from this list and feed into the scenario construction process in ‘Step 4’.

E. STEP 4 - SCENARIO CONSTRUCTION

For the scenario construction, a set of construction rules are developed to ensure the output scenarios are valid, and to eliminate certain combinations of ODD elements and behaviours that may not result into valid scenarios (e.g., turn right manoeuvre used on vehicle with junction to its left only). Within this step, there are two main sub-steps: 1) populate the required scenario parameter spaces, 2) fill in the appropriate parameter values. The filling of the parameter values are referenced to the input ODD if such specifications are available (e.g., if the system is restricted to perform at certain speeds then the parameter space values will reflect this).

The required scenario parameter spaces include all the required attributes within the dynamic and environmental condition sections as defined by the SDL level 2 grammar. The dynamic conditions include manoeuvre types (e.g., turn left/right), speed (e.g., 10 to 12 m/s), acceleration, relative locations (e.g., front side right), etc. At this sub-step, the set of rules will map to the SDL grammar requirements based on the filtered manoeuvre list, and auto-create the missing parameters. For example, initial relative positions between the other traffic participants and the Ego are required by the language grammar, however this information is not present within steps so far, they are therefore created at this sub-step.

The second sub-step is in charge of auto-complete the parameter values by incorporating a set of pre-defined rules. As an example, in the previous sub-step, Ego speed as a scenario parameter is created, if based on the ODD specification Ego speed limit is 15mph, this sub-step will create the corresponding value/value ranges based on this. The current use case only incorporates rules dealing with the dynamic aspect of a scenario, if the original input does not specify the desired scenery of the scenario, the corresponding scenario scenery construct rules will also be created using similar manner. For instance, if minor road is contained within the sampled ODD features, then parameters such as number of lanes, lane width etc. will be created and parameterised according to the language grammar and constraints.

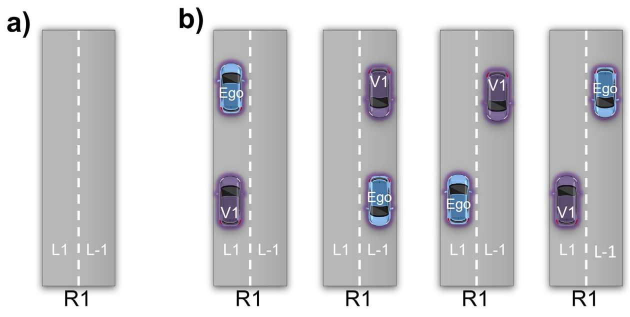


FIGURE 12. Example illustrating different possible initial conditions based on a given scenery layout.

The number of rules governing the scenario construction is significant due to the large number of the potential combinations of the constraining parameters, example contained in Figure 12 provides an illustration. The road layout in this case is a two-lane road with opposite driving directions between the lanes, as shown in Figure 12a. Assuming the scenario contains two traffic participants, the Ego vehicle and the agent vehicle (V1). Also assuming both the vehicles are obeying the rules of the road, Figure 12b lists all the four possible initial position combinations between the two vehicles. When considering the required scenario parameters based on the SDL level 2 description grammar, Table 3 illustrates three of the relevant scenario constructing rules for the scenario detailing process. Please note, in Table 3, the required scenario parameters that need to be **derived** using the rules are: 1) the agent vehicle’s initial position using Road ID and Lane ID, 2) the Ego vehicle’s initial position using Road ID and Lane ID, 3) the agent vehicle’s relative position to the Ego vehicle, and 4) the agent vehicle’s relative heading angle to the Ego vehicle. These need-to-be-derived parameters are indicated after the '**THEN**' statements in Table 3. On the other hand, those parameters within the '**IF**' statements are pre-selected scenario parameters, which are obtained either by: 1) sampling through the list of valid ODD and behaviour parameters resulted in the previous stage, 2) performing previous scenario construct rule(s), e.g., the first rule in Table 3 will select a valid initial Road ID and Lane ID for both the agent vehicle and the Ego vehicle, which will be the input for second and third rules.

TABLE 3. Individual scenario constructing rules for the example shown in Figure 12.

Example scenario constructing rules
IF the agent's relative manoeuvre is *Towards* **AND** its absolute manoeuvre is *Drive* **AND** junction is *none* **AND** number of lanes is *2* **AND** road type is *not motorway* **THEN** both the agent's and the Ego's initial positions can be **either** [R1.L-1, R1.L1]
IF the agent's and the Ego's initial positions equal **THEN** agent's relative position to the Ego is *Rear* **AND** agent's relative heading angle will be *-5 to 5* to the Ego
IF the agent's and the Ego's initial positions no equal **THEN** the agent's relative position to the Ego is *Front side right* **AND** agent's relative heading angle will be *175 to 185* to the Ego

Below are the explanations behind the three rules listed in Table 3:

- The first rule identifies the possible road and lane IDs for the initialisation of the agent vehicle and the Ego vehicle based on the layout indicated in Figure 12a, which can be equal or not equal.
- Since the agent vehicle's relative manoeuvre is *Towards* (which means one scenario actor is facing the other scenario actor), if the initial road and lane IDs of the agent vehicle and the Ego vehicle are equal, then the agent vehicle will be at the rear position of the Ego vehicle, and the agent vehicle's relative heading angle to the Ego vehicle will have a mean value of zero (i.e. the same heading). There are two cases which satisfy such conditions, they are represented by the first two initialisations (from left to right) indicated by Figure 12b.
- If the agent vehicle's and the Ego vehicle's initial road and lane IDs are not equal, given that the agent vehicle is moving *Towards* the Ego vehicle, the only valid relative position for the agent is at the *Front side right* of the Ego. There are two cases which satisfy such conditions, they are represented by the last two initialisations indicated by Figure 12b.

The complexity of the rule set increases when incorporating more junction and road types, as well as when including more traffic participants and more manoeuvre phases. However, taking a modular approach to break the intended scenario down into atomic level (i.e., using single manoeuvre phase at one time), the same mechanism can be applied to create such complex scenario output.

F. STEP 5 - SCENARIO CONCRETISATION & CONVERSION INTO ASAM OPENX FORMAT

Utilising the scenario format translation process documented in [35], all the generated logical scenarios described in SDL level 2 are then translated into ASAM OpenSCENARIO 1.x and OpenDRIVE format, using the mid-values of the logical scenario parameter ranges as the concrete parameter values. During the testing phase, such concrete scenario parameters can be overridden based on the scenario execution analysis.

V. GENERAL DISCUSSION

When evaluating the effectiveness of any scenario-based testing workflow, one of the main questions is how to ensure a 'complete' coverage of the ODD space. To answer this, refer back to the SOTIF diagram in Figure 4, any testing methods can only minimise the unknown/unsafe area, rather than fully eliminate. Scenario-based testing, with the consideration of ODD, is facilitating the testing scope by drawing the relevant 'boundary' within which the 'gaps' need to be explored. Please note that scenarios generated based on ODD should be used in conjunction with scenarios generated from other sources (such as accident, insurance claim, rules of the road etc) to form a comprehensive test suite. Different scenario generation methods complement each other by having different perspectives. In addition, the continuous improvement of the scenario set via real world trial and deployment scenarios also contribute towards the continuous V&V of the system.

In order to ensure that the scenario and its execution within the simulation environment correlate to the real world situations, there are two distinctive aspects: realistic scenarios, and credible simulation. The former one aims at the correct scenario representation of the real world situation, whereas the later one answers whether the virtual test environment used are credible enough for the type of virtual testing it is operating for. Realistic scenario includes incorporating validated scenarios both syntactically and semantically, this ensures that the correct scenarios are output. Furthermore, the scenario sources and the rareness of the scenarios should also be considered. Based on the specific study, the sources and the rareness focus might vary. Generation based on ODD can complement other scenario generation methods, for example, if the system's ODD focuses on motorway and rainy weather, but all the available accident data only contains motorway and sunny weather, ODD based scenario generation can fill in the gap left from other scenario generation approaches.

One might then ask, will scenarios purposely generated based on ODD lead to biases or over-optimisation? To answer this, we need to refer back to the definition of ODD. The ODD of a system is a design specification documenting its claimed capabilities. For example, an autonomous food delivery pod might claim to be able to operate only in residential area over pavement. Such 'claimed' capabilities need to be tested fully to provide safety arguments. In addition, knowing the ODD will also help identifying boundary cases, therefore facilitate the testing of out of ODD situations and test the vehicle's capability of safely returning to its ODD or any mitigation process. Testing a system using scenarios with mismatching ODDs will make the testing process invalid, for example testing the same food delivery pod on motorway will not ensure safety on its intended operational domain, i.e. residential ODD.

The reproducibility of scenario-based testing is another important discussion point. The scope of the dynamic element of a scenario includes both agent vehicles, and the SUT. While the SUT's behaviour is never known until the actual execution using the actual SUT, the behaviour of the other agents can be deterministic and reproducible, as they are pre-scripted. From an implementation level, the execution reproducibility is governed by two factors: the scenario description format, and the scenario execution tools. The format may or may not be independent from the tools, for example the ASAM OpenX format is independent from simulation tools whereas other proprietary simulation tools might have dedicated scenario format. But the final executed behaviour is dependent on the implementation logic designed by the tool developer against the supported format.

Since now the ODD is a key aspect for scenario-based testing and scenario generation, ensuring the correct definition of ODD, both syntactically and semantically, is crucial. The ODD specification is defined by the system developer using standardised format during the initial development stage, rather than during the testing stage, therefore ensuring the accuracy of ODD is the developer's responsibility. Ensuring the ODD spaces sufficiently explored during a scenario-based testing process is the responsibility of the testing workflow. Paper II of this work series will introduce coverage concepts linking test scenarios and ODD. In addition, multiple sources of scenario generation from different perspective are used when creating the scenario test suite. Such sources including accident data-based, insurance claims-based, system analysis-based, ODD-based, real world trial-based, rules of the road -based etc. Utilising multiple sources for scenario generation also provides the opportunity for runtime V&V, and continuous improvement evolving of the system.

VI. CONCLUSION

This paper documents a novel approach to scenario generation utilising the ODD and behaviour competency of a target system for its safety assurance process. Step 1 of the process focuses on obtaining the corresponding ODD of the system, in the paper the operating route of the system (Operational Domain) was used to extract ODD information. Based on the ODD, combined with the behaviour competency, filtering process was performed to identify the applicable behaviours for the identified ODD elements. Incorporating a set of scenario construct rules, the applicable behaviours together with the valid ODD elements are used to generate logical scenarios defined in the WMG SDL level 2 format at the logical scenario level. A further auto-conversion process then converts the logical scenarios into concrete ASAM OpenX formats (OpenSCENARIO 1.x & OpenDRIVE). Future work, which will be published in paper II of this paper series, will illustrate an scenario evaluation workflow focused around coverage analysis.

ACKNOWLEDGMENT

The work presented in this paper has been supported by UKRI Future Leaders Fellowship (Grant MR/S035176/1), Department of Transport, UK, and Transport Canada (T8080-220112). The authors would like to thank the WMG center of HVM Catapult, and WMG, University of Warwick, UK for providing the necessary infrastructure for conducting this study.

REFERENCES

[1] D. J. Fagnant and K. Kockelman, "Preparing a nation for autonomous vehicles: Opportunities, barriers and policy recommendations," *Transp. Res. A, Policy Pract.*, vol. 77, pp. 167–181, Jul. 2015.

[2] S. Le Vine, X. Liu, F. Zheng, and J. Polak, "Automated cars: Queue discharge at signalized intersections with 'assured-clear-distance-ahead' driving strategies," *Transp. Res. C, Emerg. Technol.*, vol. 62, pp. 35–54, Jan. 2016.

[3] D. J. Fagnant and K. M. Kockelman, "The travel and environmental implications of shared autonomous vehicles, using agent-based model scenarios," *Transp. Res. C, Emerg. Technol.*, vol. 40, pp. 1–13, Mar. 2014.

[4] N. Balfe, S. Sharples, and J. R. Wilson, "Impact of automation: Measurement of performance, workload and behaviour in a complex control environment," *Appl. Ergonom.*, vol. 47, pp. 52–64, Mar. 2015.

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

[6] X. Zhang, S. Khastgir, H. Asgari, and P. Jennings, "Test framework for automatic test case generation and execution aimed at developing trustworthy AVs from both verifiability and certifiability aspects," in *Proc. IEEE Int. Intell. Transp. Syst. Conf. (ITSC)*, Sep. 2021, pp. 312–319.

[7] PEGASUS Consortium. *PEGASUS Method: An Overview*. Accessed: Nov. 21, 2023. [Online]. Available: https://www.pegasusprojekt.de/en/pegasus-method

[8] European Center for Information and Communication Technologies. *VVM—Verification and Validation Methods for Automated Vehicles in Urban Environments*. Accessed: Nov. 21, 2023. [Online]. Available: https://www.vvm-projekt.de/en/

[9] S. Chen, S. Khastgir, I. Babaev, and P. Jennings, "Identifying accident causes of driver-vehicle interactions using system theoretic process analysis (STPA)," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2020, pp. 3247–3253.

[10] E. Esenturk, S. Khastgir, A. Wallace, and P. Jennings, "Analyzing real-world accidents for test scenario generation for automated vehicles," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jul. 2021, pp. 288–295.

[11] T. Menzel, G. Bagschik, L. Isensee, A. Schomburg, and M. Maurer, "From functional to logical scenarios: Detailing a keyword-based scenario description for execution in a simulation environment," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2019, pp. 2383–2390.

[12] X. Zhang, S. Khastgir, and P. Jennings, "Scenario description language for automated driving systems: A two level abstraction approach," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2020, pp. 973–980.

[13] F. Bock, C. Sippl, A. Heinz, C. Lauer, and R. German, "Advantageous usage of textual domain-specific languages for scenario-driven development of automated driving functions," in *Proc. IEEE Int. Syst. Conf. (SysCon)*, Apr. 2019, pp. 1–8.

[14] ASAM e.V. (2022). *OpenSCENARIO V1.2.0*. Accessed: Nov. 21, 2023. [Online]. Available: https://www.asam.net/standards/detail/openscenario/

[15] ASAM e.V. (2021). *OpenDRIVE V1.7.0*. Accessed: Nov. 21, 2023. [Online]. Available: https://www.asam.net/standards/detail/opendrive/

[16] *Safety Pool Database*. Accessed: Nov. 21, 2023. [Online]. Available: https://www.safetypool.ai/

[17] ASAM e.V. (2021). *OpenLABEL V1.0.0*. Accessed: Nov. 21, 2023. [Online]. Available: https://www.asam.net/standards/detail/openlabel/

[18] S. Khastgir, S. Birrell, G. Dhadyalla, and P. Jennings, "The science of testing: an automotive perspective," SAE Tech. Paper 2018-01-1070, 2018.

[19] *Taxonomy and Definitions for Terms Related to Driving Automation Systems for on-Road Motor Vehicles*, Standard SAE J3016, SAE International, Surface Vehicle Recommended Practice, 2021.

[20] *The Highway Code*, Department for Transport, Washington, DC, USA, 2022.

[21] *Automated Lane Keeping Systems (ALKS)*, UNECE, Geneva, Switzerland, 2021.

[22] *Road Vehicles—Safety of the Intended Functionality*, Standard ISO 21448:2022, 2022.

[23] P. Koopan, "The heavy tail safety ceiling," in *Proc. Automated Connected Vehicle Syst. Test. Symp.*, 2018.

[24] S. Ulbrich, T. Menzel, A. Reschka, F. Schuldt, and M. Maurer, "Defining and substantiating the terms scene, situation, and scenario for automated driving," in *Proc. IEEE 18th Int. Conf. Intell. Transp. Syst.*, Sep. 2015, pp. 982–988.

[25] BSI. (2022). *BSI Flex 1889—Natural Language Description for Automated Driving Systems*. Accessed: Jan. 12, 2024. [Online]. Available: https://www.bsigroup.com/en-GB/insights-and-media/insights/brochures/bsi-flex-1889-natural-language-description-for-automated-driving-systems/

[26] G. Bagschik, T. Menzel, and M. Maurer, "Ontology based scene creation for the development of automated vehicles," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 1813–1820.

[27] T. Menzel, G. Bagschik, and M. Maurer, "Scenarios for development, test and validation of automated vehicles," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 1821–1827.

[28] E. Esenturk, A. G. Wallace, S. Khastgir, and P. Jennings, "Identification of traffic accident patterns via cluster analysis and test scenario development for autonomous vehicles," *IEEE Access*, vol. 10, pp. 6660–6675, 2022.

[29] R. Agrawal, T. Imielinski, and A. Swami, "Mining association in large databases," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 1993, pp. 207–216.

[30] M. Brackstone, S. Khastgir, J. Martin, A. Charr, S. Navin, G. Jackman, P. Tomlinson, N. Clay, and P. Jennings, "OmniCAV: A simulation and modelling system that enables 'CAVs for all,'" in *Proc. IEEE 23rd Int. Conf. Intell. Transp. Syst. (ITSC)*, Sep. 2020, pp. 1–6.

[31] ASAM e.V. (2021). *ASAM OpenXOntology Concept*. Accessed: Nov. 21, 2023. [Online]. Available: https://www.asam.net/standards/asam-openxontology/

[32] A. A. B. da Costa, P. Irvine, X. Zhang, S. Khastgir, and P. Jennings, "Automatic generation of ADS scenarios from rules of the road (submitted)," *IEEE Trans. Intell. Vehicles*, 2024.

[33] Euro NCAP. (2020). *Euro NCAP Test Protocol-AEB VRU Systems*. Accessed: Jan. 12, 2024. [Online]. Available: https://cdn.euroncap.com/media/58226/euro-ncap-aeb-vru-test-protocol-v303.pdf

[34] *Intelligent Transport Systems—Low-Speed Automated Driving (LSAD) Systems for Predefined Routes—Performance Requirements, System Requirements and Performance Test Procedures*, Standard ISO 22737, 2021.

[35] A. Anastasio, P. Irvine, X. Zhang, S. Khastgir, and P. Jennings, "Translating automated vehicle test scenario specifications between scenario languages: Learnings and challenges," in *Proc. Driving Simulation Conf. Eur. VR*, 2022, pp. 65–72.

[36] BSI. (2020). *Operational Design Domain (ODD) Taxonomy for an Automated Driving System (ADS)—Specification*. Accessed: Jan. 12, 2024. [Online]. Available: https://www.bsigroup.com/globalassets/localfiles/en-gb/cav/pas1883.pdf

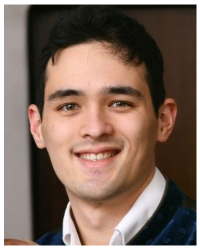
[37] *Road Vehicles—Test Scenarios for Automated Driving Systems—Taxonomy for Operational Design Domain*, Standard ISO 34503, 2023.

[38] P. Irvine, X. Zhang, S. Khastgir, E. Schwalb, and P. Jennings, "A two-level abstraction ODD definition language: Part I," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2021, pp. 2614–2621.

[39] E. Schwalb, P. Irvine, X. Zhang, S. Khastgir, and P. Jennings, "A two-level abstraction ODD definition language: Part II," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2021, pp. 1669–1676.

XIZHE (JASON) ZHANG received the B.Eng. and Ph.D. degrees from the Department of Mechanical Engineering, University of Bath, U.K. He is currently the Simulation Lead of the Validation and Verification Team, WMG, University of Warwick. He has also been leading the development of several international standardization projects with the goal of unifying and accelerating activities within the autonomous driving domain. His research interests include scenario-based and simulation-based safety assurance workflow for automated systems, ontology design and implementation, language development, and toolchain development.

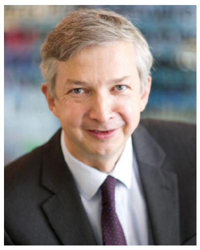
SIDDARTHA KHASTGIR received the Ph.D. degree in ADS testing and human factors (trust in automation). He is currently a Professor with WMG, University of Warwick. He is also an Expert in ADS/ADAS safety verification and has significant experience in the ADS domain include test scenarios generation, safety, simulation-based testing, standardization, and international regulations. He has over ten years' experience in safety verification and validation.



JUSTIN-KIYOSHI TIELE received the M.Eng. degree in electronic engineering with business management from the Warwick University School of Engineering, U.K. He is currently a Senior Research Engineer with the AD/ADAS Research and Development Department, Corporate R&D Unit, DENSO AUTOMOTIVE Deutschland GmbH. His research interests include safety assurance methodologies for the design, development, and testing of AI/ML-based automated driving technology. Example activities in this domain include experience with toolchains for scenario-based V&V simulation workflows, relevant standardization and regulatory frameworks, realization of field-operational-testing, perception strategies (sensing principles and algorithms), and safety architectures. He has participated in international industry collaboration activities as a consortium working group member.

KAZUHITO TAKENAKA received the B.E. degree from The University of Tokyo, in 2006, and the M.S. and Ph.D. degrees from the Graduate School of Information Science and Technology, The University of Tokyo, in 2008 and 2011, respectively. He was with DENSO Corporation, from 2011 to 2021. He is currently with DENSO AUTOMOTIVE Deutschland GmbH. His research interest includes safety assurance methodologies for automated driving systems.

TASUKU HAYAKAWA received the M.Eng. degree in systems and information engineering from the University of Tsukuba, Japan. He was with DENSO AUTOMOTIVE Deutschland GmbH, from 2018 to 2021. He is currently with DENSO Corporation. His research interests include the development of vehicle dynamics and automated driving systems for mobility solutions which include safety assurance requirements.



PAUL JENNINGS is currently a Professor and the Research Director of WMG, University of Warwick. He has over 30 years of experience in industrial collaborative research and development projects, having led research projects worth over £60M. Before taking up his role as the Research Director of WMG, where he set up and led the Intelligent Vehicles Group, with a key focus on safety. He has published over 100 academic articles and has been a principal investigator on over 30 research grants.