Black-box testing tool of railway signalling system software with a focus on user convenience

J.-G. Hwang, J.-H. Baek, H.-J. Jo & K.-M. Lee
Korea Railroad Research Institute (KRRI), Korea

Abstract

A railway signalling system is the control equipment performing vital functions, and the validation on functional safety of its software is a very important issue. Recently, various software testing tools have been applied to verify signalling system software. However, these tools are unable to support black-box testing to verify the functional safety of embedded signalling system software, and few black-box testing tools are currently commercialized. Since most of these commercialized black-box testing tools depend on a testing method through direct accessing to the target memory under test, there are a lot of difficulties when applying them to the test. In the case of using the existing tool, it is possible to test them only if the internal memory addresses values that are occupied by the executing embedded software. Therefore they are rarely utilized to verify the embedded software in the actual operating environment. To overcome these problems, this study developed a black-box testing tool using communication interfaces which are utilized when actually operating. The real interface channel is used as an input and monitoring channel for black-box testing. This approach carries out the test in a manner where the test data are input and the results are fed back to these interface channels by utilizing interface channels with other signalling equipment already operating. Boundary value analysis and equivalence class analysis modules were used to generate test cases within the developed new testing tool. The pilot of the proposed testing tool has been completed, and the feasibility study is now in progress with railway signalling system software in Korea as its test target.
Keywords: railway signalling systems, software testing, black-box testing.

1 Introduction

The railway signalling system is a vital control system in the railway system. With recent developments in computing technology, many railway control functions have increasingly depended on computer software, leading to the evolution of a more flexible and intelligent railway signalling system. At the same time, such software programs have been developed to have more functions and complexity. Meanwhile, software programs are likely to have many errors and the cost incurred by such errors has increased. Especially, if a fatal software error occurs during railway operation, it may result in loss of lives. Accordingly, software verification and validation have become more important.

As the importance of the validation of railway signalling system software is highlighted, the software validation procedures and requirements have been internationally standardized. When a new railway signalling system is developed, it is required to apply such standard to verify and validate the safety of the software [1, 2]. Recently, many research activities have been conducted to develop a railway signalling system software's verification, validation, assessment and supporting tools [3–6]. In Korea, testing and validation of railway signalling system software on the basis of international standards are required. However, software validation is only at the early stage and software validation through testing and quantitative analysis has been attempted. For validation of railway signalling system software, static testing (analysis of source code), dynamic testing and black-box testing may be used. Among these three testing methods, the black-box testing tool was developed in this study to validate the functional safety of railway signalling system software.

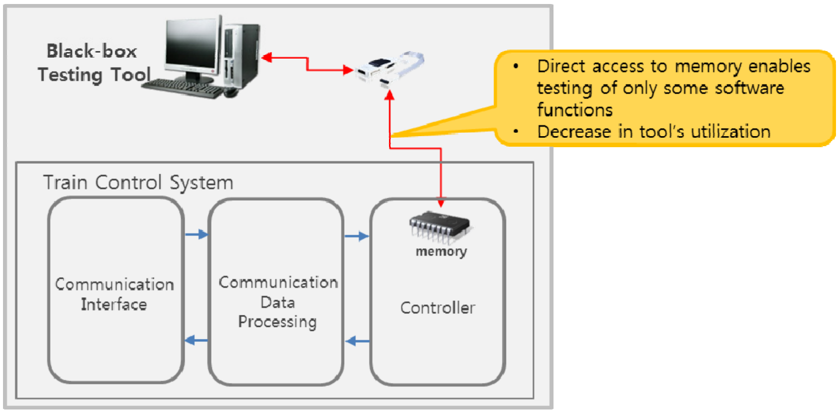

Black-box Testing Tool
Direct access to memory enables testing of only some software functions
Decrease in tool's utilization
Train Control System
Communication Interface
Communication Data Processing
memory
Controller

Figure 1: Existing testing tool.

Most commercial software testing tools cannot be used in black-box testing of railway signalling system's embedded software for functional safety. Only a few black-box testing tools for functional safety testing are commercially available. However, as shown in Figure 1, such commercial black-box testing tools depend on direct access to the system's memory, resulting in many difficulties in application to actual railway signalling system software. This is because the internal memory address values utilized or held by the system's application software have to be known for testing. Owing to such difficulties and complexity, they are rarely used in railway signalling system software validation [4, 5].

In this study, the problems of such existing commercial tools were analyzed and a new testing tool, which can be easily used in the functional testing of software, was developed. In other words, unlike existing black-box testing tools that use test cases based on internal memory access, a functional safety testing tool is proposed to allow input and analysis of test cases to the relevant software through an external interface. Such a black-box testing tool allows the development and analysis of test cases for black-box testing through analysis of actually used interface protocols, leading to increased user convenience.

2 Structure of proposed testing tool

As explained earlier, the new testing tool is the black-box testing tool for functional safety testing of railway signalling system's embedded software. It is the black-box testing tool with increased user friendliness and convenience. Existing tools are based on analysis of internal memory address values for functional safety testing (as shown in Figure 1). The new one proposed in this study is based on the communication interface actually used in system operation, to create and analyze test cases for black-box testing (as summarized in Figure 2).

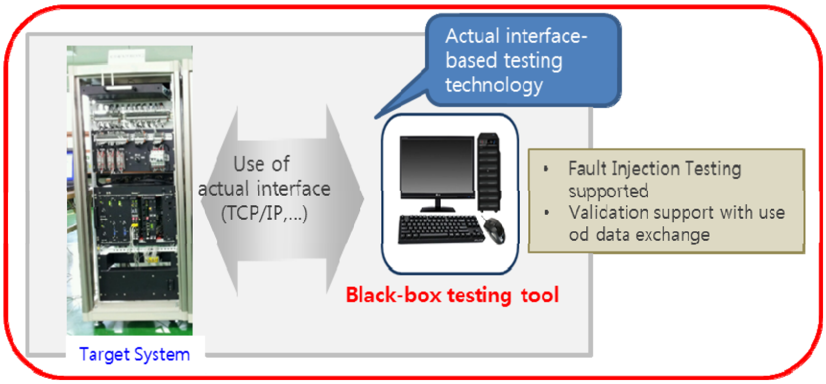

Actual interface-based testing technology
Use of actual interface (TCP/IP,...)
Fault Injection Testing supported
Validation support with use od data exchange
Black-box testing tool
Target System

Figure 2: Interface-based black-box testing tool.

As shown in Figure 2, this tool uses the existing interface channel between the target system and other devices (such as Ethernet and serial communication) and analyzes the interface specifications to create test data. Then, it inputs test data into the actually used interface, receives feedback results and analyzes them. In other words, the new testing tool can be connected to the existing interface, injects test data to the target system and receives feedback data from the target system. Therefore, it can be readily and easily used. In addition, test cases can be easily created through analysis of the target system's interface protocol specifications.

The proposed black-box testing tool was developed for black-box testing of a railway signalling system. Test cases are created through analysis of actual interface specifications. In addition, conventional software black-box testing methods, such as analysis of boundary values and equivalence area division methods, can be used to create additional test cases. Test cases created using conventional black-box testing methods are entered through the actual interface. Existing tools conduct black-box testing through analysis of the target system's internal memory values, but the new system developed in this study uses actual interfaces, which makes it more user-friendly and leading to increased convenience of the tool.

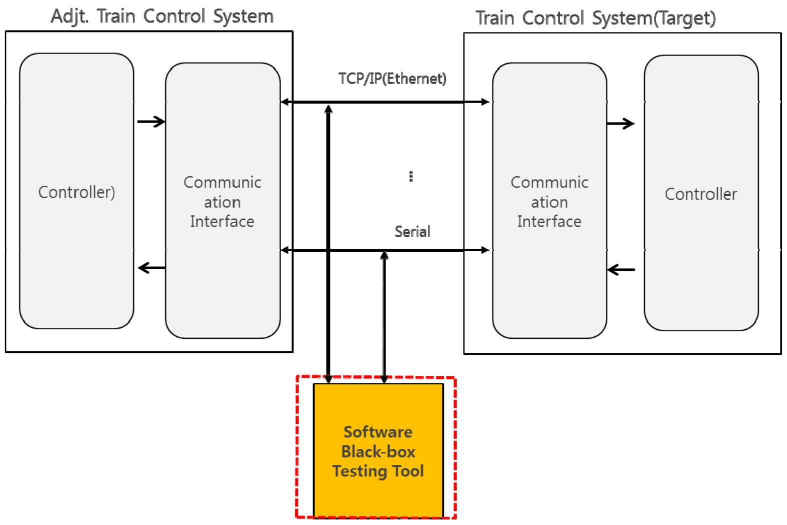


Figure 3: Interface-based black-box testing tool's structure.

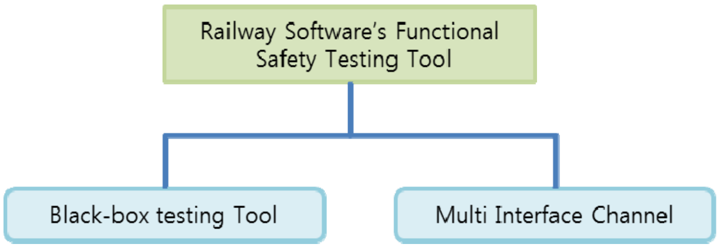


Figure 4: Testing the tool's structure.

Input value ranges are received and test data are created through analysis of such input values that are used in black-box testing. Various actual interfaces, such as Ethernet and serial communication, are used in testing. For this purpose, in addition to the black-box testing tool, the multi-interface channel is needed for connection to the actual interface. Therefore, the new black-box testing tool proposed in this study consists of a black-box testing module and interface connection module (as shown in Figure 4). The black-box testing module creates test cases and receives feedback of test results to determine acceptability. It is a core part in the actual testing. The multi-interface connection module is the one for data communication between the testing tool and target system. It sends data from the testing tool to the target system's interface and receives data from the target system to the testing tool's interface. This multi-interface channel was designed to support the expansion of interfaces to ensure testing of various systems in the future. In other words, when a new interface environment is required, the hardware interface card and data sending/receiving module can be added.

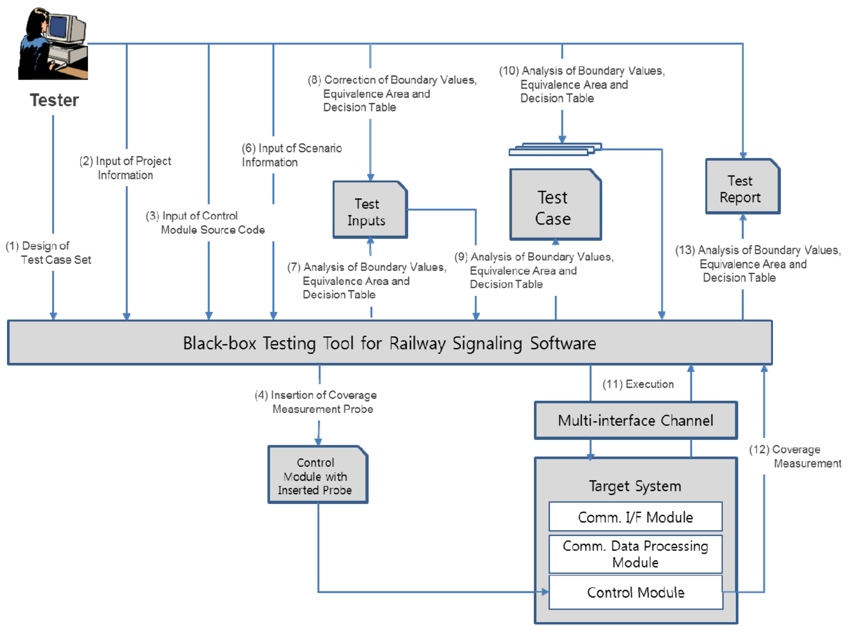

Tester
(1) Design of Test Case Set
(2) Input of Project Information
(3) Input of Control Module Source Code
(6) Input of Scenario Information
(8) Correction of Boundary Values, Equivalence Area and Decision Table
(10) Analysis of Boundary Values, Equivalence Area and Decision Table
Test Inputs
Test Case
Test Report
(7) Analysis of Boundary Values, Equivalence Area and Decision Table
(9) Analysis of Boundary Values, Equivalence Area and Decision Table
(13) Analysis of Boundary Values, Equivalence Area and Decision Table
Black-box Testing Tool for Railway Signaling Software
(4) Insertion of Coverage Measurement Probe
(11) Execution
Multi-interface Channel
(12) Coverage Measurement
Control Module with Inserted Probe
Target System
Comm. I/F Module
Comm. Data Processing Module
Control Module

Figure 5: Scenario for use of testing tool.

Figure 5 shows the scenario for use of the testing tool developed in this study. The testing tool user analyses the input data ranges for the target system and enters such data ranges to the testing tool to automatically create test data. At that time, three sub-modules, boundary value analysis model, equivalence analysis mode and decision table module, are used to analyse and generate the detailed test case based on data input by the user. These modules have already been developed prior to this study, so in this study these modules are just imported into developing the black-box testing tool. In addition, the user reviews such automatically created test cases and controls the execution of test cases. The target system's developer or administrator provides information that can be used in the analysis of input data ranges. In the figure, the coverage measurement probe insertion function is the black-box testing tool's function to analyse the target system's control module source codes and measure the coverage. This function will be implemented in the future through additional studies. At present, this function is at the concept design phase.

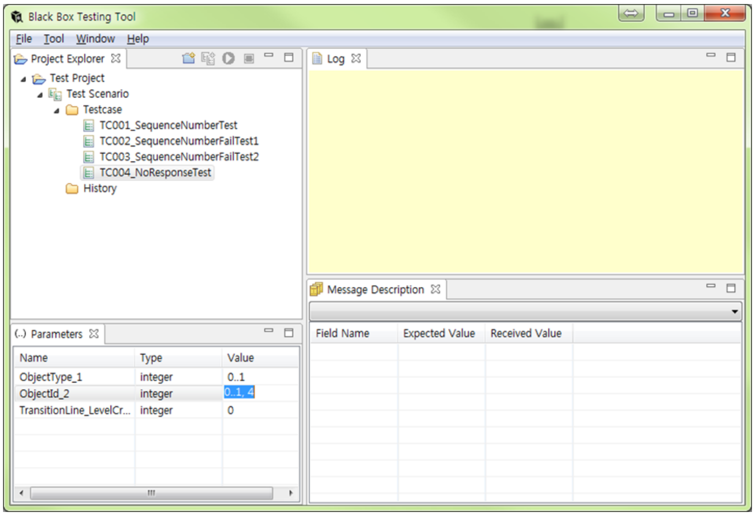
3 Implementation

The prototype of the new black-box testing tool proposed in this study was developed and is now being tested for suitability. The prototype's test case creation, testing and analysis modules were designed to be run on a lap-top computer with Windows 7 installed. The multi-interface channel was designed to support the Ethernet protocol. Channels that can support serial communication and others will be developed in subsequent studies. The module for test case creation and editing through analysis of protocols for interfaces between the target system and other control devices was based on the international standard language, TTCN-3, to ensure compatibility. In this study, "OpenTTCN 2012", which supports the international standard language, TTCN-3, was used in the prototype development. Environmental conditions for prototype development are as follows:

- Operating system: Microsoft Windows 7 Professional.
- Language: Visual Studio 6.0.
- IDE: Eclipse Indigo.
- Other: Eclipse RCP Plugin Development.
- TTCN-3 tool: OpenTTCN 2012.
- Virtual machine: JRE (Java Runtime Environment) 6.0 or higher version.

Figure 6 shows a module to create test cases on the basis of the TTCN-3 engine, which shows a window to input parameters for each test case. When types and values of individual parameters are entered into this window, they can be used in the black-box testing module.

Figure 7 shows the structure of the target system and interface for black-box testing. As shown in this figure, the new black-box testing tool inputs test case data and receives feedback data through the interface channel and snooping device. The interface channel was developed to allow expansion of connectivity with other communication protocols. The interface channel is a device for data communication between the testing tool and target system. It sends data from the testing tool to the target system's interface and it receives data from the target system to the testing tool's interface.



Test Project
- Test Scenario
  - Testcase
    - TC001_SequenceNumberTest
    - TC002_SequenceNumberFailTest1
    - TC003_SequenceNumberFailTest2
    - TC004_NoResponseTest
  - History

Name	Type	Value
ObjectType_1	integer	0..1
ObjectId_2	integer	0..1, 4
TransitionLine_LevelCr...	integer	0

Field Name	Expected Value	Received Value

Figure 6: A window to input parameter values.

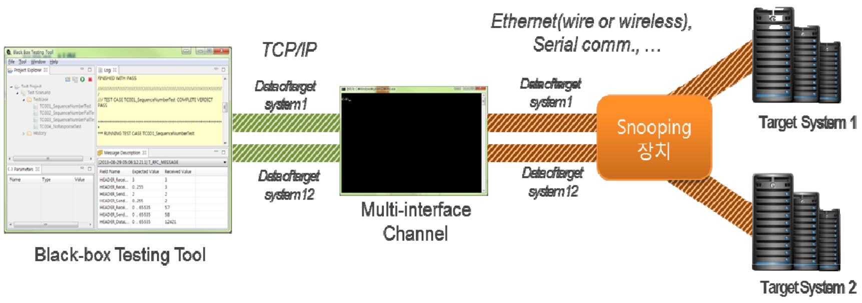
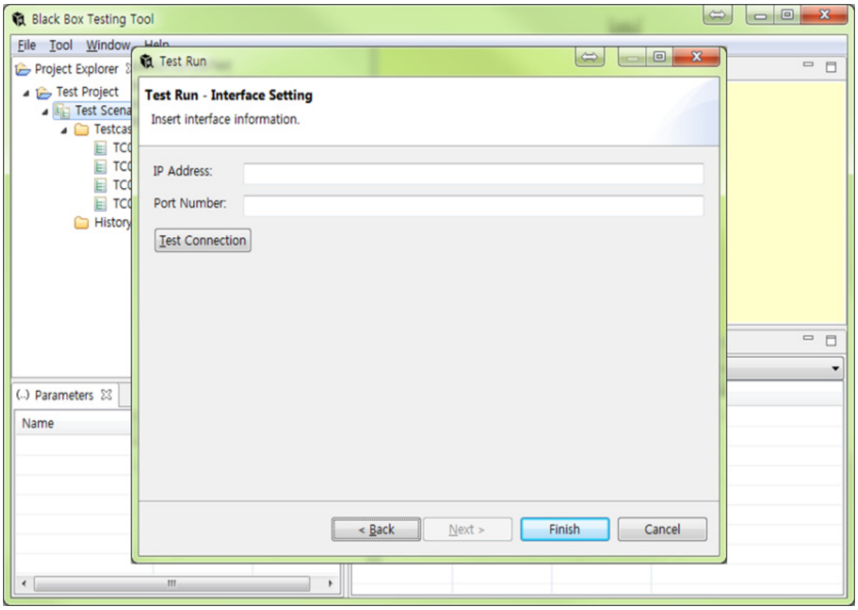


Figure 7: Connection with target system through interface channel.

The newly developed testing tool (Figure 7) is interfaced with the interface channel to input test data to the target system’s communication link and receive data from target system. For user convenience, it was designed to make access through the TCP/IP protocol. Figure 8 shows the screen for access to the interface channel. Figure 9 shows the screen for testing. On completion of each testing, log files are shown. In the bottom right of the window are listed both expected values (from analysis of protocol specifications) and actual values (from analysis of feedback data). This summarizes the test results.



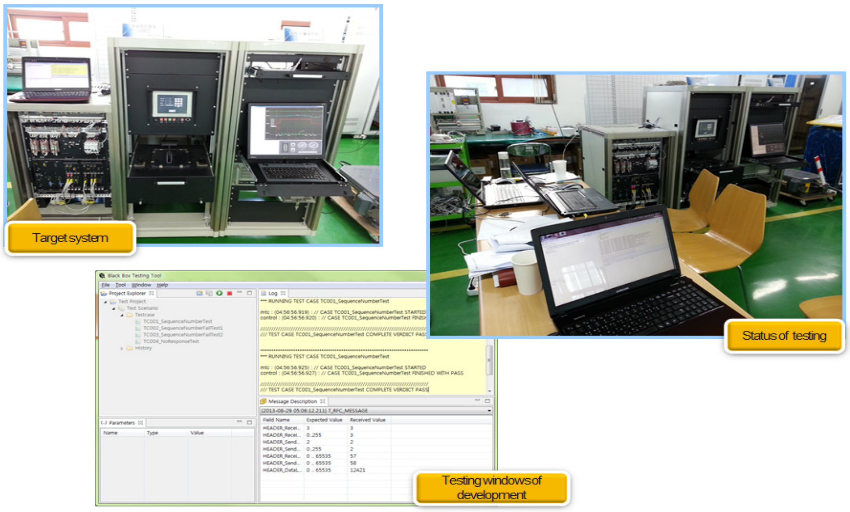
Test Run - Interface Setting

Insert interface information.

IP Address:

Port Number:

Figure 8: Screen for data input for access to multi-interface channel.



```
*** RUNNING TEST CASE TC001_SequenceNumberTest

mtc : (04:56:56.919) : // CASE TC001_SequenceNumberTest STARTED
control : (04:56:56.920) : // CASE TC001_SequenceNumberTest FINIS

//////////////////////////////////////////////////////////////////////
/// TEST CASE TC001_SequenceNumberTest COMPLETE VERDICT PAS

**********************************************************************
*** RUNNING TEST CASE TC001_SequenceNumberTest

mtc : (04:56:56.925) : // CASE TC001_SequenceNumberTest STARTED
control : (04:56:56.927) : // CASE TC001_SequenceNumberTest FINISHED WITH PASS

//////////////////////////////////////////////////////////////////////
/// TEST CASE TC001_SequenceNumberTest COMPLETE VERDICT PASS
```

[2013-08-29 05:06:12.211] T_RFC_MESSAGE

Field Name	Expected Value	Received Value
HEADER_Recei...	3	3
HEADER_Recei...	0..255	3
HEADER_Send...	2	2
HEADER_Send...	0..255	2
HEADER_Recei...	0 .. 65535	57
HEADER_Send...	0 .. 65535	58
HEADER_DataL...	0 .. 65535	12421

Name	Type	Value

Figure 9: Configuration of case study on developed tool, status of testing and testing windows.

The application testing was carried out through the target system, especially the on-board signalling system (OBC). This OBC system is operated with wayside local operation equipment (LCC) like CTC or the ATS system. Figure 9 shows the configuration of application testing using target equipment software called OBC. The developed black-box testing tool is operated in notebook as shown in Figure 9 and OBC equipment interfaced with LCC by WIFI. The other captured windowed located at the bottom of Figure 9 shows the executing windows when the tests are applied.

In applying testing, we carried out two kinds of test, as shown in Figure 10. The first is a conformity test in the case of a stand-alone type of operation of the target test and the other is the interoperability test in the case of the target equipment operated with LCC. The base inputted test case is deduced from MSC (Message Sequence Chart) in the interface protocol. The inputted test case was modified by internal sub-modules and the final test cases were generated and inputted to the real interface channel of the target equipment, and compared the feedback signals from the real interface. Several software errors imbedded in the target system were found using these tests. The effectiveness and ease of use was validated by these tests.

4 Conclusion

Recently, many vital functions of a railway signalling system have been realized by the software program, so it is important to verify the functional safety testing of such vital software. However, commercial tools designed to be used in functional safety verification of embedded software are very complex and difficult to use. Therefore, it is not easily used in functional safety testing of a railway signalling system. In this study, a new black-box testing approach is proposed to perform testing through the target system's actual interface and a pilot was developed. At present, it is being used in railway signalling software in order to verify the developed tool. Unlike other tools, it was identified that this tool can be easily and readily used in the functional safety testing of signalling software. It is expected that this tool may contribute to improve the software safety of railway signalling systems.

References

[1] IEC 62279, "Railway Applications – Software for Railway Control and Protection Systems", 2002.
[2] IEC 61508-3: Functional safety of electrical/electronic /programmable electronic safety-related systems - Part 3 Software requirements, 1998.
[3] Jong-Gyu Hwang, Hyun-Jeong Cho, Hyung-Shin Kim, "Design of the safety assessment tool for train control system software", Journal of the Korean Society for Railway, Vol. 11 Issue No. 2, pp. 139-144, 2008.
[4] M. Fewstar, D. Graham: Software Testing Automation: Effective use of test execution tools, ACM Press, Addison Wesley, 1999.

[5] J.D. Lawrence: Software qualification in safety applications, Reliability Engineering & System Safety, Vol. 70, No. 2., pp. 167-184, 2000.
[6] Korea Railroad Research Institute, "Development of Technology to Improve Operation Efficiency and Safety of Train Operation Based ICT", Annual Research Report of 2nd year, 2012.