A Method for Module Testing of Complex Safety Critical Software for Nuclear Power Plants

Sung Ho Kim*, Do Young Oh, Chang Ho Kim, Woo Goon Kim, Se Do Sohn
I&C System Engineering Department, Korea Power Engineering Co.
150 Duckjin-dong, Yuseong-gu, Daejeon 305-323
**Corresponding author: shkim9@kopec.co.kr*

1. Introduction

Software to be used for safety critical systems of nuclear power plants are developed based on a strict development process. Testing is one of the important development activities. Several steps of tests are taken for high degree of reliability to perform protection functions. This kind of software generally has complex structure and consists of tens or hundreds of modules. Each module developed needs testing on basic (module) level and integrated (unit) level. The software modules need to be tested by the developers during development step to reduce burden of defect finding-correction-regressive testing process during test step. If we consider that development environment is totally different from the target system, several points should be taken into considerations for the tests during development step and test step. This is to reduce the gap between those two environments for effective testing. In this paper, the test environments to check each module's logic during development step and test step are highlighted, and the differences in those two environments are analyzed. Resolutions to overcome those differences are presented.

2. Software Development and Tests

Software for safety critical system is developed based on one of the software life cycle models, and a widely used V-model is selected (Fig. 1).

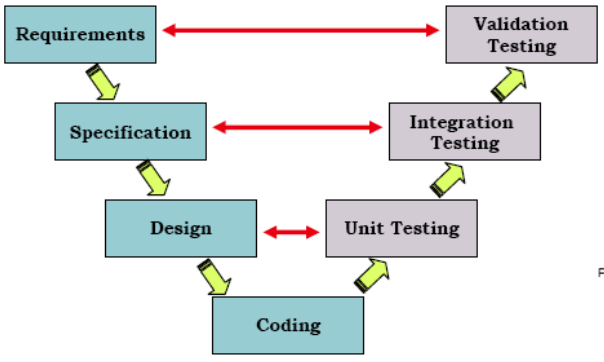

Fig. 1. A Software Development Model (V-Model)

Software is designed and documented from the requirement to the design phases. Programs are coded during the implementation phase and tested during test phase. Fig. 2 shows the software development environments of implementation and test phases.

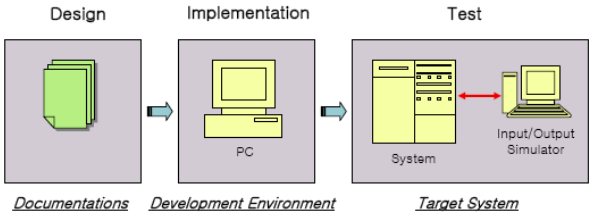

Fig. 2. Development Steps for Safety Critical Software

2.1 Software Development

Software is developed on personal computer environment having dedicated development tool provided by the platform vendor. Software architecture is decided under development environment. Documents developed for the design phase are used to implement the software. No formal testing is performed to the coding step, since verification and validation test are performed through the implementation and the test phases by the group independent of the development group.

2.2 Tests in Development Environment

Software developed under development environment is highly recommended to be tested by the developers. Logic (algorithm) incorporating the functional requirements can be checked per software module by the developers.

During the developer's module test step, static inputs and outputs are applied to each module on the personal computer. Commercial tools can be used to simulate the running conditions of the modules. Visual Studio® and Microsoft Excel® were used on the development environment in this project. Fig. 3 shows the developer's module test environment before formal module test.

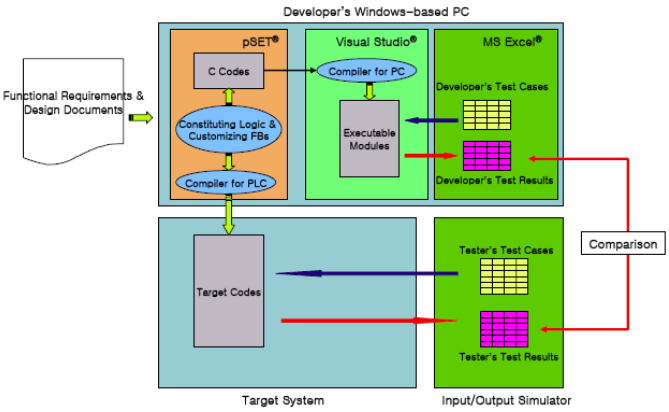

Fig. 3. Developer's Test Environment on PC

Developer's module test in development environment is very important before going ahead for formal module test by the independent verification and validation (V&V) group, since the functional requirements assigned to each module can be checked before being tested on target system. This facilitates debugging of the software and reduces the efforts for testing-exception finding process.

2.3 Tests on Target System

The main purpose of module tests on target system is to check the following features of each module:

- module logic (algorithm implementation)
- deviation in execution environment (proper function calls, etc.)
- data control (data conversion, etc.)

Test on target system for the software modules is one of the formal activities commonly followed by the development groups. A simulator is used to apply test cases to the target system and read the execution results from the target system. The software of the simulator needs to be customized to perform the module test. Interfaces between the target system and simulator are specially staged considering the input signal features to the target system. Interfaces among software modules on the target system are not tested during the module test. Logics are checked during the module test. Software defects found during this test are corrected and retested afterwards. All software modules need to pass the test acceptance criteria to go ahead for the integration (unit) test.

3. Considerations in Tests

There exist major environmental differences between the development environment and the target system as follows:

- different processors (CPU's)
- different compilers and compiling options

Special considerations should be taken to overcome those environmental differences between development and test steps as follow.

3.1 Considerations for Tests in Development Environment and on Target System

All the software modules are loaded onto the target system one by one to be tested. The target environment provides the same hardware and software configuration for actual operation. Since the personal computers used for software development are commercial-off-the-shelf products, the processors of these personal computers have different data processing features from those of target system. That is to say, the target system used in this project adopts TI DSP chips, and they follow IEEE 754 standard for floating point processing, which will result in different module test outputs for the development environment and the target system. This means that special function is coded and called by the software module during the execution on the target system. Test of the functionality for the special function is also required.

3.2 Influences of Test Support Features

Input and output simulator is used to apply inputs to the target system and get the test results. The simulator has different environment from the target system, and hence needs adjustment of data types. For this project, NI DAQ® board is used for sending and receiving data to and from the target system. Interface between these two environments is implemented using HDL serial link, which requires additional coding of interface function block modules for appropriate testing. Software control flags are also needed to apply many test case sets simultaneously. Consideration to overcome the differences of data resolutions between the target system and the simulator should also be taken.

3.3 Recommendation of enhanced module test

Module test requires many efforts for fault-finding of the software modules. Corrections of function blocks and test cases occur many times during the tests. Developer's test during development environment before the release of the software for the formal module test is very effective to reduce the required resources and test period. The differences in the features of development environment and the target system will decide the easiness of developer's test, but essentially reduce the overall efforts to successfully develop the safety critical software.

4. Conclusions

Complex safety critical software for nuclear power plants needs to be tested through several steps. Developer's test can, before verification test, reduce the trial-and-errors during test phase activities. For developer's test for the software modules, commercial tools such as Microsoft Visual Studio® and Microsoft Excel® can be used on the PC-based development environment. The environments of development and test are different from each other, and hence, it is needed to consider key factors such as data operation for applying test cases. The efforts of developers for test before verification test will also increase the reliability of the software system.

REFERENCES

[1] IEEE Std 754-1985, IEEE Standard for Binary Floating-Pont Arithmetic.
[2] John Robbins, Debugging Applications for Microsoft .net and Microsoft Windows, Microsoft Press, 2003.