An Integrated Test Design for Information Processing System of MMIS

Jaekwan Park [a*], Sangmun Seo [a], Yongsuk Suh [a], Cheol Park [a]
[a] *Korea Atomic Energy Research Institute (KAERI), Daedeok-Daero 989-111, Yuseong-Gu, Daejeon, 305-353, Korea*
[*] *Corresponding author: jkpark183@kaeri.re.kr*

1. Introduction

As advances in information technology, fully digitalized Man Machine Interface System (MMIS) has been applied to nuclear power plants and research reactor facilities. Generally, MMIS systems are verified and validated by several test activities such as unit test, software integration test, system test, and factory acceptance test [1-3] during system development process.

Power plant or research reactor facility is furnished with Information Processing System (IPS) conducting the role of information layer between control layer and operation layer of MMIS. The IPS is connected with other control systems, monitoring systems, and protection systems. The system collects thousands of interface signals through the Ethernet-based connections. Test cases for the IPS in the MMIS integration test (i.e., factory acceptance test) should include functional tests of the system interface, alarm processing, data management, network management, and primary information monitoring. This paper proposes an efficient method planning the test procedure of the system by considering the test coverage in the final test activity, factory acceptance test (FAT).

2. Test Design for Factory Acceptance Test

To establish FAT test environment of the IPS, MMIS systems should be connected physically in the same manner of the site installation as shown in figure. 1. Test procedure of the integration test should be prepared to verify following functions in detail:

(1) **System Interface Test**: signal interface test including digital communication with safety systems and non-safety systems
(2) **Primary Monitoring Function Test**: post-accident monitoring (non-safety), safety parameter display, bypassed and inoperable status indication, and alarm presentation
(3) **Alarm Test**: important alarm, system alarm, alarm list, alarm control flow, and alarm display control between display devices
(4) **Network Function and Performance Test**: performance monitoring of non-safety digital communication network
(5) **Data Management and Backup Test**: backup data or system information automatically and manually
(6) **Failover Test**: responding against single failure of system component or network component

Fig. 1. An Example of Test Environment of Research Reactor MMIS

Considering test item (1), Interface signals from the source system to the destination system have different flow paths. For example, the source system of a trip parameter, *PCS Flow Lo*, is the reactor protection system and the destinations are (a) an information display to inform the trip cause, (b) an alarm display to alert operators with reactor trip, and (c) control systems to inform change of reactor operation status to them. Thus, the interface test should be carefully designed to verify that information is transmitted to expected destinations properly. Figure 2 shows an example of information flow in point of the IPS system.

The interface signals are not tested at all before the final integration test and their flow paths are not considered as test item in other MMIS systems in the integration test. Thus, this paper recommends that all interface signals are selected as test item and verified by test procedure of the IPS. It is most important one of test items of the IPS in the integration test.

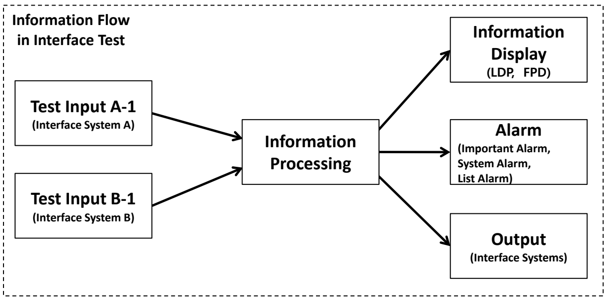
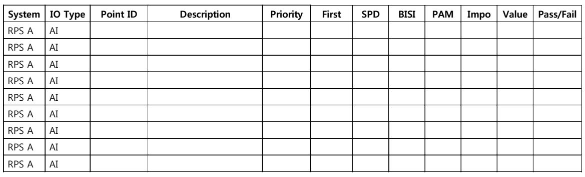


Fig. 2. An Example of Information Flow in the IPS

Next, a useful method for the interface test should be designed. There are thousands of interface signals at the system. It is not efficient to iterate above test items (1), (2), and (3) on the interface signals respectively. Actually, generating an input signal value at interface system is not simple and time-consuming task. Thus, it recommends making the test procedure based on the interface signal not based on the test item. Figure 3 shows a simple example of the test format executing tests (1), (2) and (3) together on each interface signal.



System	IO Type	Point ID	Description	Priority	First	SPD	BISI	PAM	Impo	Value	Pass/Fail
RPS A	AI										
RPS A	AI										
RPS A	AI										
RPS A	AI										
RPS A	AI										
RPS A	AI										
RPS A	AI										
RPS A	AI										
RPS A	AI										

Fig. 3. A Test Format of Interface Signals

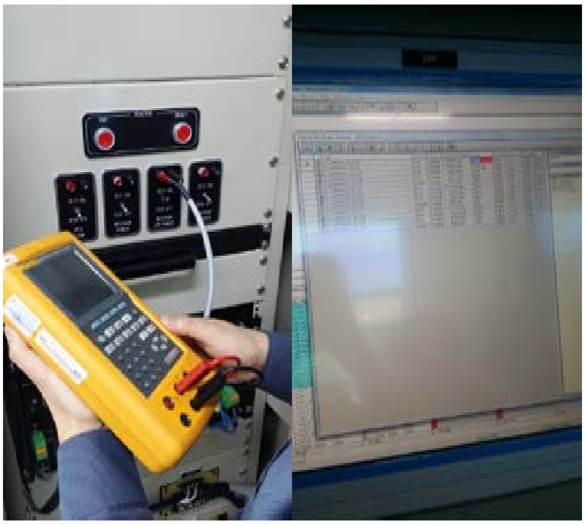


Fig. 4. Practices of Signal Generation at Interface Systems

As shown in Fig. 2, the test signals should be generated in the interface systems. Two methods for signal generation are applicable as follows:

- **Raw Signal Generation**: normally, a value within 4~20mA (for analog signal) given at an analog input port of the interface system (safety system)
- **Force Input**: making a force input at the interface system by using the maintenance computer installed in the system cabinet (non-safety system)

Figure 4 shows the signal generation practices used in the interface test.

Generally, the tests (4), (5), and (6) are done once in the previous system level test (i.e., system test). In this integration test, the tests are conducted again to find abnormal situations in whole MMIS level.

3. Conclusions

Factory acceptance test is the final activity of the development process to verify the functional and performance completeness of the system in the MMIS level. The IPS system has thousands of signal interfaces and the interface test using real signals can be only conducted in the MMIS FAT. This paper recommends that the interface test is designed as full interface test and the test steps are arranged based on the signal. It helps that data processing, recording, and display functions are easily verified.

There was a FAT for the IPS of MMIS for a research reactor at Feb. 2015. We applied our test method to the FAT. As result, we found tens of abnormal results in the test and corrected all the faults. After the test, the system stability was very improved.

Acknowledgement

This work was supported by Korea MISP (Grant Code: 2012M2C1A1026912).

REFERENCES

[1] IEEE Std. 1008-1987, Software Unit Testing, IEEE Software Engineering Technical Committee, Piscataway, NJ, 1987.
[2] IEEE Std. 829-1998: Software Test Documentation, IEEE Software Engineering Technical Committee, Piscataway, NJ, 1998.
[3] IEEE Std. 1012-2004, Software Verification and Validation, IEEE Software Engineering Standards Committee, Piscataway, NJ, 2004.