All's Well That Tests Well

Well-organized tests make sure your simulator really works for you

A testmanager's nightmare: The hardware-in-the-loop (HIL) simulators are up and running, but do not reduce the workload as hoped. So does all of that test hardware really do any good? To make sure it does, a sound strategy and good organization are needed. dSPACE not only offers tailor-made HIL simulators and the AutomationDesk test automation software. We can also provide support for setting up large, complex test environments, making sure that HIL simulators run efficiently in the long term.

HIL Simulation in the Automotive Industry

Only a few years ago, hardware-in-the-loop (HIL) simulation was just another test method for single ECUs or ECU networks. Today, it is an integral part of the software release process. HIL simulators test every system size from single components to extremely large networked systems. They cover all domains, including the engine, transmission, vehicle dynamics, chassis, infotainment, and comfort.

The usefulness of HIL simulation is no longer in question. But to run the test systems efficiently in terms of time and cost, the test strategy and test organization have to be planned carefully.

Long-Term Efficiency

The large numbers of new functions to be tested, the growing complexity of ECUs, and last but not least, the confusing number of variants all add up to an enormous quantity of test cases. Test automation is the method of choice for handling them. Test automation means test sequences can be defined once and then run automatically, overnight or on weekends. As users design more and more tests, they can build up a valuable library from which further new tests and test variants can be created. The objective is to continue using elements from the library for subsequent vehicle generations, if possible without modifications. So test automation must be viewed as a long-term task that will even outlive the simulator on which it was first implemented.

> Test automation is a long-term task that will even outlive the simulator on which it was first implemented.

Starting Out Right

A test automation tool provides support for test implementation and test execution. It is not a substitute for actual test development: the test developers still have to do this themselves. But it can help users to organize their work well from the outset, and to build a library for handling their basic function blocks. AutomationDesk, the test automation software from dSPACE, makes the task of constructing test cases a lot easier. Tests are created in graphical form, and the resulting basic blocks can be simply saved to libraries and then reused (figure 1).

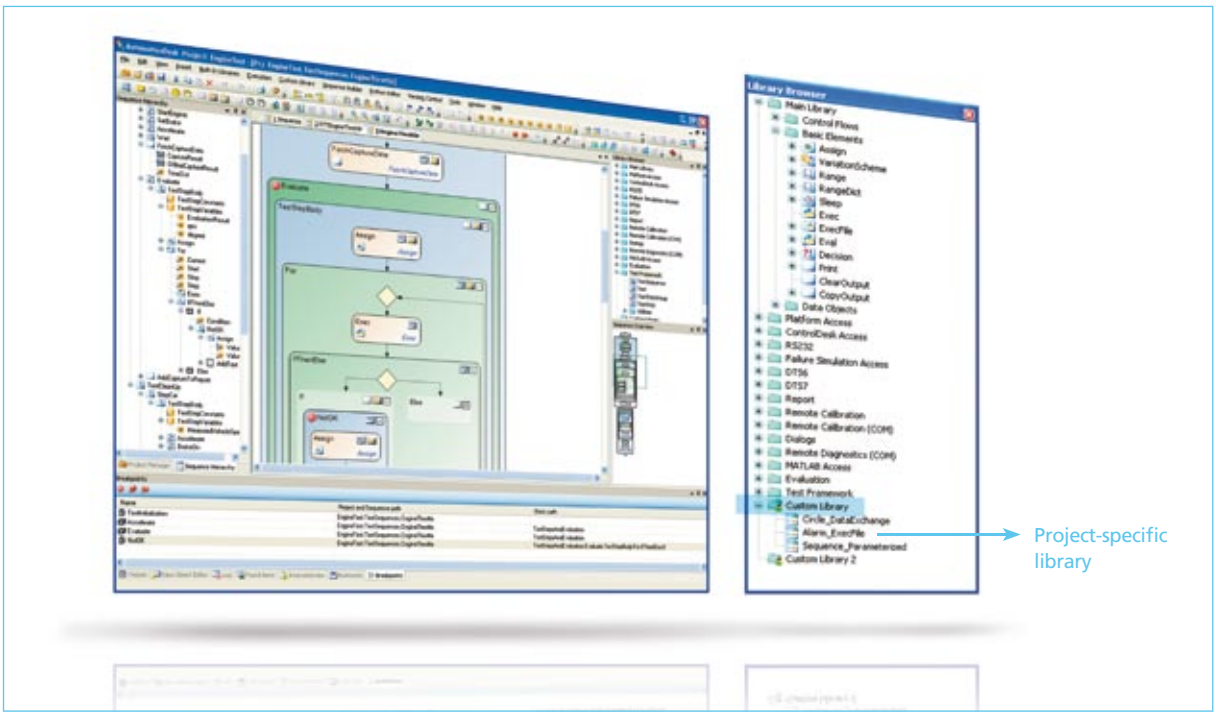
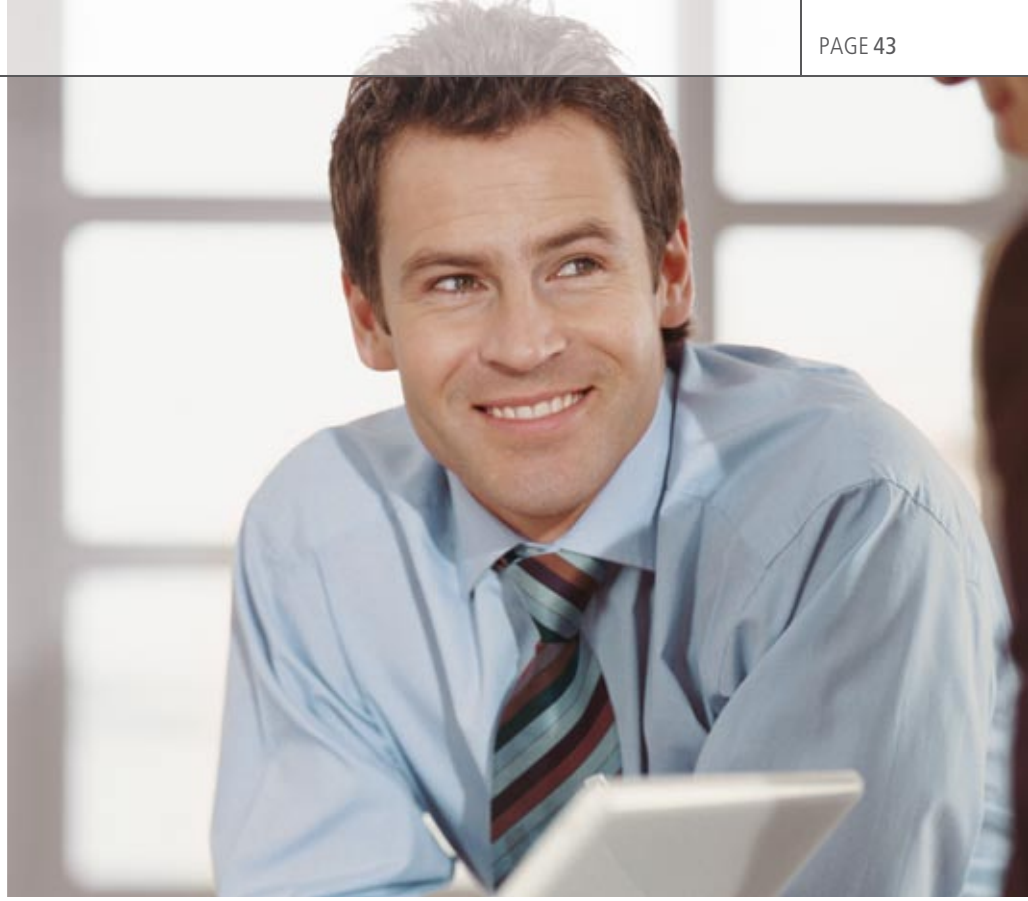


Figure 1: AutomationDesk supports the creation of project-specific libraries, for example, with Custom Libraries. The individual elements from the test cases can be added to the library by drag & drop.



Planning for Change

Some aspects can easily be overlooked, especially when test automation software is used for the first time. One of them is the fact that there may be additional vehicle series in the future, and the HIL systems may grow in size. The test cases have to be planned to allow for this. Even the test team itself might grow, and the test group structure has to accommodate that growth.

Reusing Test Cases

When testing begins, the focus is often on getting test results fast, without taking the time to set up a library. This makes it far more difficult to reuse tests later. Yet most HIL tests are so complex and elaborate that they cannot be handled and reused unless they are clearly organized.

Reuse means more than just using test sequences again in different test sets (for example, in regression testing). That type of reuse is easy to implement and involves very little work. Other types are more complex, involving the inclusion of similar components or even completely new ones.

1. Reusing a generic test for structurally similar test cases (for example, a single generic power window test for all windows, instead of four tests for four windows)
2. Reusing basic blocks on the same simulator for different test areas and issues
3. Reusing basic blocks and tests on different simulators, but for the same vehicle
4. Reusing basic blocks and tests for a new vehicle on the same simulator or a new simulator

The goal is to avoid redundancy in the library so that it is easier to maintain, and to use basic blocks that are as generic as possible to allow multiple reuse (figure 2). Generic, nonredundant basic blocks, combined with a suitable library concept, are the precondition for reusing test cases. If the test cases have to be adapted, such as when sensors have been modified, changes have to be made in only a very few places. The test cases that are derived are runnable almost immediately. If tests are created but there is no library concept, all the tests

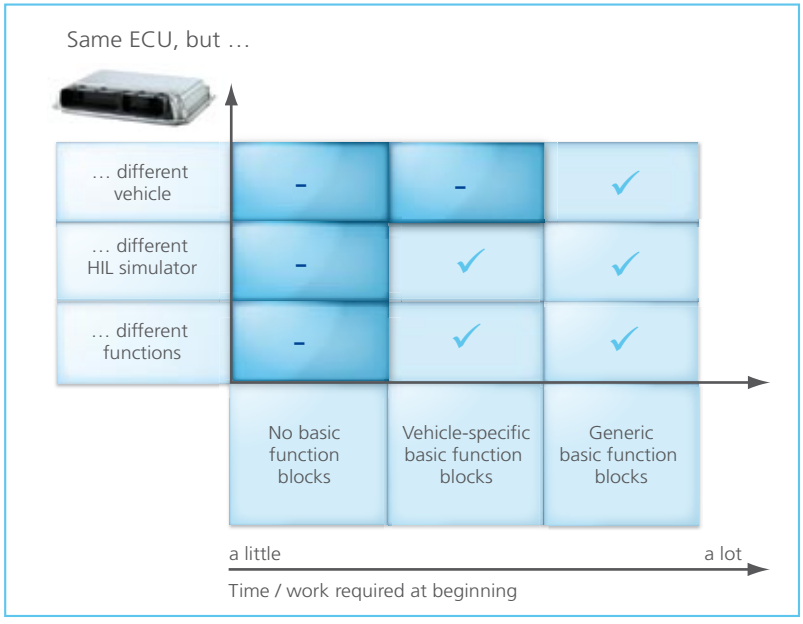
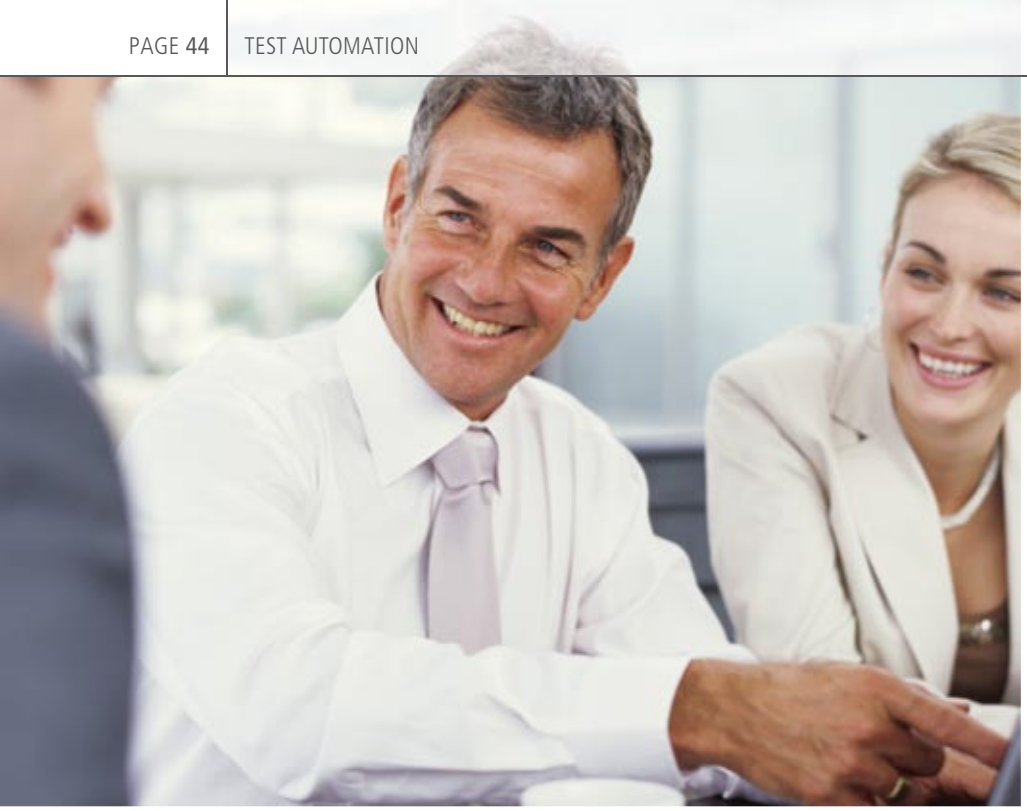


Figure 2: The initial effort invested in creating generic basic blocks simplifies test reuse later on in the process.



> A well-trained test team that is familiar with the test system is vital for the system's successful future use.

have to be reworked every time there is a change, such as when a new HIL simulator is used (figure 4). Introducing a sustainable test strategy may involve a relatively high initial workload, but the resulting libraries are the foundation for using simulators efficiently in the long term.

Quality ...

Creating, managing, and organizing test cases are not simple tasks to be performed on the side, even if there is software support. Test creation has many parallels with software development, where architecture and versioning are key elements. The quality of the work performed at the beginning largely determines how well tests can be reused later, and with how much effort. A well-trained test team that is familiar with the test system is vital for the system's future successful use.

... Means Certainty

When tests are run on a HIL simulator, developers have to be certain that errors found in a test run are really in the ECU code, and not in the tests themselves. So the first step is to specify, implement, and thoroughly test the basic blocks. Like software development, basic block development uses quality procedures such as clear function specifications, reviews, and official releases. The very first test cases are created in this way and then gradually extended. The extensions also undergo approval tests and release procedures. This provides the assurance that any errors that are found are in the ECU functions themselves, and the responsible ECU developers can classify the HIL simulation results as reliable (figure 3).

dSPACE Training and Engineering

A HIL simulator plus its associated software is a major initial investment. But this has to be seen in the right perspective: Even without a simulator, it costs a lot to run a test team over the years.
This makes it all the more important

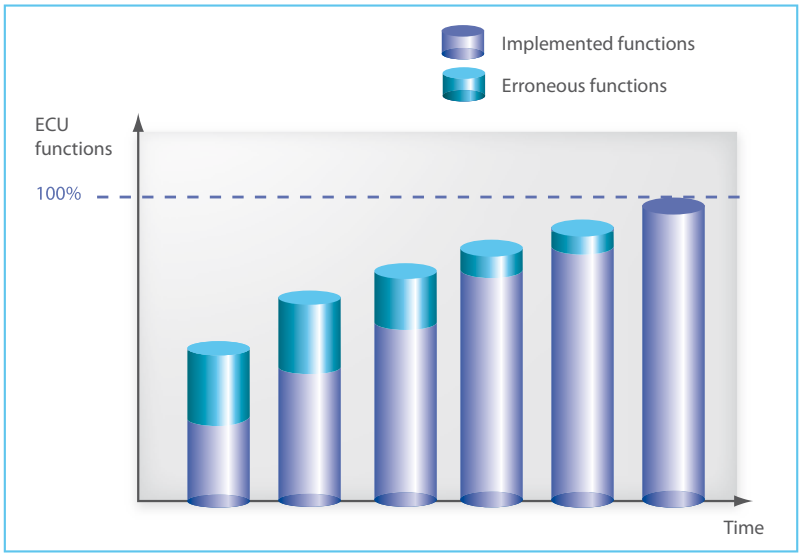


Figure 3: Using HIL simulation and test automation continuously enhances the quality of ECU functions.

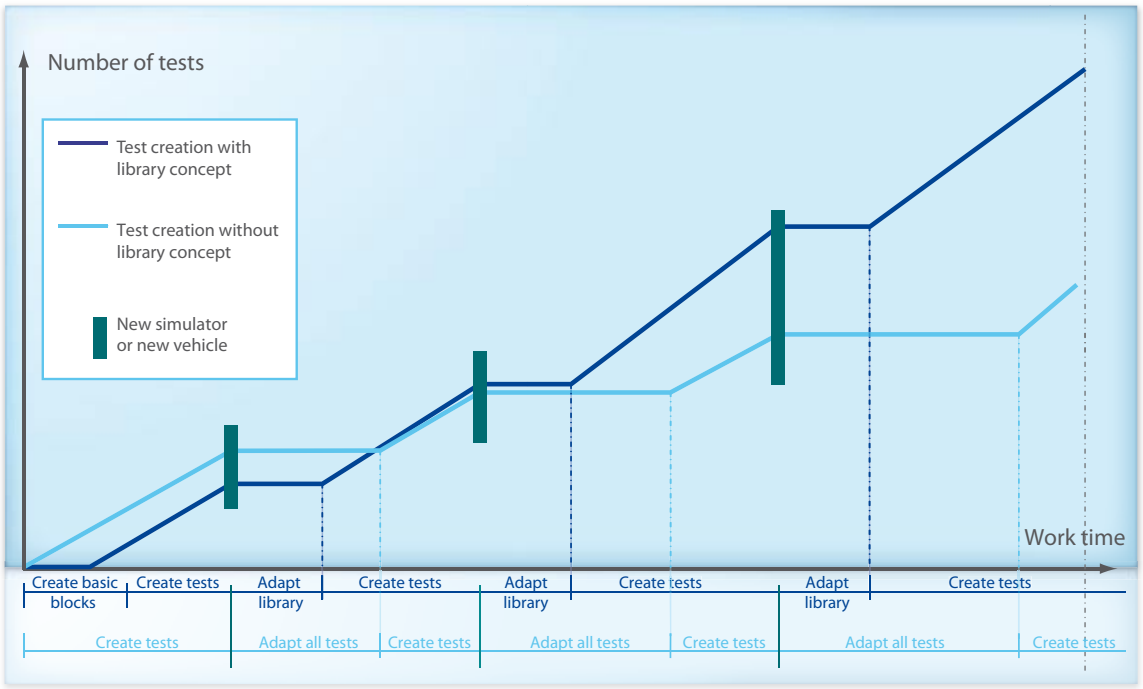


Figure 4: When a library concept is used, only the basic library blocks have to be adapted when a new simulator or a new vehicle is introduced.

to prepare test developers and managers thoroughly for working with the test software, enabling them to use it efficiently. dSPACE offers special training courses on AutomationDesk and on working with HIL simulators. The courses can be held on customers' premises if required, and can address specific requirements and needs.

For some years now, dSPACE has also been providing support for implementing test tasks. Engineering services for test automation start-up mean that users benefit from the dSPACE engineers' knowledge of the hardware and software, and from the project experience they have accumulated. The services include:

- Test processes
- Tests
- Project implementation
- Resident engineers
- Creating the test template
- Creating the library concept
- Example test implementation
- Data and result management
- Integrating third-party software
- Connecting the test software to existing tools

The knowledge and experience that dSPACE gains from engineering projects are used for further product developments, so that the products are always in tune with user requirements. Our products are market-driven, and we are actively involved in setting the trends, such as standardization formats. At the same time, users profit from our extensive experience when they set up their test environments. ■

Summary

- Test automation is indispensable to HIL simulation
- A well-structured library concept facilitates test reuse
- The initial effort pays off in the long term