Chapter 8
Model-Based Testing of Distributed Functions

Thomas Bauer
Fraunhofer IESE, Germany

Robert Eschbach
Fraunhofer IESE, Germany

ABSTRACT
The standard-compliant development of component-based embedded systems calls for systematic coverage of product requirements and for testing component interactions at the system integration stage. System functionality is represented by a set of complex distributed functions, i.e., functions that are spread across several system components. This chapter presents a novel automated model-based testing approach for distributed functions that uses informal system requirements and component behavior models. The test modeling notation makes it possible to model component interactions and composite functions with defined pre- and post-conditions. Test cases are automatically generated as scenarios of distributed functions represented by sequences of component interactions.

INTRODUCTION
In embedded systems, such as automotive or industrial automation, the majority of new innovations are realized via interactions of different system components. Examples are comfort functions in info- and entertainment systems or safety-critical driver assistance functions like park lane and brake assistants. In the past decade,
industrial development processes have been influenced by strict process and product standards, shorter development cycles, and higher quality needs. On the other hand, about 50% of all failures are caused by software faults, of which about 25% are caused by distributed software functions (Biagosch, Knupfer, Radtke, Näher & Zielke, 2005).

Complex embedded systems are often developed with third-party components provided by suppliers. The integration of third-party components requires particular efforts regarding systematic quality assurance on the part of the manufacturers, since neither the program code nor supplier-internal documentation is accessible. The manufacturer has to assure that the system assembled from externally developed components fulfills the system requirements and customer needs. A systematic integration test approach is needed to detect requirement violations and unspecified component interactions.

Integration and interoperability testing of distributed systems are essential quality assurance activities to check interactions within the system and functionalities that are spread across several system components. In integration testing, black-box testing techniques are usually applied, i.e., test cases are derived or selected based on the system specification. Integration testing is driven in practice by manual tests that highly depend on the experience of the test engineers and on their system knowledge. This approach leads to inefficient quality assurance in terms of effort, test coverage, and product quality.

By introducing model-based development approaches for software in embedded systems and standard-compliant component descriptions, different kinds of structural and behavioral models become available as valuable resources for early, systematic, and automated quality assurance on the component and subsystem levels.

Test automation usually deals with the use of automated methods supported by tools in software testing. In practice, available tools for test automation focus on the automated execution and evaluation of test cases, like x-unit frameworks for software unit testing, GUI tester and capture/replay tools for testing via the graphical user interface, and hardware-specific solutions for hardware-in-the-loop (HiL) testing of electronic control units (ECUs).

In this article, a combined model-based approach for the systematic integration testing of distributed systems is presented; it uses functional system requirements and component models to avoid the generation of large system models. The approach bridges the gap between the informal requirements from the system view and the detailed event flow from the component view. The main contribution of our work is the systematic formalization of functional system requirements and component models for testing models that enable the automated generation of integration test cases.
Implementation of the Personal Software Process in Academic Settings and Current Support Tools
www.igi-global.com/chapter/implementation-personal-software-process-academic/61213?camid=4v1a