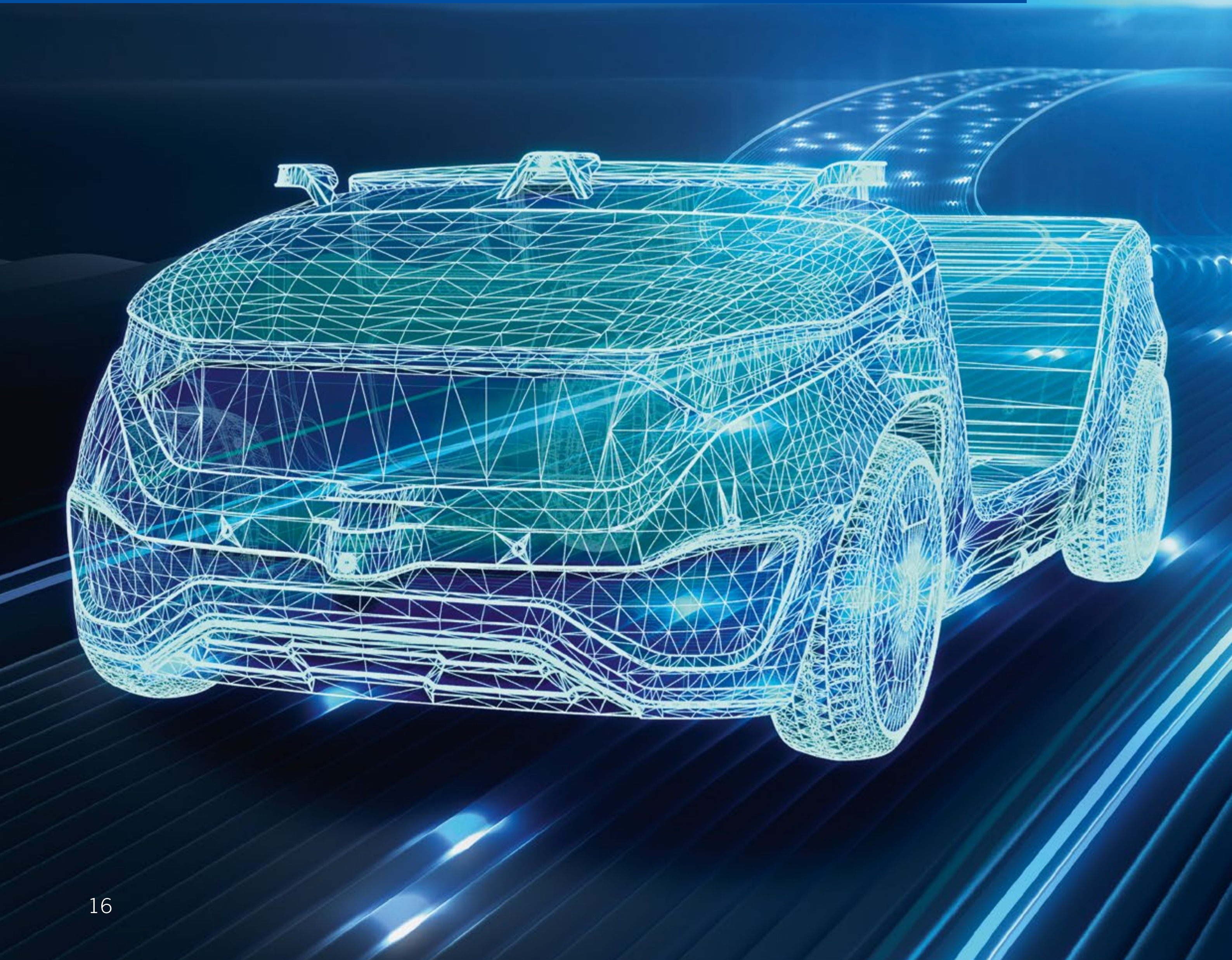
Virtual Test of Automated Driving Functions

The increasing complexity and variant diversity of automated driving functions can no longer be reliably managed by purely requirement-based test methods. In the context of its platform for highly automated driving HARRI, Bertrandt is further developing methods and tools for scenario-based testing of the corresponding functions and electronics.




© Bertrandt

AUTHORS

Dr.-Ing. Torsten Butz
is Head of Department Electronics
Development Chassis, Powertrain,
and Automated Driving at Bertrandt
in Ingolstadt (Germany).

Dipl.-Wirt.-Ing.
(FH) Simon Paleduhn
is Team Leader System
Development and Test
Management at Bertrandt
in Ingolstadt (Germany).

Dipl.-Ing. (FH) Alexander Merkel
is Head of Department Electronics
Development Diagnostics, Comfort,
ADAS, and Powertrain at Bertrandt
in Ehningen (Germany).

Dipl.-Ing. (FH)
Christian Bohner, MBA
is Team Leader Virtual Testing at
Bertrandt in Ingolstadt (Germany).

MOTIVATION

Driver assistance systems and automated driving functions are increasingly being used to improve road safety and to support the ongoing automation of road traffic. The enhanced complexity of the underlying electronics and software components, agile development methods, and the need to reduce the number of test vehicles also raise the requirements placed on suitable validation methods [1, 2]. Conventional test methods that pursue a requirement-based approach alone are no longer sufficient to cover all of the necessary validation aspects concerning function, robustness, and safety within shortening development cycles. A scenario-based test methodology and consequent virtualization make it possible to achieve a function-independent representation of the test conditions and to shift real test activities toward virtual test locations.

TEST METHODOLOGY

The further developed scenario-based test methodology, **FIGURE 1**, is derived from the research findings of the Pegasus project funded by the German Federal Ministry for Economic Affairs and Energy (BMWi) [3] and the normative standards of the Safety of the Intended Functionality [4]. The main components are functional test scenarios that are defined through an abstract linguistic description. They are parameterized by static information such as road geometries and weather conditions, as well as dynamic information such as maneuver descriptions of the ego vehicle and other road users. Translation into the universal OpenScenario format [5] and the determination of the parameter spaces result in logical scenarios aimed at guaranteeing tool-independent usability and transferability between the test locations. The scenarios are stored in a scenario database, which represents a function-independent test catalog.

Based on a criticality assessment, an automated test control system selects the relevant logical scenarios for the respective test focus and generates concrete scenarios by assigning specific values to the parameter spaces. Once the concrete test cases have been derived, they are assigned to virtual test instances, such as Model-in-the-Loop (MiL), Software-in-the-Loop (SiL) and Hardware-in-the-Loop (HiL) environments, or physical tests in the vehicle. When the test has been completed, the test case and parameter spaces are evaluated and their criticality is verified. The test results are stored in the results database and, if necessary, additional concrete scenarios are generated by a test re-adjustment in order to iteratively ensure that all important areas of the test case space have been covered.

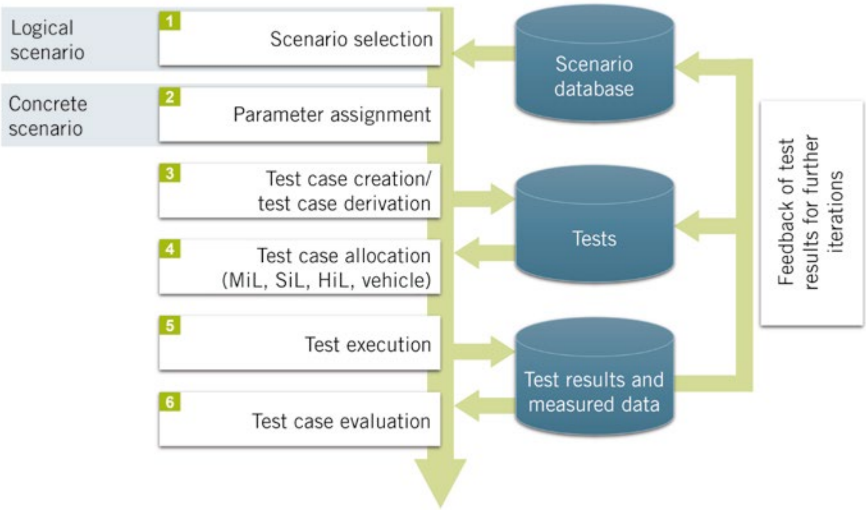


FIGURE 1 Scenario-based test methodology (© Bertrandt)

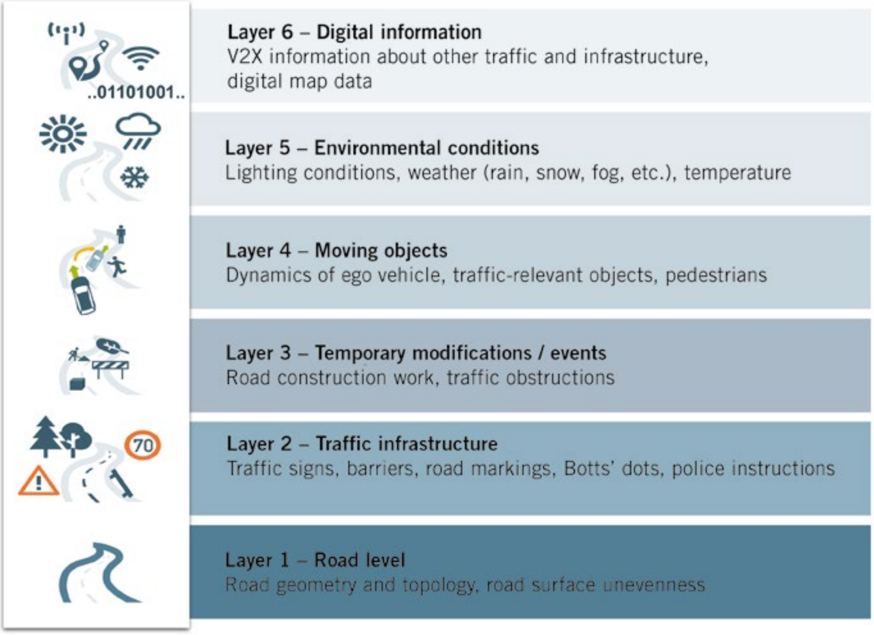


FIGURE 2 Six-layer model according to Pegasus [6] (© Bertrandt)

SCENARIO MODELING AND EVALUATION

The quality of the test results is significantly determined by the fact that the scenarios cover the relevant test case space and are based on valid data that reflect the reality on the road. Foundations for this are generic maneuvers from everyday road traffic as well as critical situations that are identified by evaluating field data or accident databases. The test scenarios are systematically described by using a six-layer model according to Pegasus [6], which consists of the levels road layout, traffic infrastructure, temporary modifications and events (for example road construction work), the dynamics of the ego vehicle and other road users, environmental conditions, and digital information (V2X), **FIGURE 2**. The information from the different layers can basically be combined arbitrarily in order to achieve a broad variation of test scenarios.

Due to the large number of possible scenarios and parameter variations, validating the complete test case space would result in exponentially high efforts, which is not feasible with regard to ever higher software release frequencies. Limiting the test to relevant concrete scenarios requires a criticality assessment that focuses the selection and value assignment of the available parameters on the parameter range that is of interest for the function to be tested. For this purpose, it is possible to use either predictive methods that do not require the test case to be carried out, or analyses that refine the parameter range as part of a test re-adjustment. Criticality measures such as time-to-collision, time-to-brake or distance-of-closest-encounter [7] can be separately defined for each scenario and stored in the scenario database.

A software module was developed for the concretization of the test scenarios, **FIGURE 3**. For the initial creation of the scenarios, the maneuver description for the dynamic road users is superimposed with static information from the other layers of the six-layer model. In the second step, the scenario parameters are combined with each other and the sampling rates for the respective parameter ranges are determined via a probability distribution. On the basis of the predictive criticality assessment, the relevant critical scenarios are identified and the test of the system boundaries is focused. Finally, a stochastic variation of the scenario parameters ensures that random scatter and deviations, which are unavoidable in real road traffic, are taken into account in the test. Within the scope of the test re-adjustment, a criticality assessment of scenarios that have already been tested results in the optimization of the dynamic and static scenario parameters, depending on how reliable the prediction of the scenario criticality proves to be.

VIRTUAL TEST TOOL CHAIN

Despite limiting the validation process to those scenarios identified as relevant, it is vital to have a test strategy that reduces the amount of testing required in the vehicle and – due to the computional time necessary for real-time simulation – in SiL and HiL environments to the necessary minimum. In fact, the aim is to completely virtualize the tool chain used, **FIGURE 4**, and, by transferring the testing process to the cloud, to achieve parallelization and simulation speeds that are faster than real-time.

While the previously mentioned tools for the scenario database, scenario concretization, results database, and test re-adjustment run on the client PC, the computing-intensive components of the virtual tool chain can be fully integrated into the cloud. Essentially, these components are the environmental simulation, the simulation of the virtual ECU (vECU), and the superordinate test automation.
In addition to the dynamics of the ego vehicle, the environmental simulation also covers the virtual traffic environment and models for environment sensors (for example camera, lidar, radar) which provide information on surrounding traffic in the form of point clouds, target lists, or object lists. The function to be tested is integrated via a vECU which allows the generation and integration of individual control algorithms or software components, up to complete ECU architectures including the related bus simulation. The vECU code is often connected via a func-

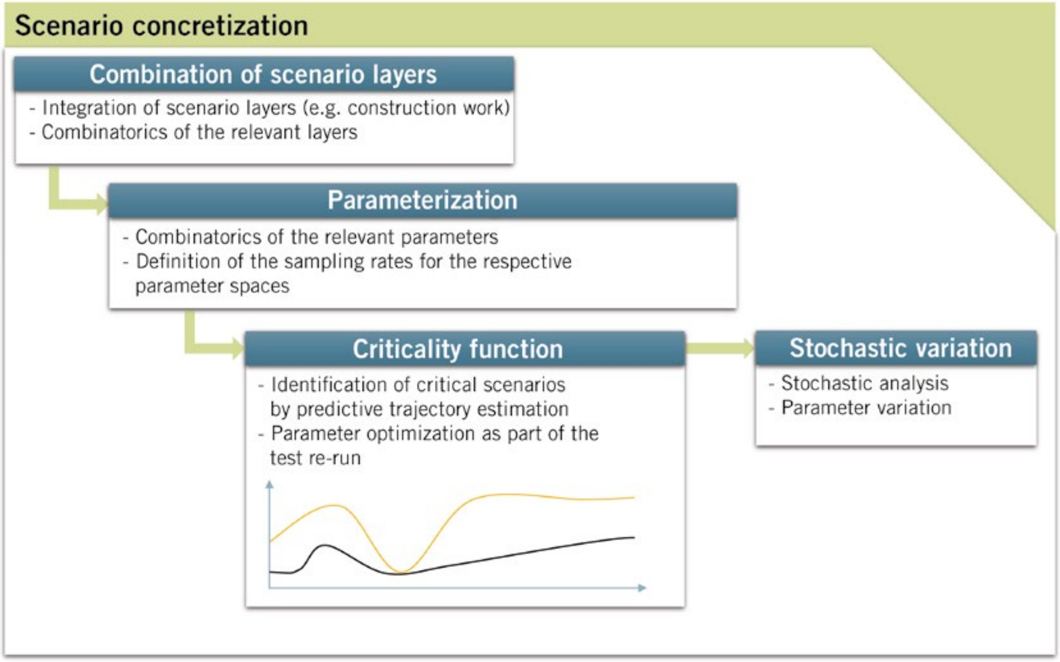


FIGURE 3 Module for scenario concretization (© Bertrandt)

tional mock-up interface. The test automation controls the simulation of the test scenarios and, at the end of the test, also performs the scenario-specific evaluation.

With regard to the functionality and maintainability of the tools employed, only commercial products were used for the relevant components of the virtual test tool chain. Two variants were implemented examplarily, which support both the Pegasus methodology and, perspectively, OpenScenario in the best possible way. On the one hand a homogeneous tool chain consisting of the simulation and vECU tools ASM, VEOS, and SystemDesk from dSpace is employed, on the other hand a heterogeneous combination of IPG CarMaker and QTronic Silver. In both cases, ECU-Test from TraceTronic serves as test automation.

There are different options for deployment into the cloud. If the tools support the use of Docker containers, porting from the PC into the cloud environment is relatively easy. Alternatively, it is also possible to operate the tool chain in a virtual machine, where additionally the operating system as well as CPU, GPU, and memory resources have to be emulated. In both cases, the virtual tool chains can be multiplied almost arbitrarily and the tests can be parallelized due to the scalability of the computing resources available in the cloud.

RESULTS

Within the context of the innovation platform for highly automated driving HARRI [8] developed by Bertrandt, the virtual test tool chain described above was used to test autonomous parking and maneu-

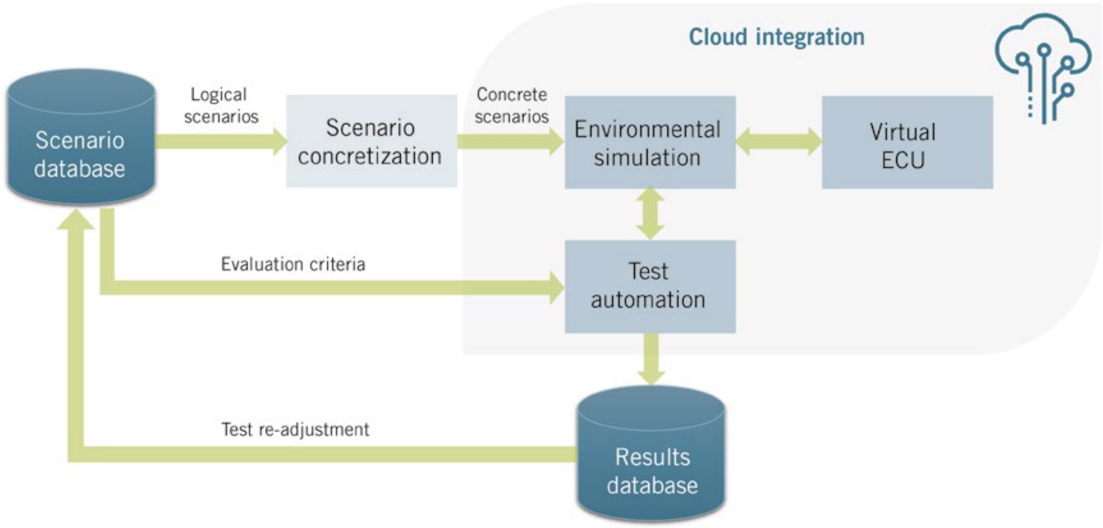


FIGURE 4 Virtual test tool chain (© Bertrandt)

vering functions. In addition, a proof of concept was implemented to test a driver assistance function for longitudinal vehicle guidance on freeways. The generic scenario catalogs that were created can also be used for higher autonomy levels. As part of customer projects, the results are being used to test predictive safety functions and other applications. The scenario description, which is independent of the test location, can also be used to generate control variables for driving robots for physical vehicle tests.

The results to date have shown that the scenario-based test methodology is more robust than conventional requirement-based testing with respect to changes to the requirements situation and the development process. In spite of the reusability of generic test scenarios and the efficient evaluation of test results, considerable effort and extent of tests are required in order to verify the correct operation and safety of automated driving functions. As functions become more complex, this can be achieved only by reducing the test scenarios on the basis of a criticality assessment and by consequent virtualization. Current studies focus on measuring and verifying the coverage of all possible test scenarios, as well as on enhancing the test case creation and evaluation by applying additional intelligence.

REFERENCES

[1] Linz, T.: Testen in Scrum-Projekten. Leitfaden für Softwarequalität in der agilen Welt: Aus- und Weiterbildung zum ISTQB Certified Agile Tester – Foundation Extension. Heidelberg: dpunkt.verlag, 2017

[2] Maurer, M.; Gerdes, J. C.; Lenz, B.; Winner, H. (ed.): Autonomes Fahren: Technische, rechtliche und gesellschaftliche Aspekte. Berlin/Heidelberg: Springer, 2015

[3] N. N.: Forschungsprojekt Pegasus. Online: https://www.pegasusprojekt.de/de/about-Pegasus, access: January 31, 2020

[4] International Organization for Standardization: ISO/PAS 21448:2019 Road vehicles – Safety of the intended functionality. Genf, 2019

[5] ASAM e. V.: OpenScenario. Online: https://www.asam.net/standards/detail/openscenario, access: January 31, 2020

[6] N. N.: Pegasus Scenario Database. Online: https://www.pegasusprojekt.de/files/tmpl/Pegasus-Abschlussveranstaltung/15_Scenario-Database.pdf, access: January 31, 2020

[7] Junietz, P.; Schneider, J.; Winner, H.: Metrik zur Bewertung der Kritikalität von Verkehrssituationen und -szenarien. 11th Uni-DAS e. V. Workshop Fahrerassistenz und automatisiertes Fahren, Walting, 2017

[8] Schiekofer, P.; Erdogan, Y.; Schindler, S.; Wendl M.: Machine Learning for Automated Driving. In: ATZworldwide 12/2019, pp. 46-49