

Formal Methods Based Acceptance Testing for AUTOSAR Exchangeability

Rickard Svenningsson and Rolf Johansson
SP Technical Research Inst of Sweden

Thomas Arts and Ulf Norell
Quviq AB

ABSTRACT

Nowadays, vehicle integrators (OEMs) put over fifty different control units (ECUs) in a vehicle. The AUTOSAR standard for Basic Software (BSW) describes in detail how the communication software components should be implemented in order to enable these ECUs to seamless work together. When integrating ECUs with BSW from different vendors, one needs to ensure that the standard is interpreted in the same way. This requires testing, but the traditional testing approach of manually crafted tests has failed due to the enormous configurability of the software. By using a novel approach based on automatically generated test cases from models and configurations, we solve this testing problem.

In addition to the obvious requirement that an acceptance test strategy must have high coverage with respect to specification requirements, it is also very important that the tests have a low life cycle cost (LCC). A low LCC implies as well low cost for developing the test as a high maintainability. Our solution does turn out to be more effective, but also more efficient than manually written test cases. This enables OEMs to allow software from different vendors in their ECUs, therewith reducing cost related to enforcing ECU vendors to buy from one specific software vendor.

CITATION: Svenningsson, R., Johansson, R., Arts, T. and Norell, U., "Formal Methods Based Acceptance Testing for AUTOSAR Exchangeability," *SAE Int. J. Passeng. Cars - Electron. Electr. Syst.* 5(1):2012, doi:10.4271/2012-01-0503.

INTRODUCTION

One of the main drivers for the AUTOSAR development partnership [3] is to reduce cost for automotive electric/electronic (E/E) systems [5]. The idea is to facilitate re-use of software and hardware components. And this can be achieved by the standardization of methodology and basic software.

Since the beginning of AUTOSAR, it has been identified that the benefits of the standardization relies on that all implementations follow the specifications. This is hard to reach in practice as the specifications are expressed in natural language and requirements are complex and non-atomic. This in turn results in room for different interpretations between vendors. It was recognized that having vendors declare conformance of AUTOSAR products by testing themselves based on their own test suites, was insufficient. To cope with this, for a number of years the strategy from the AUTOSAR consortium has been to define a conformance test process [8] [5] [7].

An AUTOSAR implementation is not just fixed source code. It consists of a configuration and tools to generate code from that configuration. Quite a lot of code is shared by

different implementations, but a substantial part is purely depending on the configuration. Therefore, a test consists of two parts. In the first place a configuration, and in the second place a sequence of API calls that stimulate the system together with a check for the produced output. The same sequence can have different outcomes, depending on the configuration parameters used. There are tens of thousands of such parameters.

As a consequence, a general conformance test suite would consist of a large number of rather small configurations accompanied by sequences of API calls and their expected outcome. AUTOSAR is a large standard containing more than 60 BSW modules, each module having their own test suite, best case a few modules being combined in a test, since it is not feasible to predict the outcome of API sequences in which many modules are involved.

Due to the fact that AUTOSAR implementations need manual effort in the code generation process, the more configurations one needs to test for, the more expensive running the tests becomes. Moreover, many OEMs would not care for their software if certain tests fail if they are not

interested in that specific feature of the system, but it is close to impossible to mine the test suite and find those tests that one really wants to pass and those for which one does not care. Thus, even with a general test suite, testing is a major effort.

We addressed this problem by building upon an existing tool; QuickCheck, to generate test cases from models. We created models for three protocol stacks and some routing functionality. These models are parameterized by the configuration. We extended QuickCheck to be able to read arbitrary valid AUTOSAR configurations and automatically create test cases for those configurations, i.e., test cases that should pass given that specific configuration. The test cases are valid sequences of API calls of which the outcome is validated against the model when executing these calls.

We tested the implementations of three different vendors with these models and found deviations from the standard in each of them, showing that this approach is effective in finding errors in well tested software. The challenging parts of the approach are to come up with a sensible configuration and QuickCheck models. The configuration was provided to us by an OEM and mimicked the essential features needed in the next generation cars. The QuickCheck models were created in much less time than traditional test suites covering the same functionality would be written. This shows that our approach is efficient as well.

TEST OF COMPOSABLE COMPONENTS

For a component-based paradigm to become successful it relies on a working standardization [11]. This is a necessary (but not sufficient) condition for independent development and seamless integration of components [12]. The traditional quality concerns are the functionality related effects of reuse, and not the non-functional properties of the components and their compositions. The latter question has been identified as important for building safety applications on AUTOSAR [10] [4]. But it mainly has implications when composing application software components [9]. For the basic software components, the problem of compositionality can still be reduced to a question of a functional compliance between the module implementations and their specifications. This is at least true as long as the set of AUTOSAR BSW specifications are consistent and unambiguous. When discussing the base for AUTOSAR conformance testing Gilberg et al. express this as: “this can only be achieved if the basic software specifications are unambiguous and mature enough to ensure a high quality of the standard...” [7].

As is shown in the paragraph on experiences below, the question of unambiguity and consistency among basic software specifications perhaps is as hard to check as the conformance between specifications and implementations.

As many software components, the AUTOSAR basic software modules each has a large configuration space. This implies that when solving the general problem of testing

conformance between the implementation and the specification, testing of both the completeness of configurability and the correctness in behavior for each configuration should be covered. This makes the scope of the AUTOSAR conformance test huge.

ACCEPTANCE TEST

At the open AUTOSAR conference 2011, the AUTOSAR consortium announced that they will not anymore only rely on conformance testing for assuring the quality of AUTOSAR implementations, but also introduce the concept of acceptance test [6].

Acceptance testing differs from conformance testing in the sense that testing is performed to ensure that implementations conforms to the specific requirements of an OEM for a specific application, rather than the general case which was the intention of the AUTOSAR conformance test concept. Hence the terminology of “acceptance test”, although slightly different from its standard use in software engineering.

By reducing the scope into something that is enough for the OEM, a manageable amount of tests needs to be performed. An obvious possibility to reduce the needed amount of tests is to neither test the completeness w.r.t. configurability nor to test the correct behavior for each possible configuration. From an acceptance test point of view it is sufficient to define a limited amount of configurations that together cover all functionality of interest.

Furthermore, by performing tests on clusters (or a set of modules) rather than on individual modules the total amount of tests can be reduced. This, however, introduces the challenge to write a test for unspecified behavior. Sure enough, the behavior of each module is specified in the standard, but the behavior of a cluster is derived and not explicitly specified. Unavoidably this leads to interpretations by the test case designer and ambiguities in the inherent specifications will need to be resolved by testing against a number of different implementations.

Similarly, when we create models for a cluster, we must interpret the standard and come up with the “combined” behavior. Models are, however, formal objects and a large part of the combined behavior can be computed automatically from the separate models.

RANDOM TEST AGAINST A FORMAL MODEL

The testing method presented in this paper is based on feeding the test object (the AUTOSAR cluster implementation) with random stimuli, and then compare its behavior with a formal model. Random stimuli should not be confused with non-sense stimuli; we only create stimuli that make sense in the present state of the system, given a certain configuration.

Instead of manually writing test suites, a formal testing model is written which is generic with respect to a