Efficient diagnostic simulation on CAN FD

Diagnostic simulation is the proven means when a test counterpart is not yet or no longer available. Whether in test preparation, in regression tests, or in training facilities. The targeted use of such a solution makes it possible to avoid mistakes.

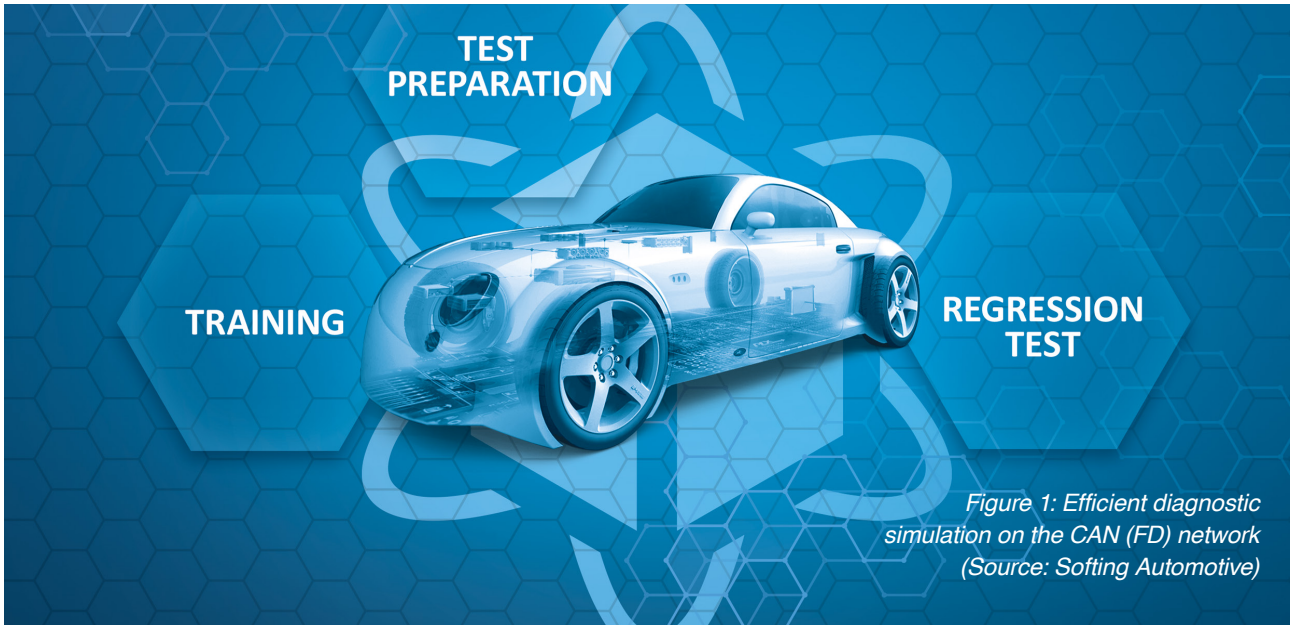

Figure 1: Efficient diagnostic simulation on the CAN (FD) network (Source: Softing Automotive)

The complexity of E/E networking is increasing all the time – and with it the necessary effort involved in testing. This is true both in terms of the validation of functionalities and in terms of ongoing regression tests of the test methods. Modern vehicles all have a large number of variants, which are usually shown by software configurations in combination with varying states of assembly. Different motorizations, for example, are generated both using a range of different engines and via coding. The number of variants also increases over the life cycle, as new software versions with changed behavior enter the field.

In diagnostic testers, this complexity should not be underestimated. After all, the test environment has to be adapted in each case, while ensuring that the existing functionality is not compromised. This has to be proven each time. As far as testing new functionalities is concerned, the question of validating the test environment is also raised on a regular basis. If in doubt, this can be left until the device under test (DuT) arrives. Troubleshooting between the test method, test environment (consisting of computer, vehicle communication interface, and cabling) and the DuT then takes considerable time, time which could well be missing later when it comes to testing.

As different as the two cases may seem, they have something in common: a lack of suitable counterparts. In the case of test preparation, this facilitates the error-free commissioning of the test setup. This means that when the DuT arrives, any error that occurs can be clearly assigned to it. This either results in a reduction of the test time or enables a much greater test depth and range in the same test time. In a regression test, as many installation and software variants as possible must be available to be able to make a valid statement. In practice, this is virtually impossible.

In both cases, a diagnostic simulation results in significant improvements as it offers a reliable, configurable counterpart. In a regression test, this basically means that all variants of all vehicles are available; in test preparation, the test environment can be approved together with the test methodology before the DuT is available. The simulation information is stored in files. The simulation files can be loaded onto the simulation device and started as part of automating the “tester test” in the program sequence. It is also possible to modify communication parameters via the interface and thus to verify the correct behavior of the tester. Such simulation files do not require much storage space and are stored centrally.

Mostly carried out via CAN

Communication between the tester and the diagnostic simulation - even though Ethernet is gaining importance in some applications - is still mostly carried out via CAN. Both, Classical CAN and CAN FD are used. For diagnostics, transmission rates of up to 1 Mbit/s must be supported for Classical CAN and 8 Mbit/s for CAN FD. Softing TCS is a modern diagnostic simulation consisting of the simulation hardware, a configuration application and an API (application programming interface) for integrating the hardware in test automations. The hardware is flexibly tailored to current and upcoming requirements thanks to the Multicore-Linux platform used. It has an ▷

Figure 2: The Softing TCS.device is a configurable diagnostic simulation and can be used as a replacement for real ECUs or vehicles (Source: Softing Automotive)

OBD (on-board diagnose) jack and thus represents a vehicle in entirety as far as diagnostics is concerned. Alternatively, CAN can be accessed in the usual way using a D-SUB jack. Simulation files for different ECUs (electronic control unit) and vehicles are loaded onto the device via a LAN connection or using a USB stick.

The behavior towards the tester is completely configured in simulation files; programming is not necessary. The simulation is generated on the ISO/OSI layer 4 level, i.e. above the CAN network. Communication mechanisms necessary in diagnostics, such as segmenting, flow control, etc., are automatically processed through the protocol stack in the simulation hardware but can be controlled via communication parameters. This is how the timing of a response can be controlled at all times, predefined in the simulation file, but also during runtime via the programming interface. As a minimum response time via the diagnostic CAN, the simulation allows approx. 1 ms. A slower response behavior can be set via the parameterization for checking the tester, even outside the protocol limits.

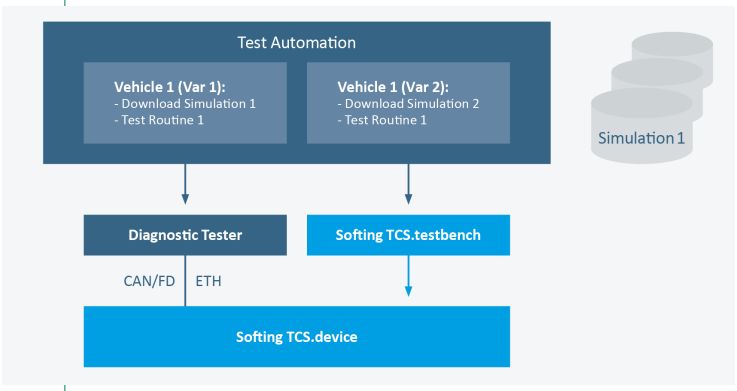

Figure 3: Integration in test automation (Source: Softing Automotive)

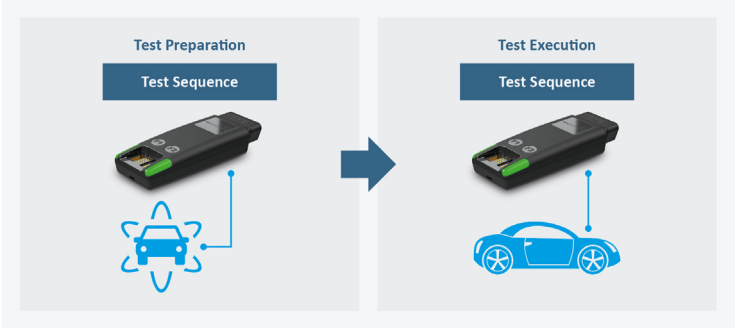

Figure 4: Simulation in test preparation (Source: Softing Automotive)

In the simplest case scenario, a message (request) sent by the tester is compared with exactly one response in the simulation file. A request for a measurement value (e.g. 22_h, 01_h, $2F_h$) is then responded by its corresponding message (e.g. 62_h, 01_h, $2F_h$, 01_hh, 23_h). Wildcards can be used as it is not always desirable to enter all requests specifically in a simulation. Using an 'X', any hex value in the request can lead to a response; a '…' would allow a variable length (e.g., 22_h, 01_h, $2X_h$, ..). Simple chains of effect have also been taken into consideration. This makes it possible to send a different response to a request each time. Changing values can thus be simulated, as can a negative response with the first request and a positive response with the following one.

Using variables is a major simplification. These can be set by a request – this is then marked accordingly in the configuration. Subsequently, the variable value in the response can be entered automatically. Thus, not all combinations have to be edited. Variables can also be used in communication control. A typical diagnostic example is session handling: Specific services can only be used in specific sessions. So, if the relevant service comes from the tester, the variable is set. In the case of relevant critical services, the variable is queried. If the variable is assigned appropriately, a valid response is received; otherwise a negative one.

The simulation files are created in different ways. The regular communication between a tester and the ECU or vehicle can be recorded for the regression test. A corresponding simulation file is then generated from the trace file at the push of a button. This works with the usual CAN formats as well as with the PCAP format. In test preparation, the diagnostic specification currently tends to take the form of an ODX file. This can be read in and the simulation can be created using the possible information. This either takes place manually for every service or automatically at the push of a button using definable rules. Both methods can also be used combined. Existing gaps or special cases can then be created manually. Overall, this enables very efficient creation to suit the particular application case.

In summary, diagnostic simulation is the proven means when a test counterpart is not yet or no longer available. Whether in test preparation, in regression tests, or in training facilities. The targeted use of a solution, such as Softing TCS, makes it possible to significantly avoid mistakes. Testing can be more specific and wider-ranging thanks to the time gained by earlier maturity. This means that the simulation not only saves a considerable amount of money, but is also the basis for improving quality.

Author

Markus Steffelbauer
Softing Automotive Electronics
info.automotive@softing.com
automotive.softing.com