Model-driven Test Case Generation in HiL Testing
Usage Models and Model-centric Testing

Finding the right test cases is a crucial task for testers, test managers, and project responsible. Audi AG tested various options in the course of several projects in order to be able to complete this task in a systematic way. In the end, it was model-driven test case generation which provided the most convincing solution: The generation of test cases on the basis of usage models, which is being further developed in the course of a collaborative project with the University of Erlangen-Nuremberg, as well as model-centric testing (.mzT) developed by the company sepp.med GmbH.
1 Introduction

In the automotive industry, it is only possible to fulfill new customer requirements and changing legal regulations with the development of increasingly distributed functionality on the basis of electronics and software. Audi AG recognized the relevance of software for the development of vehicles at a very early stage. In this area, testing is of highest significance as it is an important part of the validation of the increasingly complex functionality. The test methodology used by Audi is EXAM (Extended Automation Method). Test cases are executed automatically on HiL (hardware-in-the-loop) simulators. The test methodology EXAM requires the manual creation of individual test cases. Therefore, both the determination of coverage criteria, and the identification of scenarios that have not yet been tested are difficult tasks. It was possible to find a solution to these problems by introducing models and systematically deriving test cases from them. Audi AG’s goal is that the quality of its cars meets highest demands.

2 Extended Automation Method (Exam) Release 2.0

The test method EXAM (EXTended Automation Method) employed by Audi AG defines the process, the roles, and the tools used for the performance of test activities. Test cases are specified formally and platform-independently in UML sequence diagrams. Based on these diagrams, platform-specific code for the execution on the target system is automatically generated using reference libraries in Python. The test case specifications, the function libraries, and the test reports are stored in a central database. Therefore, the sharing of distributedly developed test cases and functionalities for test automation is supported. The test code is executed automatically on HiL test benches. Reports for the evaluation of test results are automatically generated.

However, each individual test case has to be created manually based on the specifications. Figure 1. Due to this it is difficult to figure out why a particular test case was defined or certain procedures were not tested. The determination and usage of coverage criteria or test management indicators for test case generation is hardly possible. Additionally, large effort is required for manual maintenance of existing test cases due to changes in requirements.

To overcome these flaws of the established procedure, model-driven approaches for test model description and test case generation have been adopted at the Audi AG:

- The first approach was successfully introduced in the course of a diploma thesis, and is being further developed in the course of a doctorate. This approach is based on usage models and allows the systematic distinction between usage behavior and system behavior.

- The second approach is the model-centric testing (.mzT) developed by the company sepp.med GmbH, which was successfully tested in first projects at Audi. This approach uses .mzT test models based on UML.

3 Usage-oriented Testing

A usage model describes all the usage scenarios that can be executed on the system under test (SUT). The linking between requirements and reference behavior can be realized by various ways, whereby the test designer has the option of selecting the best solution for his testing task. Test management information, such as average test case length or coverage criteria, can be determined even before test case generation. Results of executed test cases can be taken into account by the generation of new test cases, so that the test designer does not only have a view on the SUT but also on the previous test results when selecting new test cases. In this way, the usage model forms the basis for the decision which test cases should be executed and when this should be done.

A particular feature of a usage model is that the test subject is clearly presented and easy to understand as it exactly concentrates on this task. Various strategies can be used for the generation of test cases. Standard algorithms for test case generation, such as random test case generation, minimal edge coverage, or the generation of all possible paths, can be applied on usage models. Unfortunately, these algorithms are not optimal in the field of HiL testing as they do not ensure optimal utilization of the HiL test