A Cross-Platform Test System for Evolutionary Black-Box Testing of Embedded Systems

Peter M. Kruse, Joachim Wegener, Stefan Wappler
Berner & Mattner Systemtechnik GmbH, Berlin, Germany
{peter.kruse|joachim.wegener|stefan.wappler}@berner-mattner.com

When developing an electronic control unit (ECU) in a domain like the automotive industry, tests are performed on several test platforms, such as model-in-the-loop, software-in-the-loop and hardware-in-the-loop in order to find faults in early development stages. Test cases must be specified to verify the properties demanded of the developed system on these platforms. This is an expensive and non-trivial task. Evolutionary black-box testing, a recent approach to test case generation, can perform this task completely automatically. This paper describes our test system that implements evolutionary black-box testing and how to apply it to test functional and non-functional properties of an embedded system. Our test system supports the aforementioned test platforms and allows reuse of the generated test cases across them. We demonstrate the functioning of the test system in a case study with an antilock braking system.

Introduction

A great number of today’s products are based on the deployment of embedded systems. In industrial applications, embedded systems are predominantly used for controlling and monitoring technical processes. There are examples in nearly all industrial areas, for example in aerospace technology, railway and automotive technology, process and automation technology, communication technology, process and power engineering, as well as in defense electronics. In order to achieve a high quality of an embedded system, analytical quality assurance, and in particular testing, is crucial. The aim of testing is to detect errors in the system, and, if no errors are found during comprehensive testing, to convey confidence in the correct functioning of the system.

In order to find faults in the embedded control system before its deployment into the target environment, tests are usually carried out on various test platforms in the development phase. These include model-in-the-loop, software-in-the-loop and hardware-in-the-loop test platforms. In-the-loop means there is a bidirectional interaction between the embedded system and its environment: the environment stimulates the sensors of the system and in turn the system affects the environment using its actuators. With model-in-the-loop (MiL) tests, the implementation model of the system is examined in with the modeling software running on development hardware and a simulated environment. Model-in-the-loop tests are performed when model-based development is applied. With software-in-the-loop (SiL) tests, the software implementing the system behavior is examined in a simulation environment where development hardware is used and the system environment is simulated. With model-based development, the code generated from the model can be tested on the SiL platform. With hardware-in-the-loop (HiL) tests, the software integrated into the target hardware (e.g. an embedded controller) is examined in a simulation environment.

Test cases must be created to verify the specified properties of the developed system on the different test platforms. The creation of relevant test cases is a resource-consuming and challenging task when done manually. To increase the effectiveness and efficiency of the test, and thus to reduce the overall development costs for an embedded system, automatic test case generation is highly desirable. Evolutionary testing is a promising approach for fully automating test case design for various test objectives that has been shown to be effective for various domains and development paradigms [5].
In this paper, we describe the integration of our testing tool MESSINA [1] and the evolutionary testing framework ETF in order to enable evolutionary black-box testing of embedded systems on the MiL, SiL and HiL platform.

**Evolutionary Testing**

Evolutionary testing is based on evolutionary algorithms. An evolutionary algorithm (EA) is an optimization technique based on the principles of the Darwinian theory of evolution. The algorithm starts with a set of candidate solutions, typically selected randomly, called individuals. Then the EA evaluates the fitness of each candidate solution by executing a problem-specific fitness function. The fitness function rewards candidate solutions that solve the optimization problem (represented by the fitness function) better than the other candidate solutions and penalizes poor solutions by assigning lower fitness values to them. Crossover is applied to the individuals that the EA selects for offspring production. Afterwards, mutation is applied to the offspring individuals. The pool of individuals is updated by inserting the offspring individuals and eliminating already contained individuals. The process of evaluation, selection, crossover, mutation and population update is iteratively repeated unless a termination criterion applies, such as that the ideal solution is found.

Evolutionary testing transforms the test objective into an optimization problem. The input domain of the test object forms the search space in which an evolutionary algorithm searches for test data that fulfils the test objective. The fitness function for functional testing must be manually defined as it is problem-specific and not generally definable. In previous work, fitness functions for the test of various automotive systems, such as a parking controller, have been implemented, e.g. in [8], [10], [11] and [12].

**Our Evolutionary Test System**

The industrial MESSINA tool [1] supports the test of an embedded system during its development on the MiL, SiL and HiL platform. For the latter, MESSINA communicates with the MESSINA-HiL system [2]. The ECU, consisting of the embedded software and hardware, is directly connected to MESSINA-HiL. MESSINA controls MESSINA-HiL and provides the test cases and the calculations from the integrated environment models.

**MESSINA-HiL**

MESSINA-HiL [2] is a modular, general-purpose hardware-in-the-loop test system developed by Berner & Mattner. MESSINA-HiL provides the ECU physically connected to it with the corresponding input signals and reads the output signals. MESSINA-HiL offers flexible signal conditioning and uses open industry standards. It is possible to link several MESSINA-HiL systems together using optical networking technology, thereby establishing a test environment for integration testing.

**MESSINA**

MESSINA [1] is a testing tool allowing the implementation of hardware and software-independent test sequences specified in different notations, such as UML, Java, or TPT [7]. Using abstraction layers, it allows test execution on different platforms. One of these layers is the signal pool containing all system signals provided by the connected hardware or software devices. For MiL and SiL testing, MESSINA supports software devices like Simulink models, ASCET models, and AUTOSAR software components. Multiple models and software components can be run in parallel to perform virtual system integration. For HiL testing, MESSINA is directly connected to MESSINA-HiL. MESSINA downloads tests to MESSINA-HiL where the tests are executed in real-time (Figure 3 system under test). Tests implemented in MESSINA can be used seamlessly for MiL tests, SiL tests and HiL tests. As the tests are defined hardware-independently, MESSINA enables these tests to be portable across the various test platforms. Hence, the only difference between HiL, MiL and SiL testing from MESSINA's point of view is the usage of different environment models. Therefore test cases can be used for MiL, SiL, and HiL testing without any further adaptation.

**Evolutionary Testing Framework**

The EvoTest project [5] developed an extensible and open automated evolutionary testing framework that provides general components and interfaces to facilitate the automatic generation, execution, monitoring and evaluation of test cases using evolutionary computation.
The evolutionary testing framework (ETF) [4] creates an evolutionary algorithm suitable for the system under test using the algorithm configuration and generation tool GUIDE [3, 6]. During the optimization process the evolutionary testing framework provides the individuals, representing the test data for the system under test, and expects the fitness values for each individual in return. We also integrated the ETF signal generator [13] to create and optimize continuous signals.

Putting it all together

We integrated ETF with MESSINA for the automatic generation of test cases as shown in Figure 1. We developed a MESSINA plug-in to configure and run the evolutionary tests. The signals of the system under test held in MESSINA's signal pool contain type information, value range information and other metadata. These are passed to the evolutionary testing framework as a specification of an individual. The evolutionary testing framework then creates a specific evolutionary algorithm conforming to this specification. Both MESSINA and ETF support mixed data types, allowing a one-to-one transformation of individuals to test data in many cases. After the transformation of an individual to test data, MESSINA executes the test on the target system. Generic test cases are parameterized with the individual data. For MiL and SiL tests, MESSINA calls the system under test directly with the parameterized test cases. For HiL testing MESSINA downloads the tests to MESSINA-HiL.

The fitness calculation is based on an analysis of the behavior of the system under test during test execution. MESSINA records the behavior through a monitoring interface. The generic test case calculates the fitness, which MESSINA passes back to ETF.

With the implemented solution search-based testing is available for thorough model-based ECU testing.

Case Study

We applied the MESSINA-ETF integration to the test of an industrial anti-lock brake system. As this system is already in serial production, we did not expect to find real faults. The test focuses on the verification of a particular safety requirement relating to the distance a vehicle continues to move after the actuation of its brake.

Anti-lock braking system

The anti-lock braking system (ABS) is a system which prevents the wheels of a vehicle from locking while braking, in order to allow the driver to maintain steering control under heavy braking conditions and, in most situations, to shorten the braking distance. When an ABS brake is depressed hard, like in an emergency braking situation, the ABS pumps the brakes several times per second. If the angle speed of the wheels decreases rapidly, the electronic control system reports blocking danger. The pressure of the brake hydraulics is then reduced immediately and raised again to just under the blocking threshold. This process can be repeated several times per second. The goal of the anti-locking control system is to maintain the slip of the wheels at a level which guarantees the highest braking power and the highest steerability of the vehicle. Figure 2 shows the components of the simulation environment implemented.

Evolutionary Testing of the ABS

We set up the hardware-in-the-loop test environment as previously described. We implemented a complex simulation environment integrating a commercial brake model, a vehicle dynamics model, and a wheel speed sensor model [9]. The setup of the evolutionary test system is shown in Figure 3. In the first phase of the study, we specified a generic test case to expose improper system behavior.
The generic test case consists of a set of pre-actions (e.g. accelerate up to 20 m/s), the actual test action, and the check of some postconditions. The test case in MESSINA was written in Java. For the initial tests, we set the car speed as the optimization parameter. We defined the fitness function in terms of the resulting braking distance. In the second phase, we extended our setup to optimize the friction and performed measurements of wheel slip on all four wheels during a braking maneuver to calculate the fitness. This fitness function takes into account more detailed aspects of a braking maneuver.

We configured the ETF as follows:

- Mutation rate of 0.1,
- Crossover rate of 0.85,
- Population size of 100 individuals per generation, (20 individuals for initial tests)
- Elitism of 0.15, using strong elitism,
- Fertility of 0.85;
- Number of genitors 1.0;
- Selection by Deterministic Tournament of 2.0;
- Surviving Offspring 1.0;
- Surviving Parents 0.25;
- Evolutionary Programming Tournament with a selective pressure of 2.0 for Reductor Parent and Offspring;
- Sequential final redactor with a pressure of 2.0;
- Maximal number of generation of 500.

If there is no progress in the optimization process, e.g. the best fitness value found does not improve for several generations or all fitness values in one generation have a defined minimal variation, the optimization process terminates. In initial runs, these settings turned out to be suitable.

For the initial tests, we disabled the ABS functionality and performed simple brake maneuvers. After 10-12 generations with 20 individuals each, a maximum braking distance of 43 meters was found. The vehicle velocity at this point was 25m/s, the fastest possible speed in our model. Execution of one individual took about 40 seconds; the total execution time for all generations of one test run was about 90 minutes. In the first phase of the case study, we enabled the ABS functionality without changing the parameters and repeated our experiment. The resulting maximum braking distance was 39 meters. In the second phase, we applied the advanced setup for a more realistic test of the ABS system using the wheel slip sensor values for the fitness function calculation. Furthermore, individuals were transformed into curve traces for the input signals of the ABS system using the signal generator of the ETF. For each of the four wheels, a waveform is generated by the signal generator representing the friction scaling ($\mu$) of the wheel in a range of [0.5, 1.2]. To cope with the higher complexity of this search the population size was set to 100 individuals per generation, which leads to longer total execution time of the test runs. A typical optimization needed up to 37 generations. Total execution time of one run was typically around 8 hours.

Test Results

For the initial tests, we disabled the ABS functionality and performed simple brake maneuvers. After 10-12 generations with 20 individuals each, a maximum braking distance of 43 meters was found. The vehicle velocity at this point was 25m/s, the fastest possible speed in our model. Execution of one individual took about 40 seconds; the total execution time for all generations of one test run was about 90 minutes. In the first phase of the case study, we enabled the ABS functionality without changing the parameters and repeated our experiment. The resulting maximum braking distance was 39 meters. In the second phase, we applied the advanced setup for a more realistic test of the ABS system using the wheel slip sensor values for the fitness function calculation. Furthermore, individuals were transformed into curve traces for the input signals of the ABS system using the signal generator of the ETF. For each of the four wheels, a waveform is generated by the signal generator representing the friction scaling ($\mu$) of the wheel in a range of [0.5, 1.2]. To cope with the higher complexity of this search the population size was set to 100 individuals per generation, which leads to longer total execution time of the test runs. A typical optimization needed up to 37 generations. Total execution time of one run was typically around 8 hours.
Figure 3 shows a screenshot of MESSINA while execution is in progress. The lower part shows the friction scaling per wheel; the middle part shows the resulting slip values per wheel. As opposed to using the wheel slip sensor values for fitness function calculation, using the braking distance, the optimization always ran into very low frictions. To shorten total execution time of these tests, we reduced the target speed to a fixed value of 20 m/s. This reduced the test execution time of the system under test by 20% without losing any important information or characteristics, since the optimization focused on the friction scaling.

Results from these later tests are therefore not directly comparable with results from earlier tests.

Conclusion and Outlook

We presented a test system based on MESSINA, MESSINA-HiL, and the research system Evolutionary Testing Framework. This system allows full automation of black-box tests on different testing platforms (MiL, SiL, HiL) by applying search-based testing techniques. The test automation framework supports the application of evolutionary testing in very common industrial settings: testing models and software in simulation environments as well as the examination of ECUs in a hardware-in-the-loop test environment driven by a software frontend for the definition and implementation of tests. To demonstrate the application of the test system, the test of an antilock braking system was described. Test cases were generated using the evolutionary testing framework and executed in both SiL and HiL test environments. Driving maneuvers with long braking distances were found automatically.

When compared with traditional testing, the additional effort required to apply the test system with its evolutionary testing integration is relatively low in our opinion. Users only need to generalize their existing test cases by a declaration of optimization parameters and to provide a fitness function. However, the definition of a suitable fitness function remains a challenging and non-trivial task as the knowledge about the system and its constraints must be transformed into it.

Future work will be contributed to an investigation of how to configure the evolutionary testing system so as to reduce the number of pre-tests that are usually performed for parameter adjustment. Furthermore, we want to exploit the configurability of the MESSINA-HiL system for integration testing of networks of ECUs.

Acknowledgments

This work was supported by EU grant IST-33472 (EvoTest).

Bibliography

Company Profile

Berner & Mattner Systemtechnik GmbH is a Germany-based development and consulting partner for the automotive, aerospace, defence, rail and healthcare industries. Berner & Mattner’s main competencies are model-based development and quality assurance for embedded systems. The company’s services range from consultancy and design to the running of complete testing and integration labs.

Berner & Mattner offers specification and testing products such as MESSINA-HiL (a hardware-in-the-loop test system for component and integration testing of electronic control units), MESSINA (a test software for defining and executing tests for embedded systems), MODENA (a test system for testing infotainment systems), CTE XL (a graphical editor for functional test case design), PowerDiff (a tool for model comparisons) as well as the DOORS extension toolbox MERAN offering extended functionalities for text-based specifications, e.g. variant management.

In order to offer efficient and innovative test automation solutions, Berner & Mattner has performed extensive research into various automation techniques, e.g. test case generation for the Classification-Tree Method, from model-based specifications or for statistical testing. Another important test case generation approach is evolutionary testing (ET) with its many facets, such as evolutionary functional testing, evolutionary structural testing, evolutionary real-time testing and evolutionary safety testing. Evolutionary testing is able to find bugs in the system under test fully automatically which other testing techniques could not find. It can create a large number of goal-oriented test cases used to examine critical aspects of the system under test. Berner & Mattner was an important industrial partner of the EU-funded EvoTest project, which looked into evolutionary testing for complex systems. Berner & Mattner’s evolutionary testing research has resulted in new approaches to automatic test case generation and delivered successful results for the test of various automotive embedded systems, such as an anti-lock brake system, an active brake system, an adaptive headlight system and an electric window control. The results have been published in journals and scientific conferences, such as GECCO.

With the integration of ET into the MESSINA test automation framework, Berner & Mattner can offer the first industrial evolutionary testing environment enabling automatic test case generation across various test platforms like hardware-in-the-loop, model-in-the-loop and software-in-the-loop.

The contact for ET research and products is Dr. Joachim Wegener (Joachim.Wegener@berner-mattner.com)
About the authors

**Peter M. Kruse** studied computer science at Otto-von-Guericke-University of Magdeburg. He has extensive experience in software engineering and is an expert of the Eclipse tool platform. In his diploma thesis he wrote about using ontologies to improve searches in metadata. Within EvoTest he worked at the definition of the EvoTest software architecture. He is leading the current CTE XL development and programming. Additionally, he is supervising students’ activities as well as diploma theses.

**Dr. Joachim Wegener** studied Computer Science at the Technical University Berlin and obtained his PhD on the evolutionary testing of real-time systems at the Humboldt University of Berlin. This work gained him the Software Engineering Prize 2002, awarded by the Ernst Denert Foundation and the German Informatics Society. Dr. Wegener is the local representative of Berner and Mattner in Berlin, where he leads the automotive department. He previously worked for Daimler AG, where he led the development of the world’s first Industrial Evolutionary Testing System. Joachim Wegener is a pioneer of Search Based Testing and has been the first program chair for the GECCO Search Based Software Engineering track. Furthermore, he played a central role in the development of the test system TESSY and the classification-tree editor CTE. He is coordinating the "Embedded Systems Testing" research group of the German Informatics Society, and is a member of the industrial advisory board of King’s College.

**Dr. Stefan Wappler** holds a master's degree in software engineering from the University of Potsdam, Germany, and a PhD degree in engineering from the Technical University of Berlin, Germany. During his studies, Stefan Wappler focused on the development of embedded systems. He has over 5 years of experience in the field of search-based testing and researched search-based testing of object-oriented systems. Stefan Wappler is working for Berner & Mattner as a consultant specialised in testing and managing the test of various customer vehicle systems under development.

Company webpage: [www.berner-mattner.com](http://www.berner-mattner.com)