2017 IEEE International Conference on Software Quality, Reliability and Security (Companion Volume)

Towards Effective and Scalable Testing for Complex High-Speed Railway Signal Software

Chunfeng Hu*, Jin Guo*, Nan Li†, Yao Li*, Chang Rao*, Siqi Liu*

* School of Information Science and Technology, Southwest Jiaotong University, Chengdu, Sichuan, China
Email: hcfaizhz@my.swjtu.edu.cn, jguo_scce@home.swjtu.edu.cn, {Liyao, changrao, bk20122485}@my.swjtu.edu.cn

† Research and Development, Medidata Solutions, New York, USA
Email: nli@mdsol.com

***Abstract*—Signal software for high-speed railways is safety-critical, controlling communications between trains, tracks, stations, and signals. It is necessary to test railway signal software rigorously. However, ad-hoc approaches still dominate in practice. We proposed a hybrid approach that uses combinatorial testing (CT) and model-based testing (MBT). We applied this systematic approach to testing track circuit receivers (TCRs) and compared it with an ad-hoc approach. The systematic approach was more effective than the ad-hoc approach.**

Although our approach is effective in testing TCR, TCR is relatively simple. Testing complex signal software used in Chinese high-speed railways can be challenging. In this paper, we summarize our approach used in testing TCR. Moreover, we identify challenges and present directions for testing complex signal systems in a scalable manner.

***Index Terms*—Safety-critical Railway Signal Software, Combinatorial Testing, Model-based Testing**

I. Introduction

China has the world's longest High-Speed Rail (HSR) network. As of September 2016, the HSR network has over 20,000 kilometers of route in service, which is more than the rest of the worlds' high-speed rail tracks combined [1]. Signal software used in HSR is safety-critical since a software fault could lead to signal control disorder, or even train crashes.

Although safety-critical software requires rigorous testing, ad-hoc approaches are widely used in practice. Our study [1] on testing the software embedded in the track circuit receiver (TCR) found out that manually created tests missed some requirements and were not effective in finding faults, with an ad-hoc approach. A systematic approach is needed. We proposed a hybrid approach that combines combinatorial testing (CT) and model-based testing (MBT) [1], which found more faults than the ad-hoc approach, on a release version of TCR.

In this paper, we summarize our approach and report our experience in testing TCR. Furthermore, we present the problems and challenges for testing more complex systems with our approach. We show directions for applying our approach in a scalable manner.

The rest of the paper is organized as follows. Section II introduces our preliminary approach applied to TCR. Section III summarizes the challenges and presents future work.

[1] https://en.wikipedia.org/wiki/High-speed_rail_in_China

II. Methodology

This section summarizes our hybrid approach used to test TCR [1], with two techniques, combinatorial testing (CT) and model-based testing (MBT). Specifically, we applied CT independently to generate one test suite, and then generated another test suite with CT and MBT combined.

The train control system detects the presence of the train, and then send an analog signal to TCR to control the relay to be up or down. When the train is in a section, the relay must be down; otherwise, the relay must be up. The testability of the TCR software is low because the software is embedded and we have limited access to the input and output. The input of the TCR software includes a signal and the duration of the signal that lasts. The signal is composed of three parameters, *low frequency (LF)*, *carrier frequency (CF)*, and *voltage (VO)*. The analog signal is converted to the values of the three parameters for the TCR software. The output includes the status of the relay to be set and the reaction time.

With CT, we identified parameters that constitute the signal (an input to TCR) and created Input Domain Models (IDMs). The IDMs were constructed as follows. First, for each parameter, we select one or more characteristics to partition its domain. The partition should be complete, i.e., it should cover the whole domain, and disjoint, i.e., the partitioned blocks must not overlap. For example, a characteristic for a parameter may be "if this parameter is null." Based on this characteristic, the domain of this parameter is divided into two blocks: "null" and "non-null values." Second, we can select null from the first block and a random value such as "test" from the second block since values in the same block are considered to be equivalent.

Each of the three parameters has valid and invalid ranges since it represents a frequency or voltage. Each parameter has standard values. The combinations of the standard values are used to represent different train speed statuses. Therefore, we partitioned the input domain of each parameter, based on their relation to important values, which include the standard values, lower and upper bounds of the standard values, and boundaries of invalid values. From the partition for each parameter, we arbitrarily selected standard values, values near the boundaries and invalid values. We applied a combinatorial

978-1-5386-2072-4/17 $31.00 © 2017 IEEE
DOI 10.1109/QRS-C.2017.98

coverage criterion (pair-wise coverage) to the IDMs, generating a suite of tests. Each test has three values for every parameter, representing a signal.

TCR takes a few seconds to respond when receiving a signal. So we need to generate tests to cover the timing requirements. Moreover, while waiting for the action of the relay, TCR may receive more signals. Although TCR is supposed to receive one signal at one time, multiple signals could happen in unexpected scenarios. Covering unexpected scenarios is extremely important for safety-critical software. Therefore, we created UML finite state machines (FSMs) to model the system behaviors in the presence of multiple signals, with the consideration of timing.

We used a Model-Based Testing (MBT) tool, Skyfire [2], to generate test paths from the FSMs. Skyfire can convert an FSM to a generic graph with an initial node, a final node, and other nodes connected with edges. We generated test paths to satisfy edge coverage using an algorithm that reduces various costs including the number of tests [3]. These test paths are called *abstract tests* since they cannot be executed. Since each edge can be mapped to a transition of the FSMs, a test path is a sequence of transitions. For example, one abstract test path is [*InvalidFreq*, *t<5s_InvalidVO*, *t<2s*]. This test path has three transitions, each of which represents a signal or reaction time. The first two transitions represent an invalid frequency signal and an invalid voltage signal that lasts for less than 5 seconds. Then TCR should respond within 2 seconds.

The next step of MBT is to concretize the abstract tests, assigning actual test values to the signal transitions. For each transition in an abstract test, we selected signal values from the first test suite generated from the IDMs. Next, we treated each signal as an independent parameter and applied pair-wise coverage to them. We applied the same approach to other abstract tests, generating another suite of tests. Each test represents a series of signals. For the timing requirements in the transitions, a post-processing was executed to check if proper timing values were assigned to satisfy the requirements.

III. Challenges and Future Work

With the experience in testing TCR, we are under the progress of testing much more complex systems such as Train Control Center (TCC) and Automated Train Protection (ATP) systems. For example, the lines of code of and the number of functionalities of ATP is about 10 times more than that of TCR. The number of documented test scenarios of ATP is about 15 times more than that of TCR. Next, we discuss a few challenges for testing complex high-speed railway signal software and present directions for future work.

First, in the railway signal domain, most software is embedded with hardware. Inputs to embedded software are different from those to regular software. The testability of embedded software is low, as we may not directly generate parameter values for the embedded software. We need to extract parameters that constitute the inputs and create IDMs. This is very different from how we usually enter test values into a regular software program.

Second, requirements for complex systems are very complicated. Extracting IDMs for CT and creating behavioral models for MBT is difficult. Therefore, covering all requirements with models is challenging. To keep track of behavioral requirements, we start using SCADE [2], which is able to map from requirements to models.

Third, our hybrid approach with CT and MBT could generate too many tests. We generated about 14,000 tests for TCR. Our approach could easily generate hundreds of thousands of tests for a complex system like ATP. Running all the tests even on a simulation system could take a long time. In a modern agile testing environment with a continuous integration and delivery system proposed by Li et al. [4], executing lots of tests could have a long feedback cycle. Therefore, we will need to conduct more empirical studies to select the most cost-effective coverage criteria.

Fourth, safety-critical systems have lots of timing requirements. Covering these requirements is a must. We have used FSMs to describe the requirements. In addition, we would like to validate the FSMs. We will use model checking such as timed automata to detect violations on timing requirements.

Last but not least, all tests should be executed on a simulation platform, which has all the signal systems and equipment such as TCC, ATP, stations, tracks, trains, signals, relays, etc. How to construct a comprehensive suite of test scenarios used on the simulation platform is very challenging. When designing the scenarios, there are too many factors to consider such as the number and types of tracks, the number and location of signals, and the number and directions of trains. In addition, we have to consider the interactions between the sub-systems, especially when the service of a system is unavailable. CT and MBT can be useful and we plan to apply mutation testing to generating the test scenarios.

Acknowledgement

This research is partly supported by the Fundamental Research Funds for the Central Universities (2682014BR059), China Railway Corporation under Grant (2016X001-C), and the Opening Foundation of Gansu Provincial Key Laboratory of Traffic Information Engineering and Control under Grant (20161103).

References

[1] C. Rao, J. Guo, N. Li, Y. Lei, Y. Zhang, Y. Li, and Y. Cao, "Applying combinatorial testing to high-speed railway track circuit receiver," in *2017 6th Internaltional Workshop on Combinatorial Testing (IWCT2017)*, Tokyo, Japan, March 2017, pp. 199–207.

[2] N. Li, A. Escalona, and T. Kamal, "Skyfire: Model-based testing with cucumber," in *2016 IEEE International Conference on Software Testing, Verification and Validation (ICST)*, April 2016, pp. 393–400.

[3] N. Li, F. Li, and J. Offutt, "Better algorithms to minimize the cost of test paths," in *Proceedings of IEEE 5th International Conference on Software Testing, Verification and Validation*, ser. ICST'12, Montreal, Quebec, April 2012, pp. 280–289.

[4] N. Li, J. Guo, J. Lei, Y. Li, C. Rao, and Y. Cao, "Towards agile testing for railway safety-critical software," in *Proceedings of the Scientific Workshop Proceedings of XP2016*, ser. XP '16 Workshops. New York, NY, USA: ACM, 2016, pp. 18:1–18:4.

[2] http://www.esterel-technologies.com/products/scade-suite/