Current Approaches for ECU Testing

What really matters

Further development of methods and tools for comprehensive and structured testing of ECUs is necessary not only for economic reasons. New highly complex technologies such as ADAS and autonomous driving also call for suitable testing strategies. This article presents current development activities and approaches as well as appropriate strategies by Vector.

A large portion of the value created in a modern vehicle is associated with the electronic and software systems. Malfunctions of electronics and software are accordingly high-risk and are not tolerable especially when used in safety-critical systems. That is why significant investments have been made in automated testing of these systems for many years. With each new technology, the question arises about the correct test methodology. Important factors here are the effectiveness, i.e., whether and how the test objectives are achieved, and the efficiency, i.e., optimization of costs and effort.

Model-based test design

In model-based testing, test sequences and data are generated from models. These models can either be a model of the ECU functionality or the test cases themselves. The first approach, thus the derivation of test cases from function models, should be viewed somewhat critically. If both the productive software and the corresponding test cases are generated from the same model, that promises large efficiency gains – the test practically drops out during development free of charge. But one should be aware of what is actually tested in such a constellation – in the extreme case, the code generator is merely being tested against the test case generator. Moreover, a function model largely contains the "good case" and thus describes how a device is supposed to function. The corresponding operating situations for the test of these functions can still be effectively derived from this model. However, it is difficult if not impossible to derive the many different error scenarios from the function model. But it is exactly these error scenarios that make the test so valuable since most problems don't arise in the "straightforward case".

For this reason, the Vector test solution works with a test model. This means that the test designer systematically models the situations in which the System Under Test must

exhibit correct functioning. A graphical test notation enables a clear and transparent representation of the abstract test sequence **(Figure 1)**. The test cases are generated from this high-level description. Variants are visible in the diagram and are taken into account during test case generation. The test models connect the specification and implementation of the test in an abstract and easy-to-understand way. This will also facilitate the frequently neglected quality assurance measures for the tests themselves, such as reviews by developers. Abstraction enables use of the same test specifications on different test systems and the reuse of previously specified tests – both leads to significant savings.

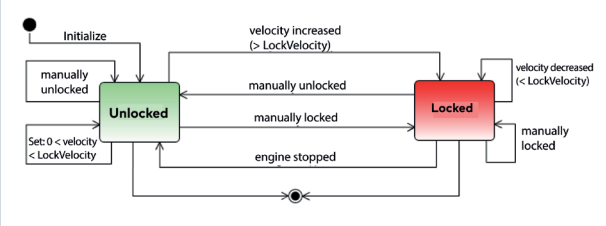

Figure 1: vTESTstudio provides various graphical test notations for test modeling

Hardware-in-the-Loop Tests

Hardware-in-the-Loop (HiL) tests are rightly considered today as a key component in assuring fully integrated ECUs. During these tests, the ECU runs in a simulated environment that takes into account the activities of the ECU which forms a closed loop with the environment model. The necessary environment model and the real-time-capable simulation system often turn out to be correspondingly complex. However, unlike the engine and vehicle dynamics controllers, for which the HiL approach was originally developed, not all ECUs today are so complex and designed as closed-loop controllers. For testing most ECUs in the vehicle, ranging from external lights to power windows, it is fully sufficient to stimulate the inputs and observe the corresponding reactions at the outputs.

As we know, it is less expensive to eliminate errors the earlier they are found. Elaborate test automation, complex models, and expensive test systems generally tendto be performed only later in the development cycle.The necessary resources are available only during the explicitly identified test phases. An agile procedure during development would be desirable that achieves high quality gains with little investment.

The art in selecting test systems is in identifying the right degree of depth, effort, and usability. With the combination of CANoe software and VT System hardware, Vector offers a test system that is scalable from the simple test tool at the developer's work station to the highly-automated HiL environment in the test lab. The core idea of the VT System is that all hardware functions that are needed for the testing of ECUs are combined into a modular system that is seamlessly integrated in CANoe. The test hardware surrounds the inputs and outputs like an envelope and includes the power supply and network connections of an ECU or a subsystem **(Figure 2)**. The pin function corresponding to stimulation, measurement, load simulation, error injection, and switching between simulation and original sensors/actuators is possible at each pin. These functions are designed universally so that a test system that is set up once can be used for different ECUs.

Besides the network environment, the physical environment can also be simulated by corresponding MATLAB/Simulink models in CANoe. A closed hardware-in-the-loop simulation is just as possible as a simple manual stimulation without a complex model. CANoe offers the same flexibility for the test automation. The list of options range from the above-described graphically notated test models

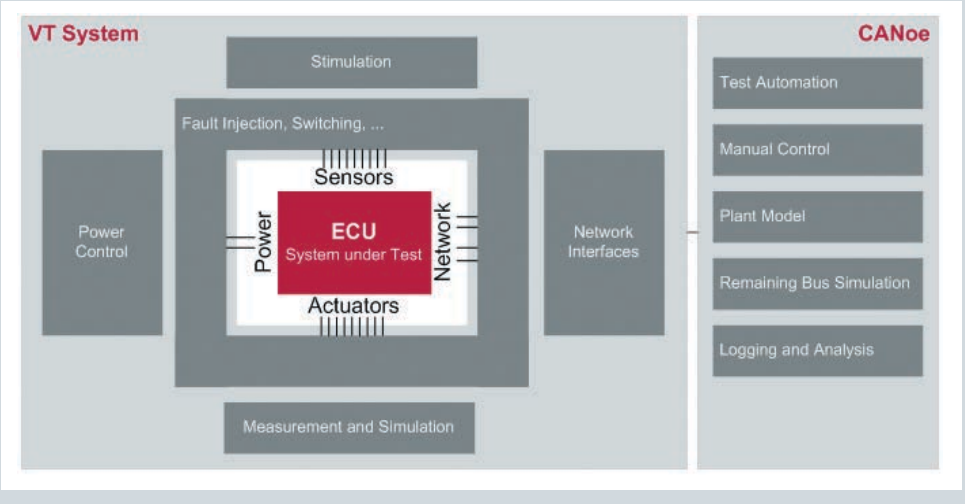

Figure 2: The VT System forms an envelope around the ECU to be tested and operates all inputs and outputs.

and the programming in different languages (CAPL, .NET/C#) to the defining of simple test sequences in table form. With vTESTstudio, a modern authoring tool is available. The manual performance of tests using the graphical user interface of CANoe and user-created panels is also possible. It is crucial that a powerful test tool with manageable complexity is available for all phases and user groups (such as ECU and software developers, component, ECU, and system testers). The tool has the same basis in all cases but has a different expansion depth.

Virtualization and Software-in-the-Loop Tests

Tests in virtual environments promise an enormous saving potential for two reasons: First, tests are expensive in real-world conditions and with real ECUs, subsystems, and vehicles. Real prototypes are needed, which requires stimulation and measurement of all inputs and outputs and physical effects by corresponding hardware. Second, systems can be tested earlier in virtual environments because it is not necessary to wait on the corresponding hardware and the integration of the various components. In addition, it is often difficult to correct the hard-to-reproduce and thus often expensive errors in rare situations with real devices. However this can be easily done in a virtual environment.

Virtual systems that reproduce real-world conditions as exactly as possible are common. For example, Vector offers an ECU simulator for integration tests of software components (AUTOSAR SWC) that even works with original ECU code. For other applications, such as system tests, model-based approach is recommended. If the functionality of an SWC is being tested, the configuration of a complete AUTOSAR stack is cumbersome. Here, it should be possible to work at a higher abstraction level, that models only the time behavior of the data exchange between the components, for example. How the data are mapped onto specific networks is of no interest here. The concentration down to the essential aspects reduces the complexity and increases the efficiency.

Another aspect concerns the possible variations in the test. What good is it to have a close-to-reality virtual environment when this reality is always changing, such as when a newer basic software version is used. For effective protection, an environment that can be variable adjusted using many parameters can be much more valuable. This leads to robust software or helps to reliably adjust or identify a difficult constellation under error conditions. The Vector software vVIRTUALtarget pro uses this approach.

Testing of Service-Oriented Communication

In automotive networking and the communication between the vehicle and the environment, a change to a service-oriented communication must be realized. The driver for this was the introduction of Ethernet in the vehicle and the "over-the-air" interfaces to the server back-end. This represents a significant paradigm shift for the test systems even if vehicle diagnostics has long used service-oriented protocols. Thus, for example, a recorded communication cannot be simply played back to the system. In contrast to signal-oriented protocols, that generally transmit signals through a simple sequence of messages, an interaction between the ECU and test system is needed for service-oriented protocols, at a minimum for registering of services.

The vehicle systems used up to now are statically designed. This means that the communication structures are specified during development and no longer changed. Service-oriented architectures offer the option of registering services dynamically and distributing them among different nodes in the network. Test systems must stay abreast of this development. However, the dynamics should not have to be programmed in the test scripts. Nor should the complexity of the protocols be visible for the test creator. With parameterizable interaction layers and OEM-specific modules for transport protocols, network management, and the like, CANoe has long provided a basis for efficient simulation and test case development in the former, signal-oriented systems. A comparable abstraction layer for the service-oriented communication is an important part of the further-developed concept of the Vector tools.

Services will become an integral component of the simulation and test automation languages with these concepts. The tool takes care of the mapping of the services onto the network level, the various protocols, and transmission mechanisms. The test author can thus concentrate on the relevant elements; an explicit and thus laborious and error-prone programming of the intermediate levels is not needed. Possible interventions on the various levels are nevertheless possible in order to provoke errors in transmission behavior for the test, for example. If corresponding ECU interfaces are present, communication on the service level in the ECU can also be accessed using these concepts.

Software Test

Another paradigm shift that is partially associated with service-oriented architectures is currently taking place in the ECU software. Up to now, ECUs have been designed as embedded systems whose firmware is trimmed for efficient use of the hardware and that are flashed on the devices with a fixed functionality and as a complete program executable. Many functions will be implemented in central ECUs in the future that are heavily oriented to classic IT systems. Various programs that can be individually replaced work together on powerful hardware on an elaborate operating system like Linux. These central ECUs serve largely as compute nodes on which a large portion of the

software runs. The control of peripherals, very time-critical routines, and probably also many safety-critical functions will still be performed by classic ECUs. AUTOSAR standardizes both systems and divides them into AUTOSAR classic and the new AUTOSAR adaptive, which is still in development.

These new software approaches for the automobile market raise many questions for the test systems and especially the test methodology. How can I ensure the quality of an ECU whose software is composed of different programs over its lifetime? The focus will shift in the future from the overall system of ECU hardware and software to the individual software functions. Where this is executed will play a subordinate role. Tests will therefore frequently take place where the software can be well observed – thus in virtual environments.

Dr. Stefan Krauß
Studied Computer Science and worked with Vector in Stuttgart/Germany since 2002 where he serves as Global Product Line Manager for the product line Tools for Networks and Distributed Systems.

Translation of a German publication in Automobil Elektronik issue 9-10/2016.