Vol. 10, No. 2, pp. 16-20, December 2011

INTERNATIONAL
JOURNAL OF SAFETY
www.kosos.or.kr

Development of Performance Testing Tool for Railway Signaling System Software

Jong-Gyu Hwang* and Hyun-Jeong Jo

Korea Railroad Research Institute, Uiwang-si 437-757, Korea

(Received October 5, 2011; Accepted December 15, 2011)

Abstract : The computer application on embedded system is recently more increased in accordance of the development of computer technology. In this trend, the depending of SW in embedded system, especially railway signaling system, is being increased further. So the testing for the safety of railway signaling system software became more important. Hence, the safety assurance of the vital software on the railway signaling system is very important but yet, not many works have been done. In this paper, we represented the implementation results of development of performance testing tool in railway signaling system. The implemented testing item had referred to the international standards in relation to the software for railway system, such as IEC 61508 and IEC 62279.

Key words : SW performance, SW metric, railway signaling system

1. Introduction

The railway signaling system is being converted to the computer system from the existing electric device, and the depending on software is being increased rapidly. As the software becomes more complicated for the intellectualization and automation in accordance with the development of computer technology, the relative importance took by software in the railway signaling system is being increased further.

Recently, the requirements for safety of railway signaling system software have been standardized internationally by IEC 61508 and IEC 62279 [1][2], and in addition, the atmosphere requiring various software testing and validation activities being required by international standards in relation to this railway system is being made. However, the validation on software mainly depends on documents for the process of development so far, and the quantitative analysis by testing is being accomplished for only an extreme fraction of it [3][4]. In addition, it is an actual circumstance that the study on criteria for testing and validation of software for railway signaling system according to the international standards or the study on technology which corresponds to them are at merely an early stage which is being started just now.

Therefore, it is a very necessary situation not only to validate documents on safety activities required by international standards in relation to the railway system but also to develop concrete technology to cope with the analysis and assessment through software testing. Especially, in the performance testing which is one of the validation items for railway software required by international standards, it is subject to the 'HR : Highly Recommended' condition in the relative international standards [1][2]. The accurate performance testing of software will be possible through automated tool rather than through validation of document or qualitative validation method, but no automated tool which enables the performance testing of software for railway system has been developed at home and abroad yet [5]-[8]. Thus, this paper designed and developed tool for performance testing to validate this software for signaling system.

2. Design of Performance testing tool

Performance testing is the test technique to verify whether a software system satisfies defined performance requirements or not, and it carries out the function to measure the performance of software at the target, after applying average loads and overloads to the software to be tested. The testing with this function was imple-

*Corresponding author: jghwang@krri.re.kr mented as an automated tool, and the following main tests will be performed in this performance testing.

- Overload and stress test
- Response time and memory restriction test
- Other test for performance requirements

It can be said that the main object of this tool development is to enable the monitoring of performance variables while observing load conditions of CPU and memory and changing the load by testing the software of signaling system desired to be developed at the target environment dynamically.

Performance testing module is consisted of the source code analysis module, load generation module, target agent, test controller and result analyzer. The following are main functions for the performance assessment of designed module, and they are as follows if arranged by function of each main component module.

• Source code analysis module: It extracts main variables which affect loads by analyzing given source codes under test automatically.

• Load generation module: It generates test cases for test by using input load generating variables. In case of the load, it makes the average load, maximum load and variable load generated.

• Target agent: It is sampling main monitoring variables through kernel module being executed at the target, and stores the result or transmits it to the target monitor.

• Test controller: It is executed at the host computer, and it sets test modes and monitoring variables, etc. and shows the test result on the screen by connecting it to the target agent. At the time of executing tests, it makes determined input events occurred at the target when external input is required among the test cases made by the load generator.

• Result analysis module: It analyzes whether performance requirements were satisfied or not by processing the monitoring results transmitted by target agent.

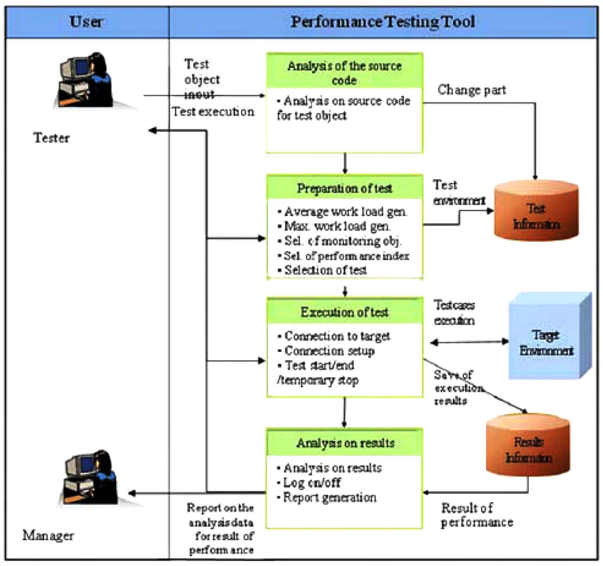


Fig. 1. Operation scenario for performance testing.

The following Fig. 1 is the one showing the outline of operation scenario for performance assessment tool.

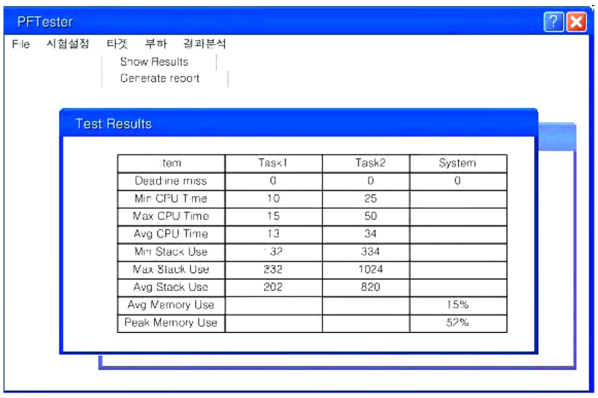
Operation scenario of performance testing tool is as follows.

- Software source code input for the test
- Test kinds setup
- Monitoring object setup
- Work load generation
- Test scenario generation
- Code compilation
- Code downloading into the target and execution
- Target agent download and execution
- Connection to the target by using test controller
- Test start
- Test end
- Test result download
- Offline analysis on the result
- Report generation

Test manager takes over source codes and test information files from the test institution under test, and provides them as the input for performance assessment tool. Performance assessment tool sets the environment and generates execution images for the test by analyzing input files. Test is started by transmitting generated images to the target board, and the data collected during the execution period of test will be transmitted to the monitor program, and the result is made to be analyzed. This performance testing tool designed as followings. Entire screen of program is consisted of the menu, status window and screen by function. Basic menu is consisted of five highest menus such as file, test setup, target, load and the analysis on result, and the submenus are designed by each highest menu. In the file menu, the project file selection, test file selection and program termination can be performed. Project file selects the makefile where source code information for the test is contained. It reads the location of source codes and header files from the makefile. Test file is the text information file provided by the institution under test for the performance test. In this file, monitoring variables, load-related information, performance assessment index, response time and test location, etc. are contained, and new items can be defined in accordance with the addition of function later on.

Test setup menu item can set main items for performance test, generate test source codes, and build the binary images possible to be executed at the target. In the Open Test Setup menu, it shows test items established currently or established as default on the screen. User can change established items displayed on the screen, and the changed contents are stored by using the Save Test Setup menu. When using the test code insert, final source codes for the test are generated by using the established test items. Generated source codes generate binary images by compiling through using the Build Target Image menu. In the target menu item, it can perform the function for controlling target boards. Down side of the screen shows the message on result of executing target control commands. By using the Connect Target Agent command, the target agent and performance testing tool are connected through Ethernet port. It can be used for checking the simple target operation status. Download Test Image menu transmits selected binary images to the target. By using the Run menu, the image downloaded at the target is executed and the test is started. Stop menu transmits the Test Stop command to the target. Transfer Results menu is the command requiring to transmit the test result data being stored by the target agent to the performance testing tool of host.

In the load menu, it contains menus to provide external loads to the target board. Show Loads menu displays load information established currently on the screen. Generate Load menu makes the target connected by driving the load generator implemented at the performance testing tool. Its own load generator transmits data to the specific port in accordance with the established load information. Spawn External Load menu calls external load generating program provided by the



Item	Task1	Task2	System
Deadline miss	0	0	0
Min CPU Time	10	25	
Max CPU Time	15	50	
Avg CPU Time	13	34	
Min Stack Use	132	334	
Max Stack Use	232	1024	
Avg Stack Use	202	820	
Avg Memory Use			15%
Peak Memory Use			52%

Fig. 2. Results of performance measurement by task.

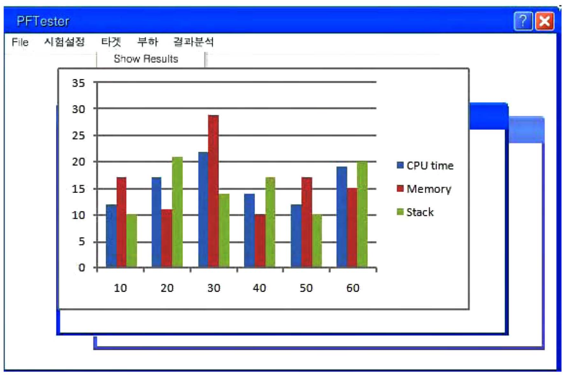


Fig. 3. Results of performance measurement by using graph.

institution under test. Stop menu stops the entire load generating tools being operated currently. Analysis On Result menu includes the function displaying the result on a screen by analyzing the result file of performance measurement transmitted from the target. The display of result is consisted of the case where it is displayed by summary-type fig. 2 and the case where the performance index being changed by task and by time is displayed by graph fig. 3.

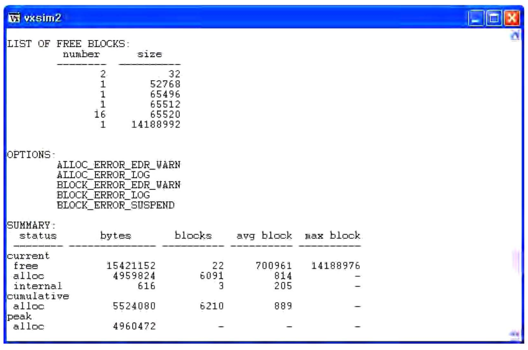
Program analysis function is consisted of the UI processing module and source code analyzer. Results of program analyzer are task structures, load information and the response time measurement information, etc., and they are used for additional test code insert modules by being connected with the result of test information file analysis modules. Test setup function inserts additional codes into source codes by using the information in relation to the monitoring object and internal load generation by using the program analyzer, the result of test information file analysis module, and the content input by the user through UI. If source codes for final test are prepared as a result of the additional code generation module, the binary images will be built in the test code build module. Target control function is implemented by the target connection, code download and test control modules by using the target communications module and UI. Target connection module connects to the target agent by using TCP/IP sockets and serial ports. Test binary images are loaded to the memory of target through code download module. Test control module processes commands such as the test start, test end and the transmission of result, etc. by communicating with target agent. Result management function performs the function displaying its result through tables and graphs on the screen by analyzing the result file transmitted from the target.

3. Development of Performance testing tool

In the real-time operation system, it is possible to check the information in relation to the task through scheduler and TCB (task control block) being managed by the operation system. Since the implemented monitoring task is operated by the kernel mode, it is possible to obtain the necessary information by approaching to the main data structure of kernel directly. By using API which collects information of task, in the performance monitoring task, the information on whole tasks in an executing or blocked status can be checked in the current real-time operation system. The system resource use rates measured at the current performance monitoring system are CPU resource use rate and the memory resource use rate. CPU resource use rate of the task is recorded at each task located at the TCB whenever ticks are counted every time at the operation system. Thus, CPU use rate can be checked through the tick value by each task at every tick value of entire TCB of the whole. To calculate CPU use rate can be included in the function which collects actual task information, but since it has the higher time complexity and work overhead compared with those for other performance monitoring information, it was implemented so as to make it collect information separately. Memory use rate provides the following information.

- Present condition of memory being used currently or available
- Present condition of memory currently allocated dynamically
- Present condition of memory cleared after having been allocated dynamically

The following Fig. 4 is the content displaying collected results on the screen while executing the monitoring task on present condition of using memories at

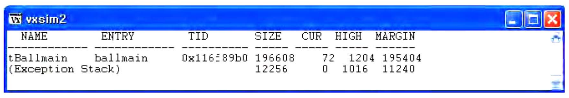


```
LIST OF FREE BLOCKS:
        number      size
        --------  ----------
               2          32
               1       52768
               1       65496
               1       65512
              16       65520
               1    14188992

OPTIONS:
        ALLOC_ERROR_EDR_WARN
        ALLOC_ERROR_LOG
        BLOCK_ERROR_EDR_WARN
        BLOCK_ERROR_LOG
        BLOCK_ERROR_SUSPEND

SUMMARY:
  status        bytes       blocks    avg block   max block
 -------- -------------- ---------- ---------- ----------
current
 free          15421152         22     700961   14188976
 alloc          4959824       6091        814          -
 internal           616          3        205          -
cumulative
 alloc          5524080       6210        889          -
peak
 alloc          4960472          -          -          -
```

Fig. 4. Results of monitoring task on present condition using target memory.



```
  NAME         ENTRY         TID        SIZE   CUR  HIGH  MARGIN
------------ ------------ ---------- ----- ----- ----- ------
tBallmain    ballmain     0x116:89b0 196608    72  1204 195404
(Exception Stack)                    12256      0  1016  11240
```

Fig. 5. Screen of monitoring function on present condition using task stack.

the target. It outputs information such as entire memories of target system, free memory and the allocated memory, etc. Fig. 5 is the one displaying the result of monitoring on the screen after performing the monitoring function on present condition of using task stacks at the target.

4. Conclusion

Recently, according the development of computer technology, the dependency of railway systems on the computer software has been increased rapidly, and the high reliability and safety for vital railway software are required in accordance with this development of technology. Accordingly, for the accurate performance testing on the vital signaling system embedded software which is required by international standards as a highly recommended matter and one of the validation items, this paper presented an automated tool which was developed for the first time at home and abroad. First of all, the functional design of developed tool for performance testing on the software dedicated to the railway was explained, and the result of its implementation was shown concretely. The tool for performance testing like this expresses results of testing on overload and stress, response time and memory restriction, others for performance requirements as the result of monitoring the present condition using target memory systems and task stacks from the viewpoint of graph so that they can be grasped easily in position of user. Users are able to check easily collection of task information, system resource use rate, memory resource use rate and results of monitoring the present condition using target memory system and task stacks. Basically, this automated tool for performance testing on the railway software is the tool to be utilized at the software validation and maintenance stage, and at the same time, it is regarded that its degree of utilization can be sufficiently high at the software development process also. If this tool is used widely at the software validation and development stages, it will be able to contribute in securing the safety and reliability greatly by preventing the error of vital software for signaling system though it before anything happens.

REFERENCES

[1] *Railway Applications - The specification and demonstration of RAMS*, IEC 61508, 1988.

[2] *Railway Application – Software for railway control and protection systems*, IEC 62279, 2002.

[3] Yong-Yoon Cho, Jong-Bae Moon, and Young-Chul Kim, *A System for Performance Evaluation of Embedded Software*, World Academy of Science, Engineering and Technology 1 2005.

[4] Bart Broekman and Edwin Notenboom, *Testing Embedded Software*, Addisson-wesley, Dec. 2002

[5] J. D. Lawrence, *Software qualification in safety applications*, Reliability Engineering & System Safety, vol. 70, no. 2, pp. 167-184, 2000.

[6] Avritzer A., Kondek J., Liu D., Weyuker E.J., *Software performance, testing based on workload characterization*, Proc. of the 3rd international workshop on software and performance, pp. 17-24, Jul. 2002.

[7] M. Fewstar, and D. Graham, *Software Testing Automation: Effective use of test execution tools*, ACM Press: Addison Wesley, 1999.

[8] Denaro G., Polini A., Emmerich W., *Early performance testing of distributed software applications*, Proc. of the 4th international workshop on software and performance, pp. 94-103, 2004.