Transactions of the Korean Nuclear Society Spring Meeting
Chuncheon, Korea, May 25-26 2006

The Testing Strategy for the Embedded Software implemented in I/O module of KNICS PLC

Jong Gyun Choi, Won Man Park, Dong Young Lee,
Korea Atomic Energy Research Institute
choijg@kaeri.re.kr, wmpark@kaeri.re.kr, dylee2@kaeri.re.kr.

1. Introduction

The safety Grade PLC (POSAFE) is being developed in the Korea Nuclear Instrumentation and Control System (KNICS) R&D project. The PLC is being designed for satisfies Safety Class 1E, Quality Class 1, and Seismic Category I. The embedded software for implementation in I/O module such as the pIAOS and pOAOS is being developed according to the safety critical software life cycle. The developed software according to the software life cycle is tested for verification and validation by an independent software testing team. This paper describes the software testing strategy to find the faults that may exist in software design and code effectively.

2. Development of a Safety PLC

The POSAFE is a PLC being developed according to the safety equipment design procedure class to apply it to the safety system in a nuclear power plant. The design, V&V, development process and quality assurance (QA) according to the configuration management of the POSAFE hardware and software satisfy the 10CFR50, Appendix B requirements.

POSAFE is composed of various modules used in the safety system of a nuclear power plant such as a sub rack, power module, processor module, communication module, digital input/output module, analog input/output module, and a high speed pulse counter module [1].

The dimensions of the POSAFE sub rack are 482.6 x 281.35 x 294mm, which satisfies the 19 inches standard and can be used with a combination of several input/output modules and communication modules.

The sub rack uses 2 independent power modules (PWR1&2), each power module has the 100% power supply capability needed for a sub rack. Accordingly, even though there is a fault in a power module, it does not affect the operation of POSAFE. An Extension sub rack is connected to the sub rack through a local bus extension module.

POSAFE provides an engineering tool (pSET) which enables users to develop application programs. The developers of the application programs can perform a programming, debugging, application program simulation and a writing of program related documents by using the pSET. The pSET is operated in Windows 2000/NT, it provides standard functions and 3 languages (LD, FBD, SFC) presented by IEC 61131-3, and it satisfies the NUREG-CR6463 criteria.

The POSAFE hardware satisfies the IEEE standard Safety Class 1E. And it is classified into seismic category I.

3. Software Design and V&V

The embedded software such as the pCOS, pIAOS and pOAOS are being developed according to the safety critical software life cycle. The software of the POSAFE is developed according to a software developing plan and procedure [2]. This plan and procedure is in accordance with USNRC Reg. Guide 1.173 and IEEE Std. 1074. Especially, the formal method is applied to design the SRS (Software Requirement Spec.) and the SDS (Software Design Spec.) to be error-free.

The developed software according to the software life cycle is verified by an independent software verification and validation (V&V) team [3]. The software of the POSAFE is verified according to a software V&V plan. This V&V plan and the procedure are in accordance with USNRC Reg. Guide 1.172 and IEEE Std. 1012.

As a part of V&V activities, the software testing is a task that executes software (program) with intent of finding errors in software design and code because errors uncovered through the testing can make the system unsafe. Therefore, it is very important to devise an effective software testing strategy for finding as many software errors as possible.

4. Software Testing Strategy

In this section software testing strategy used for finding errors in pIAOS and pOAOS is described. pIAOS and pOAOS are the software designed for analog input module and analog output module of KNICS PLC respectively [4, 5].

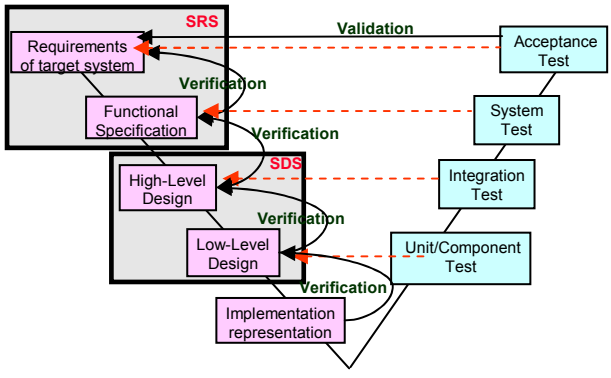

Figure 1. Software Testing Model

Figure 1 show the software testing model used for pIAOS and pOAOS testing. The software testing model includes the unit (component) test, integration test, system test and acceptance test. The software unit test is process of testing the individual subprograms, subroutines in a program. That is, rather than initially testing the program as a whole, testing is first focused on the smaller building blocks of the program. The software integration test is a process of testing interfaces between subprograms, interactions between program and hardware, and finally interfaces between program and external system. The system test and acceptance test is out of scope of this paper.

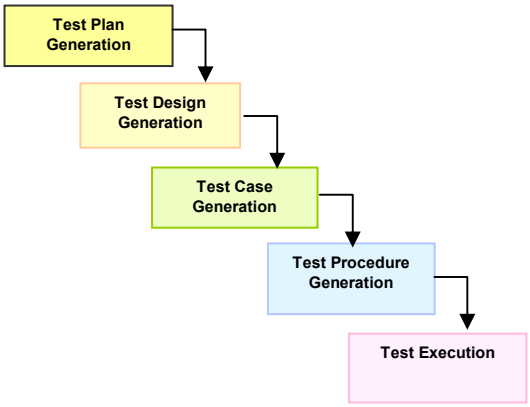

Figure 2. Software Test Procedure

Fgire 2 shows the software test procedure that should be performed in unit test and integration test.

4.1 Software Unit Test Strategy

The bottom-up testing sequence is adopted for pIAOS and pOAOS unit test [6, 7]. That is, we test the program from the bottom [8]. If the program has the structure as shown in figure 3, the sequence of unit test is (E, C, F), (B, D), (A).

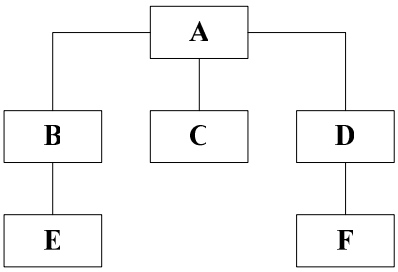

Figure 3. The Structure of Example Software

The test items include all units (subprograms) consisting of pIAOS and pOAOS and the testing methodology used is the combination of white box and black box test.

The test cases for testing each units are derived from satisfying the multi-conditions coverage criteria, and a boundary value analysis.

We stop the unit test when all the test cases execute the without detecting errors.

4.2 Software Integration Test Strategy

The test item for integration test of pIAOS and pOAOS includes the interface as follows:

- Interface between each units
- Interface with hardware components in AI/AO module
- Interface with pCOS of processor module

The test cases are derived through the equivalence partitioning method which is one of black box test methodology. Test case design by equivalence partitioning method proceeds in two steps:

- To identify the equivalence classes
- To define the test cases

We stop the integration test when all the test cases execute the program without detecting errors.

5. Conclusion

This paper described the software testing strategy which can find the errors effectively in the pIAOS and pOAOS designed for implementation in analog input module and analog output module respectively. The unit test and integration test were performed according to the proposed test strategy.

We found two errors in pIAOS through the unit and integration test. We also find three errors in pOAOS.

The information about errors was reported to software development team and QA team so that the software may be redesigned.

REFERENCES

[1] KNICS-PLC-DS301, “Safety Grade PLC Design Specification,” 2006.
[2] KNICS-PLC-SEP102, “Safety Grade PLC Software Development Plan,” 2005.
[3] KNICS-PLC-SEP110, “Safety Grade PLC Software Verification and Validation Plan,” 2005
[4] KNICS-PLC-SRS131-02, “Safety Grade PLC Analog Input Module Software (pIAOS1) Requirement Specification,” 2006.
[5] KNICS-PLC-SRS131-03, “Safety Grade PLC Analog Ouput Module Software (pOAOS1) Requirement Specification”
[6] KNICS-PLC-STP131-02, “Safety Grade PLC Analog Input Module Software (pIAOS1) Component Test Procedure,” 2006
[7] KNICS-PLC-STP131-03, “Safety Grade PLC Analog Output Module Software (pOAOS1) Component Test Procedure,” 2006
[8] Glenford J. Myers, “The Art of Software Testing,” John Wiley & Sons, 2004.