# Towards a Scenario Toolkit for Autonomous Systems

Ahmad Naja[1*], Siddhartha Gupta[1], Umut Durak[2], Sven Hartmann[1]

[1]Institute for Computer Science, TU Clausthal, Clausthal-Zellerfeld, Germany; * *ahmad.naja@tu-clausthal.de*

[2]German Aerospace Center (DLR), Braunschweig, Germany; *umut.durak@dlr.de*

**Abstract.** Scenario-based approaches have recently been widely adopted in the automotive and aviation industries. They aim to define and manage the test cases in a better way, thereby significantly reducing the risks and defining the safety argument for the system. To improve and complement the safety conditions, the Operational Design Domain (OOD) defines the operational boundaries of a driving automation system to specify the scope of the safety case, represented by human and machine-readable language. Many approaches and standards that involve essential modelling and safety concepts have been introduced to represent scenario-based simulation. In addition, some parts of these approaches were implemented as graphical tools, such as System Entity Structure (SES) and Pruned Entity Struture (PES) tools, which are based on an ontology and its derived structure. However, an extensive implementation covering scenario modelling and management, ODD, and assessment is still missing. This paper proposes an adapted scenario-based approach based on the related research and implements it using a robust GUI tool called Operational Domain Modeling Environment (ODME). ODME is progressing to fill the gap by covering the modelling functions and safety approaches in one comprehensive environment with a wide range of capabilities and features.

## Introduction

The subject of modeling aims to represent a certain aspect of reality for a particular purpose. Systems, processes, and phenomena can all be represented by models. Afterward, the system behaviors can be generated through the simulation process using the model [1]. Creating and developing models is considered as an initial step to simulate and test these systems based on different use cases, called scenarios.

A scenario describes a system's behavior based on its operating conditions and situations, changing of its parameters through time, and the mutual interaction of its components with each other [2]. Scenario modelling helps researchers better plan and lower risks by exploring a variety of prospective outcomes, comparing them using specific standard metrics, and testing decisions [i]. Developing a scenario goes through significant steps, starting by defining the scenario by the stakeholders and finishing by generating the necessary executable specifications. Scenarios as executable specifications are input to the simulation environment [3].

One of the biggest problems facing the car industry is ensuring the safety of autonomous vehicles. Scenario-based development and test methods are potential methods for testing and validating autonomous driving capabilities [4]. The scenario-generating process of the scenario-based approach carries forward a safety argument and is crucial for the system's release. So, the scenarios must be created and documented methodically. In addition, they have to be traceable throughout the development process [4]. Technology firms and automakers use the Operational Design Domain (ODD) concept to specify the safe operating conditions for their Automated Driving Systems (ADS). An ODD establishes where the ADS are intended to function correctly by definition [5]. Therefore, scenarios and OOD complement each other to provide a safety argument for an autonomous vehicle.

Organizations and researchers develop many standards to represent the scenario computationally, such as OpenSCENARIO from the Association for Standardization of Automation and Measuring Systems (ASAM). ASAM played a lead role in setting standards that cover the whole simulation pipeline. Figure 1 shows a simulation process workflow based on a family of ASAM standards, which illustrates the different steps a scenario goes through, beginning with a concept description and ending with testing results.

[i] https://www.synario.com/scenario-modeling-what-you-need-to-know/

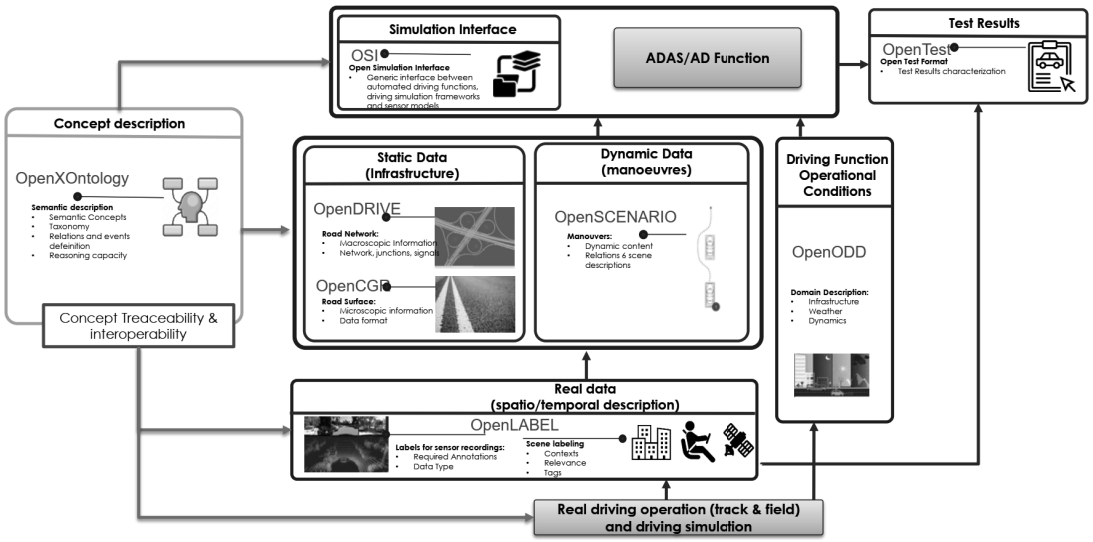


Figure 1: Simulation Process Workflow Based on a Family of ASAM Standards [6]

OpenSCENARIO is the point of interest used for driving simulation and virtual development, testing and validation of driving assistance functions, automated and autonomous driving [ii].

There are some concerns with the approach illustrated in Figure 1. Many standards are still under development and have not been shown to be compatible with each other. Many third-party tools have been shown to use one of these standards, mostly focusing on OpenSCENARIO, but none can handle the complete workflow. A more integrated and compact workflow is needed along with its associated tool-set.

This paper provides new insights to define an adapted approach based on the ASAM simulation workflow in Figure 1 and other research work. This adapted approach will compact the simulation process based on the scenario modelling, scenario generation, management and ODD definition. Furthermore, a modelling environment that implements the new approach will be introduced, called the Operational Domain Modeling Environment (ODME). The presented model-based scenario development employs System Entity Structures (SES) based metamodeling that offers the development of models and the generation of executable software entities with successive model transformations through technical spaces [7]. The SES and its related project, Pruned Entity Structures(PES) tools presented in [8], was a starting point for achieving the suggested goals. After refactoring, it will be ready to expand and implement the scenario-based approach.

The paper begins by giving a background in section I about SES, scenario representation, Operational Design Domain, ASAM Standards and SES and PES Tools. In section II, the newly suggested approach will be introduced and illustrated. Section III will implement the theoretical approach in a practical modelling environment. Finally, section IV gives a short conclusion and shows some recommended views for future work.

# 1 Background

## 1.1 System Entity Structure (SES)

SES is described as a framework for knowledge representation of system coupling, decomposition, and taxonomy [9] and is considered an enhancement in the discipline of system theory-based approaches to modelling and simulation [10]. Utilizing interactions between decomposition, coupling, and taxonomies allows for the succinct specification of a family of models [11]. SES is further described as a formal ontology framework specifying a system's components and hierarchical relationships [12]. Figure 2 shows SES nodes and relationships and a simple SES tree example.

An SES is represented as a labelled tree. The entity, Aspect, Specialization, and Multi Aspect are the four different types of nodes [14].

[ii] www.asam.net/standards/detail/openscenario/

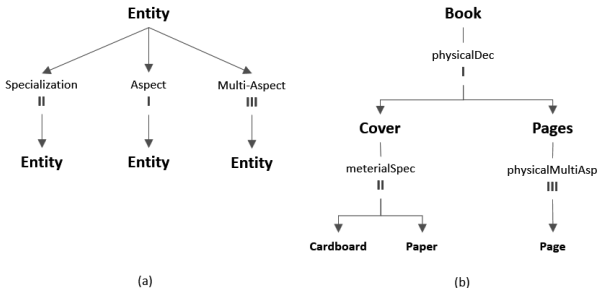


Figure 2: (a) SES Nodes and Relationships, [12] and (b) an Example.[13]

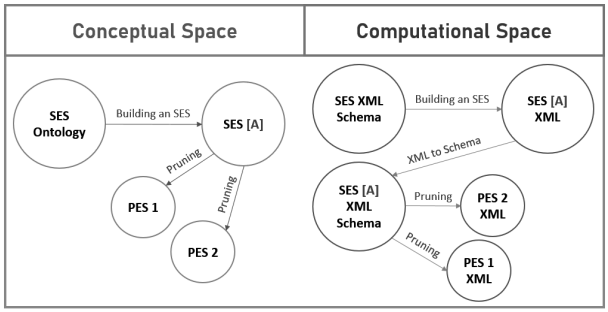


Figure 3: Computational Representation [17]

- **Entity:** It depicts system elements that are artificial or real. It is an object of interest that may have variables linked to it. Other node types are utilized to describe parent and child entities.
- **Aspect:** It indicates the decomposition relationship of an Entity node. It represents processes for breaking down larger objects into more fine-grained ones.
- **Specialization:** It represents an Entity's taxonomy. Specialization denotes groups or families of distinct forms that an object may take.
- **Multi Aspect:** This particular type of aspect describes a multiplicity connection and shows that the parent entity is composed of many entities of the same type.

Several axioms define the SES as well, which are: uniformity, strict hierarchy, alternating mode, valid brothers, attached variables, and inheritance [15].

Given its foundations in the theory of modelling and simulation and its expressive strength and clarity with only a few axioms, SES is suggested as the foundation of the proposed intermediate metamodel in the Simulation Model package. Thus, SES is appropriate as a straightforward intermediate metamodel that establishes a formal framework that is clear and understandable [16].

## 1.2 Scenario Computational Representation

The machine-readable format for SES and Pruned Entity Structure (PES) is called computational representation. There are two significant operations in the conceptual space, as shown in Figure 3.

An SES Ontology—a specific SES—is constructed utilizing the constructs, structure, and axioms of SES. The pruning process generates the Pruned Entity Structures (PESs) from this specific SES. According to [18], we may express the SES ontology with an XML Schema and the specific SES as an XML file in the computational space. Then, they proposed creating an XML that specifies a particular SES to an XML Schema. The construction and validation of PESs during pruning finally employ this schema. PESs ultimately become XML instances. The schema for the SES ontology could be defined using the XML Schema Definition Language (XSD). An XML document's restrictions and structure may be described using XSD [19].

## 1.3 Operational Design Domain (ODD)

*SAE J3016* defines the Operational Design Domain (ODD) for a driving automation system 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 the requisite presence or absence of certain traffic or roadway characteristics."* In short, the ODD establishes the limits that the driving automation system is intended to work within and, as a result, will only function when these criteria are met, as shown in Figure 4.

The ODD restricts where the automated driving system (ADS) is valid and thus confines the scope of the safety case and the verification. Use cases are required to give a strategy for a set of operating conditions (OCs) and verify that it remains within the range of the ODD [iii].

ODD can control the coverage of scenarios and provide a list of operation conditions, which can be used later as an effective input for the system assessment process. Some requirements are important when ODD is

[iii] www.claytex.com/tech-blog/

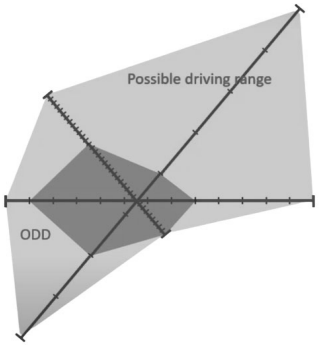


Figure 4: Operational Design Domain (ODD) [20]

defined. For instance, ODD should be generated in a format understandable for humans, such as tables. On the other hand, it has to be machine-readable, such as an XML file.

### 1.4 ASAM Standards

In the automobile sector, development and test toolchains can be typically built to ASAM standards, which are public specifications. The use of ASAM standards is optional. To determine the Standard-compliance of products, ASAM advises and promotes best practices. Throughout vehicle development, ASAM standards specify interfaces, protocols, file formats, and data models. ASAM-based tools and products provide seamless data exchange and simple integration into already-existing value chains [iv].

ASAM provides a family of standards for the simulation domain, which have repeatedly proven themselves in various development processes. These standards must interact to generate a global view of the simulation process. In the following, some related ASAM standards are mentioned [21]:

- **OpenXOntology:** An ontology-based framework for notions like roads, lanes, and traffic participants is provided by OpenXOntology. The ASAM Open XOntology comprises several interconnected components, including core, domain, and application ontologies.

- **OpenODD:** For connected autonomous cars (CAVs), OpenODD seeks to create a format that may describe a specified Operational Design Domain. The ODD defines the functional requirements for connected autonomous vehicles and outlines the environmental characteristics that CAVs must be able to control.

- **OpenSCENARIO:** The dynamic content of the world is defined by OpenSCENARIO, for instance, the anticipated behaviour of traffic participants and how they should interact with one another and their environment.

- **OpenDRIVE:** The primary goal of OpenDRIVE is to offer a description of the road network that can be used as input into simulations to create and verify advanced driver assistance systems (ADAS) and Autonomous driving features.

### 1.5 SES and PES Tools

Initial scenario workflow using SES[22] was implemented using SES tools. PES Tools provided the pruning capabilities for the tool. It added to and even finished work outlining a formal strategy for creating a scenario specification language [23]. Figure 5 shows the workflow of SES-PES projects.

In SES Tools, The user may access a wide variety of widgets using the graphical user interface that helps with modelling. It uses a collection of elements and axioms to describe knowledge about system connection, taxonomy, and decomposition [24]. While the SES model developed is pruned using PES Tools, several scenarios are produced. Pruning is a technique that creates a distinct system structure from a domain model; the outcome is known as a Pruned Entity Structure(PES).

An SES Model represents a family of models for a certain application domain. All of a system's potential configurations are taken into account when using SES modelling. An SES model tree has to be trimmed to get a specific configuration. To achieve this configuration, a domain model tree that is a selection-free tree, pruning removes extraneous structure based on the definition of a realistic frame. A domain model is often reduced by pruning by eliminating options for an entity with numerous attributes and specializations made up of several entities. A domain model tree may be pruned by giving the values of the variables, choosing one entity from the available specialization node possibilities, and indicating the cardinality in a Multi Aspect node [8].

[iv] https://www.asam.net/standards/standard-compliance/

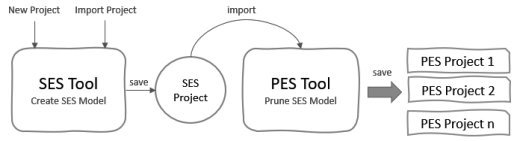


Figure 5: SES and PES Projects Structure

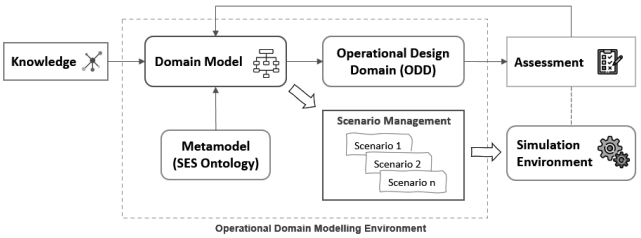


Figure 6: Proposed Scenario-based Approach Workflow

Although the tool-set provided the modelling and simulation community with many advantages, it is currently not scalable and requires several enhancements. Using two tools and managing the pruned models was a critical problem that made the project complicated and hard to use.

## 2 General Scenario-based Approach for Autonomous Systems

The proposed scenario-based approach uses the fundamental elements of many works involving scenarios such as the ASAM workflow [6] and other research works like [25], [26], integrating many features of scenario simulation principles.

The simulation model process in the proposed approach goes through a sequence of fundamental steps, from knowledge acquisition to finding the best scenario to be executed. The domain model should be created depending on some knowledge generated by experts or raw data. A domain model defines an abstract representation of all the elements of the simulated system, so it needs to be pruned to produce a particular use case or scenario. Scenario Manager aims to organize the created scenarios and their attributes and help to arrange scenarios based on different metrics. Afterward, the selected scenario will be executed physically or virtually to be tested and assessed. Finally, the metrics will be evaluated to send feedback to the domain model to be improved. Figure 6 illustrates the suggested scenario-based approach.

**Knowledge Source.** A common way to generate relevant scenarios is to use the knowledge of domain experts and formulate the scenarios manually. Data from real driving situations could be another input for scenario modelling (not the focus of our current approach).

**Metamodel (SES Ontology.)** An ontology offers a vocabulary for a specific domain by computerizing the specification of the meaning of definitions and describing the concepts and relationships that are significant in a given domain. As ontologies explain domain relationships and entities in a simple and machine-interpretable way, they serve as a bridge between humans and computer systems. Model transformations are offered as an automated way to produce an executable scenario definition, whereas metamodeling is proposed as a technique to develop a graphical modelling language. To include all the components of a scenario that may be simulated in autonomous cars, SES is used for metamodeling [23]. Upon this metamodel, a scenario modeling approach is developed.

**Domain Modelling.** In this step, the knowledge will be used to create a model based on SES ontologies that describe the system in human-readable representation. This model will be considered an abstraction of all possible scenarios, so it needs to be pruned later to generate derived models or individual scenarios. Moreover, the variables and constraints used in the model will be fundamental to defining ODD.

**Scenario Management.** The created domain model will be derived in different scenarios by the pruning process. However, the generated scenarios need a mechanism to be organized. The scenario manager aims to achieve this goal by developing a method to facilitate storing and managing the scenarios. Scenarios also needs to have certain metrics associated with them.

**Define Operational Design Domain (ODD).** In this block, the operating conditions that form the system's boundary will be defined, as shown in Table 1. These operational conditions will be used as input for the assessment process.

| Value Range | Aspect |
|---|---|
| **P**edestrian | sporadically |
| **R**oad Types | highway |
| **T**ime of Day | any |
| **S**peed Range | $\leq$ 130 kph |
| **V**isibility | $\geq$ 40 m |

Table 1: ODD Table Example

**Scenario Execution.** Once the scenario test cases have been identified, they must be executed on real physical systems or simulation tools. During the execution, the results should be recorded and stored to be analyzed and evaluated. This step is essential to specify the safety properties and is required to develop executable specifications.

**Assessment Process.** Evaluating the quality and the performance characteristics, such as safety and efficiency, is crucial in developing autonomous systems. In this block, the results and remarks of execution will be processed and analyzed to evaluate the scenario, hazard analysis and risk assessment. After that, The scenario's metrics will be updated and sent as feedback to the domain modelling block.

# 3 Operational Domain Modeling Environment (ODME)

ODME is a robust environment that contains all functionalities the user needs to create and prune models and manage pruned scenarios via the scenario manager. It is a successor to the SES tools implementing the functionalities contained in the dotted line in the proposed scenario-based approach in Figure 6. Figure 7 shows the general view of the ODME main window in domain modelling mode.

Developing the ODME project went through four important levels:

1. **Planning:** Before starting with implementation, many related projects and research were reviewed to collect ideas and create a special innovative approach. The SES and PES Tools project was also chosen as a base and starting point for ODME.

2. **Reconstruction:** two important steps were implemented in this level:

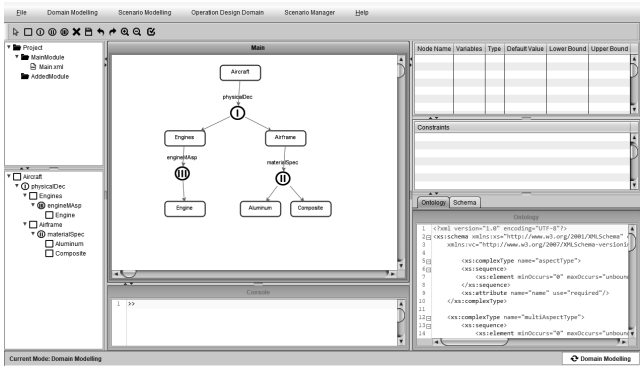


Figure 7: ODME Main Window in Domain Modelling Mode

   • Refactoring Process: cleaning the code, fixing bugs, and improving the project's structure.

   • Build Mode Switch mechanism: create a smooth and simple solution to change between Domain Modelling and Scenario Modeling modes after merging SES and PES tools.

3. **Upgrade:** We added many new features to ODME, such as importing/exporting the model as a template and saving the model as a PNG file, in addition to some improvements in GUI design. The essential new feature in ODME is the scenario manager, which allows the user to create multi-scenario models for one domain model and delete and change the metrics of scenarios. The scenario files will be managed automatically and stored in separate folders that can be moved to other devices, making the created projects portable.

4. **Test and Documentation:** by using the new environment to create real use case scenarios, documenting, and suggesting more advanced features for future work.

As mentioned before, ODME consists of two modelling modes, domain and scenario modelling, as well as the scenario manager.

## 3.1 Domain Modelling Mode

ODME facilitates the user creating a domain model as a visual representation of real situation objects by a wide set of widgets and features. The components in the domain of the problem, and the connections between them, are represented by the Domain Model. All system entity structure components introduced in section

[II] are supported. ODME added many new features which help to improve usability and make the environment reliable and user-friendly.

While ODME starts, the main window will appear in Domain Modelling mode, and a new project called Main will be generated automatically with a root node. In the Drawing Panel, the user can create a domain model using the different types of nodes in the Tool-Bar. While a node is connected to the graph model, the Synchronized Tree Window will be updated to show it. When the model is saved, the ontology and schema will be generated. Figure 8 shows a simple domain model created by ODME.

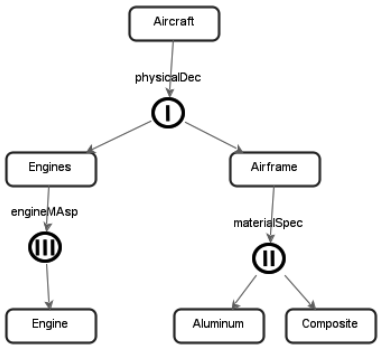


Figure 8: Domain Model Example

## 3.2 Scenario Modelling Mode

After creating a domain model, ODME provides a simple mode switch mechanism, which saves the domain model and forwards it to scenario modelling mode. In this mode, the model can only be pruned to create scenarios. The nodes which are eligible for pruning will be highlighted.

ODME support three different types of pruning [10]:

- **Multi Aspect Node Pruning** Before pruning, a Multi Aspect node must define its cardinality or a total number of aspects. Cardinality is assessed, and a certain number of aspects of the same kind are formed when a Multi Aspect node is pruned. Figure 9 shows the result after pruning the "engineMAsp" entity in Figure 8. Based on the cardinality number, which will be defined by the user (Three, for example), three engines will be generated after pruning.
- **Specialization Node Pruning** One child must be chosen to create a valid variation per the specialization requirement. Figure 8 demonstrates that the entity Airframe has two options: composite and aluminum. The entity that remains after pruning can, therefore, either be of type Aluminum or type composite. In Figure 9, the completed pruned entity Composite Airframe will be generated.

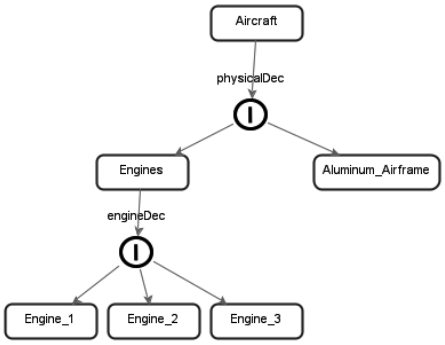


Figure 9: Scenario Model Example

- **Entity variable Pruning** Entities which have variables can only be pruned. So, they will be highlighted with green in scenario modelling mode. This pruning can be achieved by updating the value of an entity's variables. After pruning the entity variable, the variable table will also be updated with the new value. Figure 10 shows the variable table of the "Node" entity after pruning of "Var_2".

| Node Name | Variables | Type | Default Val... | Lower Bou... | Upper Bou... |
|---|---|---|---|---|---|
| Node | Var_2 | int | 5 | 0 | 10 |
| Node | Var_3 | string | none | | |
| Node | Var_1 | boolean | none | | |

Figure 10: Entity Variables Table after Pruning

## 3.3 Scenario Management

The domain model can be pruned to have many scenarios. The number of scenarios generated can extend to a large number, as the variability factors are many. It requires a mechanism to organize scenarios by labelling them, and prioritizing them before exporting and executing them. Scenario manager is one of the new features in ODME used to manage the scenarios created by the tool. The following points can summarize the main goals of a scenario manager:

- Capturing all the scenarios and creating a scenarios list to organize them, in addition, to helping to access any scenario.

- Assigning criteria as classification metrics to the scenarios manually.
- Providing run-time scripts to execute the scenario directly and show the result by linking the scenarios to the simulation environment, such as Matlab and Gazebo.
- Having the possibility of a mechanism to accept feedback from the assessment and adjust the scenarios accordingly.

The last two points are still under development. Figure 11 illustrates the main functionalities of the scenario manager.

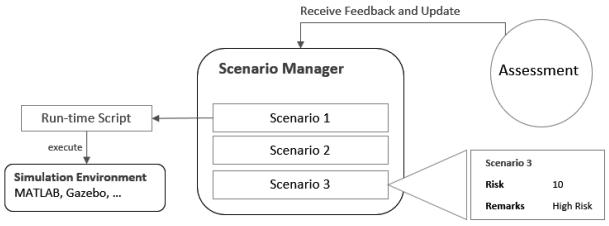


Figure 11: Scenario Manager

Scenarios List is a simple functionality in the tool to switch between scenarios, delete scenarios and change their metrics. By double-clicking on one of the scenarios, the metrics update window will be opened, where the user can update the risk value of the scenario in addition to writing some remarks. Figure 12 shows the scenarios list.

| Name | Risk | Remarks |
|---|---|---|
| InitScenario | | |
| Scenario2 | 10 | High Risk |
| Scenario3 | 2 | Low Risk |

Figure 12: Scenario Manager List

# 4 Conclusion and Future Work

In this paper, we proposed a scenario-based approach using SES ontology, which integrates many fundamental simulation concepts, like domain modelling, scenario management, and operational design domain. Furthermore, we implemented this approach in a robust tool called ODME. Nevertheless, ODME is still an active project open to many ideas and improvements. There are several directions for future work, aiming to implement ODD in our tool so that it could be used for the assessment process. In addition, the scenario manager needs to be optimized to compare different scenarios. The future work also includes developing a test data generator using machine learning algorithms. This data can help test a broader range of scenarios and define the operational conditions more precisely.

# References

[1] P. K. Davis and R. H. Anderson. Improving the composability of dod models and simulations. pages vol. 1, no. 1, pp. 517. The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology, 2004.

[2] D. A. Kononov, V.V. Kulba, S.S. Kovalevsky, and S.A. Kosjachenko. Development of scenario spaces and the analysis of dynamics of behaviour of social and economic system. Preprint, 1999.

[3] Umut Durak, Okan Topçu, Robert Siegfried, and Halit Oğuztüzün. Scenario development: A model-driven engineering perspective. 2014.

[4] Till Menzel, Gerrit Bagschik, Leon Isensee, Andre Schomburg, and Markus Maurer. From functional to logical scenarios: Detailing a keyword-based scenario description for execution in a simulation environment. 2019.

[5] S. O.-R. A. D. Committee et al. Sae j3016. taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles. tech. rep., 2016.

[6] Oihana Otaegui. Standardisation on interfaces and formats for ccam validation. connected and automated driving virtual conference, April 2021.

[7] D. Gasevic, D. Djuric, and Devedic V. Model driven engineering and ontology development. Springer, 2009.

[8] Bikash Chandra Karmokar. Application agnostic ses modeling environment and interactive pruning tool. TU Clausthal.

[9] T. Kim, C. Lee, E. Christensen, and et al. System entity structuring and model base management. pages 20: 1013–1024. IEEE Trans Syst Man Cybern, 1990.

[10] T. Ören and B. Zeigler. System theoretic foundations of modeling and simulation: a historic perspective and the legacy of a wayne wymore. pages 88: 1033–1046, 2012.

[11] J. Rozenblit and B. Zeigler. Representing and construction of system specifications using the system entity structure concepts. In: Proceedings of the 1993 winter simulation conference, Los Angeles, CA, 12–15 Dec 1993.

[12] B. Zeigler and P. Hammonds. Modeling and simulation-based data engineering: introducing pragmatics into ontologies for net-centric information exchange. London: Academic Press, 2007.

[13] H. Lee and B. Zeigler. System entity structure ontological data fusion process integrated with c2 systems. pages 4: 206–225. J Defense Model Simul Appl Methodol Technol, 2010.

[14] B. Zeigler. Object-oriented simulation with hierarchical, modular models: inteligent agents and endomorphic systems. San Diego, CA: Academic Press Professional, 1990.

[15] B. Zeigler. Multifaceted modeling and discrete event simulation. Orlando, FL: Academic Press, 1984.

[16] Umut Durak. Extending the knowledge discovery metamodel for architecture-driven simulation, modernization. page Vol. 91(12) 1052–1067. Simulation: Transactions of the Society for Modeling and Simulation International, 2015.

[17] Bikash Chandra Karmokar, Umut Durak, Sven Hartmann, and Bernard P. Ziegler. Towards a standard computational representation for system entity structures.

[18] B. P. Zeigler and P. E. Hammonds. Modeling and simulation-based data engineering: introducing pragmatics into ontologies for net-centric information exchange. Elsevier, 2007.

[19] H. S. Thompson, N. Mendelsohn, D. Beech, and M. Maloney. W3c xml schema definition language (xsd) 1.1 part 1: Structures. page W3C Working Draft Dec vol.3. The World Wide Web Consortium (W3C), 2009.

[20] Bernhard Kaiser. Application story of odd as part of safety assurance: The significance of a well-structured odd specification for the ad safety and sotif process. ANSYS, June 2021.

[21] Asam sim:guide, standardization for highly automated driving. ASAM e.V.

[22] U. Durak, S. Jafer, R. Wittman, S. Mittal, S. Hartmann, and B. P. Zeigler. Computational representation for a simulation scenario definition language. page page 1398. In 2018 AIAA Modeling and Simulation Technologies Conference.

[23] B. Chandra Karmokar, U. Durak, S. Jafer, B. N. Chhaya, and S. Hartmann. Tools for scenario development using system entity structures. page page 1712. In AIAA Scitech 2019 Forum.

[24] B. P. Zeigler and P. E. Hammonds. Modeling and simulation-based data engineering: Introducing pragmatics into ontologies for net-centric information exchange. Elsevier, 2007.

[25] Stefan Riedmaier, Thomas Ponn, Dieter Ludwig, Bernhard Schick, and Frank Diermeyer. Survey on scenario-based safety assessment of automated vehicles.

[26] Demin Nalic, Tomislav Mihalj, Maximilian Baeumler, and Matthias Lehmann. Scenario based testing of automated driving systems: A literature survey.