Transactions of the Korean Nuclear Society Autumn Meeting
Gyeongju, Korea, November 2-3, 2006

Module Testing Techniques for Nuclear Safety Critical Software Using LDRA Testing Tool

Kwon-Ki Moon, Do-Yeon Kim, Hoon-Seon Chang, Young-Woo Chang, Jae-Hee Yun, Jee-Duck Park, Jae-Hack Kim
Korea Power Engineering Company Inc.,150 Dukjin-dong, Yuseong-gu, Daejeon, Korea

1. Introduction

The safety critical software in the I&C systems of nuclear power plants requires high functional integrity and reliability. To achieve those requirement goals, the safety critical software should be verified and tested according to related codes and standards through verification and validation (V&V) activities.

The safety critical software testing is performed at various stages during the development of the software, and is generally classified as three major activities: module testing, system integration testing, and system validation testing. Module testing involves the evaluation of module level functions of hardware and software. System integration testing investigates the characteristics of a collection of modules and aims at establishing their correct interactions. System validation testing demonstrates that the complete system satisfies its functional requirements [1].

In order to generate reliable software and reduce high maintenance cost, it is important that software testing is carried out at module level. Module testing for the nuclear safety critical software has rarely been performed by formal and proven testing tools because of its various constraints.

LDRA testing tool is a widely used and proven toolset that provides powerful source code testing and analysis facilities for the V&V of general purpose software and safety critical software. Use of the toolset is indispensable where software is required to be reliable and as error-free as possible, and its use brings in substantial time and cost savings, and efficiency.

2. Module testing techniques and results using LDRA testing tool

In this section, the techniques and the results used to test modules for the nuclear safety critical software are described.

2.1 LDRA testing tool overview

LDRA Testbed is a powerful and fully integrated tool suite, which enables advanced software analysis techniques to be applied at key stages of the software development lifecycle. It has evolved over a number of years to run on a range of platforms, to analyze code developed in a wide variety of languages. It provides powerful analysis facilities applied in the two main testing domains of static analysis and dynamic analysis. Static analysis analyzes the code and provides an understanding of code structure. Dynamic analysis involves execution with test data, through an instrumented version of the source code, to pinpoint defects at run time [2].

TBrun is the automated LDRA test harness generator and automatically generates test drivers and harnesses (wrapper code). It facilitates the execution of source code procedures or functions (modules) with test data. It enables users to specify inputs to the modules under test, and accept or reject the outputs generated when executing them with test data. Code coverage for the test can also be measured [3].

Standard LDRA Testbed results are available for the source files containing modules from the LDRA Testbed results menus. Figure 1 shows the architecture of the combined LDRA testing tool.

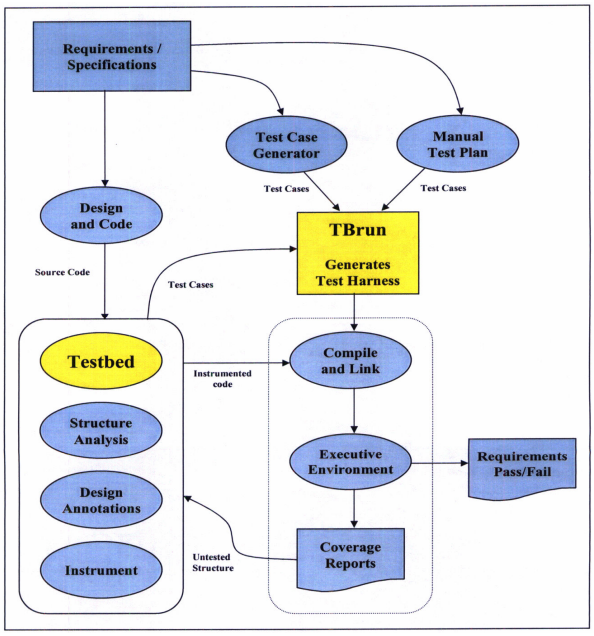

Figure 1. Architecture of the combined LDRA testing tool

2.2 Module testing methods

The objective of module testing is to find bugs in individual modules at the early stage of software lifecycle. Module testing requires a significant resource [4].

Module testing using LDRA Testbed is performed using the white-box or black-box analysis. White-box

analysis deals with the detailed structural analysis of the code under test by utilizing coverage analysis to measure the effectiveness of the test data in exercising the module under test. Black-box analysis deals with the testing of code based on its externally perceived interface. The tester presents inputs to the module, then examines the outputs, and need not know any of fine details of the structure of the code [3,5].

2.3 Module testing procedures

Module testing using LDRA Testbed is performed under Window NT environment. The various tests are performed as the following procedures: 1) create the shell file for the test module using script file under MS DOS command environment; 2) compile the created shell file using Microsoft Visual C++ 6.0 compiler; 3) launch and execute LDRA Testbed; 4) generate the test harness using TBrun; 5) move test cases file into LDRA Testbed environment using script file under MS DOS command environment; 6) execute TBrun; 7) when done, the results will appear in the lower left windowpane.

2.4 Module testing results

After the successful completion of the module testing, two kinds of test reports are generated for check and analysis purposes. They are dynamic coverage analysis report and TBrun regression report. Dynamic coverage analysis report is the output of white-box analysis and TBrun regression report is the output of black-box analysis.

Figure 2 shows dynamic coverage analysis report. It represents the overall coverage results by the percentage of coverage for all statements or branches in the module. "Pass" with green color in the overall coverage results of the report means that the coverage testing in the module is successful. In case of "Fail" with red color in the overall coverage results of the report, the module and test cases are required to be analyzed in detail, and modified according to the circumstances.

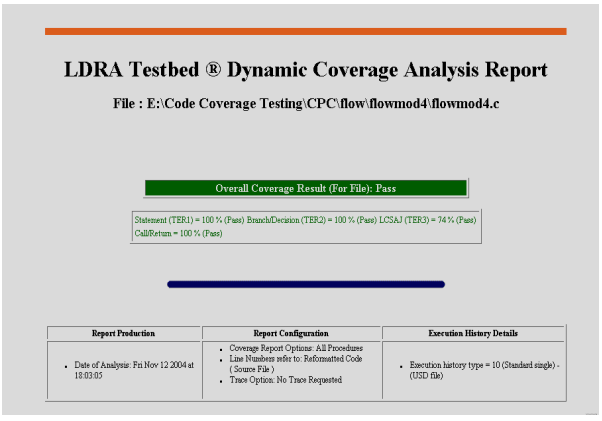

Figure 2. Dynamic coverage analysis report

Figure 3 shows TBrun regression report. It represents the overall results by black-box analysis of the module testing. "Pass" with green color in the overall results of the report means that the outputs at the module testing completion are the same with the expected results included in test cases, and black-box analysis of the module testing is successful. In case of "Fail" with red color in the overall results of the report, the module and test cases are required to be analyzed in detail, and modified according to the circumstances.

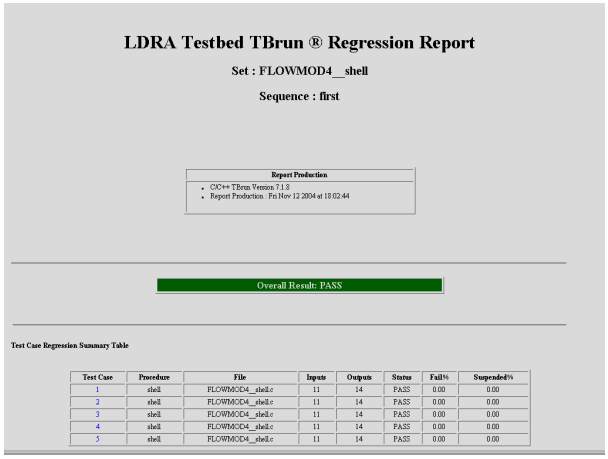

Figure 3. TBrun regression report

3. Conclusion

The nuclear safety critical software is required to be verified, validated, and tested from the early stage of software lifecycle according to related codes and standards.

LDRA testing tool is widely used and a proven toolset that provides powerful source code testing and analysis facilities for the V&V of the safety critical software. It also enables advanced software analysis techniques to be applied at key stages of software lifecycle. This study has shown that testing efficiency in the module code and test arena can be greatly improved using LDRA testing tool, compared to manual techniques.

LDRA testing tool is easy to use and automatically produces accurate and up-to-date detailed displays and reports which enable module testing results to be documented for the V&V of the nuclear safety critical software.

REFERENCES

[1] Neil Storey, Safety-Critical Computer Systems, 1996
[2] LDRA Ltd., LDRA Testbed Technical Description, 2000
[3] LDRA Ltd., TBrun Manual, 2003
[4] Steven R. Rakitin, Software Verification and Validation: A Practitioner's Guide, 1997
[5] Ian Sommerville, Software Engineering: 6th Edition, 2001