Scenario-based X-in-the-Loop Test for Development of Driving Automation

Felix Reisgys*, Johannes Plaum*, Andreas Schwarzhaupt* and Eric Sax**

Abstract: Even though scenario-based X-in-the-Loop testing has been evaluated in multiple research projects, there are still few application examples in series development projects. Based on current challenges in ADAS/AD testing, five requirements for scenario-based XiL testing are derived. Subsequently, we define and discuss the following three use cases along a reference ADAS/AD development process, each supplemented by respective example applications: Automated concept evaluation during Software (SW) Requirements Analysis, support of SW Unit Verification and automated validation in SW Qualification Test. All three use cases can contribute significantly to requirement fulfillment. This especially applies to use case 3.

Keywords: Driving Automation, Scenario-based testing, V&V, XiL

1 Introduction

Validation & Verification (V&V) of Advanced Driver Assistance Systems (ADAS) and Highly Automated Driving (HAD) has become more and more challenging due to the increase of testing width and depth as driving automation advances. Consequently, it is crucial to adjust and extend the existing state-of-the-art V&V processes. [1] [2] A promising way currently discussed in multiple research projects [3] [4] [5] is to combine so-called scenario-based testing with X-in-the-Loop (XiL) environments. However, only few industry applications have been presented so far. We will define a set of requirements for new test approaches and discuss the potential of scenario-based XiL test utilizing three possible use cases.

2 State of the Art

2.1 Existing Development Process Models

Current ADAS development projects deploy systems engineering and SW development process models which can be combined and adjusted depending on project needs.

The **Stage Gate Model** divides the project in several stages and connects them via gates. A stage contains activities to produce deliverables and gather information to reduce project risk. A gate represents the end of the previous stage and the beginning of

*Felix Reisgys, Johannes Plaum and Andreas Schwarzhaupt are with Daimler Truck AG (e-mail: {felix_manuel.reisgys, johannes.plaum, andreas.schwarzhaupt}@daimlertruck.com).

**Eric Sax is with Institut für Technik der Informationsverarbeitung at Karlsruher Institut für Technologie (e-mail: eric.sax@kit.edu).

the next stage and requires a decision on whether the project will be continued, paused or terminated. [6] Development processes in automotive industry may contain overlaps between stages [7] and involve samples (A-sample to D-sample) to represent gates with increasing system maturity [8].

The **V-Model** describes a well-established development process. In its initial publication, the V-Model 97 was composed of submodels project management, software development, quality assurance and configuration management [9]. Its major principle is to divide the system into multiple subsystems which are developed in parallel. Subsystem testing and integration are performed to verify that the system fulfills all requirements. The V-Process referenced in other publications such as ISO 26262 [10] and Automotive SPICE [11] is described in the submodel software development.

Agile Software Development follows an iterative and flexible process to increase efficiency and manage uncertainties better by leveraging developer competencies [12]. The Agile Manifesto published in 2001 includes four values and twelve leading principles [13]. Among different methods, Scrum has become the most commonly used process [14].

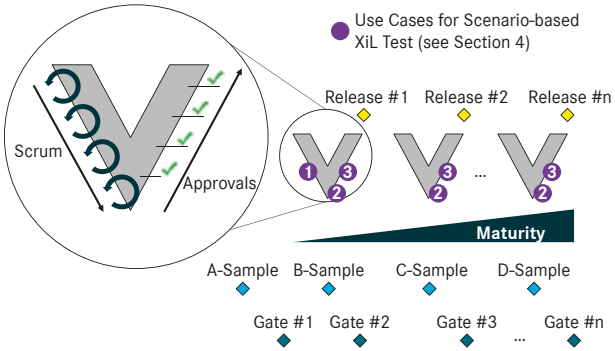

Figure 1: Custom ADAS Development Process

By combining all three models, the **custom industry process for ADAS development** as displayed in Figure 1 is derived and shall serve as reference for the following sections. On the highest level, a Stage Gate Model based process defines maturity gates for system releases. Each release is following the design and test process as defined by the V-Model. On the working level, implementation of SW units ("SW component that is not subdivided into other components" [15]) are performed based on a Scrum process.

2.2 Scenario-based Testing

As an approach applied in research projects such as PEGASUS [3], VVM [4] or SET Level [5], scenario-based testing has gained increasing relevance. Its general concept is to use a scenario as part of a test case description in order to aggregate redundant test inputs into equivalence classes [16]. A scenario is defined as a "temporal development between several scenes in a sequence of scenes" [17], where a scene is a snapshot of the ego vehicle's environment including static and dynamic objects [17]. The information aggregated in

a scenario can be structured as defined in the six-layer model: street layer (1), traffic infrastructure (2), temporal modification of layers 1 and 2 (3), movable objects (4), environment conditions (5) and digital information (6) [18]. Scenarios can be categorized as follows [19]:

- **Functional scenarios**: A consistent natural language scenario description of entities and their relations on a semantic level with a use-case dependent varying degree of detail (e.g. movement of vehicles, road topology). Descriptions may be supplemented by illustrations.
- **Logical scenarios**: A formal scenario description providing scenario parameter ranges in a state space to represent entities and their relations (e.g. velocity, distance). It is optionally possible to define distributions and constraints for parameters (e.g. $v_1 > v_2$).
- **Concrete scenarios**: A formal scenario description based on a logical scenario, additionally providing explicit scenario parameter values (e.g. $v_1 = 10m/s$).

Test cases to be reused consistently in every release (see Figure 1) can be composed from a scenario and respective evaluation criteria as well as requirements for test execution [17]. A potential approach for scenario-based testing is to generate concrete test cases from a logical scenario description by applying a parameter variation [3].

2.3 X-in-the-Loop Testing

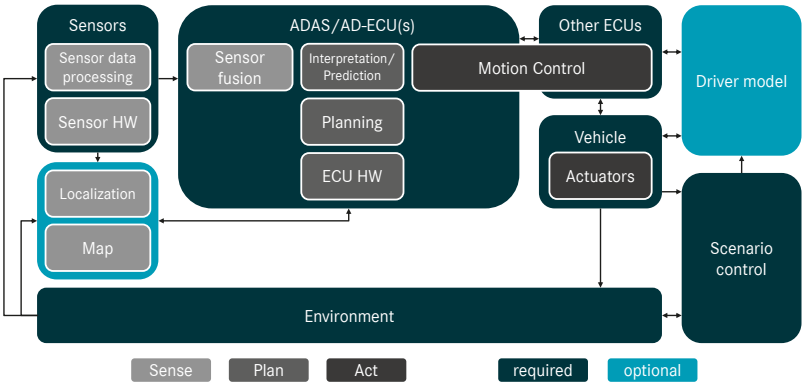

Figure 2: XiL Structure for Driving Automation Systems

By definition, X-in-the-Loop test environments stimulate a System under Test (SUT) by processing system outputs in a feedback loop [20]. They can be distinguished from open loop environments which do not contain a closed feedback loop [21]. The "X" refers to the SUT representation which can be a model, a SW or hardware (HW) for example [20]. A XiL environment for ADAS/AD testing can be structured as illustrated

in Figure 2. A driver model is only required up to SAE Level 3 to represent driver control acitivity, however is not part of the SUT. Depending on the representation (simulated, emulated or real) [22] of the systems interfacing with the SUT, the following XiL variants are commonly used [20]:

- Model-in-the-Loop (MiL): A fully virtual environment to test SUT simulation models created from a model-based design process.
- Software-in-the-Loop (SiL): A fully virtual environment to test SUT SW code.
- Hardware-in-the-Loop (HiL): A partly virtual environment to test a SUT on target HW (ECU).

3 Requirements for Future Test Methods

Even though real-world testing capacities are limited due to economical and practical reasons, it still plays a major role in the development process [10]. Nevertheless, there are multiple trends in testing of driving automation leading to a growth of the overall testing scope:

- Both new driver assistance systems and the extension of their sensor suite lead to a higher variance of scenarios to be tested [2]. Consequently, testing width increases.
- In addition, customer expectations in regards to quality impose additional requirements in the testing depth of each system.
- All systems assigned to SAE Level 3 and above need to be tested in scenarios where (temporarily) no driver is present as fall-back for the system [23]. This increases both testing width and depth.

Given the challenges, future testing methods need to meet the following requirements:

1. **Reduction of real-world testing share**: Since real-world testing capacities are limited, their overall share of all testing efforts should be reduced.
2. **Increase of test coverage**: The number of known and untested scenarios should be quantified and minimized.
3. **Independence of test validity**: Whenever simulation is used, it has to be ensured that its validity is evaluated and considered in the overall test statement.
4. **Identification of new scenarios**: The share of unknown scenarios should be minimized by extending the existing database by new logical scenarios.
5. **Efficient scenario prioritization**: Test plans and the overall test statement should consider the relevance of different scenarios.

The combination of scenario-based testing and XiL environments allows to collect and aggregate scenarios from sources such as expert knowledge, driving data or accident databases [24] in a structured way to test them in different test environments. This especially facilitates reproducibility and consistency of test execution on different platforms. Furthermore, it is possible to scale up the number of test cases and eliminate redundant scenarios, leading to an increased test efficiency.

4 Example Use Cases for Scenario-based XiL-Test

In this section, we will introduce three potential use cases to deploy scenario-based XiL testing in a development process for all degrees of driving automation. Each use case includes an example application in an industry reference development project and can be allocated to the example development process as indicated by the numbers in Figure 1. Use cases 2 and 3 can be repeated for every release. Within scenario-based XiL testing, the representation of the SUT and its environments may vary [22]. Based on the XiL structure in Figure 2, Table 1 provides an overview of the subsystem representation in each use case. Furthermore, Table 1 indicates which subsystems can be part of the SUT depending on the use case.

Use Cases / Subsystems	1 SW Requirements Analysis	2 SW Unit Verification	3 SW Qualification Test
Sensor data processing	simulated	simulated	simulated or real
Sensor HW	simulated	simulated	simulated or real
Localization	simulated	simulated	simulated or real
Map	simulated	simulated	simulated or real
Sensor fusion	simulated	simulated or real	real
Interpretation/ Prediction	simulated	real	real
Planning	simulated	real	real
ECU HW	simulated	simulated or real	simulated or real
Motion Control	simulated	real	real
Actuators	simulated	simulated	simulated

SUT | optional SUT | not SUT

Table 1: XiL SUT Representation by Use Case

4.1 Use Case 1: Software Requirements Analysis

Scenario-based XiL test can already support the definition of system requirements and the feasibility evaluation of a specific solution as long as there is no series SUT SW or HW available.

4.1.1 Overview

In this use case, varying concepts of the SUT based on a behavioral model are evaluated for different logical scenarios derived from the intended Operational Design Domain (ODD). Both concept parameters and ODD can be adjusted during the requirements analysis process. The SUT is represented by simulation models to be integrated into MiL or SiL environments. Scenarios can be reused along the whole subsequent development

process [25]. Overall, this use case represents an addition to existing processes in the SW Requirements Analysis.

4.1.2 Example Application

The example application supports the model-based specification of a Moving Off Information System (MOIS) on SW module (multiple units) level. A MOIS is a driver warning system for Vulnerable Road Users (VRUs) at low vehicle speeds [26]. The MOIS function concept is implemented as a Functional Mock-up Unit (FMU) to be integrated as SUT into a SiL environment containing surrounding AUTOSAR SW components (SWC) of the target ECU as well as an environment and sensor simulation. Technically, there is no feedback loop in this example application as the SUT is a warning-only system.

Scenario-based testing is used here to vary parameters of the MOIS simulation model and consequently optimize the SUT specification. In addition, it is possible to perform error injections to sensor signals to evaluate the robustness of the concept. Finally, a back-to-back proving ground test can be used to validate the SiL results and yields data for repeatable open-loop tests. Pass-fail criteria (e.g. MOIS information/warning issued) allow a concept evaluation for each logical scenario.

4.2 Use Case 2: Software Unit Verification

SW Unit Verification does not necessarily require execution of all possible test cases after each unit change by the developer, but rather focus on specific aspects of the system behavior to ensure fast verification of units.

4.2.1 Overview

Depending on whether a single or multiple SW units are tested, the SUT may vary in this use case (see Table 1). Since scenario-based testing focuses on the functional system behavior, MiL or SiL environments are sufficient in most cases, while HiL is generally feasible as well if HW samples are available. Parameters can be varied both manually or by an automated algorithm.

4.2.2 Example Application

The example application deploys a SiL test environment for the SW Unit Verification of an Advanced Emergency Braking System (AEBS) which is integrated on a dedicated ADAS ECU using AUTOSAR. Consequently, the SUT is represented by those SWCs required for the AEBS functionality. This includes interpretation and planning algorithms, as well as sensor fusion and motion control. All remaining parts of the ECU such as AUTOSAR base SW and HW are represented by simplified simulation models. The same applies to environment perception sensors, other ECUs and vehicle actuators.

Parameter variation to generate concrete scenarios is conducted via a Python script. Developers can define and adjust parameter sets manually and deterministically repeat identical scenarios multiple times as back-to-back tests. Inputs by the ego vehicle driver and other objects are fed into the simulation loop by Python modules. Test evaluation by the developer is supported by criticality metrics (e.g. Time-to-Collision) or pass/fail criteria (e.g. collision yes/no), both depending on the respective SUT and scenario.

4.3 Use Case 3: Software Qualification Test

SW Qualification Test targets an increased testing width and depth compared to use case 2. The goal is a comprehensive V&V of the SW modules in the relevant logical scenarios through a parameter variation that covers the possible parameter space.

4.3.1 Overview

A typical SUT to test an ADAS or HAD system includes subsystems sensor fusion, interpretation, prediction, planning and motion control SW (see Table 1). SiL environments leverage high scalability of scenario numbers to test a real SW module, while MiL is relevant to test single SW units. HiL environments are deployed to test the SW integrated on ECU(s), optionally even in combination with real sensor HW. A major approach for scenario-based XiL in SW Qualification Tests is parameter variation to generate concrete scenarios [27]. Furthermore, SW Qualification Tests can also be executed as regression tests in a continuous testing and integration workflow to track SUT performance.

4.3.2 Example Application

The application example deploys scenario-based testing for a risk evaluation for the release of an HAD system, as introduced in [28]. Parameter variation is applied using sampling algorithms from reliability analysis and parameter distributions derived from real world data. For the safety assessment of the concrete scenarios, different criticality metrics such as TTC or Time Headway (THW) are combined into an overall scenario criticality score.

As an example, an entering highway scenario of a heavy duty vehicle is implemented in a Python environment to generate concrete scenarios. Sampling and analysis are performed in an optimization SW. In addition to the safety metrics, additional observers (e.g. "merge performed") provide a qualitative overview of vehicle behavior in different parameter regions. The two evaluation levels enable the identification of potentially hazardous scenarios occurring in the logical scenarios when testing new functions.

5 Discussion

The contribution of each use case towards the fulfillment of the requirements defined in Section 3, is discussed in this section and summarized in Table 2.

Regarding the first requirement, both use case 1 and 2 are only used as a supplement to existing methods with low impact on the share of real-world testing. In contrast, the SW Qualification Test use case is supposed to reduce the scope of real-world testing by focusing on those scenarios with high relevance.

Through the continuous use of test cases along the development process, all use cases contribute to the second requirement. Especially the systematic test case generation in use cases 1 and 3 provides high potential to improve test coverage, while the manual scenario selection process in use case 2 leads to a "medium" rating.

Both use cases 1 and 3 are highly dependent on XiL validity [29]. This is especially critical for use case 1 as there is no real SUT test data available for XiL validation and reliability of simulation results (e.g. criticality metrics) may be limited. SW Unit

Verification does not impose high XiL validity demands as it is executed via white box tests focusing on specific SUT aspects.

Use cases 1 and 3 have a high potential to fulfill the fourth requirement. The combination with established processes like a hazard and risk analysis in use case 1 and the integration of data-driven approaches for use case 3 enable the enhancement of a scenario database [16]. In contrast, use case 2 concentrates on a fixed set of scenarios.

Especially in use cases 1 and 3, e.g. coverage-based [30] or criticality-based [28] selections facilitate efficient scenario prioritization to meet requirement 5. SW Unit Verification can also make use of these prioritization methods, however typically involves a manual selection which limits its potential [31].

Overall, all use cases are beneficial for development of both ADAS and highly automated driving. While use case 1 and 2 can be considered as an addition to existing approaches, use case 3 offers significant potential to meet the requirements defined in Section 3 (see Table 2), especially for SAE level 3 and above. The ability to manage growing ODDs as well as the consistency and traceability of scenarios along all use cases and the whole development process are major benefits of scenario-based testing.

Requirements \ Use Cases	1 SW Requirements Analysis	2 SW Unit Verification	3 SW Qualification Test
1 Reduction of real-world testing share	low	low	high
2 Increase of test coverage	high	medium	high
3 Independence of test validity	low	high	medium
4 Identification of new scenarios	high	low	high
5 Efficient scenario prioritization	high	medium	high

Table 2: Degree of Requirement Fulfillment

6 Summary and Outlook

The main contribution of this work is the systematic integration of three use cases for scenario-based XiL testing into the development process of driving automation: SW Requirements Analysis, SW Unit Verification and SW Qualification Test. Each use case includes a concrete industry application example and is evaluated against previously defined requirements. Overall, all use cases can be regarded as a reasonable supplement for existing development processes. Especially the use of scenario-based XiL testing for SW Qualification Test is considered to meet the requirements to a high degree.

In future work, scenario-based XiL testing should be evaluated for additional use cases such as system application and vehicle integration testing to further facilitate continuous use of scenarios along the development process. Moreover, it is possible to derive multiple variants for each use case.

References

[1] W. Wachenfeld, *How Stochastic can Help to Introduce Automated Driving*. PhD Thesis, TU Darmstadt, 2017.

[2] D. A. Weitzel, *Absicherungsstrategien für Fahrerassistenzsysteme mit Umfeldwahrnehmung: Bericht zum Forschungsprojekt FE 82.0546/2012*, vol. 98 of *Berichte der Bundesanstalt für Straßenwesen Fahrzeugtechnik*. Bremen: Fachverl. NW, 2014.

[3] DLR, "PEGASUS Abschlussbericht Gesamtprojekt," 2020.

[4] European Center for Information and Communication Technologies – EICT GmbH, "VVM - Projekt," 2021. `https://www.vvm-projekt.de/projekt`.

[5] DLR, "SET Level - Projekt," 2021. `https://setlevel.de/projekt`.

[6] R. G. Cooper, "Perspective: The Stage-Gate Idea-to-Launch Process—Update, What's New, and NexGen Systems," *Journal of Product Innovation Management*, vol. 25, no. 3, pp. 213–232, 2008.

[7] G. Hab and R. Wagner, *Projektmanagement in der Automobilindustrie*. Wiesbaden: Springer Fachmedien Wiesbaden, 2017.

[8] E. Sax, ed., *Automatisiertes Testen Eingebetteter Systeme in der Automobilindustrie*. München: Hanser Verlag, 1. auflage ed., 2008.

[9] T. Binder, D. Winkler, and S. Biffl, "Quo Vadis V-Modell," 2006.

[10] ISO, "ISO 26262:2018: Road vehicles - Functional safety," 2018.

[11] VDA QMC Working Group 13 / Automotive SIG, "Automotive SPICE: Process Reference Model Process Assessment Model," 2017.

[12] T. Dyba and T. Dingsoyr, "What Do We Know about Agile Software Development?," *IEEE Software*, vol. 26, no. 5, pp. 6–9, 2009.

[13] Beck et al., "Manifest für Agile Softwareentwicklung," 2001. `https://agilemanifesto.org/iso/de/manifesto.html`.

[14] Digital.ai, "14th Annual State of Agile Report," 2020.

[15] ISO/IEC/IEEE, "ISO/IEC/IEEE 24765: International Standard - Systems and software engineering - Vocabulary," 2017.

[16] C. King, L. Ries, J. Langner, and E. Sax, "A Taxonomy and Survey on Validation Approaches for Automated Driving Systems," in *2020 IEEE International Symposium on Systems Engineering (ISSE)*, pp. 1–8, IEEE, 2020.

[17] S. Ulbrich, T. Menzel, A. Reschka, F. Schuldt, and M. Maurer, "Defining and Substantiating the Terms Scene, Situation, and Scenario for Automated Driving," in *2015 IEEE 18th International Conference on Intelligent Transportation Systems (ITSC 2015)*, (Piscataway, NJ), pp. 982–988, IEEE, 2015.

[18] J. Bock, R. Krajewski, L. Eckstein, J. Klimke, J. Sauerbier, and A. Zlocki, "Data Basis for Scenario-Based Validation of HAD on Highways," in *27th Aachen colloquium automobile and engine technology*, 2018.

[19] G. Bagschik, T. Menzel, A. Reschka, and M. Maurer, "Szenarien für Entwicklung, Absicherung und Test von automatisierten Fahrzeugen," in *11. Workshop Fahrerassistenzsysteme und automatisiertes Fahren* (Uni-DAS e. V., ed.), 2017.

[20] K. v. Neumann-Cosel, *Virtual Test Drive*. PhD Thesis, TU München, 2014.

[21] J. Bach, *Methoden und Ansätze für die Entwicklung und den Test prädiktiver Fahrzeugregelungsfunktionen*. PhD Thesis, Karlsruher Institut für Technologie, 2018.

[22] F. Schuldt, *Ein Beitrag für den methodischen Test von automatisierten Fahrfunktionen mit Hilfe von virtuellen Umgebungen*. PhD Thesis, TU Braunschweig, 2017.

[23] SAE, "Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles," 2018.

[24] A. Pütz, A. Zlocki, J. Bock, and L. Eckstein, "System validation of highly automated vehicles with a database of relevant traffic scenarios," in *12th ITS European Congress*, 2017.

[25] T. Braun, L. Ries, F. Körtke, L. Turner, S. Otten, and E. Sax, "Collection of Requirements and Model-based Approach for Scenario Description," in *Proceedings of the 7th International Conference on Vehicle Technology and Intelligent Transport Systems - Volume 1: VEHITS*, pp. 634–645, SciTePress, 2021.

[26] United Nations, "Proposal for a new UN Regulation on uniform provisions concerning the approval of motor vehicles with regard to the Moving Off Information System for the Detection of Pedestrians and Cyclists (Moving Off Information Systems (MOIS)): ECE/TRANS/WP.29/2020/122," 2020.

[27] S. Riedmaier, T. Ponn, D. Ludwig, B. Schick, and F. Diermeyer, "Survey on Scenario-Based Safety Assessment of Automated Vehicles," *IEEE Access*, vol. 8, 2020.

[28] M. Rasch, P. Ubben, T. Most, V. Bayer, and R. Niemeier, "Safety Assessment and Uncertainty Quantification of Automated Driver Assistance Systems using Stochastic Analysis Methods," in *NAFEMS World Congress 2019*, 2019.

[29] F. Reisgys, M. Elgharbawy, A. Schwarzhaupt, and E. Sax, "Argumentation on ADAS Simulation Validity using Aleatory and Epistemic Uncertainty Estimation," in *Proceedings of the Driving Simulation Conference 2021 Europe VR* (A. Kemeny, J.-R. Chardonnet, and F. Colombet, eds.), (Munich, Germany), pp. 25–32, 2021.

[30] Foretellix Ltd., "M-SDL Specification," 2021.

[31] D. Baumann, R. Pfeffer, and E. Sax, "Automatic generation of critical test cases for the development of highly automated driving functions," in *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*, pp. 1–5, 2021.